UNIVERZITA MATEJA BELA
Fakulta prírodných vied, Katedra informatiky

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta přírodovědně-humanitní a pedagogická

MEDZINÁRODNÁ KONFERENCIA O VYUČOVANÍ INFORMATIKY

DidInfo 2019

25. ROČNÍK KONFERENCIE

ISBN: 978-80-557-1533-9
ISSN: 2454-051X

BANSKÁ BYSTRICA
2019

Predsedovia programového výboru/General Chairs:
doc. Ing. Jarmila Škrinárová, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
doc. PaedDr. Jiří Vaníček, PhD. *Jihočeská Univerzita, České Budějovice, CZ*

Programový výbor konferencie/Programm Commitee:
doc. RNDr. Gabriela Andrejková, CSc. *Univerzita Pavla Jozefa Šafárika, Košice, SK*
Sonya Armstrong, PhD. *West Virginia State University, USA*
Mgr. Jan Berki, PhD. *Technická univerzita v Liberci, CZ*
Mgr. Martin Cápay, PhD. *Univerzita Konštantína Filozofa, Nitra, SK*
doc. RNDr. Miroslava Černochová, CSc. *Univerzita Karlova, Praha, CZ*
prof. Dr. Valentina Dagiene *Vilnius University, LT*
Ing. Dana Horváthová, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
prof. Dr. Mirjana Ivanovic *University of Novi Sad, SR*
Ing. Jana Jacková, PhD. *Katolícka univerzita, Ružomberok, SK*
prof. RNDr. Ivan Kalaš, PhD. *Univerzita Komenského, Bratislava, SK*
doc. RNDr. Zuzana Kubincová, PhD. *Univerzita Komenského, Bratislava, SK*
doc. RNDr. Gabriela Lovászová, PhD. *Univerzita Konštantína Filozofa, Nitra, SK*
Ing. Božena Mannová, PhD. *České vysoké učení technické, Praha, CZ*
RNDr. Alžbeta Michalíková, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
RNDr. Pavel Pešat, PhD. *Univerzita J. E. Purkyně, Ústí nad Labem, CZ*
Prof. Dr. Kate Sanders *Rhode Island College, USA*
prof. Ing. Veronika Stoffová, CSc. *Trnavská univerzita v Trnave, SK*
doc. RNDr. Petr Šaloun, PhD. *Technická univerzita Ostrava, CZ*
doc. RNDr. Ľubomír Šnajder, PhD. *Univerzita Pavla Jozefa Šafárika, Košice, SK*
doc. RNDr. Pavel Töpfer, CSc. *Univerzita Karlova, Praha, CZ*
doc. Ing. Ľudovít Trajteľ, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
Dr. Livia Tudor *Petroleum-Gas University, RO*
PaedDr. Patrik Voštinár, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*

Organizačný výbor konferencie/Organizing Commitee:
Ing. Dana Horváthová, PhD.
RNDr. Alžbeta Michalíková, PhD.
PaedDr. Patrik Voštinár, PhD.
Mgr. Adam Dudáš
Mgr. Nika Klimová
Katarína Gavaldová

Recenzenti/Reviewers:
doc. RNDr. Gabriela Andrejková, CSc. *Univerzita Pavla Jozefa Šafárika, Košice, SK*
Mgr. Jan Berki, PhD. *Technická Univerzita v Liberci, CZ*
PaedDr. Ivan Brodenec, PhD. *Úrad podpredsedu vlády SR pre investície a informatizáciu, Bratislava, SK*
Mgr. Martin Cápay, PhD. *Univerzita Konštantína Filozofa, SK*
Mgr. Adam Dudáš. *Žilinská univerzita, Žilina, SK*
Ing. Dana Horváthová, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
PaedDr. Roman Hrušecký, PhD. *Univerzita Komenského, Bratislava, SK*
RNDr. Milan Hudec, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
Ing. Jana Jacková, PhD. *Katolícka univerzita, Ružomberok, SK*
doc. RNDr. Ľudmila Jašková, PhD. *Univerzita Komenského, Bratislava, SK*
prof. RNDr. Ivan Kalaš, PhD. *Univerzita Komenského, Bratislava SK*
Ing. Alžbeta Kanáliková, PhD. *Žilinská univerzita, Žilina, SK*
doc. Mgr. Ján Karabáš, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
Mgr. Nika Klimová. *Univerzita Konštantína Filozofa, Nitra, SK*
doc. RNDr. Zuzana Kubincová, PhD. *Univerzita Komenského, Bratislava, SK*
doc. RNDr. Gabriela Lovászová. PhD. *Univerzita Konštantína Filozofa, Nitra, SK*
Ing. Janka Majherová, PhD. *Centrum vedecko-technických informácií, Liptovský Mikuláš, SK*
RNDr. Miroslav Melicherčík, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
PaedDr. Viera Michaličková, PhD. *Univerzita Konštantína Filozofa, Nitra, SK*
RNDr. Alžbeta Michalíková, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
RNDr. Pavel Pešat, PhD. *Technická Univerzita, Liberec, CZ*
doc. RNDr. Ľubomír Salanci, PhD. *Univerzita Komenského, Bratislava, SK*

prof. Ing. Veronika Stoffová, CSc. *Trnavská univerzita v Trnave, SK*
doc. Ing. Jarmila Škrinárová, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
doc. RNDr. Ľubomír Šnajder, PhD. *Univerzita Pavla Jozefa Šafárika, Košice, SK*
RNDr. Peter Tomcsányi, PhD. *Univerzita Komenského, Bratislava, SK*
doc. PaedDr. Monika Tomcsányiová, PhD. *Univerzita Komenského, Bratislava, SK*
doc. RNDr. Pavel Töpfer, CSc. *Univerzita Karlova, Praha, CZ*
doc. Ing. Ľudovít Trajteľ, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*
doc. PaedDr. Jiří Vaníček, PhD. *Jihočeská Univerzita, České Budějovice, CZ*
PaedDr. Patrik Voštinár, PhD. *Univerzita Mateja Bela, Banská Bystrica, SK*

Editori/Editors: Ing. Dana Horváthová, PhD., RNDr. Alžbeta Michalíková, PhD., doc. Ing. Jarmila Škrinárová, PhD., PaedDr. Patrik Voštinár, PhD.
Copyright © 2019 autori/authors
ISBN: 978-80-557-1533-9
ISSN: 2454-051X
Recenzovaný zborník/Reviewed Proceedings

Tento zborník vznikol s podporou nasledovných projektov a organizácií:

- Projekt KEGA 009KU-4/2017 Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní
- Projekt IT Akadémia IT AKADÉMIA - vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje, ITMS kód projektu: 312011F057
- Slovenská e-akadémia BB, n.o.

Predhovor

Znie to až neuveriteľne, ale v tomto roku sa uskutočňuje už 25. ročník konferencie DidInfo, ktorá je zameraná na žiaka a učiteľa predmetu Informatika. To ma inšpiruje k malému pohľadu do minulosti.

Hoci prvá konferencia mala len regionálny charakter, zavčasu sa stala celoslovenským a postupne aj medzinárodným podujatím, ktorého sa zúčastnili mnohé domáce aj zahraničné osobnosti, ktoré môžeme považovať za „stálice“ konferencie. Veľmi ma teší, že sú tu s nami aj dnes. Rada by som podčiarkla najmä tvorivého ducha a priateľskú atmosféru, ktorú nám tu pomáhate vytvárať.

Na prelome tisícročí, keď sa zavádzal Internet do škôl a veľkú pozornosť sme venovali využívaniu výpočtovej techniky nielen na informatike, ale aj na iných predmetoch (matematika, fyzika, cudzie jazyky, ...).

Ako sa vyvíjalo programovanie, programovacie jazyky a prostredia, tak sa v priebehu rokov menili aj konkrétne ciele konferencie. Všetkými ročníkmi konferencie DidInfo sa vinie stuha vyučovania programovania a algoritmického myslenia. A my sme boli (a stále sme) svedkami overovania nových metodík, príkladov a postupov.

Nasledovala éra programátorských a rôznych robotických súťaží pre žiakov rôznych vekových kategórií. Inšpiratívne boli a sú vystúpenia žiakov, ktorí spolu so svojimi učiteľmi prezentovali výsledky svojej práce.

V poslednom období nastal veľkých rozvoj robotických a mikroprocesorových stavebníc, dronov a iných mobilných, webových a cloudových aplikácií a pomôcok. Mnohé sú cenovo dostupné, obsahujú množstvo senzorov a zaujímavých výstupných zariadení. Sú určené pre rôzne vekové kategórie. Preto našli svoje opodstatnenie na hodinách modernej informatiky. Žiaci a učitelia všetkých vekových kategórií ich majú radi. Prácu na hodinách považujú skôr za hru ako za vyučovanie. Aj na tohtoročnom DidInfe učitelia v hojnej miere prezentujú vytvorené učebnice, metodiky a príklady, vo forme 49 príspevkov a 7 workshopov. Pomocou týchto metodík je možné nielen naučiť sa pracovať s jednotlivými pomôckami a učiť sa ich programovať, ale najmä rozvíjať výpočtové myslenie.

Netreba zabúdať ani na množstvo projektov, s ktorými sme žili, pracovali sme na nich, diskutovali a prezentovali ich na mnohých ročníkoch konferencie. Spomeňme si na niektoré z nich, napr. projekty Tempus, Infovek, Ďalšie vzdelávanie učiteľov informatiky, aktuálna IT akadémia a priebežne veľa projektov KEGA.

Na základe dlhodobej spolupráce a stretávaní sa na konferencii s účastníkmi z Českej republiky sa konferencia DidInfo 2017 spojila s konferenciou DidactIG (ČR). V roku 2018 sa spoločná konferencia DidInfo konala v priestoroch Technickej univerzity v Liberci a veríme, že tomu bude tak aj na budúci rok. Tým by sme naplnili náš plán, podľa ktorého sa konferencia bude konať striedavo na Slovensku (na Fakulte prírodných vied Univerzity Mateja Bela v Banskej Bystrici) a v Českej republike (na Fakulte prírodovedne humanitnej a pedagogickej Technickej univerzity v Liberci).

Apríl 2019

doc. Ing. Jarmila Škrinárová, PhD.

Predsedkyňa programového výboru
konferencie DidInfo 2019

Předmluva

Vzhledem ke kulatému výročí naší konference si dovolím být poněkud osobní a omlouvám se za to. Když jsem před těmi více než dvaceti lety začal na naší fakultě učit didaktiku informatiky, byl to zcela jiný předmět, než jak jej známe dnes, a to jak svým významem v přípravě učitele, tak svou náplní. Byla to ovšem doba velkých výzev, objevů, změn, setkávání s inspirativními lidmi. Když se podívám na dnešní situaci v našem oboru, s potěšením mohu konstatovat, že stále poskytuje velké výzvy, přináší nové objevy a myšlenky a nepozoruji v tomto klesající dynamiku. Máme štěstí, že didaktika informatiky se stále nachází ve stavu, kdy není plně zformována a usazena, že stále poskytuje možnosti a nabídky pro bádání a inovace, že neztrácí na důležitosti a aktuálnosti.

Právě letos je to asi dobře patrné na velkých realizačních projektech inovujících školskou informatiku, právě probíhajících v obou našich zemích. Projekty zavádějí do škol poznatky, které na naší konferenci byly objeveny, prezentovány a v rámci komunity didaktiků informatiky sdíleny, v podobě ucelených kurikulárních materiálů, podstatně formujících vzdělávání v této oblasti. Konference poskytuje základnu pro komunitu učitelů, vědců a inovátorů, kteří netvoří pouze pro poznání samotné, ale mají na očích a na mysli děti, které na školách jsou nebo do nich přijdou. Patří jí za to dík.

Když jsem s výukou didaktiky informatiky začínal, dozvěděl jsem se o této tehdy vznikající konferenci a zaujal mě její název, který mi připadal jako vtipná zkratka. Pojmenoval jsem touto zkratkou svoji sdílenou složku s materiály pro studenty na fakultních učebnách. Později jsem začal na tuto konferenci jezdit, vracet se do Banské Bystrice a zkratka DidInfo se mi spojila s konkrétními lidmi, se zážitky, s fascinujícími myšlenkami, které pro moji hlavu zpřístupnili vystupující na přednáškách a v disputacích. Dodnes tuto složku při výuce používám a pokaždé, když ji otevřu, si vzpomenu právě na milé lidi a zážitky z Banské Bystrice a mé srdce se ohřeje. Věřím, že jsem pak chvilku na své studenty milejší než obvykle.

Vážení kolegové, přeji Vám inspirativní čtení.

Duben 2019

doc. PaedDr. Jiří Vaníček, PhD.

Předseda programového výboru
konference DidInfo 2019

Obsah

Príspevky

Michal Anderle
Porovnanie slovných riešení teoretickej úlohy žiakov strednej školy a riešiteľov súťaže PRASK..10
Gabriela Andrejková, Katarína Kimáková, Anna Mišianiková, Zuzana Tkáčová
Príprava učiteľov na výučbu predmetu Informatika v prírodných vedách a matematike.....16
Daniela Bezáková, Andrej Blaho, Andrea Hrušecká, Roman Hrušecký
Informatika v cirkuse.....22
Miroslav Biňas, Matej Madeja
Využitie minipočítača Raspberry Pi v informatických predmetoch na stredných školách.....26
Andrej Blaho
Python pre stredné školy.....32
Lucia Budinská, Karolína Mayerová
Ako žiaci piateho a šiesteho ročníka základnej školy riešili vybrané grafové úlohy.....35
Iveta Csicsolová, Zuzana Kubincová
Nájdeš chybu v mojom programe?.....41
Martin Cápay, Magdaléna Bellayová
Aktivity s BBC micro:bit podporujúce kolaboratívnu prácu žiakov ZŠ.....47
Martin Cápay, Nika Klimová
Dokáže programovanie hardvéru zaujať študentov?.....52
Rostislav Fojtík
Analýza obsahu výuky informatiky na základních a středních školách.....58
Ján Guniš, Ľubomír Šnajder, Viera Michaličková
Nový informatický predmet Riešenie problémov a programovanie.....62
Dana Horváthová, Miroslav Blšák
Rozšírená realita vo vzdelávaní.....68
Dana Horváthová, Jaroslav Knežník
Blokové programovanie v rámci informatického krúžku.....73
Jana Jackova, Janka Majherová, Patrik Sitiarik, Roman Stančo, Veronika Motýľová
Robotika v neformálnom vyučovaní informatiky.....77
Ľudmila Jašková, Natália Kováčová, Mária Karasová
Dva programovacie jazyky priateľské k nevidiacim žiakom nižšieho sekundárneho vzdelávania..82
Zdeněk Lomička, Zbyněk Filipi
Řešení úloh v blokovém programování jako součást projektového dne založeného na deskoherních principech.....88
Gabriela Lovaszová, Viera Michaličková, Ján Guniš, Ľubomír Šnajder
Nový informatický predmet Programovanie mobilných zariadení: Tvorba projektov.....94
Marek Nagy
Using a text adventure game approach to train GNU/Linux command line skills.....100
Ingrid Nagyová
Automatická kontrola programových kódů ve výuce programování.....104
Jiří Pech, Milan Novák
Použití Lego Mindstorms jako platformy pro výuku robotiky za pomoci Linuxu a Pythonu.....110
Tomáš Průcha, Zbyněk Filipi, Lucie Rohlíková
Problémy v přípravě učitelů informatiky: případová studie.....115

Ľubomír Salanci
Matematická zložka vo vyučovaní programovania.....119
Mária Stankovičová
Informatika bez bariér.....125
Veronika Stoffová, Veronika Gabaľová
IKT vo vyučovaní na prvom stupni základnej školy.....130
Miroslav Svítek
Inovativní výuka chytrých měst.....136
Martin Šechný
Informatika v Slovenskej terminologickej databáze.....140
Václav Šimandl
Ochrana osobních údajů podle GDPR v soutěži Bobřík informatiky.....146
Jarmila Škrinárová, Adam Dudáš
Procesy operačného systému vo vyučovaní informatiky.....152
Ľubomír Šnajder, Ján Guniš, Gabriela Lovászová, Viera Michaličková
Nový informatický predmet Programovanie mobilných zariadení – koncepcia jeho výučby a ukážky etúd.....157
Zuzana Tkáčová
Umelá inteligencia ako súčasť kurikula informatiky na základných a stredných školách.....164
Peter Tomcsányi
Úspešnosť súťažiacich s rôznym počtom vynechaných úloh v súťaži iBobor.....169
Monika Tomcsányiová
Programovanie v JavaScripte v príprave učiteľov informatiky.....172
Ľudovít Trajteľ, Dana Horváthová, Patrik Voštinár, Miroslav Melicherčík, Alžbeta Michalíková, Jarmila Škrinárová, Vladimír Siládi, Michal Vagač
Budúci „Chief Cybercrime Officer“ dnes ešte derie školské lavice.....178
Jiří Vaníček, Ingrid Nagyová, Monika Tomcsányiová
Tvorba učebnice programování a vzdělávací cíle.....186
Michaela Veselovská, Zuzana Kubincová, Karolína Mayerová
Robotické modely s LEGO WeDo 2.0 v treťom a štvrtom ročníku.....192
Michaela Veselovská, Karolína Mayerová
Vývoj vysokoškolského kurzu edukačnej robotiky.....197
Patrik Voštinár, Martin Kováčik
Programovanie mobilných aplikácii – natívny, webový alebo hybridný vývoj?.....201

Krátke príspevky

Ľudmila Jašková
Súťaž iBobor – kategória Nevidiaci ZŠ - prvýkrát online.....208
Alžbeta Kanáliková
Výučba komunikačnej bezpečnosti a kryptografie.....210
Nika Klimová, Patrik Voštinár, Jarmila Škrinárová, Dana Horváthová
Využitie didaktických pomôcok v mimoškolskej činnosti.....212
Karolína Mayerová, Lucia Budinská, Monika Tomcsányiová
Prieskum o výučbe informatiky na 1. stupni ZŠ na Slovensku v kontexte súťaže iBobor.....214
Miroslav Melicherčík, Michal Vagač, Patrik Voštinár, Michal Povinský
Programovanie trochu inak s použitím mikrokontroléra Arduino.....216

Workshopy

Magdaléna Bellayová, Martin Cápay
Vzbuďme v študentoch chuť programovať! Aktivity s BBC micro:bit.....219
Angelika Haneszová
Osvedčené úlohy na krúžku robotiky pre LEGO Mindstorms EV3.....220
Marek Mansell
Tvorba medzipredmetových projektov pomocou mikropočítača BBC micro:bit.....221
Zuzana Tkáčová
Využitie Raspberry Pi na hodinách informatiky.....222
Jiří Vaníček
Kódování obrázků s Microbitem.....223
Mária Večeríková, Jana Ferencová
Uvádzame vedu do života s robotikou LEGO Education WeDo 2.0.....224
Patrik Voštinár, Nika Klimová, Jarmila Škrinárová, Dana Horváthová
Ukážka robotov z informatického krúžku pre žiakov 2. stupňa ZŠ.....225

Organizátori

Sponzori a mediálni partneri

Príspevky

Papers

Porovnanie slovných riešení teoretickej úlohy žiakov strednej školy a riešiteľov súťaže PRASK

Comparison of theorethical task´s solutions written by high school students and PRASK competition contestants

Michal Anderle
KI, FMFI Univerzita Komenského
Mlynská dolina 1
842 48, Bratislava
Slovensko
anderle@dcs.fmph.uniba.sk

ABSTRACT

Main goal of the computer science lesson is not to teach programming or how to use office tools, but to develop pupils' computational thinking and their ability to solve problems. This is exactly the goal of many different algoritmic competitions, including PRASK – an algoritmic competition for middle schoolers. The interesting question is, whether the tasks from these competitions could be used in regular school lessons.

In our research, one task from PRASK competition was selected and proposed as homework to two groups of pupils from bilinqual high school in Bratislava – pupils from the fourth and the fifth (the last) grade. The selected task delved into problematics of Eulerian graphs. In this paper, pupils solutions, both from high school and PRASK competition, are analyzed and the main similarities and differencies are outlined. That gives us preliminary results, which can be further expanded in further research.

Keywords

Theoretical task. Descriptive solutions. High school. PRASK competition. Eulerian graphs.

ABSTRAKT

Hlavným cieľom predmetu informatika na školách nemá byť výučba programovania či používanie kancelárskych nástrojov, ale rozvíjanie žiackeho myslenia, schopnosti riešiť problémy a algoritmicky rozmýšľať. Práve o to sa dlhodobo snažia rôzne informatické súťaže, medzi ktoré patrí aj základoškolská súťaž PRASK. Otázkou však je, nakoľko sa úlohy z takýchto súťaží dajú využívať pri klasickej výučbe.

V našom výskume sme vybrali úlohu zo súťaže PRASK zaoberajúcu sa eulerovskými grafmi a zadali sme ju žiakom dvoch ročníkov na bilingválnom gymnáziu v Bratislave – štvrtákom a piatakom (maturitný ročník) – ako domácu úlohu. V príspevku analyzujeme riešenia žiakov gymnázia aj riešiteľov súťaže PRASK a zameriavame sa na podobnosti a rozdiely, ktoré sa v týchto riešeniach vyskytujú. To nám poskytuje prvotné výsledky, na ktoré vieme nadviazať v ďalších prácach.

Kľúčové slová

Teoretická úloha. Slovné riešenia. Stredná škola. Súťaž PRASK. Eulerovské grafy.

1 ÚVOD

V posledných rokoch sa do popredia dostáva pojem informatického myslenia (angl. *computational thinking*), ktoré predstavila Jeannette Wing v roku 2006 [1] a označila ho za fundamentálnu zručnosť, ktorú by mali ovládať všetci ľudia. Naskytá sa preto dôležitá otázka, ako túto zručnosť rozvíjať.

Pomerne bežný prístup je prostredníctvom výučby programovania. Existuje však aj opačný pohľad, Lu a Fletcher tvrdia, že programovanie nie je nevyhnutné v rozvoji informatického myslenia. Je totiž náročné vyjadrovať sa vo formálnom programovacom jazyku, ak nie sú dostatočne uchopené procesy, ktoré sa v nich využívajú [2].

Zaujímavou alternatívou by mohlo byť zapojenie úloh z rôznych algoritmických súťaží. Tie totiž okrem programovania často obsahujú aj teoretické úlohy predstavujúce zaujímavé koncepty informatiky a mohli by slúžiť na propagovanie informatiky ako vedy [3].

Jedna pomerne často využívaná a skúmaná súťaž je iBobor [4]. V našom výskume sa však zameriame na algoritmickú súťaž pre žiakov základných škôl – PRASK [5]. Hlavný rozdiel oproti súťaži typu iBobor je, že na riešenie úloh je vyhradený dlhší čas, nie sú ponúknuté správne možnosti, ale od žiakov sa vyžaduje, aby svoje výsledky zdôvodnili. Žiaci sú navádzaní k vlastnému skúmaniu a vďaka tomu sa môžu ponoriť hlbšie do skúmanej problematiky. Na druhej strane, takáto súťaž nie je určená pre široké publikum a snaží sa cieliť na úzku skupinu talentovaných detí. Aj takéto úlohy by sa však mohli dať využiť pri rozvoji informatického myslenia širšej populácie.

V tomto článku prezentujeme výskum, v ktorom sme dvom skupinám stredoškolákov zadali úlohu zo súťaže PRASK. V nasledujúcich častiach predstavíme metodológiu výskumu, zadanie danej úlohy a analýzu získaných riešení.

2 METODOLÓGIA

V spojení s algoritmickými súťažami sa naskytá zaujímavá otázka. Bolo by možné zaradiť úlohy z takýchto súťaží do bežného vyučovania? A ak áno, aké benefity by to prinášalo žiakom? Samozrejme, takúto komplexnú otázku nie je možné zodpovedať pomocou jedného výskumu. Rozhodli sme sa preto spraviť prvotné testovanie, ktorého hlavným cieľom bolo zistiť, nakoľko sa líšia riešenia súťažiacich a žiakov strednej školy a ako skúsenosť s riešením takýchto úloh ovplyvňuje formu týchto riešení.

V prvej fáze výskumu sme vybrali jednu teoretickú úlohu zo súťaže PRASK a k tejto úlohe sme získali odovzdané riešenia. Následne sme túto úlohu v rovnakej forme zadali dvom skupinám stredoškolákov na bratislavskom gymnáziu ako domácu úlohu. Vyžadovaná forma riešení bola totožná s riešeniami v súťaži.

V druhej fáze sme všetky zozbierané riešenia analyzovali pomocou metód kvalitatívneho výskumu, používali sme najmä kódovanie, ktoré nám umožnilo nájsť a upresniť najčastejšie sa vyskytujúce postupy [6]. K dispozícii sme mali tri skupiny riešení, všetky sme vyhodnocovali podľa tých istých kritérií.

Prvú skupinu tvorili riešenia 23 základoškolských žiakov, 6 dievčat a 17 chlapcov, ktorí úlohu riešili vo februári 2015 v súťaži PRASK. Väčšina z nich navštevovala 7 až 9 triedu ZŠ, poväčšinou mali 13 až 15 rokov. Je potrebné si uvedomiť, že títo žiaci sa do súťaže zapojili dobrovoľne. Preto musíme brať do úvahy, že informatika ich zaujímala a prirodzene k nej inklinovali. Taktiež však, veľa z nich neodovzdalo riešenia ku všetkým častiam úlohy. V časti výskumu sme preto brali do úvahy iba menšiu podskupinu riešiteľov, ktorí odovzdali riešenie ku všetkým podúlohám. Túto podskupinu tvorilo 11 riešiteľov, 5 dievčat a 6 chlapcov, pričom piati z nich navštevovali 9. ročník ZŠ a šiesti 7. ročník ZŠ. A hoci táto úloha bola zaradená v historicky prvom ročníku súťaže PRASK, väčšina týchto žiakov mala predchádzajúce skúsenosti s korešpondenčnými seminármi, keďže boli riešiteľmi iných, najčastejšie matematických, seminárov ako Pikomat, Riešky či UFO.

Druhú skupinu tvorilo 10 žiakov štvrtého ročníka päťročného štúdia na bilingválnom gymnáziu v Bratislave, 4 dievčatá a 6 chlapci, vek 17 až 18 rokov. Títo žiaci navštevujú voliteľný seminár z programovania s časovou dotáciou dve hodiny týždenne. Výuka prebieha v slovenčine, obsahom predmetu je výučba programovania v jazyku Python. Učiteľom tohto predmetu je autor článku a túto triedu učí prvý rok.

Tretiu skupinu tvorilo 14 žiakov piateho ročníka päťročného štúdia na bilingválnom gymnáziu v Bratislave, 6 dievčat a 8 chlapcov, vek 18 až 19 rokov. Títo žiaci si v tomto roku zapísali maturitné semináre z informatiky, hoci nie všetci z nich z nej idú aj maturovať. Časová dotácia týchto predmetov je dokopy štyri hodiny, z toho dve sú vyhradené programovaniu v Python-e vyučovanému v slovenčine a dve teórii informatiky vyučovanej v angličtine. Učiteľom oboch týchto seminárov je autor článku a túto skupinu žiakov učil už minulý rok seminár z informatiky. Počas predošlého roka riešili viacero príkladov, ku ktorým odovzdávali slovné popisy a návrhy algoritmov, dokonca na hodine riešili aj niekoľko úloh priamo zo súťaže PRASK.

3 TESTOVANÁ ÚLOHA

Za 5 rokov existencie súťaže PRASK vzniklo viac ako 70 unikátnych úloh, ktoré sú voľne prístupné na webovej stránke tejto súťaže [7]. Tieto úlohy sa v závislosti od formátu riešenia delia na tri skupiny – teoretické, praktické a programátorské. Programátorské vyžadujú odovzdanie odladeného programu, praktické sa zase zameriavajú na využívanie rôznych technológií a sú väčšinou interaktívne. Teoretické úlohy predstavujú žiakom informatické koncepty a prostredníctvom podúloh navádzajú žiakov, aby ich preskúmali a sami dospeli k všeobecným záverom. Svoje zistenia potom žiaci spisujú do riešení, v ktorých argumentujú, prečo sú ich pozorovania správne a platné, poprípade navrhujú všeobecné algoritmy zapísané v prirodzenom jazyku.

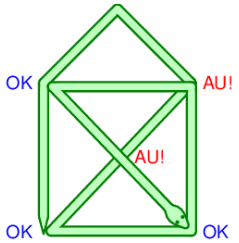
Pri výbere úlohy, ktorú sme použili na testovanie sme hľadali úlohu, ktorá obsahuje viacero postupne sa stupňujúcich podúloh, počnúc konkrétnymi príkladmi slúžiacimi na „ohmatanie“ modelu, končiac všeobecným riešením, ktoré od žiakov vyžaduje návrh algoritmu. Naviac sme chceli, aby daná úloha nevyžadovala od žiakov žiadne predchádzajúce vedomosti a všetko potrebné sa dozvedeli zo zadania.

Nakoniec sa nám ako vhodná úloha, spĺňajúca všetky požiadavky, zdala vôbec prvá teoretická úloha súťaže PRASK s názvom *Pytón s umeleckými sklonmi*[1] V tejto časti si priblížime jej zadanie, očakávané riešenie a tiež sa zamyslíme nad tým, k čomu nabáda výsledná forma zadania.

3.1 Zadanie

Cieľom tejto úlohy je predstaviť žiakom koncept Eulerovských grafov, teda grafov, v ktorých existuje ťah prechádzajúci všetkými hranami. Žiaci by mali objaviť nutnú podmienku, ktorá musí v tomto type grafov platiť a sformulovať algoritmus riešiaci zadaný problém na takýchto grafoch.

Samozrejme, tento koncept môže byť pre žiakov pomerne náročný, najmä ak sa s grafmi dovtedy nestretli. Našťastie, eulerovské grafy sú obrázky, ktoré sa dajú nakresliť jedným ťahom a to je princíp, ktorý je žiakom pravdepodobne dôverne známy. Namiesto grafu preto zadanie používa metaforu hada, ktorý sa stáča a svojim telom vytvára na zemi rôzne útvary. Navyše si žiadnou časťou tela nemôže ľahnúť cez seba a občas si dokonca zahryzne do chvosta čím čiaru svojho tela uzavrie.



Obrázok 1: Obrázok zo zadania prvej podúlohy

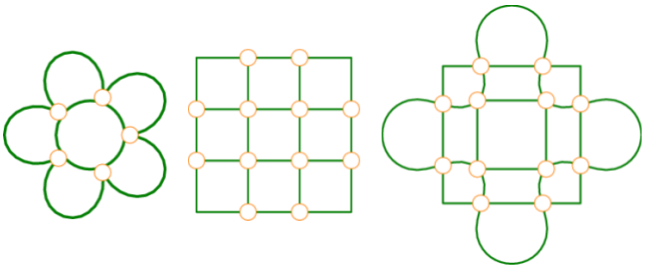
Týmto spôsobom zadanie predstavuje, že nás budú zaujímať uzatvorené a nekrižujúce sa čiary. Tieto pojmy dokonca neskôr, pri presnejšej formulácii podúloh, aj použije. V informatických súťažiach je však zvykom úlohu „zarozprávkovať“. Jednak úloha pôsobí zaujímavejšie a motivujúcejšie, jednak prirodzená metafora pomáha lepšie porozumieť presnejšej formálnej definícii.

V úvode úlohy je teda predstavený had a problémy, ktoré rieši. K tomu pomáha aj ilustračný obrázok, viď. Obrázok 1, ktorý ukazuje, čo znamená kríženie hada a ako sa mu vie potenciálne vyhnúť. Nasledujú samotné podúlohy. V prvej podúlohe majú opraviť vyššie spomenutý obrázok tak, aby sa had nekrížil. V druhej podúlohe dostanú tri vzory z Obrázku 2, ktoré majú pomocou hada vymodelovať. V tejto podúlohe má dokonca platiť, že had si zahryzol do chvosta. K tejto podúlohe bol poskytnutý aj príklad možného zlého nakreslenia – čiara sa križovala a dokonca nebola súvislá, tvorili ju dva cykly. V prípade, že sa daný obrázok nedá nakresliť jednou čiarou, žiaci boli vyzvaní, aby odôvodnili, prečo to tak je.

Posledná podúloha je všeobecnejšia. Had v nej už leží v nejakom neznámom uzavretom útvare, na niektorých miestach sa však križuje. Úlohou žiakov je navrhnúť postup, pomocou ktorého vie upraviť svoju polohu tak, aby bol výsledný obrazec rovnaký, ale nikde sa nekrížil cez seba. Zadanie je následne ešte konkrétnejšie

[1] https://prask.ksp.sk/ulohy/zadania/995

a snaží sa riešiteľom ukázať, ako tento postup môže vyzerať. Naznačuje, že sa im stačí pozerať na jednotlivé krúžky (namiesto pojmu vrchol sa v zadaní objavuje pojem krúžok ako je ukázané aj na Obrázku 2, budeme ho preto používať aj my) a prekresľovať iba tie, kde sa čiary križujú. Okrem toho, zadanie uvádza, že pre jednoduchosť môžu predpokladať, že do každého krúžku vchádzajú práve 4 čiary.



Obrázok 2: Obrázok zo zadania druhej podúlohy

3.2 Očakávané riešenie

Pri tvorbe zadaní a rovnako pred analýzou riešení je dobré si rozmyslieť, aké riešenia od žiakov očakávame, poprípade chceme. Vie nám to totiž pomôcť aj pri tvorbe zadania. Zadanie sme v tomto prípade netvorili my, iba sme ho prebrali zo súťaže PRASK[2]. Očakávané riešenie preto odhadujeme iba na základe toho, ako je štruktúrované zadanie a čo sme si prečítali vo vzorovom riešení.

V prvej aj druhej podúlohe je hlavným cieľom, aby si žiaci vyskúšali tvorbu takýchto obrázkov. Obrázok domčeka je navyše klasickým príkladom kreslenia jedným ťahom, úloha mu však pridáva malý zvrat, v ktorom si treba dať pozor aj na to, aby sa čiary nekrížili. Najdôležitejším bodom tohto obrázku je stred, ktorý má žiakom ukázať, že čiara môže zatočiť aj uprostred, nie len na okrajoch obrázka. Dva obrázky v druhej podúlohe, ktoré sa dajú nakresliť sú takisto pomerne jednoduché a k ich riešeniu sa dá dopracovať postupným skúšaním.

V druhej podúlohe je však aj obrázok, ktorý sa jedným ťahom nakresliť nedá. Ak si žiak skúša tento obrázok kresliť, takmer okamžite narazí na situáciu, z ktorej je zjavné, že obrázok nenakreslí. A problémom budú vždy vrcholy s nepárnym stupňom, na čo majú žiaci prísť. Okrem toho si všimnime, že tretí obrázok je zámerne iba rozšírením druhého obrázku. Mriežka ostala v podstate rovnaká, iba sa niektoré čiary zaoblili, ale pridali sa oblúky na bokoch, ktoré spôsobili, že už žiaden vrchol nemá stupeň 3. Riešiteľa, ktorý si túto zmenu všimne, to vie nasmerovať správnym smerom v odôvodňovaní, prečo sa daný obrázok nedá nakresliť.

Pri odôvodňovaní sme očakávali dva typy riešení. Prvý, v ktorom si riešiteľ uvedomí fakt, že vrcholy nepárneho stupňa sa v eulerovskom grafe nachádzať nemôžu a odôvodní to tým, že pri uzavretej čiare musí vždy platiť, že do vrcholu vojdeme a následne vyjdeme. V druhom type sa žiak odvolá na skúšanie rôznych možností, pri ktorých sa mu nepodarilo obrázok nakresliť. Toto odôvodnenie môže byť absolútne korektné v prípade, že žiak naozaj preukáže, že vyskúšal každú potenciálnu možnosť.

[2] Autor článku je spoluzakladateľom súťaže PRASK, na tvorbe tejto konkrétnej úlohy sa však nepodieľal.

Pri tretej podúlohe je v zadaní naznačené, že sa stačí zameriavať na krúžky, do ktorých vchádzajú práve 4 čiary, a že stačí prekresľovať iba tieto krúžky. Očakávané je teda riešenie, v ktorom si riešiteľ všimne, že je iba jeden spôsob, ako sa v takýchto krúžkoch môžu čiary krížiť a existujú dva spôsoby, ako ich prekresliť bez kríženia. Dôležité však je, aby pri prekreslení zvolil správnu z týchto možností – tú, pri ktorej nevzniknú dve uzatvorené čiary. Na to, že sa také niečo sa môže stať, upozorňuje pomocný obrázok k druhej podúlohe, kde je jedno takéto nesprávne zakreslenie zaznačené.

4 ANALÝZA RIEŠENÍ

Úloha popísaná vyššie bola zadaná v prvom ročníku súťaže PRASK vo februári 2015. Naše testovanie na strednej škole prebehlo v decembri 2018. Zadanie aj formát riešenia zostali nezmenené, zhruba sa zachoval aj čas vyhradený na riešenie, približne tri týždne. Úlohu mali žiaci riešiť samostatne, nedá sa však vylúčiť, že si navzájom pomáhali. Pri opravovaní riešení a ich analýze sme však nenadobudli pocit, že by niektorí žiaci od seba odpisovali, poprípade ich výkony nekorešpondovali s výkonmi na hodinách. Sme si preto skoro istí, že výsledné formulácie boli originálne.

Všetky tri sady riešení sme analyzovali rovnakým spôsobom metódou kvalitatívneho výskumu – kódovaním. V tejto časti si predstavíme použité kódy, bližšie popíšeme charakteristiky riešení jednotlivých skupín a výsledky navzájom porovnáme.

4.1 Kódovanie riešení

Všetkých riešení bolo pomerne málo, dokopy 47. Rozhodli sme sa ich preto spracovať kvalitatívne a to za pomoci kódovania. Na začiatku sme si prešli všetky riešenia a priradili im niekoľko kódov. Tie sme následne v ďalších kolách opravovania bližšie špecifikovali a spájali. Každá z podúloh 2 a 3 má priradené vlastné kódy. Niektoré z nich hovoria o tom, ako žiak riešil danú podúlohu, niektoré, aké prostriedky pri tom využil.

Po vytvorení finálnych kódov sme ešte raz prešli všetky riešenia a priradili im prislúchajúce kódy. Jednému riešeniu bolo možné priradiť viacero kódov. V niekoľkých prípadoch sa dokonca stalo, že žiak v riešení spomenul viacero prístupov, akými sa snažil dopracovať k riešeniu.

Predstavme si použité kódy, keďže ich význam bude dôležitý pri záverečnom porovnaní aj celkovej analýze. Pri podúlohe 2 sme sa zameriavali iba na to, či žiaci vedeli odôvodniť, prečo sa druhý obrázok nedá nakresliť neprerušovanou čiarou, keďže nájsť zakreslenie zvyšných obrázkov zvládli väčšinou bez problémov. Pri tejto podúlohe sme použili nasledovné kódy:

- *Skúšanie možností bez popisu* – sem zapadali všetky riešenia, v ktorých žiaci uviedli, že sa obrázok nedal nakresliť, neuviedli žiadne odôvodnenie, alebo sa jednoducho odvolali na to, že veľa skúšali a nepodarilo sa im to.
- *Skúšanie možností s vysvetlením* – do tohto kódu sme zahrnuli riešenia, v ktorých žiaci postupovali pri skúšaní možností systematicky a preukázateľne zdôvodnili, že už vyskúšali všetky možnosti a ani jednou sa to nakresliť nepodarilo.
- *Nepárnosť bez odôvodnenia* – žiaci si uvedomili, že nepárny počet čiar vchádzajúcich do krúžku

spôsobuje problém, ale do popisu neuviedli, v čom tento problém spočíva. Dá sa predpokladať, že po opätovnom vyzvaní by žiaci vedeli sformulovať skutočný dôvod, nepovažovali však za nutné ho uviesť, napriek tomu, že zadanie ich vyzývalo k tomu, aby svoj názor zdôvodnili.

- *Nepárnosť s odôvodnením* – žiaci správne odôvodnili, prečo je nepárny počet čiar v krúžku problém a vedie k neexistencii riešenia. Najčastejšie toto vysvetlenie spočívalo vo formulácii „keď vojdeme, musíme aj vyjsť“.

Pri tretej podúlohe sme rozlišovali tieto kódy:

- *Prekreslenie bez uvedomenia si nesúvislosti* – žiaci navrhli správny postup prekresľovania krúžkov, v ktorých sa čiary pretínajú, neuvedomili si však, že ľubovoľným prekreslením môžu čiaru rozdeliť na dva cykly.
- *Prekreslenie s uvedomením si súvislosti* – žiaci si uvedomili, že iba jeden spôsob prekreslenia zachováva celistvosť čiary a popísali, ako si vybrať správnu možnosť.
- *Kreslenie po vrstvách* – v tomto type riešení sa žiaci snažili kresliť obrázok postupne, takpovediac po vrstvách. Nesprávne predpokladali, že v každom obrázku budú vedieť pri kreslení systematicky postupovať od stredu alebo zvonka.
- *Spájanie cyklov* – pri tomto type postupu si žiaci uvedomili, že im môže vzniknúť viacero cyklov, ktoré sa následne snažili spájať. Tento prístup je zaujímavý najmä preto, že sa ponáša na dôkaz existencie eulerovských ťahov.
- *Použitie ilustračného obrázku* – v riešení žiaci použili vlastnoručne vytvorený obrázok, na ktorom ilustrovali svoj postup. Najčastejšie tento obrázok znázorňoval, ako sa bude prekresľovať krúžok s križiacimi sa čiarami.

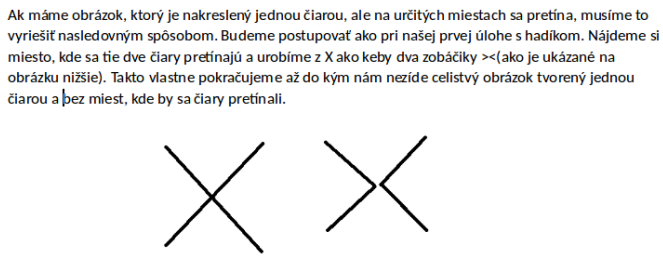
Nakoniec by sme ešte radi dodali, že samotné kódy nehovoria nič o tom, či žiak úlohu úspešne vyriešil, poprípade nakoľko presne sa vyjadroval. Tieto kódy zaznamenávajú najmä prístup, ktorý pri riešení uplatnil.

4.2 Riešenia žiakov štvrtého ročníka bilingválneho gymnázia

Pri kreslení nenastali takmer žiadne problémy. Jedna žiačka nenakreslila správne prvý domček a dvom iným žiakom sa čiara križovala v začiatku, čo sa dá pripísať tomu, že si poriadne neprečítali, že čiara má byť uzavretá, poprípade to nepochopili správne. Pomerne špeciálnym prípadom bol tiež žiak, ktorý na internete našiel eulerove grafy a podmienku, ktorú majú spĺňať. To využil pri riešení podúlohy 2 aj 3, neuvedomil si však, že eulerove ťahy nehľadia na kríženie a vo svojom riešení to ďalej nešpecifikoval, úlohu preto nevyriešil.

V podúlohe 2 väčšina žiakov neuviedla žiaden presvedčivý argument, pre ktorý sa obrázok nedá nakresliť, iba 4 (vrátane vyššie spomínaného žiaka) zistili, že problémom sú vrcholy nepárneho stupňa a iba jeden to v riešení aj odôvodnil.

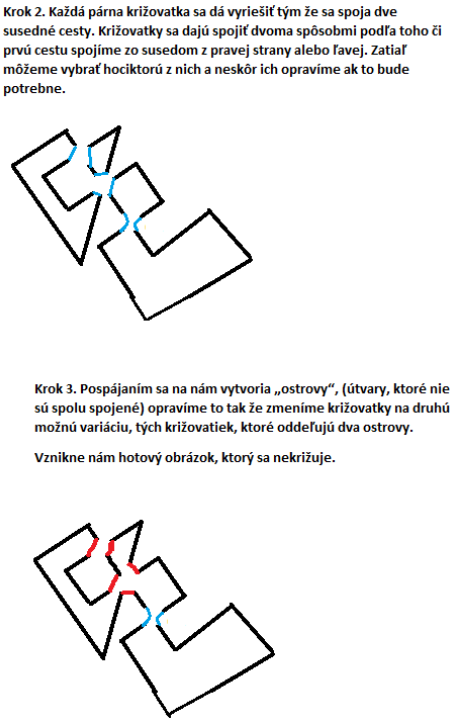
V podúlohe 3 iba jeden žiak dokázal úlohu presvedčivo vyriešiť. Väčšina žiakov sa buď snažila obrázok kresliť po úrovniach, alebo si neuvedomili, že si treba dať pozor na súvislosť čiary. Celkovo sa v riešeniach takmer neobjavili ilustračné obrázky a vyjadrovanie bolo nepresné, zmätočné a ťažko čitateľné.



Obrázok 3: Ukážka riešenia žiaka štvrtého ročníka

4.3 Riešenia žiakov piateho ročníka bilingválneho gymnázia

Aj v tejto skupine sa objavuje pár špecifík. Jeden žiak nenakreslil správne obrázok domčeka (zvyšné obrázky má správne) a jedna žiačka našla na internete eulerovské grafy. Rovnako ako v prípade štvrtáka, táto vedomosť jej nepomohla pri riešení tretej podúlohy, keďže ju nevedela upraviť a predpokladala, že iba kontrola eulerovej podmienky bude dostatočná.



Obrázok 4: Ukážka riešenia žiaka piateho ročníka

V podúlohe 2 mali všetci obrázky nakreslené správne. Takisto všetci prišli na to, že stredný obrázok sa nakresliť nedá, pričom takmer všetci si uvedomili, že problematické sú krúžky s nepárnym počtom hrán. Zhruba polovica z nich dokonca sformulovala dôvod, pre ktorý sú takéto krúžky nevyhovujúce.

Okrem jedného žiaka sa všetci zvyšní podúlohu 3 pokúšali vyriešiť. Až siedmim z nich sa to podarilo a aj svoje riešenie popísali dostatočne kvalitne. Skoro všetci pochopili, ako sa budú

krúžky prekresľovať, niektorí si však nedali pozor na súvislosť výsledného obrázku.

Oproti skupine štvrtákov bolo cítiť značné rozdiely. Riešenia boli v drvivej väčšine popísané oveľa lepšie a zrozumiteľnejšie, mali štruktúru a nepoužívali nedefinované pojmy či operácie. Taktiež sa oveľa častejšie objavovali ilustračné obrázky ukazujúce, ako bude vyzerať prekresľovanie krúžkov. V dvoch prípadoch tieto ilustračné obrázky dokonca pomohli žiakovi v akceptovaní popisu ako správneho. V týchto dvoch prípadoch žiak v popise špeciálne nedefinoval, ako zachováva súvislosť čiary, dokonca sa nedalo určiť, či si uvedomuje, že táto možnosť tam je. Odvolával sa však na obrázok, z ktorého jasne vyplývalo, že prekresľovanie bude zachovávať súvislosť.

V tejto skupine sa takisto nachádza žiačka, ktorá rieši Korešpondenčný seminár z programovania (KSP), stredoškolskú obdobu súťaže PRASK. Úlohu vyriešila správne a jej riešenie bolo podľa nášho názoru napísané najlepšie.

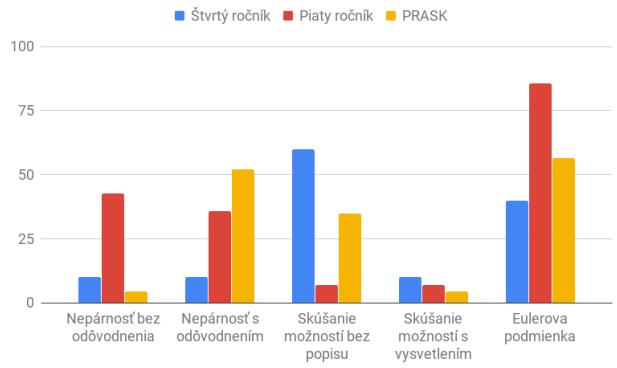
4.4 Riešenia riešiteľov súťaže PRASK

Prvú podúlohu a obrázky v druhej správne nakreslili skoro všetci, našli sa iba traja riešitelia, ktorí mali problém s prvým obrázkom. Pri odôvodňovaní toho, že mriežka sa nakresliť nedá, zhruba polovica iba skonštatovala, že to nejde, druhá polovica však prišla na to, že problémom sú krúžky s nepárnym počtom hrán a skoro každý z nich to aj odôvodnil. Celkovo 13 riešiteľov si uvedomilo, že v takomto obrázku nemôžu byť vrcholy s nepárnym stupňom.

K podúlohe 3 poslalo riešenie iba 11 riešiteľov. Z tejto skupiny sa piatim podarilo vyriešiť úlohu a svoje riešenie popísať na dostatočnej úrovni. Väčšina prišla na správny spôsob prekresľovania, opäť však nastal problém v tom, či si uvedomili, že si treba dať pozor na súvislosť čiary. V riešeniach sa naozaj často objavovali ilustračné obrázky a takisto tieto popisy boli veľmi dobre napísané, zrozumiteľné a jasné.

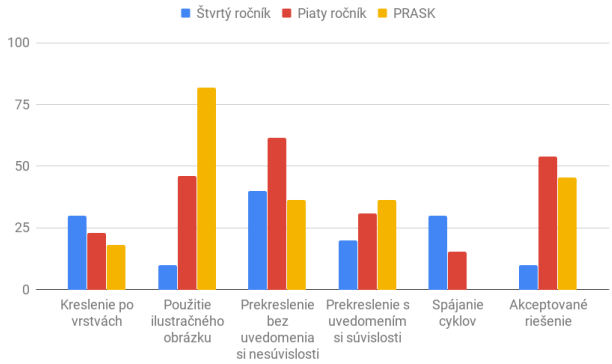
4.5 Porovnanie

Skúsme porovnať jednotlivé skupiny a ich používanie kódov naprieč riešeniami. Nižšie uvedené grafy vznikli tak, že sme pre každú skupinu spočítali počet žiakov, ktorých sa týkal konkrétny kód a následne sme tento počet previedli na percentá, pričom základom boli všetci žiaci, ktorí danú úlohu riešili. V prípade skupiny PRASK sme pri druhej podúlohe použili vzorku 23 žiakov, pri tretej však len 11, ktorí k úlohe skutočne aj niečo poslali.



Obrázok 5: Graf výskytu kódov v riešeniach druhej podúlohy

Na grafe z Obrázku 5 vidíme, že najväčší rozdiel je medzi štvrtákmi a piatakmi. Kým skoro všetci štvrtáci sa odvolávali na vágne skúšanie všetkých možností, poprípade nemali potrebu svoje riešenie zdôvodniť, aj keď ich k tomu zadanie vyzývalo, drvivá väčšina piatakov uviedla správny dôvod neexistencie a polovica z nich ho dokonca obhájila. Pri riešiteľoch PRASKu je zhruba vyrovnané, či objavili hľadanú vlastnosť, a v takom prípade ju aj odôvodnili, alebo ju nenašli či neuviedli. Riešitelia, ktorí svoj postup odôvodnili, však poväčšine riešili aj tretiu podúlohu a ako sme už spomínali, neriešili takýto typ úlohy po prvýkrát.



Obrázok 6: Graf výskytu kódov v riešeniach tretej podúlohy

Z grafu na Obrázku 6 vieme vyčítať ešte viac. Úspešnosť riešenia a vyhýbanie sa chybným úvahám, napríklad kreslenie po vrstvách, bola výrazne väčšia u piatakov a riešiteľov PRASK. Obe tieto skupiny sa od štvrtákov odlišujú v tom, že podobný typ úloh riešili už predtým. Položka *Akceptované riešenie* hovorí o tom, koľko žiakov malo úlohu správne a zároveň aj dostatočne jasne popísanú. Vidíme, že obe tieto skupiny boli v popise lepšie a takisto mali väčšiu šancu všimnúť si, že čiara sa môže rozpadnúť na viacero častí.

A samozrejme asi najväčší rozdiel vidíme v používaní ilustračných obrázkov. Kým takmer žiaden štvrták nemal potrebu použiť pri vysvetľovaní obrázok, zhruba polovica piatakov a takmer všetci riešitelia PRASK ich použili. Ich použitie je pritom vítané, keďže sprehľadňujú riešenie a ľahšie sa na nich prezentujú žiakove myšlienky.

5 DISKUSIA

V predchádzajúcej časti sme videli analýzu riešení, z ktorých si vieme vyvodiť niekoľko zaujímavých záverov. Tento výskum bol však spravený iba na pomerne malej vzorke žiakov, ktorí sa informatike a programovaniu venovali dobrovoľne. Závery preto nemôžeme pokladať za univerzálne platné, naznačujú však trendy, ktoré by sa dali skúmať a ďalej potvrdzovať v nadväzujúcich prácach.

Najväčšie rozdiely sa ukazujú medzi štvrtákmi a piatakmi spolu s riešiteľmi súťaže PRASK. Sme názoru, že tento rozdiel je spôsobený najmä množstvom skúseností s riešením úloh podobného typu. Mnoho z účastníkov PRASKu riešilo predtým matematické súťaže, kde sa dôraz kladie tiež na argumentáciu, formuláciu a dokazovanie tvrdení. Piatakov sme počas predchádzajúceho roka na seminári z informatiky tiež viedli týmto smerom. Riešili sme s nimi podobné úlohy, dávali im spisovať ich riešenia a následne sme im poskytovali spätnú väzbu podobnú tej, ktorú dostávajú riešitelia korešpondenčných seminárov. Výsledkom je, že títo žiaci boli oveľa úspešnejší v riešení

vybranej úlohy, takmer všetci odhalili vlastnosť eulerových grafov a možnosť prekresľovať krúžky.

Tento výsledok je samozrejme prirodzený a očakávaný. Trénovaním riešenia takýchto úloh sa v tom stali lepšími. Podstatnejšie však je, že títo žiaci oveľa lepšie rozumeli potrebe odôvodňovať svoje tvrdenia. A takisto v tom boli oveľa lepší. Ich popisy boli kvalitnejšie, čitateľnejšie a prehľadnejšie, oveľa častejšie využívali v riešeniach ilustračné obrázky.

Tento výsledok nadväzuje na náš predchádzajúci výskum [8], v ktorom sme so skupinou piatakov riešili inú úlohu zo súťaže PRASK so záverom, že takáto úloha nebola pre nich náročná na vyriešenie, problémom však bolo formálne vyjadrovanie. Už v tomto článku sme uviedli niekoľko prípadov, v ktorých sa spôsob vyjadrovania zlepšil v priebehu hodiny po tom, ako im učiteľ predstavil riešenie predchádzajúcej podúlohy. A z tohto článku vyplýva, že ich schopnosť vyjadrovať sa trénovaním ešte viac zlepšila.

Myslíme si, že toto je pomerne dôležitý prvok vyučovania informatiky. Programovanie nie je totiž nič iné ako formálny zápis postupu v špeciálnom jazyku. Je preto zmysluplné, aby sme žiakov najskôr viedli k formálnemu vyjadrovaniu v jazyku im prirodzenom. Táto schopnosť je tiež uplatniteľná v ich ďalšom živote bez ohľadu na to, čo sa rozhodnú študovať – jasné a formálne vyjadrovanie sa vyžaduje pri písaní záverečných prác, na pohovoroch či následne v zamestnaní.

V prípade štvrtákov vidíme, že na takýto typ úlohy neboli vôbec zvyknutí a pravdepodobne sa s niečím takým počas štúdia nestretli. Časťou problému bude, že takéto úlohy je náročné pripraviť a následne vyhodnotiť. Korešpondenčné semináre a olympiády z matematiky, fyziky či informatiky však takéto príklady vytvárajú už roky a vo všetkých sa kladie veľký dôraz práve na teoretickú časť, kde žiaci dokazujú a odôvodňujú. K týmto príkladom sú zverejňované aj vzorové riešenia, nie je preto nemysliteľné, aby ich učitelia preberali a zapájali do vyučovania.

6 ZÁVER

Už mnoho rokov sa rôzne algoritmické súťaže snažia zdokonaľovať informatické myslenie a schopnosti programovať u úzkej cieľovej skupiny talentovaných detí. Naskytá sa však otázka, či by tieto úlohy nevedeli byť použité aj počas bežnej vyučovacej hodine na stredných či základných školách. V tomto článku sme prezentovali výskum, v ktorom sme zobrali jednu úlohu zo základoškolskej súťaže PRASK a dali sme ju riešiť dvom skupinám stredoškolákov. Odovzdané riešenia, spolu s riešeniami zo súťaže, sme následne podrobili kvalitatívnej analýze.

Zistili sme, že s rastúcimi skúsenosťami s podobnými úlohami, rástla aj šanca, že žiaci danú úlohu vyriešia a zvolia vhodný spôsob riešenia. Keďže sa jednalo o úlohu teoretickú, dôležitou súčasťou riešenia bol samotný popis argumentov a odôvodnení. Na takýto typ zadaní žiaci nie sú zvyknutí a bolo vidieť, že sa s formuláciou v mnohých prípadoch trápili. Takisto však bolo vidieť, že čím viac skúseností mali, tým bol ich prejav jasnejší a zrozumiteľnejšie, v ich riešeniach sa začali objavovať ilustračné obrázky, na ktoré sa mohli odvolávať a rozumeli, čo znamená odôvodniť svoje riešenie platnými argumentami.

Schopnosť adekvátne sformulovať riešenie, postup či algoritmus je v informatike veľmi dôležitá a je nevyhnutné aby sme ju v žiakoch správnym spôsobom rozvíjali. Tento článok ukazuje, že vhodným médiom na tento rozvoj by mohli byť práve úlohy z rôznych algoritmických súťaží.

Samozrejme, tento výskum bol iba počiatočný, spravený na malej a špecifickej vzorke a v budúcnosti by sme naň chceli nadviazať. Radi by sme naše výsledky overili aj na ďalších úlohách z tejto súťaže a takisto by sa dalo bližšie pozrieť na vzťah skúsenosti a schopnosti vyjadrovať sa. Tento výskum by sa dal takisto testovať na ďalších vzorkách žiakov, líšiacimi sa vekom, školou či povinnosťou informatiky.

BIBLIOGRAFICKÉ ODKAZY

[1] Jeannette M Wing. Computational thinking. Communications of the ACM, 49(3):33–35, 2006.

[2] Lu, James J., and George HL Fletcher. "Thinking about computational thinking." ACM SIGCSE Bulletin 41.1 (2009): 260-264.

[3] Valentina Dagiene and Gerald Futschek. Introducing informatics concepts through a contest. In IFIP Working Conference: New Developments in ICT and Education. Universite de Picardie Jules Verne, Amiens, 2010.

[4] BUDINSKÁ, L. MAYEROVÁ, K. WINCZER M. Analýza grafových úloh zo súťaže iBobor – Juniori a Seniori. In *Didinfo 2018*. [online]. Liberec, 2018. s. 237-242. ISBN: 978-80-7494-424-6. Dostupné z: https://didinfo.net/images/DidInfo/files/didinfo_2018.pdf

[5] M. Anderle (2018) PRASK – an Algorithmic Competition for Middle Schoolers in Slovakia, Olympiads in Informatics, 2018, Vol. 12, pp. 147-157.

[6] John W Creswell. Educational research: Planning, conducting, and evaluating quantitative. Prentice Hall Upper Saddle River, NJ, 2002.

[7] PRASK. http://prask.ksp.sk

[8] M. Anderle (2018) Transformation of tasks from Competition to High School Lessons – Binary Search Trees, ICERI2018 Proceedings, pp. 6549-6557.

Príprava učiteľov na výučbu predmetu Informatika v prírodných vedách a matematike

Professional Development for Teaching New Subject Informatics in sciences and mathematics

Gabriela Andrejková
Ústav informatiky PF UPJŠ
Jesenná 5
04001 Košice
Slovensko
gabriela.andrejková@upjs.sk

Anna Mišianiková
Katarína Kimáková
Ústav biologických a ekologických vied
Mánesova 23, 04001 Košice
Slovensko
katarina.kimakova@upjs.sk,
anna.misianikova@upjs.sk

Zuzana Tkáčová
Ústav informatiky PF UPJŠ
Jesenná 5, 04001 Košice
Katedra techniky a informačných technológií PF UKF
Dražovská cesta 4, 94974 Nitra
Slovensko
zuzana.tkacova1@upjs.sk

ABSTRACT

Natural science subjects and mathematics bring interesting problems that can be solved by means of informatics and informatics provides more precise means of solving them. Teaching natural science subjects and mathematics in their connection with informatics can be a significant contribution to these subjects. In this article, we describe in details the training of teachers for teaching of the subject "Informatics in Natural Sciences and Mathematics", which is implemented in the project IT Academy - Education for the 21st Century. We will also highlight the results and views of the first graduates of this education.

Keywords

Informatics. Natural Science. Mathematics. Teachers Training.

ABSTRAKT

Prírodovedné predmety a matematika prinášajú zaujímavé problémy riešiteľné pomocou informatiky a informatika poskytuje exaktnejšie prostriedky na ich riešenie. Výučba prírodovedných predmetov a matematiky v ich prepojení s informatikou môže byť významným prínosom pre tieto predmety. V tomto článku detailne popíšeme vzdelávanie učiteľov pre výučbu predmetu „Informatika v prírodných vedách a matematike", ktoré je realizované v projekte IT Akadémia – vzdelávanie pre 21. storočie. Uvedieme tiež výsledky a postrehy prvých absolventov tohto vzdelávania.

Kľúčové slová

Informatika. Prírodné vedy. Matematika. Vzdelávanie učiteľov.

1 ÚVOD

V príspevku [1] bola uvedená koncepcia nového pripravovaného motivačného predmetu pre stredné školy Informatika v prírodných vedách a matematike (IPVaM). V odpovediach na tri základné otázky o zaradení tohto predmetu do výučby [2] je jednoznačne vyjadrené, že predmet je potrebný, pretože prispieva k poznaniu tohoto sveta a rozširuje jeho poznanie svojím interdisciplinárnym charakterom. Prispieva k rozvíjaniu myslenia a spôsobilostí žiaka a podporuje schopnosť úspešne študovať na univerzite, prispieva hlavne k správnemu výberu študovaného odboru. Potreba predmetu podobného zamerania je vyjadrená tiež v [3].

Predmet je navrhnutý pre výučbu na gymnáziách a jeho štruktúra je orientovaná na riešenie jednoduchých problémov z rôznych oblastí prírodovedných predmetov a matematiky. Preto je použiteľný vo výučbe aj na iných stredných školách. Napríklad, fyzikálne alebo chemické problémy môžu zaujať žiakov na stredných priemyselných školách.

V súvislosti s výučbou tohto predmetu vznikla otázka prípravy učiteľov, pretože do výučby prichádzajú nové témy patriace do oblasti zobrazovania objektov, modelovania procesov a databázových systémov. Potreba prípravy učiteľov pre výučbu tohto predmetu vychádza tiež z analýzy dotazníkového prieskumu realizovaného v rámci realizácie národného projektu IT Akadémia – vzdelávanie pre 21. storočie [4], ktorý školám ponúka možnosť implementácie tohto predmetu. Z analýzy dotazníka v danej oblasti vyplýva, že zo 102 gymnázií a 95 stredných odborných škôl, ktoré dotazník vyplnili má o zaradenie nového predmetu záujem spolu 128 škôl.

Obsah prípravy učiteľov je tvorený témami predmetu IvPVaM, ktoré budú vyučované na stredných školách. Témy boli vyberané tak, aby žiakov zaujali, a aby aktuálne vedomosti žiakov stačili na ich zvládnutie. K príprave učiteľov bolo vytvorené akreditované inovačné vzdelávanie s názvom daného predmetu, ktoré veľmi uľahčí prácu učiteľa, pretože vo vzdelávaní učiteľ preštuduje väčšinu pripravených tém. Predpokladáme, že absolvent tohto vzdelávania bude okrem poznatkov o vyučovaných témach vedieť

- zhodnotiť a aplikovať metodické postupy vo výučbe prírodovedných predmetov na úrovni využívania hotových aktivít a ich modifikácií, napríklad bude vedieť upraviť hotový program pre použitie vo výučbe,
- demonštrovať prácu s dostupnými informáciami v prírodovedných databázach,
- metodicky pristúpiť k výučbe študovaných tém tak, aby to vedel naučiť svojich žiakov.

V tomto príspevku uvedieme detailnejšie obsah niektorých vzdelávaných tém a skúsenosti zo vzdelávania učiteľov. V texte sa bude prelínať metodická príprava jednotlivých tém pre žiakov a vzdelávanie učiteľov, pretože to spolu súvisí. Tiež popíšeme niektoré výsledné prezentácie učiteľov, ktoré sú ukážkou toho, že vzdelávanie prispelo k zmenám vo vyučovaní už vzdelávaných

učiteľov a ku hľadaniu ďalších zaujímavých tém súvisiacich s prírodnými vedami.

2 OBSAH A FORMA VZDELÁVANIA

Príprava učiteľov na výučbu predmetu IPVaM je prispôsobená jeho obsahu, ktorý je zameraný na témy z prírodných vied a na matematiku v celkovom rozsahu 60 hodín, z toho je 48 hodín prezenčných a 12 hodín dištančných. Rozdelenie hodín podľa predmetov je uvedené v Tabuľke 1.

Tabuľka 1. Počty hodín pre študované predmety

Blok Informatika	Prvý predmet	Druhý predmet
8 prezenčných + 2 dištančné hodiny	20 prezenčných + 5 dištančných hodín	20 prezenčných + 5 dištančných hodín

Učiteľ si vyberá predmety podľa svojej aprobácie, prvý blok Informatika je povinný pre všetkých učiteľov (Informatika je ešte aj ako samostatný predmet). V prezenčnej časti sú preberané témy, ktoré sú orientované na výučbu žiakov. Predovšetkým sú vysvetlené metodiky (prípadne sú doplnené odborné poznatky) a prechádzané pracovné listy, ktoré si učitelia overujú kvôli prípadnému použitiu vo výučbe.

V dištančnej časti sa sústreďujeme na tvorbu jednoduchých úloh a ich riešenie. Učitelia môžu neskôr použiť tieto úlohy vo svojej vlastnej výučbe. V Tabuľke 2 uvádzame zoznam preberaných tém v jednotlivých predmetoch. Tieto témy sú už spracované a vo vzdelávaní učiteľov boli otestované.

Zoznam tém bude ešte doplnený o ďalšie témy, napríklad v geografii. Počet pripravovaných tém je stanovený tak, aby témy v ľubovoľnej kombinácii dvoch predmetov pokryli výučbu jedného polroku. To znamená učiteľ, ktorý má kombináciu dvoch aprobačných predmetov, je pripravený na jeden polrok výučby.

V ďalšom prírodovedný predmet a matematiku spolu s témami budeme nazývať modul.

3 VYBRANÉ TÉMY DETAILNEJŠIE

Uvedieme detailnejší obsah úvodného bloku Informatika a dvoch ďalších vybraných tém z biológie a informatiky.

3.1 Úvodný blok Informatika

Tento blok je zameraný na vysvetlenie základnej štruktúry programu v jazyku Python. Sústreďujeme sa na to, aby sme uviedli niekoľko jednoduchých úloh, na ktorých ukážeme použitie niekoľkých základných príkazov a tiež na to, aby sme učiteľov naučili krátke programy čítať a porozumieť, akú činnosť vykonávajú. Za dôležité považujeme vysvetliť aj postup pri inštalácii prostredia a interpretéra pre programovací jazyk Python. Pri vysvetľovaní štruktúry programu sú zaradené témy:

1. Práca v interaktívnom režime (v príkazovom riadku)
2. Vytvorenie programu v súbore s koncovkou .py
3. Textové reťazce, základné typy hodnôt, príkaz cyklu for
4. Grafická plocha, práca s korytnačou grafikou
5. Vetvenie programu, príkaz if
6. Funkcie

Prvé skúsenosti z výučby tohto bloku ukázali, že je potrebný pomalý postup pri preberaní tém, aby učitelia získali istotu, že to vedia a postupujú správne. Nejde tu o samostatnú tvorbu programu ale o to, aby porozumeli, čo jednoduchý program urobí.

Tabuľka 2. Zoznam preberaných tém v jednotlivých moduloch

Predmet	Preberané témy a metodiky k nim
Biológia	*Zobrazovanie:* Koľko vtákov letí v kŕdle? Ako vieme, koľko krviniek je v krvi? Čo prezradí svetlo o bunkách? *Modelovanie:* Kedy má chirurg prácu? Zajace a líšky Kedy je les zdravý? Dáte sa zaočkovať? *Databázy:* Zoznámte sa s rastlinami Kam letia sťahované vtáky? Nie je vírus ako vírus
Chémia	Kreslíme molekuly uhľovodíkov Tvoríme molekuly derivátov uhľovodíkov a reakčné schémy Izoméria v organickej chémii Kreslíme molekuly látok v živých organizmoch Kreslíme chemické aparatúry a vypracovávame chemické protokoly Kyslé dažde - tvorba databáz Periodicita vlastností prvkov - práca s databázami Genetický kód
Fyzika	*Úvod do modelovania:* Ako sila ovplyvní pohyb? Ako prekonať rýchlosť zvuku? Ako štartuje raketa? Ako udržať teplotu v dome? *Zobrazovacie metódy* Ako fotiť pohybujúce sa objekty Termografia-bezkontaktné meranie teploty Zobrazovanie ultrazvukom
Geografia	Nezamestnanosť na Slovensku Vekové pyramídy
Informatika	*Zobrazovanie* - vykresľovanie obrázkov pomocou korytnačej grafiky Spracovanie reálneho obrázka, úprava farieb pixelov Modifikácia rovinného obrázku – generovanie infikovaných ložísk Modifikácia rovinného obrázku na základe stanovených podmienok *Modelovanie* - nameraných hodnôt Modelovanie lineárnych javov Modelovanie nelineárnych javov *Práca s databázami* - navrhnúť a vytvoriť vlastnú štruktúrovanú reprezentáciu dát pomocou dátového typu slovník Vyhľadávanie biologických, chemických a geografických informácií v dostupných online databázach
Matematika	Geometrické konštrukčné úlohy, programovací jazyk Asymptote Body a úsečky Zobrazenia Geometria trojuholníka Témy sú systematický naviazané na seba, a takto je pripravený študijný materiál.

V dištančnej časti boli zadané dve úlohy, ktorými sme sledovali tiež to, aby učitelia hľadali možnosti použitia programovania vo výučbe:

> **Úloha 1:** Sformulujte zadanie úlohy orientovanej na zvolený prírodovedný predmet alebo matematiku, ktorej riešením bude program s číselným výstupom.
>
> **Úloha 2:** Sformulujte zadanie úlohy orientovanej na zvolený prírodovedný predmet alebo matematiku, ktorej riešením bude program s grafickým výstupom.

Kvalita vypracovania týchto úloh v skupine 31 učiteľov bola rôznorodá. Učitelia prikladali malý význam formulácii úlohy. Sústredili sa na riešenie, t. j. na vytvorenie programov. Prvé úlohy boli väčšinou zamerané jednoduché fyzikálne, chemické a matematické výpočty (jednoduché vzorce). Druhé úlohy na vykresľovanie trojuholníkov, kružníc a len niekoľko úloh bolo trochu zložitejších. Výnimkou boli úlohy učiteľov informatiky, ktorí vytvárali trochu zložitejšie zadania a programy.

3.2 Modul Biológia

Obsah biologického modulu nadväzuje na zručnosti nadobudnuté v úvodnom informatickom bloku. Je rozdelený do 10 metodík, z ktorých učiteľ biológie odučí 8 podľa vlastného výberu. Obsahujú úlohy zaujímavé zo spoločenského hľadiska, vyžadujú samostatné uvažovanie a posilňujú interdisciplinárne vzťahy.

3.2.1 Zobrazovanie

V rámci bloku *Zobrazovanie* sú zaradené úlohy na počítanie a meranie objektov na digitálnom obraze pomocou softvéru. Mnohí žiaci si neuvedomujú, čo všetko môže byť zdrojom dát. Keď počujú o meraní, napadnú im súvislosti skôr s matematikou a fyzikou, miery, jednotky a ich fyzikálna podstata. S dátami sa však pracuje aj v biológii. Biológ samozrejme môže merať priamo objekt alebo používať monitorovacie metódy aplikované priamo v teréne. Vyžaduje to však veľa času a pozornosti, a predovšetkým byť fyzicky prítomný v správnom čase na správnom mieste, najmä keď sa jedná o živočíchy. Zaujímavým problémom je aj monitorovanie a vyhodnocovanie počtu vtákov, čo rieši metodika *Koľko vtákov letí v kŕdle?*

Úlohy sú tiež zamerané na počítanie a meranie ďalších objektov na digitálnom obraze, napríklad veľkosť peľových zŕn, počet červených krviniek človeka alebo analýzu fyziologicky významných rozdielov sfarbenia listov a pod. Získajú sa tak dáta, ktoré sa dajú ďalej analyzovať a interpretovať, čím sa rozvíjajú zručnosti vedeckej práce. Ak rozumieme podstate digitálneho obrazu a princípu softvéru, lepšie pochopíme, ako získať čo najviac presných a užitočných informácií.

Ďalšie témy rozširujú analýzu obrazu o vysvetlenie algoritmu rozpoznávania objektov a biologické obrazové databázy. V medzinárodných biologických databázach sa vyhľadávajú informácie o zadaných organizmoch.

3.2.2 Modelovanie

Úlohy predstavujú biologické problémy, ktoré je možné vyriešiť dostupnými informatickými prostriedkami či pomocou vlastných programov. Príkladom je metodika *Dáte sa zaočkovať?*, kde sa vysvetľuje prenos a šírenie sa infekčného ochorenia.

Metodika vychádza z rastúceho trendu v spoločnosti odmietať povinné očkovanie detí a podceňovať následky nebezpečných ochorení. Vzniká ilúzia, že ak sa nemoc niekoľko rokov u nás nevyskytla, celkom vymizla a nie je potrebné chrániť sa pred ňou očkovaním. Žiaci formulujú hypotézu, kedy hrozí prepuknutie ochorenia a jeho šírenie v populácii vo forme epidémie a následne modelujú šírenie infekcie pred a po zásahu vakcináciou. Učiteľ má k dispozícii súbory s vytvoreným modelom šírenia chrípky v škole v programovacom jazyku Python. Žiaci tiež analyzujú graf šírenia epidémie zmenou parametrov, čím získajú predstavu, ako sa epidémia správa a aplikujú model na riešenie reálnej situácie epidémie chrípky v školskom prostredí. Žiaci modelovaním šírenia infekcie pri rôznej miere zaočkovania zistia, že približne 96 % jedincov v populácii musí absolvovať očkovanie, aby sa infekčná choroba nevyskytovala ani v jej nezaočkovanej časti populácie. Model ako názorný príklad účinnej miery ochrany populácie vakcináciou môže ovplyvniť ich postoje k povinnému očkovaniu.

Princíp modelovania pomáha analyzovať aj priebeh tých biologických dejov, ktoré sa uskutočňujú veľmi pomaly alebo rýchlo, v molekulárnych až gigantických biosférických rozmeroch. Modelovanie je aplikované aj pri ďalších témach, a to pri raste buniek a nádoru, obnove lesa a vzťahoch dravca a koristi. Cieľom je, aby žiaci rozpoznali za modelom príslušný algoritmus, vedeli do jeho fungovania zasiahnuť a interpretovať danú situáciu.

Učitelia biológie hodnotili vzdelávanie tohto bloku pozitívne a ich spätná väzba naznačila, že vypracované metodiky a úlohy sú pre nich zrozumiteľné. Ocenili tiež využívanie a prelínanie rôznych informatických prístupov, čo prispelo k rozvoju informatického myslenia. Prvé praktické skúsenosti učiteľov biológie potvrdili motiváciu žiakov pre využívanie informatiky pri riešení úloh.

3.3 Modul Informatika

3.3.1 Zobrazovanie

Počítačová grafika je oblasť informatiky zaoberajúca sa metódami a technikami konštrukcie, manipulácie, ukladania a zobrazovania pomocou počítačov (ISO definícia).

V súčasnosti existuje mnoho aplikácií, ktoré nám poskytujú možnosti manipulácie a úpravy fotografií a iných obrázkov. Vizualizácia dát sa vykonáva v každej oblasti, preto je dobré porozumieť, ako to prebieha, o čom je potrebné pri spracovaní uvažovať. Naším cieľom bude pochopiť princípy tvorby a úprav obrázkov, a naučiť sa niektoré základné pojmy, ktoré sa tu používajú.

Riešené problémy:

1. Rovinný obrázok vyhodnotiť na základe farebnosti použitých pixelov. Naučiť sa meniť farby pixelov, pridať pixely do objektu.
   Poznámka: Obrázok bude získaný ako výrez skutočného obrázku. Na vyhodnotenie je pripravený program v jazyku Python.
2. Metódou Pixel ART vytvárať a zobrazovať rôzne obrázky (farbením pixelov) v programovacom jazyku Python.
   Poznámka: Vytvorený program a niekoľko obrázkov je pripravených, nové obrázky sa žiaci naučia vytvárať a v programe použiť.
3. Metódou Pixel ART vytvoriť obrázok živočícha v rovine, ktorý rastie na základe daných pravidiel, v programovacom jazyku Python.
   Poznámka: Pravidlá rastu budú rozanalyzované, program bude k dispozícii. Žiaci budú robiť malé modifikácie pravidiel. Cieľom je predpovedať, ako živočích narastie v nejakom čase.

3.3.2 Modelovanie

Vytváranie modelov vychádzajúcich z nameraných údajov (alebo inak získaných dát), na základe ktorých je možné predpovedať

ďalší vývoj v dátach. Modelovanie by sme mohli brať ako spôsob pre začlenenie udalostí každodenného života a rozhodovania do matematického problému (modelu). Umožňuje otvorenú diskusiu a tvorivosť žiakov.

Modelovanie by malo robiť abstraktnú matematiku zmysluplnejšou a relevantnejšou pre stredoškoláka. Preto budeme riešiť jednoduché úlohy, na ktorých si vysvetlíme metódu vytvárania modelov.

Modelovací cyklus pre stredoškolskú výučbu je možné popísať v nasledujúcich krokoch [9]:

- Identifikovať a priradiť premenné - vybrať podstatné časti problému a reprezentovať ich symbolicky;
- Sformulovať model - geometrický, tabuľkový, grafický, alebo algebraický;
- Analyzovať a vykonať operácie - vypočítať časti procesu, použiť formuly, algebraické výrazy, a pod.
- Interpretovať výsledky - dosiahnuté výsledky vysvetliť v rámci originálneho problému;
- Vyhodnotiť výsledky a spracovať závery - vyhodnotiť, či majú dosiahnuté výsledky zmysel v pôvodnom kontexte, skontrolovať výsledky použitím rôznych matematických metód;
- Publikovať a zdôvodniť závery - vysvetliť publikované závery a urobiť predikcie do budúcnosti.

Riešené problémy:

1. Prípravné úlohy na vykresľovanie rôznych závislosti. Inštalácia potrebných knižníc.
2. Športové rekordy v atletike. Údaje o športových rekordoch získané z rôznych webových stránok budú vizualizované, lineárne modelované a na základe modelu bude urobená predikcia roku, kedy sa očakáva nový rekord.
   Poznámka: Program v jazyku Python je pripravený pre použitie viacerých dát.
3. Nelineárne modelovanie.
   Poznámka: Program v jazyku Python je pripravený pre modelovanie skutočne nameraných dát v experimentoch kognitívneho laboratória.

3.3.3 Práca s databázami

Organizovať, vyhľadávať, triediť, či spracovať dáta patrí k základným vedeckým zručnostiam. Preto sa požiadavka na prácu s dátami a databázami vyskytla ešte pri návrhu predmetu IPVaM od všetkých prírodovedných predmetov. Inovovaný Štátny vzdelávací program pre informatiku [5] pre gymnáziá však problematiku databáz explicitne nerieši, okrajovo sa k nim môžeme dostať len v novozavedenom tematickom bloku **Reprezentácie a nástroje – štruktúry**, v rámci ktorého má žiak vedieť [5] a na toto je potrebné učiteľov pripraviť:

- organizovať informácie do štruktúr – vytvárať a manipulovať so štruktúrami, ktoré obsahujú údaje a vzťahy (tabuľky, grafy, postupnosti obrázkov, čísel, ...),
- orientovať sa, vyhľadávať a získavať informácie zo štruktúry podľa stanovených kritérií,
- voliť vhodnú štruktúru pre reprezentáciu informácie a zdôvodňovať ju (napr. kedy voliť čísla, alebo kedy meno a priezvisko evidovať v dvoch samostatných stĺpcoch v tabuľke, ...),
- posudzovať vlastnosti operácií s rôznymi štruktúrami (napr. možnosť mazania, vkladania, vyhľadávania, ...),
- interpretovať údaje zo štruktúr – odvodzovať vzťahy zo zadaných údajov v štruktúre, prerozprávať informácie uložené v štruktúre vlastnými slovami.

S ohľadom na reálnu, spravidla len niekoľkohodinovú časovú dotáciu, ktorá môže byť pre tento tematický blok vo všeobecnovzdelávacom predmete Informatika vyčlenená, bolo potrebné aj v predmete IPVaM sa k tejto téme vrátiť a ponúknuť učiteľom aj žiakom nadstavbové úlohy, ktoré by prepojili základné vedomosti s vybranými praktickými príkladmi a ktorých cieľom bolo oboznámiť sa s databázami ako nástrojmi pre zbieranie a usporiadanie informácií – preskúmať rôzne podoby dát a existujúcich dátových súborov, prakticky vyskúšať základné možnosti vyhľadávania dát v biologických, chemických a geografických online databázach, ako aj ich analýzu a vizualizáciu a následne, pomocou programovacieho jazyka Python, sa naučiť efektívne využívať dátový typ slovník pre ukladanie informácií. Nakoľko slovníky nepatria k dátovým typom, ktoré sa v základnom programátorskom kurze Pythonu na gymnáziách učia, bolo potrebné postupne a systematicky pristupovať k práci s týmto dátovým typom, kde využívajú žiaci aj zručnosti, ktoré už majú z práce s číslami, reťazcami a zoznamami.

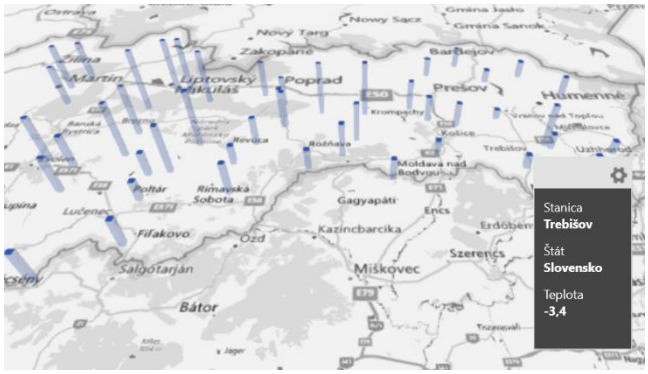
Vyučovacie hodiny sú navrhnuté tak, aby na ich zvládnutie žiakovi postačili vstupné vedomosti pokrývajúce prácu s internetom a webovým prehliadačom, vyhľadávanie na webe, prácu s tabuľkovým kalkulátorom a základy programovania v jazyku Python (vo verzii 3.x). Na prácu na hodinách sa používa **metóda EUR**, ktorá člení hodinu na tri fázy:

1. časť **EVOKÁCIA** – formou práce v skupinách s pracovným listom si žiaci urobia prehľad svojich doterajších vedomostí, príp. vlastných skúseností z problematiky využívania rôznych typov databáz, resp. programovania v Pythone a využívania rôznych dátových typov (číslo, reťazec, zoznam),
2. časť **UVEDOMENIE SI VÝZNAMU** – v rámci tejto časti učiteľ facilituje žiacke bádanie pri ich práci s odbornými online databázami, resp. s dátovým typom slovník podľa postupu v pracovnom liste,
3. časť **REFLEXIA** – žiaci by mali byť schopní na základe predošlých aktivít porovnať možnosti rôznych druhov databáz a posúdiť ich využitie pri vedeckej práci, resp. porovnať možnosti a vlastnosti dátového typu slovník, posúdiť jeho využitie pri vedeckej práci a naprogramovať vlastný projekt využívajúci slovník.

Najdôležitejšou časťou prvej tematickej dvojhodinovky, pri ktorej žiaci skúmajú možnosti vedeckých online databáz, je práca s databázou *National Center for Biotechnology Information (NCBI) [6]*, kde sa žiaci na príklade neznámeho génu BRCA1 oboznámia vyhľadávacím nástrojom *Entrez*, databázou *Nucleotide* a *PubMed*, naučia sa vyhľadať referenčnú DNA sekvenciu vo formáte *FASTA* a na praktickej úlohe si vyskúšajú nástroj *BLAST* (Basic Local Alignment Search Tool) na porovnávanie DNA sekvencií.

Na nástrojoch NCBI databázy tak môžu žiaci zistiť, že verejne dostupné online biologické databázy môžu obsahovať viaceré rôzne databázy, ale aj vyhľadávacie nástroje, ktorými je možné získať veľké množstvo výsledkov a doplňujúcich informácií. Taktiež disponujú užitočnými nástrojmi, ktoré umožňujú vyhľadávanie aj pomocou presných DNA sekvencií, pričom tieto všetky nástroje sú dostupné zadarmo. Podobné možnosti ponúkajú aj ďalšie databázy s chemickými alebo geografickými informá-

ciami, čo si žiaci taktiež prakticky vyskúšajú. Pri spracovaní geografických dát si žiaci pomocou doplnku *Microsoft 3D mapy pre Excel* vytvoria v tabuľkovom kalkulátore 3D mapu aktuálnych teplôt na Slovensku, obrázok 1.



Obrázok 1. Ukážka vytvorenej 3D mapy teplôt

V druhej tematickej dvojhodinovke sa pozornosť sústreďuje na programovanie v Pythone. Žiaci si najprv na náročnejších úlohách zopakujú a precvičia prácu s rôznymi dátovými typmi (Tabuľka 3).

Tabuľka 3. Úlohy na použitie dátových typov

Čo chceme dostať	***Z čoho to chceme získať***	***Ako to získame***
18.3	nameraná_teplota=’18.3°C’	
10	datum=‘2018-10-23‘	
[‘so‘, ‘ne‘]	dni=[‘po’,‘ut’,’st’,’št’,’pi’,’so’, ’ne’]	
18.3	zaznam=[‘2018-10-23‘, ‘18.3‘]	
[‘18.3°C‘, ‘15.1°C‘, ‘17.5°C‘]	merania=[[‘2018-04-23‘, ‘18.3‘], [‘2018-04-24‘,‘15.1‘], [‘2018-04-25‘, ‘17.5‘]]	
Najnižšia teplota bola 15.1°C Najvyššia teplota bola 18.3°C	teploty=[18.3, 15.1, 17.5]	

Následne na príkladoch skúmajú vlastnosti dátového typu slovník, základné príkazy pre prácu s ním a pomocou série praktických úloh tieto poznatky hneď precvičujú pri návrhu a vytvorení slovníka pre informácie o chemických prvkoch a pri vytváraní slovníka na určenie aminokyseliny na základe kodónu podľa priloženej tabuľky, ktorú budú mať k dispozícii. Toto je prípravná úloha pre riešenie komplexnejšej bioinformatickej úlohy pre prácu s DNA reťazcami v rámci špecializovaného biologického bloku predmetu IPVaM.

Finálnym výstupom práce s databázami v priebehu uvedených dvoch vyučovacích tém je samostatný programátorský projekt, v ktorom žiaci navrhnú jednoduchú verziu nástroja BLAST na porovnávanie sekvencií DNA, ktorá obsahuje referenčnú DNA (ako reťazec) a niekoľko testovaných DNA (v slovníku, kde kľúčmi sú mená testovaných osôb a hodnotami ich DNA reťazce). Program porovná DNA testovaných osôb s referenčnou DNA a pre každú osobu vypíše, či v testovanej DNA nastala alebo nenastala mutácia.

Napriek tomu, že takto spracované vyučovacie hodiny venované práci s databázami obsahujú veľmi silné väzby medzi predmetmi, na základe skúseností z výučby sa ukazuje, že pri citlivom spracovaní je takto možné veľmi prístupnou formou prepojiť prírodovedné témy s informatikou, čo podporuje motivačný charakter nášho predmetu.

4 ZAUJÍMAVÉ ÚLOHY

Prvé vzdelávanie učiteľov už prebehlo a bolo ukončené záverečnými prezentáciami vlastných tém, ktoré by bolo možné zaradiť do výučby. Uvedieme ukážky prezentovaných tém, ktoré boli pripravené na vysokej úrovni.

4.1 Úloha – Molekulové základy genetiky

Spracované podľa RNDr. M. Vavrovej, Gymnázium. sv. košických mučeníkov v Košiciach, ktorá riešila nasledujúci didaktický problém [7]: V každom okamihu v každom organizme v každej jednej bunke sa uskutočňujú procesy transkripcie, translácie, prípadne replikácie. Študenti si mnohokrát neuvedomujú dôležitosť týchto dejov a ich nevyhnutnosť pre existenciu a správne fungovanie buniek, organizmov. Pochopenie princípov prenosu genetickej informácie môže byť pre niektorých študentov spočiatku náročné, keď treba správne rozlíšiť, čo je matricou pre syntézu DNA, RNA a bielkovín a ako správne uplatniť princíp komplementarity v závislosti od toho, či vzniká RNA alebo DNA.

Na základe uvedenej aktivity žiaci svoje teoretické poznatky o ústrednej dogme molekulovej biológie uplatnia v praxi pri riešení komprimovaných súborov v Pythone, ktoré môžu aj modifikovať a použiť pri tvorbe vlastných programov.

V rámci riešenia problému učiteľka sledovala nasledujúce ciele:

- zdôvodniť odlišnosť štruktúry DNA a RNA v súvislosti s ich významom pre prenos genetickej informácie,
- vysvetliť proces replikácie, transkripcie a translácie (syntézy bielkovín),
- uplatniť princíp komplementarity a použiť ho,
- použiť jazyk Python pri riešení úloh z genetiky,
- modifikovať program v jazyku Python a aplikovať ho pri riešení úloh z genetiky.

Pre žiakov bol pripravený pracovný list, v ktorom sa nachádzali úlohy rôznych typov.

4.2 Úloha – Modelovanie kmitania/vlnenia

Spracované podľa G. Kacejovej (učiteľka fyziky) a V. Kopaničákovej (učiteľka informatiky), Gymnázium Janka Franciscihо-Rimavského, Levoča, ktorých cieľom bolo vytvoriť model na zobrazenie kmitania, alebo postupného vlnenia ako funkciu okamžitej výchylky od času, na pochopenie veličín vystupujúcich v rovniciach [8]:

$$y = y_m sin(2\pi ft + \varphi) \qquad y = y_m sin2\pi\left(\frac{t}{T} - \frac{x}{\lambda}\right)$$

a preskúmať závislosti okamžitej výchylky od amplitúdy, frekvencie a vlnovej dĺžky. Na základe skúmania v prostredí IP Coach vytvoriť model (matematický popis fyzikálneho javu) na skladanie dvoch rôznych kmitaní alebo vlnení, ktorý by žiaci

modelovali v programátorskom prostredí podľa ich výberu (Python, Excel, IP Coach, OverLeaf).

Závery a postrehy z výučby tejto témy:

Fyzika: žiaci sa pri nastavení rôznych parametrov kmitania/ vlnenia presvedčili, že výsledná amplitúda je väčšia/menšia ako amplitúdy jednotlivých zložiek, príp. aj nulová.

Aj to, že časový priebeh môže mať veľmi zložitý priebeh (napr. pri skladaní, ak zložky nemajú rovnakú frekvenciu, nemajú stály fázový posun...).

Informatika: žiaci sú schopní samostatne naštudovať a následne použiť nový typ softvéru; počas tvorby modelu sa vyskytli väčšie ťažkosti s inštaláciou knižníc v Pythone; spoločné úvahy o tom, ako navoliť príkazy, pomohli k lepšiemu pochopeniu fyzikálneho modelu kmitania/vlnenia

5 ZÁVER

26 učiteľov ukončilo vzdelávanie obhajobou záverečných prezentácií pred trojčlennou komisiou do konca januára 2019. Prezentácie boli zamerané na prípravu a vysvetlenie jednej témy pre výučbu, pričom východiskom boli poznatky získané v rámci vzdelávania. Rozdelenie prezentácií podľa predmetov, resp. v akých moduloch ich učitelia plánujú vyučovať je v tabuľke 3. Číselné hodnoty v tabuľke 3 vyjadrujú síce nerovnomernosť v počte prezentácií v moduloch, ale učitelia si vyberali predmet na prezentáciu a absolvovali vzdelávanie aj v druhom module. Aj keď počet informatických tém je rovný 1, teší nás, že informatika sa stála súčasťou v témach prezentácií v iných moduloch. Vo všeobecnosti učitelia považovali informatický modul za ťažší.

Tabuľka 4. Počty prezentovaných tém podľa modulov

Modul	Počet tém
Biológia	3
Chémia	9
Fyzika	6
Geografia	2
Informatika	1
Matematika	5

Ak by sme porovnávali úroveň prezentácií z hľadiska použiteľnosti vo výučbe v budúcnosti, konštatujeme, že v každej prezentácii bol prvok inovácie výučby splnený.

V module Biológia demonštrovali použitie programovania pri riešení biologických problémov, napríklad [7].

Témy v module Fyzika boli zamerané na modelovanie, napríklad Pohyb lyžiara dole svahom, Ako vzlieta lietadlo? Automobilové preteky. Modelovanie kmitania/vlnenia [8]. Modelovanie bolo urobené použitím programovacieho prostredia Coach. Spracovanie týchto prezentácií považujeme za výborné tiež vďaka ich zanieteným učiteľom.

Témy v module Matematika nadväzovali na konštrukčné úlohy, s ktorými sa možno stretnúť v stereometrii pri hľadaní rôznych výsekov kocky. Pri vykresľovaní bolo použité programové prostredie Asymptote v nadväznosti na TEX.

V module Chémia boli témy zamerané na vytvorenie zbierky protokolov na rôzne témy.

Po prvých skúsenostiach zo vzdelávania učiteľov budú niektoré časti pripravených materiálov modifikované a hlavne vylepšované.

POĎAKOVANIE

Tento článok bol vytvorený v rámci národného projektu IT Akadémia - vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje.
Vďaka patrí všetkým ďalším spolutvorcom predmetu – M. Ganajovej, V. Hubeňákovej, Z. Ješkovej, J. Kaňukovi, M. Kírešovi, M. Kožurkovej, S. Krajčímu, J. Sekerákovi.

BIBLIOGRAFICKÉ ODKAZY

[1] ANDREJKOVÁ, G., et al. *Koncepcia predmetu Informatika v prírodných vedách a matematike.* In: DRÁBKOVÁ Jindra a Jan BERKI, eds. Sborník konf. Didinfo 2018, Liberec, [online]. ISBN: 978-80-7494-424-6, ISSN: 2454-051X. Dostupné z: didinfo.net/images/DidInfo/files/didinfo_2018.pdf

[2] HROMKOVIČ, J., STEFFEN, B. *Why Teaching Informatics in Schools Is as Important as Teaching Mathematics and Natural Sciences.* In Proc. ISSEP 2011, Springer-Verlag, Berlin, Heidelberg, LNCS 7013, pp. 21-30

[3] Informatics Education in Europe: *Are We All on The same Boat? Report by* The Committee on European Computing Education (CECE), May 2017. Dostupné na: http://10.1145/3106077

[4] Národný projekt IT Akadémia - vzdelávanie pre 21. storočie, http://*itakademia.sk/sk/o-projekte/*

[5] http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_g_4_5_r.pdf

[6] Databáza *National Center for Biotechnology Information* https://www.ncbi.nlm.nih.gov/

[7] VAVROVÁ, M. Molekulové základy genetiky (Využitie Pythonu vo výučbe biológie). Záverečná prezentácia vzdelávania IPVaM, PF UPJŠ, Košice, 2019.

[8] KACEJOVÁ, G., KOPANIČÁKOVÁ, V. *Modelovanie kmitania/vlnenia.*. Záverečná prezentácia vzdelávania IPVaM, PF UPJŠ, Košice, 2019.

[9] BRABBHAM, K., *Does Exploring Non-Linear Models Address High School Students' Misconceptions of Linearity and Rate of Change?* 2014. Louisiana State University, Master's Theses. 3260.

$http://digitalcommons.lsu.edu/gradschoo\l_theses/3260$

Informatika v cirkuse
Informatics in the Circus

Daniela Bezáková
Katedra didaktiky matematiky, fyziky a informatiky, FMFI UK
Mlynská dolina
842 48 Bratislava
Slovensko
bezakova@fmph.uniba.sk

Andrej Blaho
Katedra aplikovanej informatiky, FMFI UK
Mlynská dolina
842 48 Bratislava
Slovensko
andrej.blaho@fmph.uniba.sk

Andrea Hrušecká
Katedra didaktiky matematiky, fyziky a informatiky, FMFI UK
Mlynská dolina
842 48 Bratislava
Slovensko
hrusecka@fmph.uniba.sk

Roman Hrušecký
Katedra didaktiky matematiky, fyziky a informatiky, FMFI UK
Mlynská dolina
842 48 Bratislava
Slovensko
hrusecky@fmph.uniba.sk

ABSTRACT
In our paper we present an educational - just being developed - software Circus, designed for preschool children. Its aim is to get in (first) touch with some informatics concepts like interpreter, action/command, sequence, order, record. We present the software itself and also some early experiments in two kindergartens.
The Circus software is a part of upcoming conception of informatics from preschool to upper secondary school.

Keywords
Informatics. Kindergarten. Circus. Educational software.

ABSTRAKT
Článok sa venuje vznikajúcemu edukačnému softvéru Cirkus, ktorý je určený pre deti predškolského veku. Jeho cieľom je propedeutika niektorých základných informatických pojmov ako vykonávateľ, akcia/príkaz, postupnosť, poradie, záznam. V článku predstavíme samotný softvér a jeho overovanie v dvoch materských školách.
Softvér Cirkus je súčasťou pripravovanej koncepcie informatiky od predškolského veku až po maturitu.

Kľúčové slová
Informatika. Materská škola. Cirkus. Edukačný softvér.

1 ÚVOD
Keď sa deti hrajú (napr. s edukačným softvérom), niekedy si rozvíjajú informatické koncepty, o ktorých ani netušia. A možno ani autori softvérov to nemali za cieľ. Takýchto softvérov existuje malé množstvo aj pre materskú školu. My sme sa však rozhodli navrhnúť edukačný softvér, ktorý by cielene rozvíjal informatické myslenie u detí predškolského veku.

Navrhli sme softvér Cirkus, ktorého cieľom je propedeutika niektorých základných informatických pojmov ako akcia/príkaz, postupnosť, poradie, záznam (jeho čítanie, vykonávanie a tvorba). Podľa nášho návrhu ho realizuje náš bývalý kolega Milan Moravčík so svojím tímom.

Softvér je momentálne vo fáze pilotného overovania. Overovali sme ho v dvoch materských školách v Bratislave, v každej s triedou predškolských detí.

2 KONCEPCIA INFORMATIKY
Edukačný softvér Cirkus je súčasťou pripravovanej koncepcie informatiky od predškolského veku až po maturitu [1]. Súčasťou tejto koncepcie je aj prostredie Robot Emil, ktoré sa zameriava na programovanie [2]. Momentálne Robot Emil pokrýva celú oblasť programovania pre 3. ročník základnej školy. V súčasnosti sa pripravuje overovanie nových prostredí pre druhý a štvrtý ročník základnej školy.

Koncepcia vzniká v spolupráci Katedry didaktiky matematiky, fyziky a informatiky na Fakulte matematiky, fyziky a informatiky Univerzity Komenského v Bratislave[1] a neziskovej organizácie Indícia[2].

3 O SOFTVÉRI CIRKUS
Celý softvér je navrhnutý tak, aby s ním pracovali deti pri interaktívnej tabuli v malej skupine (odporúčame 4-6 detí), pričom aktivity manažuje učiteľka. V celom softvéri nie je žiadne testovanie ani overovanie riešení a celá „kontrola“ je výlučne na deťoch a ich diskusii a argumentácii medzi sebou prípadne s učiteľkou. Táto metóda je inšpirovaná tzv. Hejného metódou [3].

Tému cirkusu sme zvolili zámerne, nakoľko je deťom známa a majú ju rady. Hlavnou postavou v našom cirkuse je šašo, ktorý vie vykonávať rôzne akcie. Vybrali sme niekoľko takých akcií, ktoré sú typické pre cirkus, deti sa s nimi mohli, resp. majú šancu stretnúť, a aby boli pre deti atraktívne. V neposlednom rade bolo rozhodujúcim aj to, ako veľmi náročná bude realizácia týchto akcií. Snahou bolo, aby väčšinu týchto akcií vedeli deti napodobňovať, resp. zahrať.

Momentálne sú k dispozícii nasledujúce akcie[3]: hra na bubon, hra na trúbku, nafukovanie balónov, jazda na monocykli, potlesk, výskok, čarovanie z klobúka, mlynské koleso, púšťanie bubliniek.

Obrázok 1: Akcie šaša

V prvých verziách softvéru sa všetky akcie vždy vykonávali rovnako. V aktuálnej verzii už je pri vykonávaní týchto akcií istá náhodnosť - rôzny štýl vyskakovania, potlesku, či bubnovania, rôzny počet bubliniek, rôzne farby balónov, čarovanie rôznych

[1] https://fmph.uniba.sk/pracoviska/katedra-didaktiky-matematiky-fyziky-a-informatiky/

[2] https://www.indicia.sk/

[3] Zoznam akcií ešte nie je úplný. Sú navrhnuté a rozpracované aj ďalšie. Neplánujeme ho však priveľmi rozšíriť.

vecí. Takisto sa niekedy akcia nevykoná, resp. nie úplne, ak ide za sebou viacero rovnakých akcií.

4 ČLENENIE SOFTVÉRU CIRKUS

Softvér Cirkus je rozdelený do 4 úrovní:

1. Šašo so základnou množinou akcií
2. Šašo s rozšírenou množinou akcií
3. Šašo a Šašinka
4. Šašinka predcvičuje Šašovi

Každá z úrovní je rozdelená do 3 režimov:

1. Tréning
2. Predstavenie
3. Šatňa

4.1 Úrovne Cirkusu

V prvej a druhej úrovni deti trénujú Šaša jednotlivé akcie. Trénovanie prebieha v režime *Tréning*. Kliknutím na akciu ju Šašo **ihneď vykoná** a táto akcia sa pridá do záznamu akcií. Ten je obmedzený na 7 akcií. Záznam môžu deti vymazať. Natrénovanú zostavu akcií si deti môžu kedykoľvek prehrať ako predstavenie (režim *Predstavenie*). Prvá a druhá úroveň sa líšia len množinou akcií, ktoré môže Šašo robiť. V druhej úrovni pribudnú akcie, v ktorých Šašo mení pozíciu.

Obrázok 2: Druhá úroveň – Šašo s rozšírenou množinou akcií

V tretej úrovni sú od začiatku v manéži Šašo aj Šašinka, pričom každý má svoj vlastný záznam akcií. Počas tréningu môže byť aktívny len jeden z nich, čo meníme kliknutím na Šaša, resp. Šašinku (alebo na ikonu pred záznamom).

Obrázok 3: Tretia úroveň – Šašo a Šašinka

V prvých troch úrovniach si deti môžu vytvárať ľubovoľné záznamy akcií.

V štvrtej úrovni je Šašinka v roli predcvičovateľky. Úlohou detí je natrénovať Šaša podľa toho, aké akcie predvedie Šašinka. Záznam akcií je len jeden – pre Šaša. Aj keď úlohou detí je natrénovať Šaša podľa predcvičovateľky Šašinky, neexistuje v softvéri žiadne overovanie, či sú ich akcie rovnaké. Ako Šašo a Šašinka cvičia naraz môžu deti vidieť v režime *Predstavenie*, a či cvičia to isté je len na diskusii medzi deťmi, prípadne detí s učiteľkou.

Šašinka má pripravených niekoľko postupností akcií (zadaní) od jednoduchých (3 akcie) až po zložitejšie (7 akcií). Medzi jednotlivými zadaniami sa deti prepínajú pomocou šípok v softvéri.

Na rozdiel od predchádzajúcich úrovní, v tejto sa akcia nevykoná ihneď po kliknutí, len sa pridá do záznamu akcií. Cieľom tejto úrovne je, aby sa deti učili **plánovať** postupnosť akcií.

Obrázok 4: Štvrtá úroveň – Šašinka predcvičuje Šašovi

4.2 Režimy Cirkusu

Keď sa spustí niektorá úroveň Cirkusu, spustí sa v režime *Tréning* (manéž bez divákov). V tomto režime dieťa trénuje niektorého šaša podľa „pravidiel“ úrovne. Šašo, prípadne Šašinka, majú na sebe tréningové oblečenie. V tomto režime je dostupná ponuka akcií, na ktoré môžu deti klikať a tak trénovať Šaša, resp. Šašinku.

Keď sú Šašo, resp. Šašinka, natrénovaní, môžu deti úroveň prepnúť do režimu *Predstavenie*. Tento režim je zobrazený ako manéž s divákmi, pričom už nie sú dostupné žiadne akcie. Po vstupe do tohto režimu začnú Šašo aj Šašinka vykonávať svoje akcie, pričom vykonávaná akcia sa zvýrazňuje na príslušnom zázname akcií. Šašo aj Šašinka sú už (pekne) oblečení (oblečenie je buď preddefinované alebo ho môžu deti zmeniť v režime *Šatňa*).

V každej úrovni, z režimu *Tréning* a režimu *Predstavenie*, môžu deti prejsť do režimu *Šatňa*. V tomto režime môžu deti prezliecť Šaša, resp. Šašinku (závisí od úrovne). Ak deti prejdú do režimu *Tréning*, zvolené oblečenie je na vešiaku. Ak prejdú do režimu *Predstavenie*, tak Šašo, prípadne Šašinka, majú zvolené oblečenie na sebe a v ňom predvádzajú predstavenie.

Deti môžu v šatni prezliekať tričko, nohavice, topánky, klobúk. Ponuka oblečenia sa bude ešte rozširovať.

Obrázok 5: Režim Šatňa

5 OVEROVANIE SOFTVÉRU

Overovanie prebehlo v dvoch materských školách v Bratislave s deťmi predškolského veku. Bolo zamerané predovšetkým na samotný softvér, jeho funkčnosť, ovládanie, dizajn. V každej z týchto materských škôl prebehli tri stretnutia.

5.1 Organizácia overovania

Na stretnutiach boli vždy prítomní dvaja až štyria členovia nášho tímu (ďalej výskumníci), učiteľka príslušnej triedy a riaditeľka materskej školy. Jedného stretnutia sa zúčastnili aj dve praktikantky. Detí bolo na každom stretnutí približne 20.

Z organizačného hľadiska boli všetky stretnutia rovnaké. Na začiatku sme všetci (deti, učiteľky, výskumníci) sedeli spoločne na koberci a rozprávali sa s deťmi - snažili sme sa navodiť atmosféru. Potom nasledovala práca v skupinách, zvyčajne štyroch. Jedna skupina detí pracovala pri interaktívnej tabuli - túto aktivitu viedol vždy jeden z výskumníkov. Ďalšie skupiny pracovali na iných aktivitách mimo interaktívnej tabule. Ich prácu manažovala pani učiteľka. Po približne 10-12 minútach sa skupiny vymenili. Po necelej hodine práce si tak všetky deti vyskúšali všetky aktivity, vrátane práce so softvérom pred interaktívnou tabuľou. Náplň aktivít mimo interaktívnej tabule navrhovali a pripravili učiteľky z príslušnej materskej školy. Aktivity prispôsobili téme Cirkus, alebo téme, ktorou sa práve v danom týždni zaoberali, prípadne skĺbili obe témy dohromady. Pre jedno stretnutie sme jednu mimotabuľovú aktivitu pripravili my - išlo o doplňanie chýbajúceho obrázka v postupnosti, pričom postupnosť tvorili obrázky akcií zo softvéru. Na záver sme sa zase všetci spoločne posadili na koberec, zopakovali sme si, čo sme dnes robili a zisťovali sme, aké dojmy mali deti z tejto hodiny. V oboch materských školách ešte nasledovala krátka pohybová aktivita - tanec na nejakú pesničku, ktorú zvolila pani učiteľka.

5.2 Reakcie detí

Deťom sa práca so softvérom Cirkus páčila. Veľmi rady predvádzali akcie, ktoré robili Šašo a Šašinka, teda spolu s nimi tlieskali, vyskakovali, napodobňovali fúkanie balónov či bubliniek, čarovanie z klobúka. Poradili si aj s akciami ako mlynské koleso, či monocykel - vymysleli si za ne vlastnú zjednodušenú náhradu, teda len naznačili, že robia mlynské koleso, resp. že krútia pedálmi.

Obrázok 6: Šašo deti inšpiroval k hraniu na bubnoch

Deti boli veľmi vnímavé a postrehli zmeny medzi jednotlivými verziami softvéru (napr. Šašo už netlieska len nad hlavou, kúzlenie rôznych predmetov...).

Posledná (štvrtá) úroveň bola pre niektoré deti menej zaujímavá, lebo už museli plniť konkrétne úlohy (ktoré predvádzala Šašinka) a nemohli voliť akcie podľa seba. Pri nej však najviac spolupracovali, najmä vtedy, ak si museli zapamätať viacero akcií, ktoré predvádzala Šašinka.

Veľký úspech mala šatňa. Dievčatá by v nej však ocenili viac dievčenského oblečenia pre Šašinku.

Obrázok 7: Deti obliekajú Šaša

5.3 Reakcie učiteliek

Miernou nevýhodou bolo, že učiteľky v oboch materských školách len málo stíhali sledovať skupinu pracujúcu pri interaktívnej tabuli, zväčša sa venovali ostatným skupinám. Tiež po skončení aktivity sa ďalej museli venovať deťom. Preto ich názor na softvér a prácu s ním vlastne nebolo možné získať. Nejakú spätnú väzbu sme však v oboch materských školách dostali od riaditeliek, s ktorými sme vždy po ukončení aktivity v triede zhodnotili stretnutie.

Učiteľky si uvedomili, že práca so softvérom pri interaktívnej tabuli vyžaduje prítomnosť učiteľa, čo znamená, že aktivity pre ostatné skupiny treba pripraviť tak, aby ich riadenie bolo minimálne.

Ďalej prejavili záujem o to, aby sme pre každé stretnutie jednu z mimotabuľových aktivít navrhli my - mohla by nejako súvisieť so softvérom.

5.4 Závery z overovaní

Deti nemali žiadne problémy s ovládaním softvéru. Práca so softvérom bola pre väčšinu detí intuitívna. Milo nás prekvapilo, ako deti rozumeli obrázkom jednotlivých akcií pre Šaša a Šašinku - podľa obrázka uhádli takmer všetky akcie. Taktiež úplne

prirodzene vedeli vymazať záznam pomocou gumy (možno sa už s nástrojom guma stretli v nejakom detskom kresliacom softvéri). Nástroje na prechody medzi režimami *Predstavenie*, *Tréning* a *Šatňa* im stačilo raz vysvetliť a ďalej ich už používali úplne prirodzene. Iba v štvrtej úrovni nie je pre deti intuitívne, čo sa od nich očakáva. Zo začiatku bolo pre nich mätúce, že akcie sa nevykonávali hneď. Po vysvetlení už dokázali s touto úrovňou pracovať úplne samostatne.

Pri ovládaní šatne sa prejavilo, že dnešné deti sú zvyknuté na prácu so zariadeniami ako tablet či smartfón, pretože dvere na skrini sa snažili otvárať posúvaním (v softvéri sa to robí klikaním). Uvažujeme, že pridáme tento spôsob ovládania ako alternatívny.

Overili sme si, že pre náš cieľ je ideálny počet detí v skupine pracujúcej s interaktívnou tabuľou 4 až 6. V jednom prípade sme pracovali so skupinami, v ktorých boli len tri deti. Ukázalo sa, že je to až zbytočne málo, pretože (pri rovnakom celkovom počte detí) tak vzniká veľa skupín, je potrebné pripraviť viac iných aktivít a zvyšuje sa náročnosť riadenia práce jednotlivých skupín. Pri skupinách väčších ako šesť detí zase nastáva problém, že deti pri interaktívnej tabuli sa nestihnú vystriedať, resp. predlžuje sa čas, kedy sa dieťa "dostane k slovu" a predlžuje sa aj celkový čas aktivity.

Obrázok 8: Overovanie tretej úrovne

6 METODIKA

K softvéru Cirkus pripravujeme aj metodický materiál pre učiteľa. Prácu so softvérom odporúčame realizovať na štyroch stretnutiach, ktoré zodpovedajú jednotlivým úrovniam softvéru. V úvode metodického materiálu popisujeme spôsob organizácie týchto stretnutí, ktorý je rovnaký pre všetky štyri stretnutia. Ďalej uvádzame metodické listy ku každému stretnutiu. V závere metodického materiálu učiteľ nájde námety na ďalšie aktivity. Prílohou materiálu budú aj súbory s obrázkami určené na tlač, ktoré sa využívajú v jednotlivých aktivitách.

Činnosť na každom stretnutí navrhujeme rozdeliť na tri etapy. V prvej a tretej etape pracuje učiteľ s celou triedou v strede učebne, neďaleko interaktívnej tabule. V druhej etape sú deti rozdelené na štyri približne rovnako veľké skupiny po cca 4-6 detí:

- jedna skupina detí pracuje (s učiteľom) pri interaktívnej tabuli,
- jedna skupina rieši za stolom úlohy čiastočne súvisiace so softvérom, ktorých zadania sú nahovorené na hovoriacich štipcoch[4] - ide prevažne o úlohy, v ktorých deti strihajú, lepia a dopĺňajú, resp. vytvárajú postupnosť obrázkov podľa určitého zadania,
- zvyšné dve skupiny organizuje učiteľ podľa svojho uváženia: môže pre každú z nich pripraviť samostatnú činnosť alebo môžu obe skupiny pracovať spoločne na tej istej aktivite.

Obrázok 9: Aktivita, v ktorej deti dokončovali a vytvárali postupnosti akcií lepením

Asi po 10 - 12 minútach sa skupiny vymieňajú. Po necelej hodine by si každá skupina mala vyskúšať prácu pri interaktívnej tabuli, aj všetky ostatné aktivity.

V metodických listoch k jednotlivým úrovniam softvéru navrhneme:

- spoločnú úvodnú etapu a záverečnú etapu, ktoré si učiteľ môže doplniť podľa potreby,
- postup pre skupinu pri interaktívnej tabuli,
- jednu aktivitu za stolom mimo interaktívnej tabule, ktorá súvisí s tým, čo sa robí na tabuli,
- pomôcky k aktivite so softvérom aj k aktivite za stolom mimo tabule.

7 ZÁVER

Navrhnutý softvér bol pre deti zaujímavý a aj pri opakovaných overovaniach mali chuť s ním pracovať. Overovania nám ukázali ako softvér vylepšiť a čo zaradiť do metodických materiálov pre učiteľky.

Po dokončení softvéru a metodík ich plánujeme overiť v materských školách, pričom by hodiny viedli samotné učiteľky.

BIBLIOGRAFICKÉ ODKAZY

[1] O Informatike s Emilom. https://www.indicia.sk/aktualne-skolenia/informatika-s-emilom/o-informatike-s-emilom

[2] Robot Emil. https://www.robotemil.com/sk/

[3] Hejného metoda. https://www.h-mat.cz/hejneho-metoda

[4] http://www.stiefel-eurocart.sk/digitalne-interaktivne-pomocky/408-hovoriace-stipce.html

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN ..... ISSN .....

Využitie minipočítača Raspberry Pi v informatických predmetoch na stredných školách

Use of Raspberry Pi Minicomputer in Computer Science Lessons at High Schools

Miroslav Biňas
Katedra počítačov a informatiky,
Technická univerzita Košice
Letná 9
04200 Košice
Slovensko
miroslav.binas@tuke.sk

Matej Madeja
Katedra počítačov a informatiky,
Technická univerzita Košice
Letná 9
04200 Košice
Slovensko
matej.madeja@tuke.sk

ABSTRACT

In this paper, we describe the impact of 6 days long training of highschool teachers about the *Internet of things*. We used *Raspberry Pi* as the main building block of the training and *Python* as a programming language. During the training we tried to present interesting resources and libraries for *Python* language to the teachers, and show them that with the choice of the right technologies, the lessons can be fun for the teachers and for the students too. The impact on the teachers was visible during the training. Another interesting impact was visible on the group of high school students, which we had oportunity to train after the training.

Keywords

Raspberry Pi. Python. Programming. Internet of things.

ABSTRAKT

V tomto článku opíšeme dopad 6-dňového školenia stredoškolských učiteľov v oblasti *Internetu vecí*. Ako hlavný stavebný prvok sme použili minipočítač *Raspberry Pi* a programovací jazyk *Python*. V rámci školenia sme sa snažili učiteľom prezentovať zaujímavé zdroje a knižnice v jazyku *Python*, a ukázať im, že voľbou vhodných technológií dokážu byť vyučovacie hodiny zábavné ako pre učiteľa, tak aj pre študentov. Dopad kurzu sme na učiteľoch mohli sledovať už počas školenia. Nemenej zaujímavý dopad mal kurz aj na skupine stredoškolských študentov, ktorých sme mali možnosť školiť po skončení školenia učiteľov.

Kľúčové slová

Raspberry Pi. Python. Programovanie. Internet vecí.

1 ÚVOD

Problémy so vzdelávaním (nielen) informatických predmetov na stredných školách sú nám na Slovensku dôverne známe už dlhé roky. Medzi štandardné problémy dlhodobo patria:

- nedostatok (až úplná absencia) učebníc;
- nedostatočné (až žiadne) vybavenie potrebné pre výučbu konkrétnych odborných predmetov;
- nákup vybavenia, ktoré sa nakoniec z rozličných dôvodov nepoužíva;
- veľké množstvo odporúčaní na tvorbu metodík bez vhodných príkladov;
- metodiky, ktoré autori vypublikovali bez toho, aby ich predtým odskúšali; a
- nedostatok školení, ktoré by učiteľom pomohli v aplikovaní nových postupov a technológií, poprípade by ich inšpirovali k zmene v predmetoch, ktoré vyučujú.

Medzi štandardné problémy je možné zaradiť aj mnohé projekty, ktoré síce prinášajú dobré myšlienky a riešenia, ale ich podpora po ukončení projektu okamžite skončí. Poprípade absencia portálu, kde by si učitelia mohli nájsť články prezentujúce nové postupy, návody, technológie alebo softvér, ktoré by mohli použiť vo svojej výučbe a čo je dôležité – mali by dôvod sa na takýto portál vracať pravidelne. Rovnako je možné medzi problémy zahrnúť aj absenciu časopisov ako *ABC* alebo *Zenit*, poprípade neskorší *Elektrón Zenit* z 90-tych rokov minulého storočia, ktoré by dokázali obohatiť ako učiteľov, tak aj študentov.

Jeden z ďalších projektov, ktorý sa snaží pomôcť stredoškolským učiteľom informatických predmetov zlepšiť ich kvalitu, ale aj zaviesť do výučby nové predmety, je projekt *IT Akadémia – vzdelávanie pre 21. storočie*[1]. V rámci tohto projektu máme (na *Katedre počítačov a informatiky*, *Fakulty elektrotechniky a informatiky*, na *Technickej univerzite v Košiciach*) možnosť pomôcť učiteľom stredných škôl zaviesť do výučby nový predmet *Internet vecí*. Na tomto predmete je zaujímavé to, že prepája (minimálne) tri rozličné oblasti navzájom: programovanie (softvér), elektroniku (hardvér) a komunikáciu v rámci internetu (počítačové siete). Pri príprave tohto kurzu sme si stanovili dva hlavné ciele:

1. ukázať učiteľom, že vhodným spôsobom vie byť tento predmet veľmi zábavný; a
2. prezentovať minipočítač *Raspberry Pi* ako pomôcku pre absolvovanie (nielen) tohto predmetu.

K stanoveniu týchto cieľov nás viedlo niekoľko dôvodov. Tým prvým bol ten, že minipočítač *Raspberry Pi* bol vytvorený práve s cieľom vytvoriť lacný počítač pre školy. Druhým dôvodom je, že minipočítač *Raspberry Pi* bol úspešne nasadený do škôl vo *Veľkej Británii*, ale rovnako tak aj v *Estónsku* [2]. Tretím dôvodom je veľké množstvo projektov a materiálov, ktoré vytvára *Raspberry Pi Foundation*[2] a poskytuje ich zdarma.

1 http://itakademia.sk/

2 https://www.raspberrypi.org/

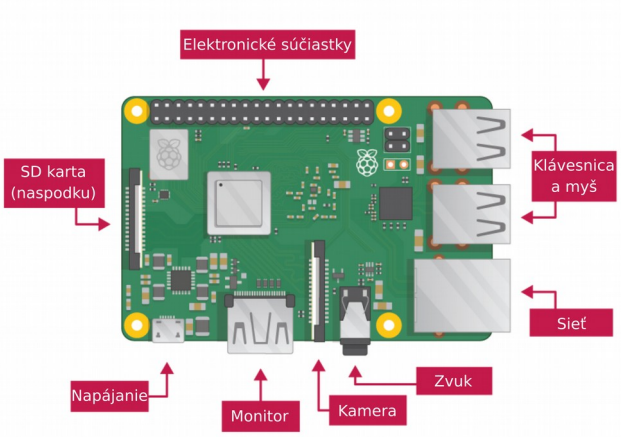
ISBN ..... ISSN .....

V tomto článku teda opíšeme skúsenosti, ktoré sa nám podarilo po absolvovaní dvoch behov tohto kurzu získať. V rámci týchto dvoch behov sme mali možnosť sa spoznať a pracovať s 28 učiteľmi stredných škôl. Okrem toho sme mali možnosť pracovať aj s dvoma skupina študentov, či už v našich priestoroch, alebo priamo na stredných školách. Rovnako v článku predstavíme možnosti využitia minipočítača *Raspberry Pi* pre rozličné použitie v rámci výučby na stredných školách, ako aj podporu, ktorú je možné získať priamo z výstupov *Raspberry Pi Foundation*.

Jeden beh kurzu mal spolu 6 stretnutí. Účastníci ho absolvovali v troch dvojdňových stretnutiach s odstupom obyčajne dvoch týždňov.

2 MINIPOČÍTAČ RASPBERRY PI

Minipočítač Raspberry Pi (viď obrázok 1) je jednodoskový štvorjadrový minipočítač určený primárne pre školy. Najmä vďaka svojej cene (svetovo okolo *$35*, u nás ho je možné kúpiť za zhruba *40€*) sa stal jedným z najpopulárnejších a najpredávanejších minipočítačov na svete vôbec. Jeho použitie zahŕňa okrem školstva množstvo tzv. „hobby“ projektov, kedy sa používa pre riešenia inteligentnej domácnosti alebo rozličných domácich zábavných/multimediálnych centier.



***Obrázok 1: Raspberry Pi 3 Model B+* [8]**

Práve cena zariadenia môže byť veľmi zaujímavá pre použitie v samotných školách. Hardvérové parametre síce nepredurčujú použitie *Raspberry Pi* pre náročné aplikácie, ale ako pomôcka vo výučbe IT predmetov je ideálna. Okrem programovania je vďaka vyvedeným *GPIO pinom* a sieťovým rozhraniam možné *Raspberry Pi* použiť aj na tvorbu rozličných zariadení pre *Internet vecí*.

3 PROFIL STREDOŠKOLSKÝCH UČITEĽOV A ICH ŠTUDENTOV

Ako bolo uvedené skôr, mali sme možnosť pracovať s dvoma skupinami stredoškolských učiteľov. Vzhľadom na to, že ide o kurz *Internet vecí*, učitelia mali skúsenosť buď s programovaním v aspoň jednom programovacom jazyku alebo s elektronikou. V tom lepšom prípade mali skúsenosti v oboch týchto oblastiach.

Pozitívnym zistením bolo, že drvivá väčšina týchto učiteľov mala aspoň základné skúsenosti s programovacím jazykom *Python*[3]. V tejto súvislosti bola najčastejšie spomínaná učebnica pána *Kučeru – Programujeme v Pythone* [1]. Aj napriek tomu, že sa mnohí z nich zhodli, že táto učebnica nie je ideálna, pretože vo veľkej miere len popularizuje *Python* pomocou grafickej knižnice *Tkinter*[4], používajú ju, pretože inú nemajú. To vlastne potvrdzuje jeden z problémov uvedených v úvode článku – nedostatok vhodných učebníc a študijných materiálov. Rovnako tak niektorí učitelia absolvovali školenie organizované priamo autorom tejto publikácie.

Jazyk *Python* sa stal komunikačným jazykom kurzu. Všetky úlohy, ktoré boli v rámci kurzu riešené, boli realizované v tomto jazyku, čo značne uľahčovalo prácu. V prvom behu kurzu sme absolvovali odbočku do jazyka *C* v rámci predstavenia mikrokontroléra *Arduino*[5], ale v druhom behu sme túto časť zamenili za komunikáciu pomocou protokolu *Firmata*[6] tiež v jazyku *Python*.

Po absolvovaní prvého behu kurzu sme nadviazali spoluprácu s dvoma učiteľkami. Buď k nám chodili so svojimi študentmi alebo sme my chodili k nim do škôl. Títo študenti mali taktiež základné znalosti s programovaním v jazyku *Python*, ale chýbali im skúsenosti a znalosti s elektronikou. To práve poskytovalo priestor pre pomoc s výučbou a popularizáciou *Raspberry Pi*.

4 ZDROJE RASPBERRY PI FOUNDATION

Raspberry Pi Foundation je nezisková organizácia, ktorá stojí za vývojom minipočítača *Raspberry Pi*. Okrem toho v súčasnosti zastrešuje rozličné ďalšie organizácie ako je *Code Club*[7] alebo *CoderDojo*[8]. Svojimi aktivitami sa snažia podporovať mladých ľudí v štúdiu technických oblastí, ako aj učiteľov, ktorí tieto oblasti vyučujú. Či už oficiálne v školách alebo ako dobrovoľníci v rámci voľnočasových aktivít.

Jednou z činností, ktoré *Raspberry Pi Foundation* ponúka, je aj tvorba vzdelávacích materiálov. Všetky materiály, ktoré vyrobili, sú k dispozícii zdarma. Študenti alebo jednotlivci sa teda môžu vzdelávať sami v rámci zdrojov, ktoré majú k dispozícii online na stránke projektov[9]. Učitelia sa zasa môžu pomocou týchto projektov inšpirovať a použiť ich priamo vo svojej výučbe.

Okrem online zdrojov vydáva *Raspberry Pi Foundation* už 4 časopisy (*MagPi*[10], *HackSpace*[11], *Hello World*[12] a *Wireframe*[13]). Tie je možné objednávať v tlačenej podobe, ale aj voľne stiahnuť v elektronickej podobe. Podobným spôsobom už vydali aj niekoľko kníh.

Okrem toho organizujú aj školenia pre učiteľov s názvom *Picademy*, na ktorých školia priamo ľudia z *Raspberry Pi*

3 https://www.python.org/

4 https://wiki.python.org/moin/TkInter

5 https://www.arduino.cc/

6 https://github.com/firmata/protocol

7 https://codeclub.org/en/

8 https://coderdojo.com/

9 https://projects.raspberrypi.org/en/

10 https://www.raspberrypi.org/magpi/

11 https://hackspace.raspberrypi.org/

12 https://helloworld.raspberrypi.org/

13 https://wireframe.raspberrypi.org/

Foundation a používajú materiály, ktoré sú voľne dostupné na ich stránkach.

V roku 2018 vznikol projekt na preklad vzdelávacích materiálov do rozličných jazykov [3]. Pomaly teda pribúdajú preklady aj do slovenčiny, čo umožňuje aj našim učiteľom používať existujúce materiály priamo na stránkach portálu *RaspberryPi.org*.

Okrem uvedených projektov sa samozrejme *Raspberry Pi Foundation* venuje aj správe a udržiavaniu distribúcie *Raspbian*, ktorá predstavuje odporúčaný linuxový operačný systém pre *Raspberry Pi*.

5 MOŽNOSTI VYUŽITIA RASPBERRY PI VO VÝUČBE NA STREDNÝCH ŠKOLÁCH

Operačný systém *Raspbian* je vo verzii s odporúčaným softvérom vybavený obrovským množstvom softvéru určeného pre výučbu. Okrem štandardných aplikácií, akými sú internetový prehliadač alebo kancelársky balík, obsahuje aj komerčné nástroje, ktoré majú licenciu pre použitie na minipočítači *Raspberry Pi* zdarma. Jedná sa napríklad o *Wolfram Mathematica* alebo o hru *Minecraft*, ktorú je možné využiť ako nástroj na popularizáciu výučby programovania (v jazyku *Python*).

Spolu s učiteľmi sme počas kurzu preberali rozličné možnosti využitia minipočítača *Raspberry Pi* v ich výučbe, resp. sme hľadali alternatívy k tomu, čo aktuálne robia a používajú v systéme *Raspbian*. V nasledujúcom texte budú zhrnuté niektoré z nich.

5.1 Programovanie v jazyku Python

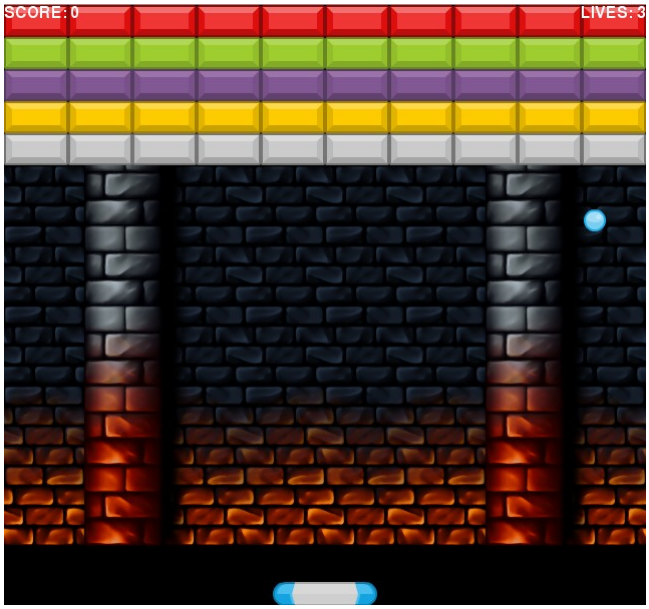
Jazyk *Python* sa už na stredných školách vyučuje. Častokrát sa však vyučuje rovnako, ako sa programovanie vyučovalo v 90-tych rokoch minulého tisícročia – tvorbou textových aplikácií, na ktorých sú prezentované jednotlivé preberané témy. Často taktiež používajú štandardný editor *IDLE*, ktorý síce poskytuje základné rozhranie, ale rozhodne nejde s dobou.

V oblasti editorov sa však v roku 2018 zmenila situácia, pretože boli vytvorené dva nové editory vhodné pre začiatočníkov programujúcich v jazyku *Python*. Prvým z nich je editor *Thonny*[14], ktorý je postavený rovnako ako editor *IDLE* na knižnici *Tkinter*. Výhodou tiež je, že je súčasťou distribúcie *Raspbian*.

Druhým editorom je editor *Mu*[15]. Ten síce nie je súčasťou *Raspbian*-u, ale je možné ho do systému doinštalovať. Tento editor umožňuje pracovať v niekoľkých režimoch podľa toho, čo chce programátor programovať, napr. hry pomocou knižnice *Pygame Zero*[16].

A tu je práve jedna z výhod použitia *Raspberry Pi* a jeho *Raspbian*-u – v štandardnej verzii sa nachádza veľké množstvo predinštalovaných knižníc. Jednou z nich je aj uvedená knižnica *Pygame Zero*, ktorú vytvoril v roku *2015 Daniel Pope* ako odpoveď učiteľom na ich reakciu, že používanie hernej knižnice *Pygame* vo výučbe je ťažkopádne [4]. Pomocou knižnice *Pygame Zero* je totiž možné vytvárať počítačové hry veľmi jednoducho a rýchlo. Takýto prístup môže zvýšiť záujem a motiváciu o programovanie.

V rámci kurzu sme túto knižnicu aj prezentovali. V prvom behu sme s učiteľmi vytvorili jednoduchú verziu hry *Space Invaders* a v druhom zasa *Arkanoid*. V oboch prípadoch ide o kód, ktorý nepresiahol 100 riadkov a spoločné programovanie (s analýzou a návrhom) nebolo dlhšie ako 3 hodiny, pričom výsledkom bola fungujúca hra. V druhom prípade sme dokonca vytvorili aj herný ovládač pomocou dvoch tlačidiel, čo viedlo k zmene piatich riadkov kódu, ale jednalo sa o jednoduchú ukážku prepojenia elektroniky s programovaním. Snímka obrazovky z výslednej hry *Arkanoid* sa nachádza na obrázku 2.



Obrázok 2: Hra Arkanoid vytvorená pomocou knižnice Pygame Zero

Popularizáciu tejto knižnice je možné vidieť priamo v časopise *MagPi* a v súčasnosti aj v časopise *Wireframe*. Zaujímavú ukážku knižnice je tiež možné nájsť v knihe [5].

Podobný efekt môže mať aj využitie hry *Minecraft*, ktorej verzia *Minecraft Pi* sa nachádza v distribúcii *Raspbian*. V tomto prípade je možné staviť na skutočnosť, že študenti majú s touto hrou skúsenosti a vo svete *Minecraft*-u strávili určite hodiny času. Pomocou knižnice *mcpi* získajú možnosť ovládať hru, resp. zasahovať do nej pomocou jazyka *Python*. Tým pádom sa stane pre nich výučba programovania zábavnejšou. Ukážka programu *„Hello world!“* s použitím knižnice *mcpi* sa nachádza nižšie. Výstup je zobrazený na obrázku 3.

```
from mcpi.minecraft import Minecraft
mc = Minecraft.create()
mc.postToChat("Hello world!")
```

Pomocou knižnice *mcpi* je okrem iného možné napr. umiestňovať do sveta *Minecraftu* stavebné bloky a vytvárať tak v programe rozličné stavby alebo elegantne ilustrovať, ako pracujú cykly postupným zobrazovaním jednotlivých stavebných blokov vytvárajúc hranoly (jednoduchý cyklus), plochy (cyklus v cykle) alebo celé kvádre (cyklus cez tri súradnice).

14 https://thonny.org/

15 https://codewith.mu/

16 https://pygame-zero.readthedocs.io/en/stable/

Obrázok 3: Hello world! v jazyku Python v hre Minecraft

V súvislosti s hrou *Minecraft* a programovaním v jazyku *Python* ponúka *Raspberry Pi Foundation* niekoľko zdrojov, ktoré je možné použiť. Pre výučbu programovania v jazyku *Python* v hre *Minecraft* je však možné použiť aj knihu *Adventures in Minecraft* [6] od *Martina O'Hanlona*, ktorý publikuje články na túto tému aj v časopise *MagPi* a prevádzkuje stránku *<Stuff about="code"/>*[17], ktorá obsahuje veľké množstvo príkladov toho, čo všetko je možné v jazyku *Python* v hre *Minecraft* vytvoriť.

V oboch prípadoch sa nadšenie dostavilo aj na učiteľov, aj keď sa im viac páčilo programovať hru pomocou knižnice *Pygame Zero*. V prípade programovania s hrou *Minecraft* sa prejavila absencia ich skúseností s hraním tejto hry.

5.2 Operačný systém Linux

Ako bolo uvedené, systém *Raspbian* je linuxový systém. Znalosť linuxového systému je dnes žiadaná alebo aspoň výhodná aj na trhu práce. Hlavne na pozíciách označovaných ako *DevOps*, ktorá predstavuje zlúčenie schopností programátora a administrátora.

Pre bežnú prácu so systémom *Raspbian* si však vo veľkej miere používatelia vystačia len so základnou znalosťou operačného systému *Linux*. Nakoľko je *Raspbian* grafickou distribúciou, na väčšinu bežných operácií nie je potrebná žiadna znalosť príkazov z príkazového riadku.

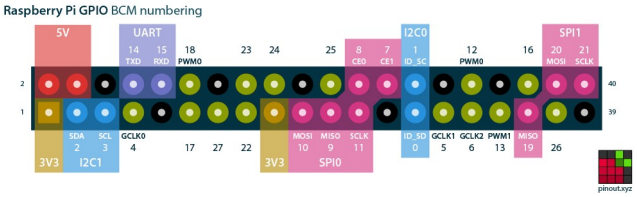
5.3 Práca s elektronikou

Minipočítač *Raspberry Pi* obsahuje *40 GPIO pinov* (ich rozloženie sa nachádza na obrázku 4) pre pripojenie rozličných externých *senzorov* a *akčných členov*. Prístup k nim je možné opäť zabezpečiť pomocou rozličných knižníc jazyka *Python*. Pre rýchle zoznámenie sa s problematikou je možné použiť knižnicu *GPIO Zero*[18], ktorá mapuje mnohé akčné členy a senzory na objekty v jazyku *Python*.

Tento prístup sa však líši oproti tomu, ktorý je známy z programovania mikrokontroléra *Arduino*. Miesto polling-u alebo systému prerušení je veľa vecí skrytých. Pokročilí používatelia majú samozrejme stále možnosť siahnuť po knižniciach, ktoré poskytujú nízkoúrovňovejší prístup práce s pinmi ako poskytuje knižnica *GPIO Zero*.

17 https://www.stuffaboutcode.com/

18 https://gpiozero.readthedocs.io/en/stable/



***Obrázok 4: Označenie GPIO pinov na minipočítači Raspberry Pi* [9]**

5.4 Tvorba dynamických webových stránok

Niektorí učitelia z oboch skupín sa zúčastňujú aj ďalších školení v rámci projektu *IT Akadémia*. Jedným z ponúkaných kurzov je aj kurz tvorby dynamických webových stránok. Ako jazyk je v tomto kurze zvolené *PHP*. Na tom by nemuselo byť nič zlé, ale spôsob výučby sa veľmi podobá tomu, ako boli dynamické weby vytvárané okolo roku 2000, keď ešte neexistovali voľné softvérové rámce na zjednodušenie tvorby webov a aj ponuka voľne dostupných redakčných systémov bola nízka. Príznakom tvorby stránok v jazyku *PHP* v tomto období bolo totiž neaplikovanie návrhového vzoru *MVC* (z angl. *Model View Controller*), resp. *MTV* (z angl. *Model Template View*). Výsledkom tohto prístupu bolo často kombinovanie *HTML* kódu s *PHP* kódom v jednom súbore.

V rámci kurzu *Internet vecí* sme sa tejto téme okrajovo venovali práve v časti venujúcej sa prepojeniu vecí s internetom. Vytvorili sme jednoduché webové rozhranie, ktoré bolo dynamicky generované v jazyku *Python* pomocou micro webového rámca *Flask*[19]. Pre mnohých z učiteľov bola nová samotná informácia o možnosti vytvárania dynamických webových stránok v tomto jazyku. Rovnako tak aj jednoduchosť tvorby. Micro webový rámec *Flask* je taktiež súčasťou inštalácie distribúcie *Raspbian*.

5.5 Databázové systémy

Na niektorých školách sa učia aj základy jazyka *SQL*. Na výučbu základov častokrát nie je potrebná najnovšia verzia veľkého databázového systému, ale je možné si vystačiť napríklad s *SQLite*[20]. Jedná sa o relačnú súborovú databázu, kedy nie je potrebné inštalovať a spúšťať serverové riešenie, ale stačí len ovládač v niektorom programovacom jazyku (napr. *Python*), ktorý vie vykonávať dopyty nad súborom s údajmi.

Podpora pre *SQLite* sa v jazyku *Python* nachádza už v štandardnej knižnici. Tým pádom je možné prácu s *SQLite* databázou integrovať do ľubovoľného riešenia. Alebo je možné doinštalovať knižnicu *SQL Alchemy*[21], ktorá poskytuje možnosť mapovať relácie do objektov, tzv. *ORM* (z angl. *Object Relation Mapping*).

5.6 Ďalšie nástroje

Samozrejme obsah distribúcie *Raspbian* nie je vybavený len nástrojmi určenými pre prácu s jazykom *Python*. V základnej inštalácii sa nachádza aj jazyk *Java* spolu s nástrojmi *BlueJ*[22] a

19 http://flask.pocoo.org/

20 https://www.sqlite.org/index.html

21 https://www.sqlalchemy.org/

22 https://www.bluej.org/

Greenfoot[23], ktoré aplikujú metodiku *začnite s objektmi* (z angl. *Objects-first Approach*).

Okrem toho je možné medzi softvérom nájsť *Scratch* na grafické (blokové) programovanie vhodné najmä pre žiakov základných škôl, poprípade začiatočníkov, alebo *Sonic Pi* pre výučbu programovania pomocou tvorby hudby. V základnej výbave sa nachádza aj nástroj *Node-RED*, ktorý je čiastočným zástupcom grafického programovania a umožňuje rýchlo prototypovať riešenia nielen pre oblasť *IoT*.

6 SCENÁRE POUŽITIA

Implementovanie minipočítača Raspberry Pi do výučby na stredných školách prináša so sebou niekoľko scenárov použitia. Dokonca, niektoré z nich sme mohli vidieť či už počas priebehu školenia na učiteľoch, alebo na študentoch po jeho skončení.

Prvý scenár použitia by bolo možné charakterizovať ako *„nemám vlastné Raspberry Pi“*. S týmto scenárom sme vlastne začínali samotné školenie, kedy smeli jeho účastníci používať *Rasbperry Pi* len v škole ako pracovnú pomôcku. Tým pádom sa však nemohli jednotlivým témam venovať aj doma a celé experimentovanie sa muselo diať výlučne počas kurzu.

Druhý scenár by bolo možné charakterizovať ako *„mám vlastné Raspberry Pi a nosím si ho so sebou“*. Tento scenár bolo možné sledovať hlavne na študentoch po skončení kurzu. Skupina, ktorá si začala nosiť vlastné zariadenia sa pomaly zväčšovala. V princípe každý, kto mal vlastné *Raspberry Pi*, ho nenechal doma, ale si ho vždy na spoločné stretnutie doniesol so sebou. Tomu určite napomáha samotná cena zariadenia, ktorá nie je veľmi vysoká, ako aj dostupnosť na trhu. Študenti si teda so sebou priniesli svoj vlastný počítač a v učebni použili len monitor, klávesnicu a myš, poprípade napájanie.

Tretí scenár je modifikáciou predchádzajúceho a bolo by možné ho charakterizovať ako *„mám vlastné Raspberry Pi, ale nenosím si ho so sebou“*. Tento prístup je bežný napr. v prípade použitia domácich stolových počítačov, ktoré sú príliš veľké na to, aby si ich študenti nosili so sebou do školy, aj keď ho majú aj doma. V našom prípade však nedošlo k aplikovaniu tohto scenára, pretože študenti podobne ako pri práci s prenosnými počítačmi preferujú prácu na vlastnom zariadení, pokiaľ ho majú. Čakali sme, či niektorý zo študentov si nebude nosiť len SD kartu s vlastným systémom, ktorú potom len zasunie v učebni do poskytnutého zariadenia. Takýto prístup je totiž tiež možný, nakoľko hardvérové vybavenie každého zariadenia je identické. Nikto z nich však tento prístup neaplikoval.

7 DOPADY ŠKOLENIA INTERNET VECÍ

Dopad kurzu bolo možné sledovať na oboch skupinách – ako na učiteľoch, ktorí sa ho priamo zúčastnili, tak aj na študentoch, ktorých sme následne mali možnosť vďaka nim učiť.

7.1 Dopad kurzu na učiteľov

Dopad na učiteľoch bol viditeľný veľmi rýchlo. Už počas úvodných dvoch stretnutí, keď sme im ukazovali neštandardný a nie veľmi tradičný spôsob výučby programovania, bolo vidieť ich nadšenie pre vec. Diskutovali sme aj o možnostiach použitia knižnice *Pygame Zero* a *Minecraft*-u bez *Raspberry Pi*.

Na tématicky druhé stretnutie venované *hardvéru* a *elektronike* si niektorí jednotlivci už doniesli svoje vlastné sady pre prácu s elektronikou. Aj napriek tomu, že mnohí z nich nemali predchádzajúcu skúsenosť, priznali sa, že prvýkrát zapájali LED diódy s odpormi pomocou kontaktného poľa až v ten deň. Nijako ich to neodradilo, čomu pomohlo aj to, že každý mal možnosť pripájať elektronické súčiastky ku vlastnému kontaktnému poľu a ku vlastnému *Raspberry Pi*.

Posledné stretnutie sa zasa nieslo v duchu ovládania vytvorených *vecí* na diaľku. Najprv pomocou protokolu MQTT, potom pomocou webového prehliadača, kde sme vytvorili jednoduché webové rozhranie pre ovládanie príslušného zariadenia (tzv. *HTTP Switch*). Nakoniec sme všetky zariadenia vzájomne prepojili pomocou *Mozilla IoT Gateway*, ktorý ich umožňoval ovládať na diaľku aj pomocou mobilného telefónu. Samozrejme všetko bolo vytvorené v jazyku *Python* a potrebných knižniciach.

S učiteľmi sme zostali v kontakte aj po skončení kurzu. Niektorí z nich dokonca boli aj na akcii *Linuxový víkend v Košiciach*, kde sa aktívne zúčastnili školenia s *BBC micro:bit*-mi. Jedna učiteľka dokonca zapojila do svojich aktivít s *Raspberry Pi* aj svojho manžela.

7.2 Dopad kurzu na študentov

Podobne ako učitelia, sa správali aj študenti. Stretnutia s nimi neboli vždy úplne pravidelné, ale darí sa nám s nimi na základe možností stretávať už takmer pol roka.

Podobne ako pri učiteľoch bolo vidieť nadšenie aj u nich. Dokonca si mnohí z nich začali postupne nosiť na spoločné hodiny vlastné *Raspberry Pi*. V rámci spoločných hodinových stretnutí, ktoré sme mávali k dispozícii, sme sa venovali najmä práci a programovaniu elektronických prvkov. Postupne sa vďaka zadaniam od pani učiteľky začali venovať aj ďalším oblastiam využitia a témy na rozhovory pred alebo po stretnutí išli častokrát nad rámec toho, o čom sme sa bavili spolu.

Podobne ako učitelia, aj niektorí z týchto študentov sa zúčastnili víkendovej akcie *Linuxový víkend*. Hlavne kvôli prednáške o programovaní počítačových hier v jazyku *Python*, kde si jeden z autorov tohto článku pripravil príspevok na základe skúseností s knižnicou *Pygame Zero*. Efekt ako pre nich, tak aj pre účastníkov, kedy sme spolu za približne *45* minút naprogramovali takmer fungujúcu hru *Arkanoid*, bol zrejmý. Vyhli sa akurát možnosti zapojiť sa do súťaže v tvorbe počítačových hier s názvom *Pyweek*[24].

8 ZÁVER

S použitím minipočítača *Raspberry Pi* sa vo veľkej miere v okolí stretávame hlavne u nadšencov, ktorí s jeho pomocou realizujú svoje vlastné projekty. Častokrát sa jedná o jeho využitie pre rozličné riešenia inteligentnej domácnosti alebo zábavných (multimediálnych) centier.

Na Slovensku však jeho použitie v školách ako bežnej pomôcky v procese výučby je skôr výnimočné. O popularizáciu sme sa začali snažiť už na podujatí *Namakaný deň 2018*[25], kedy sme väčšinu sprievodných pracovných dielní (workshop-ov) realizovali práve pomocou *Raspberry Pi*.

Ten pravý efekt sa však prejavil až pri príprave a realizovaní školenia *Internet vecí* z projektu *IT Akadémie*. Bolo naozaj vidieť, že pokiaľ sú učitelia správne motivovaní a podmienky v domácej škole im to umožňujú, dokážu zrealizovať a aplikovať

23 `https://www.greenfoot.org/door`

24 `https://pyweek.org/`

25 `http://www.namakanyden.sk/2019/`

prezentované veci už veľmi skoro. Napomáha tomu aj nákupná cena zariadení a aj ďalšej výbavy, ktorá je pomerne nízka, keďže cena *Raspberry Pi* v porovnaní so stolovým počítačom je zhruba 8x nižšia.

Použitie *Raspberry Pi* so sebou prináša aj ďalší efekt, ktorým je samoštúdium učiteľa, ktoré je v mnohých prípadoch bádateľské – napr. pri príprave metodík s elektronikou, kedy si učiteľ potrebuje najprv naštudovať a vyskúšať, ako veci pracujú a až potom ich prezentovať pred študentmi a následne spolu s nimi. Vedie však k tomu, aby sa sami učitelia zlepšovali a vytvárali pre svojich študentov zaujímavé a zábavné, ale v prvom rade náučné úlohy.

Na študentoch sme dokonca videli ešte jeden efekt, ktorý sme neplánovali a dopredu ani nepredpokladali – došlo k zvýšeniu záujmu o hodiny tohto typu aj u dievčat. To bolo možné vidieť najmä na ich aktívnom zapájaní sa do činností počas stretnutí. Dokonca si v rámci záverečných projektov niektoré dievčatá vybrali aj nie úplne triviálne problémy, ako napr. svetelná závora (brána) na bežeckých pretekoch pomocou prototypovacej dosky *Arduino* alebo počítačová hra pomocou knižnice *Pygame Zero.* Podobný efekt bolo možné vidieť napríklad po zavedení *BBC micro:bit*-ov do výučby vo *Veľkej Británii,* kde sa záujem o štúdium IKT odborov zvýšil u dievčat o *70%* [7]. Aj keď ide o inú technológiu v porovnaní s minipočítačom *Raspberry Pi*, atraktivita takéhoto typu výučby pre študentov je zrejmá. Aj tento fakt nás uistil, že zavedenie *Raspberry Pi* do výučby informatických predmetov na stredných školách je rozhodne dobrý nápad.

PODAKOVANIE

Príspevok bol spracovaný v rámci projektu *OP Ľudské zdroje "IT Akadémia –vzdelávanie pre 21. storočie"* (ITMS: 312011F057).

BIBLIOGRAFICKÉ ODKAZY

[1] KUČERA, Peter. *Programujeme v Pythone: učebnica informatiky pre stredné školy*. 2016. ISBN 978-80-972320-4-7.

[2] NUTALL, Ben. *Raspberry Pi in Estonia project launch*. [online] [cit. 2019-01-29] https://www.raspberrypi.org/blog/raspberry-pi-in-estonia-project-launch/

[3] SZYMOR, Nina. *Celbrating our translators!* [online] [cit. 2019-01-29] https://www.raspberrypi.org/blog/celebrating-our-translators/

[4] MCMANUS, Sean. *Pygame Zero: Game coding made easy* [online]. 2015, 4 (35) [cit. 2019-01-31]. ISSN 2051-9990. Dostupné z: https://www.raspberrypi.org/magpi/issues/35/

[5] MCMANUS, Sean. *Mission Python: code a space adventure game*. San Francisco: No Starch Press, [2018]. ISBN 978-159-3278-571.

[6] O'HANLON, Martin a WHALE, David. *Adventures in Minecraft*. Second edition. Indianapolis, IN: Wiley, [2018]. ISBN 978-1119439585.

[7] *Microbit girls impact*. [online] [cit. 2019-01-29] https://microbit.org/en/2017-07-31-microbit-girls-impact/

[8] *Setting up your Raspberry Pi* [online] [cit. 2019-03-17] https://projects.raspberrypi.org/en/projects/raspberry-pi-setting-up

[9] *Raspberry Pi GPIO Pinout* [online] [cit. 2019-03-17] https://pinout.xyz/

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN ..... ISSN .....

Python pre stredné školy
Python for high schools

RNDr. Andrej Blaho, PhD.
KAI FMFI UK
Mlynská dolina
842 48 Bratislava
Slovensko
andrej.blaho@fmph.uniba.sk

ABSTRACT
Language Python is one of most popular programming languages among professionals or professional higher education courses. This language starts to appear also in high education courses.

This contribution describes text and activity book which was developed in 2018 and offers introduction to programming in Python for beginners in high school.

Keywords
programming, high (secondary upper) school, text and activity book

ABSTRAKT
Jazyk Python je jedným z najpopulárnejších programovacích jazykov medzi profesionálmi, ale tiež v profesionálnych vysokoškolských kurzoch a začína sa objavovať aj na stredných školách.

Tento príspevok opisuje materiál, ktorý vznikol v roku 2018 a ponúka úvod do programovania v Pythone najmä pre úplných začiatočníkov na strednej škole.

Kľúčové slová
programovanie, stredná škola, učebnica

ÚVOD
Jazyk Python je jedným z najpopulárnejších programovacích jazykov nielen medzi profesionálmi, ale čím ďalej, tým sa častejšie objavuje nielen v profesionálnych vysokoškolských kurzoch, ale úspešne nahrádza iné jazyky na stredných školách, ale niekde aj vo vyšších ročníkoch základných škôl. Učitelia na stredných školách sú väčšinou odkázaní na vysokoškolské kurzy a preto je veľmi cenné, keď sa objavujú aj materiály určené priamo pre stredoškolskú informatiku. V príspevku opisujeme vznik materiálu, ktorý vznikol v roku 2018 a ponúka úvod do programovania v Pythone najmä pre úplných začiatočníkov na strednej škole.

PYTHON NA MATFYZE

Jazyku Python vo vyučovaní sa venujeme na FMFI UK už piaty rok, pričom na webových stránkach je každoročne publikovaný aktuálne bežiaci kurz. Ešte do roku 2013 bol prvým jazykom budúcich informatikov Pascal (v priebehu vyše 25 rokov to boli verzie TurboPascal, Delphi, Lazarus), ale od 2013 sme úspešne prešli na jazyk Python.

Hlavnými dôvodmi opustenia Pascalu boli:

- každoročne nižšia a nižšia popularita Pascalu na vysokých školách,
- ťažkopádne vývojové prostredie,
- chýbajúce moderné dátové štruktúry.

Napriek tomu, že Pascal je stále jeden z didakticky najvhodnejších programovacích jazykov, hľadali sme niečo aktuálnejšie. Ukázalo sa, že práve Python je výborná voľba, hlavne z týchto dôvodov:

- už niekoľko rokov sa stáva najčastejším programovacím jazykom vo vysokoškolských kurzoch na celom svete,
- pre jeho rýchle prototypovanie je veľmi populárnym nielen vo veľkých IT firmách ale aj v malých, prípadne v startupoch,
- obrovskú popularitu si získava napr. aj v dátových vedách (spracovanie veľkých dát, strojové učenie), ale pracujú s nim aj vedci mimo IT a to najmä vďaka obrovskej škále kvalitných knižníc,
- existujú stovky (rôzne) kvalitných učebníc a materiálov, z ktorých veľká časť je voľne prístupná,
- nielen pre začiatočníkov sú veľmi užitočné komunitné portály, na ktorých sa riešia nielen jednoduché problémy, ale aj náročné programátorské témy.

Asi najdôležitejším dôvodom bola jeho didaktická vhodnosť – v súčasnosti sa Python považuje za didakticky vhodný zo súčasných najlepších programovacích jazykov hlavne pre jeho jednoduchosť zápisov a čitateľnosť výsledného kódu.

Úvodný kurz programovania

Zhrňme hlavné charakteristiky úvodného kurzu programovania na našej fakulte:

- kurz je určený najmä budúcim profesionálnym programátorom (študenti aplikovanej informatiky),
- väčšina študentov, ktorí prichádzajú do tohto kurzu, absolvovala nejaké stredoškolské programovanie (najčastejšie v Pascale) a dokonca z informatiky maturovali,
- vďaka tomu sú hlavne úvodné časti, ktoré sa prekrývajú so stredoškolskou úrovňou, preberané vo zvýšenom tempe, hoci aj úplný začiatočník s minimálnymi programátorskými skúsenosťami má šance takéto tempo zvládať,
- kľúčové programátorské koncepty sú v priebehu štúdia rozložené tak, aby študenti mali čo najviac priestoru sa s nimi nielen zoznámiť, ale hlavne s nimi preriešiť čo najviac úloh od jednoduchých až po tie náročnejšie,
    - každej náročnejšej téme sa venujeme na samostatnej prednáške, prípadne je rozložená do viacerých prednášok,
- osvedčilo sa nám veľmi rýchlo už od prvých prednášok zaviesť prácu s grafikou a na tomto ďalej ilustrovať a precvičovať aj netriviálne programátorské témy,

    - sme presvedčení, že táto grafická vizualizácia je pre študentov veľmi dôležitá a sme sa ešte nestretli s podobne realizovanými kurzami vysokoškolského programovania vo svete, je to inovatívny a originálny metodický príspevok autora článku,
- hoci je Python objektový jazyk, študenti sa zoznámia so skutočným OOP až v druhej polovici prvého semestra (v 15. prednáške), dovtedy sa naučia riešiť základné konštrukcie imperatívnym programovaním,
    - hneď po oboznámení sa s OOP sa od študentov vyžaduje riešiť väčšinu problémov prísne objektovo,
- celkovo je kurz pripravený tak, aby sa študenti zoznamovali s novými náročnejšími konceptami až vtedy, keď sú na to pripravení a majú za sebou dostatočné skúsenosti s predchádzajúcimi konštrukciami, na ktorých sa budujú nové poznatky.

Každý študent má na cvičeniach počas úvodného kurzu šancu riešiť skoro 350 menších úloh, vďaka čomu získava neoceniteľné prvé skúsenosti, ktoré sú nevyhnutným základom chápania programátorských konceptov. Okrem toho rieši okolo 10 väčších projektov, ktoré sú na úrovni záverečnej skúšky kurzu. Náš vysokoškolský kurz je publikovaný na stránkach [1] a [2], je voľne prístupný s otvorenou licenciou Creative Commons (CC BY-SA 4.0).

PYTHON NA STREDNÝCH ŠKOLÁCH

Python získava popularitu nielen vo firmách a na vysokých školách, ale veľmi úspešne nahrádza staršie programovacie jazyky aj na stredných a čiastočne aj na základných školách. Zatiaľ sú v tomto najúspešnejšie najmä stredné školy, ktoré sú zamerané buď na informatiku alebo matematiku, alebo pracujú s nadanými žiakmi. Aj bežné stredné školy sledujú takéto novinky a preto by sme im mali pomôcť v ich orientácii a naštartovaní zmeny.

Keďže vhodných kurzov programovania v Pythone pre stredné školy je vo svete dosť málo, často sa učitelia inšpirujú vysokoškolskými učebnicami, kurzami, webovými portálmi a pod. Lenže stredoškolské programovanie by malo mať svoje špecifiká.

V prvom rade by to mali byť iné ciele a iné metódy. Treba vychádzať z toho, že vysokoškolské kurzy sú zamerané na vybranú vzorku záujemcov, ktorí sa rozhodli študovať nejaký väčšinou technický smer. Na rozdiel od toho je informatika na stredných školách určená všetkým žiakom, teda aj tým, ktorí nebudú pokračovať v takto zameranom štúdiu.

Cieľom programovania na vysokých školách je pripraviť budúceho profesionála do IT praxe, prípadne mu nastaviť jeho odbornú úroveň tak, aby zvládal ďalšie informatické štúdium. Pritom na stredných školách je cieľom to, čo je špecifikované v štandardoch v štátnom vzdelávacom programe.

Okrem toho aj štýl vyučovania na vysokých školách je diametrálne rozdielny od prístupu na stredných školách.

Teda zhrňme:

- žiak na SŠ si má vybudovať informatické / logické myslenie, pochopiť jeho princípy (koncepty),
- podobne ako v školskej matematike by mali byť najcennejšie práve schopnosti riešiť problémy, pričom na to potrebujeme veľmi dobré a dlhodobé objavovanie základných konceptov,
- cieľom teda nie je ani syntax, dokonca ani konkrétny programovací jazyk, ale chápanie základných princípov – toto sa nedá naučiť (ani pochopiť) učením sa definícií, ale len osobnou skúsenosťou fungovania tohto nového myšlienkového sveta,
- žiaci by mali mať dostatočný časový priestor na získavanie vlastných skúseností s fungovaním všetkých stavebných prvkov a najmä dostatok času na objavovanie čo najväčšej množiny chybových situácií,
- keďže nám ide hlavne o informatické myslenie, učiteľ by mal dať žiakom šancu objavovať princípy a minimalizovať to, čo im prezradí výkladom.

Veľkým problémom mnohých predmetov nielen na stredných školách je snaha žiakom „upratať“ informácie o tom, ako tá ktorá oblasť funguje. Napr. pomocou vzorcov upratať fyzikálne správanie sveta, pomocou chemických zápisov upratať fungovanie chemickej podstaty, upratať zákonitosti slovenského jazyka a pod. Predpokladáme, že žiaci majú dlhoročné vlastné skúsenosti s fungovaním sveta, s jazykovým prejavom, preto cítime potrebu im toto kvalitne upratať.

O čo horšie je to v programovaní – je to úplne inak fungujúci svet, v ktorom sa všetko popisuje nejakými základnými stavebnými kameňmi (typy a konštrukcie) a zatiaľ s týmto svetom žiak nemá skoro žiadne skúsenosti. Žiaľ, veľakrát sa učiteľ snaží tieto skúsenosti upratať skôr, ako sa tohto sveta žiaci čo i len dotknú.

Programovanie na strednej škole

Takže programovanie na strednej (zrejme aj na základnej) škole by malo byť hlavne o

- osobnom objavovaní základných koncepcií (dát a spôsoboch manipulácií)
    - namiesto definícií konštrukcií objavovať správanie vlastnými skúsenosťami,
- veľmi dobre zvolenej postupnosti konceptov, aby bol naozaj časový priestor sa s nimi zžiť
    - dôležité koncepty roztiahnuť na viac hodín, nechať dostatok času na samostatné riešenie všetkých prípadov,
- osobných skúsenostiach so správaním sa počítača v prípade chybových situácií
    - pochopiť, že robiť chyby v programovaní nie je zlyhaním, ale bežnou súčasťou pri riešení problémov,
- vhodne zvolených úlohách, na ktorých sa ilustruje riešenie programátorských úloh
    - pri témach úloh používať len vhodný aparát, s ktorým nemajú žiaci problémy (napr. matematika zo ZŠ),
    - nesnažiť sa používať náročnejšiu matematiku, ktorú mnohí žiaci ešte nemajú tak vžitú, aby ju využili na pochopenie informatických konceptov.
- vyvarovať sa na danej úrovni nevhodným konceptom, ktoré vysoko presahujú bežnú stredoškolskú úroveň informatiky,

- niektoré témy sú vhodné až pre maturitu, iné až na vysokej škole, pre bežných žiakov sú zbytočne náročné, napr. zoznamy, globálne premenné, funkcie ako parametre.

Vychádzame z toho, že vysokoškolský kurz Pythonu na matfyze má didakticky zaujímavé princípy objavovania programovania pomocou úloh s grafickým výstupom. Zvolili sme si prostredie tkinter, toto prostredie je súčasťou štandardnej inštalácie a pritom je pre potreby vyučovania úvodného programovania dostatočné.

Návrh materiálu pre úvod do Pythonu na stredných školách

Z materiálu ŠVP (Štátny vzdelávací program) pre gymnázia [3] môžeme vybrať tieto **kľúčové koncepty**:

- príkaz, parameter príkazu, postupnosť príkazov, program,
- premenná, priradenie, výraz,
- vstup a výstup,
- opakovanie pomocou cyklu,
- vetvenie na základe podmienky,
- interpretovanie, krokovanie, hľadanie chýb.

Niektoré z týchto kľúčových konceptov v sebe obsahujú veľké množstvo ďalších menších konceptov, niekedy hovoríme **mikrokonceptov**. Dobrý učebný materiál by na tieto mikrokoncepty mal myslieť, lebo len ich zvládnutím má zmysel hovoriť o zvládnutí vyššieho konceptu. Napr. aj koncept premennej a priradenia v sebe obsahuje napr. tieto mikrokoncepty:

- predstava premennej na uchovávanie nejakej hodnoty,
- mechanizmus priradenia,
- zápis a vyhodnocovanie výrazu,
- priradenie hodnoty výrazu s viacerými premennými,
- sekvencia priradení,
- priradenie do premennej, ktorá už má nejakú hodnotu,
- priradenie do premennej, keď výraz obsahuje túto premennú,
- krokovanie,

Je otázne, či všetky tieto mikrokoncepty naučíme naraz, alebo niečo z toho sa objaví až vtedy, keď na to príde čas. Zrejme niektorý učiteľ sa rozhodne týmto mikrokonceptom venovať len pár minút (na všetko sa dá predsa prísť so sedliackym rozumom), iný tomu venuje niekoľko vyučovacích hodín.

Presne takýto pohľad sme mali pri návrhu vlastného materiálu pre úvod do programovania: dať šancu aj učiteľovi, ktorý berie vážne dôsledné precvičenie všetkých mikrokonceptov.

Náš materiál sme postavili na kreslení do grafickej plochy a popritom sa žiaci veľmi pomaly zoznamujú s najdôležitejšími konceptami. Rozdelili sme ho do niekoľkých kapitol, pričom predpokladáme, že učiteľ si zvolí vlastné tempo, pri ktorom jednu kapitolu môže preberať buď jednu vyučovaciu hodinu alebo dve aj tri. Závisí to od množiny úloh, ktoré riešia žiaci a od ich úspešnosti.

Momentálna verzia materiálu [4] má tieto kapitoly:

1. Zoznámme sa s jazykom Python.
2. Python ako kalkulačka, premenné, priradenie.
3. Prvý program, výpisy, výpočty.
4. Grafická plocha, obdĺžniky.
5. Farby pri kreslení obdĺžnikov.
6. Použitie premenných a výrazov pri kreslení.
7. Vytvárame podprogramy.
8. Náhoda.
9. Text v grafickej ploche.
10. Fonty a farby pre texty v grafike.
11. Program s opakovaním.
12. Riešenie úloh s for-cyklom.
13. Kreslenie elíps a kruhov.
14. Kruhy a cykly.
15. Parameter podprogramu.
16. Parametre podprogramov.
17. Znakové reťazce.
18. Desatinné čísla.

ZÁVER

Materiál [4] je voľne prístupný na internete a v súčasnosti sa dopĺňa a prispôsobuje podľa pripomienok učiteľov. Najvýraznejšie využitie a potvrdenie prínosu tejto verzie učebnice je pri tvorbe iného výučbového materiálu (pracovné listy a metodika) v bežiacom národnom projekte „Informatické myšlení“ v Českej republike. V rámci tohto projektu vznikli špecifické pracovné listy pre žiaka aj metodika pre učiteľa (viac na stránkach imysleni.cz). V súčasnosti sa tento materiál overuje na 9 stredných školách v Čechách a veríme, že po testovaní a dopracovaní bude prístupný aj pre slovenských učiteľov a žiakov.

BIBLIOGRAFICKÉ ODKAZY

[1] BLAHO, A. *Programovanie v Pythone* [online]. *Bratislava 2019:* http://python.input.sk

[2] BLAHO, A. Algoritmy a dátové štruktúry [online]. *Bratislava 2019:* http://struct.input.sk

[3] *Inovovaný ŠVP pre gymnáziá s osemročným vzdelávacím programom* [online]. http://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/inovovany-svp-gymnazia-osemrocnym-vzdelavacim-programom/matematika-praca-informaciami/

[4] BLAHO, A. *Učebnica Pythonu pre stredné školy* [online]. *Bratislava 2018:* http://input.sk/ucebnica

[5] *Informatické myšlení* [online]. *České Budějovice 2019: https://imysleni.cz/ucebnice/zaklady-programovani-v-jazyce-python-pro-stredni-skoly*

Ako žiaci piateho a šiesteho ročníka základnej školy riešili vybrané grafové úlohy

How Pupils in Fifth and Sixth Grade Were Solving Selected Graph Tasks

Lucia Budinská
KDMFI, Fakulta matematiky, fyziky a informatiky UK
Mlynská dolina 1
842 48 Bratislava
Slovensko
Lucia.Budinska@fmph.uniba.sk

Karolína Mayerová
KDMFI, Fakulta matematiky, fyziky a informatiky UK
Mlynská dolina 1
842 48 Bratislava
Slovensko
mayerova@fmph.uniba.sk

ABSTRACT
Innovated State Educational Programme for informatics states that pupils in lower secondary school should learn how to work with graph data structure. Graph task categorisation was created in the previous research based on graph tasks from Slovak Bebras contest, with focus on methods used in solving process. In this paper, there are shown main results from mixed methods research where we analyse three tasks from Bebras contest in which girls were more successful than boys. The tasks are firstly described together with results from the contest. The main aim of this paper is qualitative phase of research in which we let fifth- and sixth-graders solve these tasks. We analyse group differences (grade and gender) in their answers, group discussions about tasks and results of second test. Based on our results some changes to the next phase of research are proposed.

Keywords
Graph tasks. Bebras. Grah data structure. Gender differences.

ABSTRAKT
Inovovaný štátny vzdelávací program pre predmet informatika určuje, že žiaci by sa mali v škole učiť pracovať aj s grafovými štruktúrami. Pri predchádzajúcich výskumoch zo slovenskej súťaže iBobor vznikla kategorizácia grafových úloh, v ktorej sa autorky zamerali aj na metódy riešenia týchto úloh. Tento článok sa venuje prvotným výsledkom zmiešaného výskumu, v ktorom sme analyzovali tri úlohy, ktoré v súťaži riešili lepšie dievčatá ako chlapci. Bližšie tieto úlohy opisujeme a sumarizujeme výsledky súťažiacich z ich zaradenia do súťaže. Následne sa bližšie venujeme kvalitatívnej fáze výskumu, ktorá prebehla so žiakmi piateho a šiesteho ročníka, pričom sme sa zamerali na rozdiely medzi medzi jednotlivými skupinami (podľa ročníka a pohlavia) pri riešení, diskusiách a následnom testovaní. Na základe výsledkov navrhujeme zmeny do ďalšej fázy výskumu.

Kľúčové slová
Grafové úlohy. IBobor. Grafové štruktúry. Rozdiely medzi chlapcami a dievčatami.

1 ÚVOD
Práca s grafovými štruktúrami sa vyskytuje ako tematická oblasť v inovovanom Štátnom vzdelávacom programe [1], no často sa stáva, že učitelia nevedia, čo si pod týmto slovným spojením predstaviť a ako včleniť túto tému do vyučovania. Informatická súťaž iBobor [2] ponúka veľké množstvo úloh, mnohé z nich sa venujú práve práci s grafovými štruktúrami. Počas našich predchádzajúcich výskumov vznikla kategorizácia grafových úloh zo súťaže [3], v ktorej sme vychádzali zo zadaní úloh a obrázkov v nich použitých. Analýza súťažných výsledkov kategorizovaných úloh ukázala, že v niektorých typoch úloh sú úspešnejší chlapci, v iných zas dievčatá. Zaujímalo nás, prečo takéto rozdiely vznikajú, a taktiež, či žiaci naozaj riešia grafové úlohy tak, ako výskumníci pri kategorizácii predpokladali.

Tento článok sa preto venuje prvým výsledkom kvalitatívnej štúdie, pri ktorej sme sa zamerali na rozdiely chlapcov a dievčat pri riešení úloh, v ktorých je potrebné porozumieť informáciám zaznamenaným v grafovej štruktúre.

2 METODOLÓGIA
Cieľom nášho dlhodobého výskumu je lepšie porozumieť, ako jednotlivé skupiny žiakov riešia grafové úlohy zo súťaže iBobor, aké sú rozdiely medzi žiakmi a ako žiaci pri riešení týchto úloh postupujú. V tomto článku sme sa zamerali na porovnanie spôsobov, akými chlapci a dievčatá v piatom a šiestom ročníku základnej školy pristupujú k riešeniu vybraných grafových úloh. Úlohy sme vybrali na základe predchádzajúcich etáp výskumu [3] a výsledky sledovanej skupiny žiakov porovnávame s výsledkami súťažiacich zo súťaže iBobor.

Pre každú z úloh zaradenú do súťaže máme k dispozícii dáta – ako žiaci riešili danú úlohu a základné údaje o nich (pohlavie, ročník). Tieto dáta slúžili ako prvotné informácie pri vyváraní predstavy o úrovni vedomostí žiakov a ich najčastejších nesprávnych odpovedí. Ďalej nás zaujímalo, kde konkrétne v úlohách nastávajú problémy, v čom sa líši spôsob riešenia chlapcov a dievčat a ako nad úlohami premýšľajú. Na toto skúmanie pokladáme za najvhodnejší kvalitatívny prístup, preto sme zvolili zmiešané metódy výskumu a to konkrétne *vysvetľujúci sekvenčný dizajn* [4].

Prvá, kvantitatívna, fáza výskumu spočívala v štatistickej analýze dát zo súťaže. Okrem jednoduchej opisnej štatistiky sme použili aj Pearsonov Chi-kvadrátový test nezávislosti [5]. Úspešnosť v úlohe sme rozdelili do troch kategórií – správna odpoveď, nesprávna odpoveď, žiadna odpoveď, a pre každú kategóriu sme zisťovali, či je úspešnosť úlohy závislá od pohlavia alebo nie. T. j. či jedna skupina žiakov riešila úlohu štatisticky významne lepšie ako druhá.

Úlohy sme analyzovali aj podľa kategorizácie [3], ktorú stručne popíšeme v nasledujúcej kapitole. Z prvej fázy sme tak dostali prehľad o tom, ako boli úlohy úspešné a zároveň aj ako by ich žiaci

mali riešiť, na základe navrhnutých kategórií. Vznikla tak skupina tvrdení, ktoré sme sa rozhodli skúmať so žiakmi pomocou ohniskového neriadeného rozhovoru [6]. Žiaci sa v skupinách rozprávali o riešení úloh, pričom rozhovory neboli priamo riadené učiteľom, ale skôr pracovným listom a samotnými zadaniami úloh v nich. Rozhovory sme nahrávali na diktafón, následne prepisovali a okódovali. Pred skupinovou prácou a aj po nej ešte žiaci sami riešili vybrané úlohy vo forme pracovných listov. Tie sme zozbierali a následne kódovali a analyzovali. Triangináciu dát zabezpečili terénne poznámky učiteľa z hodiny.

3 GRAFOVÉ ÚLOHY

Pod označením *grafová úloha* rozumieme úlohu, v ktorej sa nachádza graf ako dátová štruktúra obsahujúca informácie, respektíve graf skladajúci sa z vrcholov a hrán ako v teórii grafov. Vrcholy aj hrany môžu byť ohodnotené, a hrany môžu byť orientované. Kategorizácia grafových úloh, ktorú sme bližšie opísali v článku [3] uvádza 5 spôsobov riešenia grafových úloh v súťaži iBobor, ktoré mali aj podľa následného výskumu na úlohách zo stredoškolských kategórii [7, 8] vplyv na úspešnosť jedného z pohlaví. Sú to:

- vytváranie stratégie,
- objavovanie stratégie,
- prehľadávanie grafu s obmedzeniami (podmienkami),
- metóda „pozriem – vidím“,
- vyskúšanie všetkých možností.

Prvé dve metódy sa vyskytujú v úlohách, v ktorých je prítomný aj nejaký algoritmus, ktorý musia súťažiaci objaviť alebo vytvoriť, aby úlohu vyriešili. Zvyšné tri metódy sa naopak vyskytujú najmä v úlohách, kde je najdôležitejšie porozumieť usporiadaniu informácii v grafe, dopĺňanie informácii do štruktúry, či extrahovanie informácií zo štruktúry.

Podľa predchádzajúcich výsledkov pre základné školy [3], chlapci boli úspešnejší pri vytváraní vlastnej stratégie a pri prehľadávaní grafu s obmedzeniami alebo viacnásobnými podmienkami. Obrázky v týchto úlohách boli pomerne abstraktné ale graficky jednoduché (štvorčeková mriežka alebo neorientovaný graf), a úlohy obsahovali dlhšie textové zadanie. Dievčatá boli na druhej strane úspešnejšie v úlohách s komplexnými grafovými štruktúrami, v ktorých boli pomenované alebo špecificky označené vrcholy a/alebo hrany a obrázky boli preto komplikovanejšie. Pri riešení týchto úloh bolo potrebné vykonať jednoduchú operáciu – čítanie z grafu, či vyskúšanie malého množstva možných odpovedí, t. j. hlavne posledné dva spôsoby spomenuté vyššie.

So zvyšujúcim sa vekom sa zvyšoval počet úloh, v ktorých boli úspešnejší chlapci, čo súvisí aj s väčším počtom úloh, ktoré vyžadovali vytváranie či objavovanie stratégie [8]. Predtým dievčenské úlohy sa v stredoškolských kategóriách stávali neutrálnymi – t. j. v úspešnosti chlapcov a dievčat nebol významný rozdiel.

3.1 Vybrané grafové úlohy

Pre náš pilotný kvalitatívny výskum sme zo všetkých analyzovaných úloh vybrali tri, pričom sme dbali na to, aby každá reprezentovala nejakú z operácii spomínanú aj v iŠVP a bola vhodná pre nižšie ročníky druhého stupňa základnej školy. Rozhodli sme sa najprv sa zamerať na jednoduchšie úlohy, takže sme vybrali úlohy bez nutnosti objavovania alebo vytvárania stratégie. Takéto úlohy by totiž mohli byť pre piatakov a šiestakov náročné.

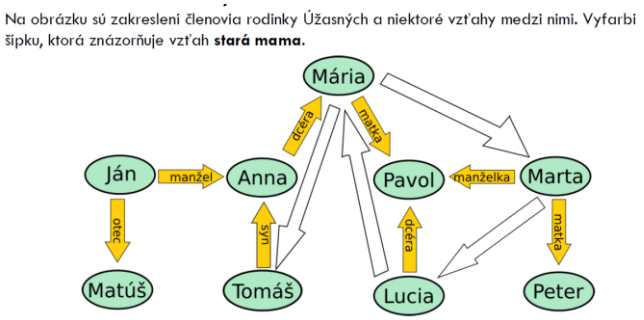
V iŠVP sa v oblasti Reprezentácie a nástroje – štruktúry [1] píše, že žiak by mal na konci 6. ročníka vedieť:

- „orientovať sa v jednoduchej štruktúre – vyhľadávať a získať informácie zo štruktúry podľa zadaných kritérií,
- organizovať informácie do štruktúr – vytvárať a manipulovať so štruktúrami, ktoré obsahujú údaje a jednoduché vzťahy (tabuľky, grafy, postupnosti obrázkov, čísel, ...),
- interpretovať údaje zo štruktúr – vyvodiť existujúce vzťahy zo zadaných údajov v štruktúre, prerozprávať informácie uložené v štruktúre vlastnými slovami.“

Zamerali sme sa práve na tieto operácie na grafoch a v ďalšej časti bližšie opíšeme vybrané úlohy spolu s ich kategorizáciou a výsledkami z ich zaradenia v súťaži iBobor.

3.1.1 Rodinný graf

Prvou z úloh, ktorú sme pre tento výskum vybrali, bola úloha Rodinný graf, ktorá bola použitá v súťaži v školskom roku 2013/14 v kategórii Benjamíni. Táto úloha bola zaradená medzi ľahké úlohy a jej úspešnosť bola 48,8 %. Ako je vidieť na Obrázku 1, v úlohe sa vyskytuje graf znázorňujúci rodinné vzťahy, ktoré by žiaci druhého stupňa základnej školy mali poznať. Pre správne riešenie je dôležitý smer šípok, ktoré označujú, kto má ku komu aký rodinný vzťah.



Obrázok 1: Úloha Rodinný graf (Rodinka Úžasných)

Správna odpoveď bola šípka smerujúca od Márie k Tomášovi. Túto odpoveď označilo 46,4 % chlapcov a 51,8 % dievčat, čo je štatisticky významná odchýlka (Pearsonov koeficient bol 7,08 v prospech dievčat), bližšie rozdiely sú uvedené v Tabuľke 1. Vidíme z nej, že rozdiely medzi pohlaviami boli významné pre všetky vekové skupiny súťažiacich, ale aj to, že so zvyšujúcim sa vekom sa chybovosť súťažiacich znižovala. Najčastejšou nesprávnou odpoveďou bolo označenie šípky od Lucie k Márii (27,5 %), potom označenie šípky od Márie k Marte (15 %), iba necelých 5 % súťažiacich označilo šípku smerujúcu od Marty k Lucii. Úlohu neriešilo 4,69 % súťažiacich.

Tabuľka 1: Výsledky úlohy Rodinný graf

Ročník	Úspešnosť chlapci	Úspešnosť dievčatá
5. ročník	40,59 %	44,60 %
6. ročník + prima	43,40 %	49,12 %
7. ročník + sekunda	54,55 %	61,50 %
dokopy	**46,38 %**	**51,75 %**

Z pohľadu kategorizácie riešenia grafových úloh spomínanej v predchádzajúcej podkapitole predpokladáme, že žiaci používajú najčastejšie metódu „pozriem – vidím“, a teda získavajú informácie z grafovej štruktúry, ktoré následne interpretujú.

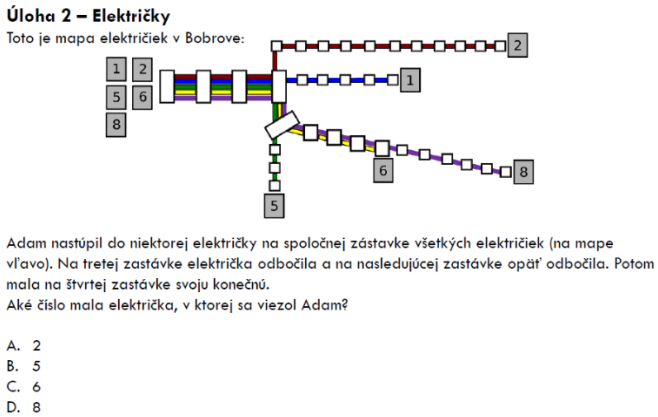
3.1.2 Električky

Druhou skúmanou úlohou v tomto výskume bola úloha z kategórie Benjamíni v školskom roku 2015/16 s názvom Električky. Obrázok v úlohe zobrazuje mapu električkových tratí a súťažiaci mali zistiť, akou električkou sa viezol chlapec, viď. Obrázok 2. Popis jeho trasy (odbočovanie a konečná zastávka) boli popísané v textovom zadaní. Dievčatá v tejto úlohe dosiahli úspešnosť 51,9 % a chlapci 49,6 %. Tento rozdiel je stredne významný (Pearsonov koeficient 3,38 v prospech dievčat), celkové výsledky sú znázornené v Tabuľke 2. Z výsledkov vidno, že kým v piatom ročníku sú rozdiely medzi chlapcami a dievčatami zanedbateľné, v siedmom ročníku sú výrazné. Taktiež s vekom výrazne vzrástla úspešnosť.

Tabuľka 2: Výsledky úlohy Električky

Ročník	Úspešnosť chlapci	Úspešnosť dievčatá
5. ročník	41,53 %	42,32 %
6. ročník + prima	49,40 %	51,32 %
7. ročník + sekunda	57,58 %	61,94 %
dokopy	**49,69 %**	**51,93 %**

Najčastejšou nesprávnou odpoveďou bola odpoveď *B* (17,95 % všetkých odpovedí), potom odpoveď *D* (14,08 %). Nesprávnu odpoveď *A* označilo 13,60 % súťažiacich a iba 3,62 % súťažiacich neodpovedalo.



Obrázok 2: Úloha Električky

Žiaci pri riešení museli používať textový opis trasy, ktorý obsahoval viacero podmienok, preto za vhodnú metódu na riešenie tejto úlohy pokladáme *prehľadávanie grafu s obmedzeniami (podmienkami)*, z pohľadu iŠVP sa jedná o získavanie informácií podľa zadaných kritérií.

3.1.3 Deň matiek

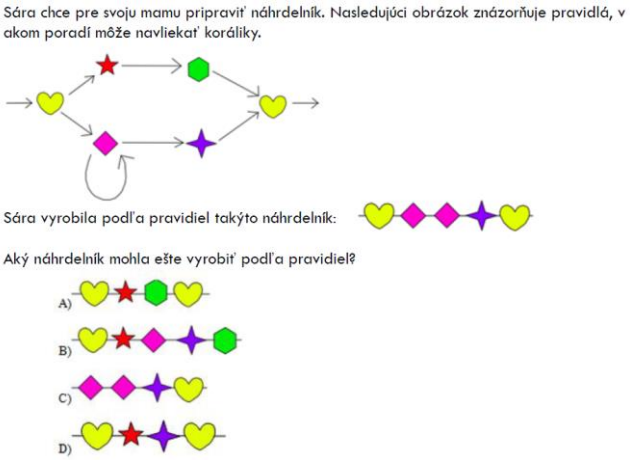
Úloha Deň matiek, ktorá bola použitá v roku 2013/14 v kategórii Kadeti, bola treťou a poslednou skúmanou úlohou v tejto časti výskumu. Úspešnosť chlapcov v tejto úlohe bola 82,0 % a úspešnosť dievčat 85,6 %. Rozdiel medzi pohlaviami bol štatisticky významný s Pearsonovým koeficientom 5,6 v prospech dievčat. Pre celkové výsledky pozri Tabuľku 4. Rozdiely medzi pohlaviami sú v oboch vekových kategóriách približne rovnako významné.

Tabuľka 3: Výsledky úlohy Deň matiek

Ročník	Úspešnosť chlapci	Úspešnosť dievčatá
8. ročník + tercia	81,69 %	85,50 %
9. ročník + kvarta	91,16 %	85,62 %
dokopy	**82,00 %**	**85,55 %**

Hoci bola úloha určená pre vyššiu vekovú kategóriu, jej veľmi dobré výsledky a malá miera chybovosti nás presvedčili, že táto úloha môže byť vhodná aj pre mladších žiakov. Úlohu v súťaži neriešilo iba 0,79 % a každú z nesprávnych odpovedí vybralo v priemere 5 % súťažiacich.

V úlohe je použitý model konečného automatu (značne zjednodušeného) a žiaci mali porozumieť pravidlám tvorby náramkov, ktoré sú znázornené v grafe, pozri Obrázok 3. Na pochopenie, ako sa náramky tvoria, slúžil ukážkový príklad. Keďže úloha obsahovala 4 možné odpovede, bolo vhodné ju riešiť vylučovaním jednotlivých možností, alebo inými slovami *vyskúšaním všetkých možností*. V tejto úlohe žiaci manipulovali s údajmi (obrázkami) v grafe a na základe pravidiel vyvodzovali výsledky (náramky).



Obrázok 3: Úloha Deň matiek

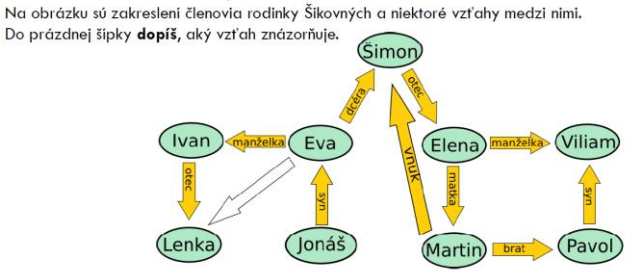
3.2 Podobné úlohy

Úlohy opísané v predchádzajúcej časti sme zadali žiakom vo forme pracovného listu. Po jeho riešení nasledovala skupinová diskusia o riešení a aby sme overili, či žiaci pochopili, ako majú úlohy riešiť, sme pre každú z úloh vytvorili jej obmenu. Ponechávali sme vizuálne rovnaké obrázky štruktúr, aj opisné informácie k úlohe, menili sme len otázku respektíve spôsob odpovede. V stručnosti ich opíšeme.

3.2.1 Rodinka Šikovných

Graf z úlohy *Rodinný graf* (v pracovnom liste nazvanej *Rodinka Úžasných*) sme upravili iba minimálne – zmenili sme mená a šípky s rodinnými vzťahmi. Na rozdiel od pôvodnej úlohy, žiaci

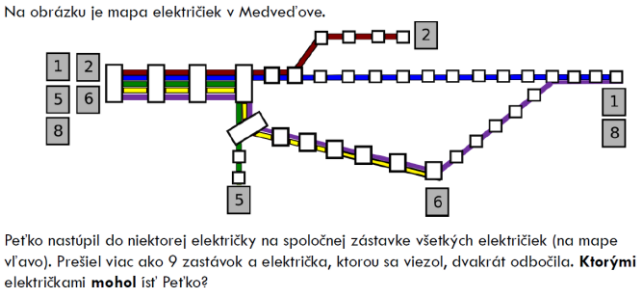
nehľadali šípku znázorňujúcu konkrétny vzťah, ale ku konkrétnej šípke zisťovali, aký vzťah znázorňuje. Zadanie úlohy sa nachádza na Obrázku 4. Zvolili sme pomerne jednoduchý vzťah *matka*, keďže našich skúseností vieme, že s pomenovaním širších rodinných vzťahov zvyknú mať mladší žiaci (a výraznejšie chlapci) problémy. Naším zámerom bolo zistiť, či žiaci porozumeli, čo znázorňuje smer šípky a dokázali správne vydedukovať odpoveď.



Obrázok 4: Úloha Rodinka Šikovných

3.2.2 *Električky v Medveďove*

Pri úprave úlohy Električky sme iba mierne zmenili graf a namiesto konkrétnych podmienok sme ponechali iba dve. Žiakom sme neponúkli možnosti, odpoveď bola otvorená, viď. Obrázok 5. Správne riešenie mohlo obsahovať dve (resp. tri) možné električkové linky, podľa toho, čo žiaci pokladali za odbočenie (zámerne sme tento pojem nevysvetľovali). Na túto informáciu sa učiteľ žiakov pýtal pri vysvetľovaní správnych odpovedí, pričom väčšina z nich vedela zdôvodniť svoju odpoveď práve pomocou toho, čo považovali za odbočenie. Zaujímalo nás tak, či sa vedia orientovať v grafovej štruktúre (znázorňujúcej mapu električkových tratí) a zároveň, ako sa vysporiadajú s nie jednoznačne definovanou podmienkou odbočenia.

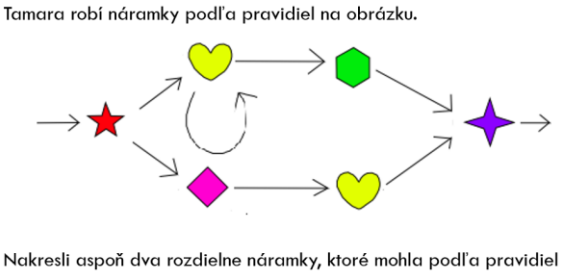


Obrázok 5: Úloha Električky v Medveďove

3.2.3 *Náramky*

Graf z úlohy *Deň matiek* sme upravili iba minimálne, tentoraz sme žiakom neponúkli žiaden ukážkový príklad (keďže také už videli v predchádzajúcej úlohe), ani možné odpovede, ale nechali sme ich nakresliť dva náramky, ktoré podľa pravidiel môžu vzniknúť, pozri Obrázok 6. Sledovali sme tým ich porozumenie grafového zápisu pravidiel a taktiež nás zaujímalo, ako budú postupovať pri hľadaní aspoň dvoch odpovedí.

[1] keďže niektorí zo žiakov sú bilingválni, resp. slovenčina nie je ich materinským jazykom



Obrázok 6: Úloha Náramky

4 VÝSLEDKY

Kvalitatívny výskum prebehol v jesenných mesiacoch na základnej škole v Bratislave v štyroch skupinách z dvoch ročníkov. Zúčastnilo sa ho dokopy 17 žiakov, z toho 7 piatakov (3 chlapci a 4 dievčatá) a 10 šiestakov (4 chlapci a 6 dievčat).

4.1 Priebeh výskumu

Výskumu bola v každej triede venovaná jedna vyučovacia hodina. Po úvodných pokynoch každý žiak sám riešil pracovný list s troma úlohami opísanými v predchádzajúcej kapitole, pričom mal okrem odpovede napísať aj vysvetlenie, prečo takú odpoveď zvolil. Táto fáza trvala približne 10 minút, počas nej sa žiaci nemohli radiť a od učiteľa dostali iba rady typu „prečítaj si ešte raz zadanie“, „pozri sa do obrázku“, prípadne odpovede na otázky súvisiace s jazykom[1].

Úlohy v pracovnom liste mali rovnaké zadanie ako úlohy v súťaži, obrázky úloh sme mierne upravili, aby na prvý pohľad nepripomínali úlohy zo súťaže.

Po samostatnej fáze sa žiaci dali do skupín podľa pohlavia a mali porovnať svoje odpovede. Každá skupina bola nahrávaná na diktafón, aby sme vedeli určiť, aký slovník zvolili pri vzájomnom vysvetľovaní, ale aj aby sme zabezpečili, že zistíme, kto mal správne a kto nesprávne odpovede. Táto fáza trvala v závislosti od skupiny približne 10 až 15 minút. Žiaci mali napísať do pracovných listov odpoveď, na ktorej sa v skupine zhodli, a vysvetlenie, prečo ich pôvodné riešenie bolo prípadne nesprávne.

Treťou fázou bolo samostatné riešenie podobných úloh. Žiakov sme k vzájomnému vysvetľovaniu motivovali tým, že v prípade, že každý žiak zo skupiny (chlapčenskej alebo dievčenskej) vyrieši všetky úlohy druhého pracovného listu správne, dostane celá skupina jednotky z informatiky. Túto metódu sme mierne modifikovali od metódy spomínanej v knihe *Mathematical Mindset* [9 str. 137]. Tento prístup sme zvolili preto, že inak žiaci iba preberajú odpovede tých v skupine, ktorých považujú za najšikovnejších (alebo najhlasnejších) a boja sa priznať, že odpovedi nerozumejú.

Na konci hodiny po odovzdaní pracovných listov učiteľ prešiel jednotlivé úlohy s celou triedu a vysvetlil, ako sa malo postupovať pri riešení a prečo sú danú odpovede správne/nesprávne.

4.2 Úloha Rodinný graf

Úloha Rodinný graf (v pracovnom liste nazvaná Rodinka Úžasných) sa ukázala ako najproblematickejšia úloha z pracovného listu. Pri samostatnej práci na nej žiaci trávili buď veľmi veľa času, alebo naopak pomerne málo (čo ale neznamená, že ju mali vyriešenú správne). Vo všetkých štyroch skupinách prišli žiaci

na to, že starou mamou je Mária. Chlapci šiestaci dokonca pôvodne za odpoveď považovali vrchol s menom Mária a nehľadali šípku znázorňujúcu vzťah. V tomto prípade učiteľ musel usmerniť ich riešenie, pričom ich upozornil na to, že majú vybrať šípku.

Obe chlapčenské skupiny nakoniec zvolili (nesprávnu) odpoveď a to šípku smerujúcu od Lucie k Márii. Viackrát zaznelo, že „Lucia je stará mama Márie“ a zároveň aj „Mária je stará mama“. Z oboch debát je zrejmé, že niektorí žiaci pochopili, čo znamená smer šípok, ale akoby si nespojili tieto dve protichodné informácie, ukážka takéhoto riešenia je na Obrázku 7. Niektorí chlapci mali pôvodne označenú aj šípku smerujúcu od Márie k Tomášovi, pri debate sa však skupinovo zhodli na druhom smere. V oboch chlapčenských skupinách viac riešili rodinné vzťahy („Lucia je Máriina vnučka, lebo...“) a menej smer šípok.

Dievčenské skupiny viac riešili smer šípok ako rodinné vzťahy. V šiestackej dievčenskej skupine správne vyčítali z grafu, že Mária má (aspoň) dve vnúčatá – Luciu a Tomáša, boli rozdelené približne na polovicu v tom, ktorý vzťah označovali. Diskusia sa niesla v smere, či šípka označuje „vzťah OD koho alebo vzťah KOMU,“ pričom si pomáhali zvyšnými znázornenými vzťahmi v grafe. Piatacké dievčatá sa tiež zamerali na smer šípok, časť si myslela, že smer šípky znamená „že je to ona“, časť opačne, následne na príklade šípky od Lucie k Pavlovi usúdili, že je to prvá možnosť, lebo „Pavol predsa nie je dcéra“. Obe dievčenské skupiny našli správne odpovede.

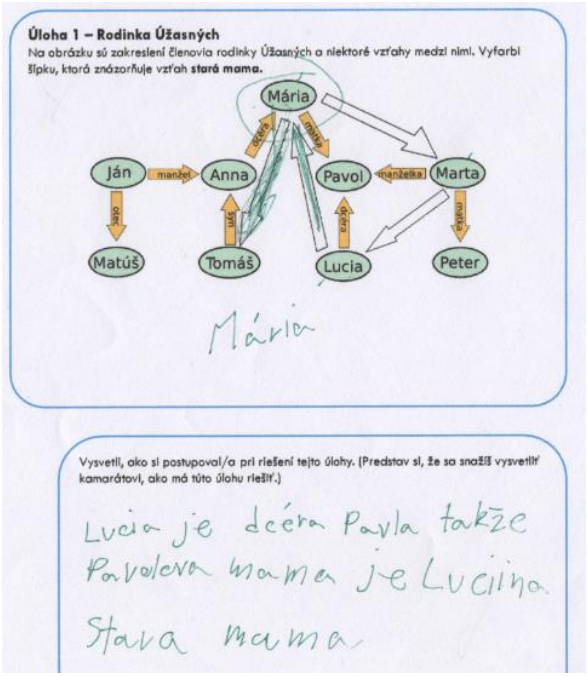
Úloha v druhom pracovnom liste sa volala Rodinka šikovných. Opäť bol na nej zobrazený graf s rodinkou a rovnakými pravidlami smeru šípok, tentoraz ale mali žiaci na prázdne miesto do šípky napísať, aký vzťah znázorňuje. Zaujímavé je, že tri skupiny odpovedali správne („mama“) – obe dievčenské a chlapci piataci. Piatacký chlapci mali všetci správnu odpoveď aj napriek tomu, že v predchádzajúcej úlohe označili zlý smer. V šiestackej skupine chlapcov mal iba jeden z nich správnu odpoveď. Tento chlapec mal správnu odpoveď aj v prvom pracovnom liste, do debaty sa ani na pobádanie učiteľom nezapájal, nemal vhodné argumenty, respektíve ho zvyšok skupiny „prekričal“. Traja zvyšní chlapci zo skupiny mali zhodne napísané „dcéra“.

4.3 Električky

Pri riešení úlohy Električky viacerí žiaci spočiatku nerozumeli nákresu. Po ozrejmení, že biele obdĺžniky a štvorce sú zastávkami, pochopili, čo majú robiť. Medzi sebou si dávali aj pomocné rady s príkladmi z reálneho prostredia, ako napríklad: „Toto máš ako zastávka Molecova. Tiež tam stojí viac spojov a môžu ísť každý inam.“

Zopár žiakov tipovalo, zvyšní mali správne odpovede. Pri tejto úlohe nevedeli zdôvodniť, prečo je ich odpoveď správna. Pri vzájomnom vysvetľovaní používali názorné ukážky typu: „odbočil sem“, „tu a tu a tu išiel“, „skončil tu“. Textové zadanie úlohy tak postupne odkrokovali na grafe. Najväčšie problémy mali šiestacký chlapci, v tejto skupine všetci tipovali. Až pri vzájomnom vysvetľovaní si vzájomne vylučovali zvyšné odpovede.

Problematické v tejto úlohe sa ukázalo to, že východzia zastávka sa neráta a preto sa mnohí žiaci zasekli už pri prvom výroku „na tretej zastávke električka odbočila“. Nakoniec sa všetky skupiny zhodli na správnej odpovedi.



Obrázok 7: Zlá odpoveď ale dobré zdôvodnenie úlohy Rodinka Úžasných (chlapec, 6. ročník)

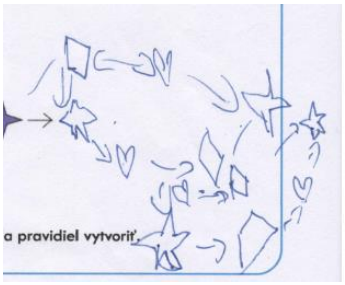
Obmenou úlohy v druhom pracovnom liste bola úloha Električky v Medveďove, v ktorej mali žiaci podľa pravidiel určiť, ktorými električkami mohol chlapec ísť. Rozdiely medzi chlapcami a dievčatami sme nezaznamenali, obe skupiny pre každé pohlavie riešili úlohu správne. Rozdiel bol iba medzi piatakmi a šiestakmi – z piatakov iba jedno dievča označilo dve možné linky (2 a 6), v šiestom ročníku ich označilo šesť žiakov (z desiatich). Nesprávne odpovede sa pri tejto úlohe nevyskytli. Linku 8 nevybral nikto zo žiakov (hoci v dvoch pracovných listoch od šiestakov sa objavila preškrtnutá), po následnej diskusii s učiteľom vyplynulo, že si neboli istí, či vystúpil na konečnej danej linky, alebo nie a preto odpoveď 8 nakoniec zamietli.

4.4 Deň matiek

Úloha Deň matiek robila žiakom najviac problémov – veľa z nich nevedelo, čo znamenajú pravidlá na obrázku a ako ich interpretovať. Najčastejším vysvetlením, prečo je odpoveď A správna bolo: „predtým išla dolnou cestičkou, teraz pôjde hornou.“ Niektorí žiaci zvolili odpoveď B s argumentom, že obsahuje všetky tvary ako sú v obrázku. Argumentom pre C bolo, že sú v správnom poradí. Chlapčenské skupiny zvolili stratégiu vyraďovania možností, v šiestackej pomerne jasne jeden z chlapcov zadefinoval pravidlá: „musí začínať srdcom, musí končiť srdcom a môžu tam byť iba štyri [tvary], iba toto [kosoštvorec] môže byť dvakrát“. Práve pri slučke pod ružovým kosoštvorcom zaznelo v každej skupine, že to znamená „že môže ísť tadeto dvakrát“, žiadna zo skupín tak neuvažovala nad tým, že môže ísť aj viac ako dvakrát.

Z diskusií bolo zjavné, že žiaci nákresu rozumejú skôr intuitívne a nevedeli úplne vysvetliť, prečo je ich odpoveď správna. To sa ukázalo ako problém pri následnom samostatnom riešení úlohy *Náramky*, pri ktorom nevyriešili príklad správne tí žiaci, ktorí sa do predchádzajúcej diskusie nezapájali alebo bolo z ich slov zrejmé,

že tomu nerozumejú. V každej skupine okrem piatackých dievčat sa vyskytol žiak, ktorý iba prekreslil diagram – pokladal to za náramok, keďže, ako to všetci z nich neskôr všetci vysvetľovali „náramok je kruhový“, pozri Obrázok 8. Všetky piatacké dievčatá napísali viac možností ako dve (tri, štyri), v každom prípade boli aspoň dve z nich správne. Chybou bolo vynechanie tvaru zo začiatku alebo konca náramku, prípadne zopakovanie tvaru, ktorý nad sebou nemal slučku. Zvyšní žiaci mali správne odpovede – všetci vybrali obe možné cesty, ak na ceste bola slučka, tak ju použili.



Obrázok 8: Zlá odpoveď pri úlohe Náramky (chlapec 6. ročník)

5 DISKUSIA

Pilotné testovanie ukázalo, že žiaci vo veľkej miere používajú metódy riešenia vybraných grafových úloh, tak ako sme očakávali na základe predchádzajúceho výskumu [3]. Najviditeľnejšie sa rozdiely medzi chlapcami a dievčatami ukázali pri úlohe s rodinným grafom, v ktorej bol aj najväčší rozdiel medzi chlapcami a dievčatami v súťaži iBobor. Rozdiely sme videli aj pri spôsobe, akým jednotlivé skupiny pracovali, chlapci oveľa častejšie pri vysvetľovaní, ako zvolili odpoveď, používali spojenia typu „tipol som si“, „[prišiel som na to] logicky“, alebo „neviem“. Dievčatá sa viac snažili opísať, ako k riešeniu prišli a aj pri vzájomnej interakcii postupovali viac demokraticky a vysvetľujúco. Samozrejme, toto môže byť ovplyvnené konkrétnou vzorkou žiakov a jej malým rozsahom. Preto plánujeme toto testovanie zopakovať na žiakoch piateho a šiesteho ročníka z inej školy.

Problémy, ktoré ukázalo pilotné testovanie, sa budeme snažiť eliminovať malou úpravou úloh. Konkrétne pri úlohe *Rodinka Úžasných (Rodinný graf)* pridáme krátku textovú informáciu, aby sledovali smer šípok. Zadanie úlohy *Električky* bude obsahovať informáciu, že zastávky sú znázornené bielymi štvorčekmi/obdĺžnikmi a taktiež dodatočnú informáciu, či vystúpil na konečnej zastávke. A pri poslednej úlohe *Deň matiek*, aby sme ju prispôsobili mladším žiakom, viac popíšeme, čo znamenajú pravidlá navliekanie (t. j. kde stroj začína navliekať korálky a kde končí) a na čo sa majú sústrediť v grafe.

Okrem upravených úloh budeme v ďalšej iterácii sledovať podobnosti a rozdiely v zdôvodňovaní odpovedí pri jednotlivých skupinách. Ďalšie výsledky použijeme na úpravu úloh (z textovej aj grafickej stránky) a ich transformáciu do učebných materiálov a príručky pre učiteľa, ako takéto úlohy včleniť do učebného procesu a ako pomocou nich rozvíjať kľúčové kompetencie na prácu so štruktúrami z iŠVP.

6 ZÁVER

Grafové štruktúry sú súčasťou informatiky, ktorá však nie je veľmi známa žiakom na základných a stredných školách. Tieto úlohy poskytujú širokú škálu námetov a nápadov, ako ich využiť vo vyučovaní. S mnohými z nich sa žiaci stretnú dokonca aj v reálnom živote (mapa liniek MHD, diagramy s návodmi a podobne). Informatická súťaž iBobor ponúka učiteľom a žiakom veľké množstvo úloh, ktoré sa tejto tematike venujú. Predchádzajúce výskumy na týchto úlohách ukázali, že existujú rozdiely v úspešnosti chlapcov a dievčat pri ich riešení, a tie súvisia s metódami, ktoré sa pri ich riešení používajú.

Tento článok sa venuje pilotnému overovaniu troch metód riešenia grafových úloh, konkrétne metóde „pozriem, vidím“, prehľadávaniu grafu s podmienkami (obmedzeniami) a vyskúšaniu všetkých možností. Pre každú z metód sme vybrali jednu zo starých úloh súťaže (na základe ktorých spomínaná kategorizácia vznikala), mierne sme ju upravili a otestovali so žiakmi piateho a šiesteho ročníka. Testovanie potvrdilo, že výskumná skupina žiakov pri riešení týchto úloh využívala metódy z kategorizácie. Rozdiely medzi chlapcami a dievčatami sme zaznamenali skôr pri prístupe k úlohám a k spoločnej diskusii, v ktorej mali hľadať spoločné riešenie. Tieto rozdiely však nepovažujeme za významné, pokiaľ sa nám ich nepodarí overiť aj na väčšej vzorke. Pilotné testovanie tiež odhalilo niekoľko nedostatkov v testovaných úlohách, ktoré mohli spôsobiť žiakom problémy. V ďalšej etape výskumu ich plánujeme eliminovať a viac skúmať spôsoby, akými žiaci ku grafovým úlohám pristupujú a metódy, ktoré pri ich riešení používajú. Výsledky budú slúžiť na tvorbu didaktických materiálov určených na vyučovanie oblasti práca s grafovými štruktúrami.

BIBLIOGRAFICKÉ ODKAZY

[1] ŠTÁTNY PEDAGOGICKÝ ÚSTAV. *Štátny vzdelávací program - Informatika*. http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_nsv_2014.pdf

[2] IBOBOR. http://ibobor.sk/

[3] BUDINSKÁ, L., MAYEROVÁ, K. Graph Task in Bebras Contest: What does it have to do with gender? In *Proceedings of the 6th Computer Science Education Research Conference*. ACM, 2017, s. 83-90.

[4] CRESWELL, J. W. *A Concise Introduction to Mixed Methods Research*. SAGE, 2015. 132 s. ISBN 978-1-4833-5904-5.

[5] AGRESTI, A. Categorical Data Analysis. 2nd edition. John Willey & Sons, 2012. ISBN 0-471-36093-7.

[6] ŠVAŘÍČEK, R. ŠEĎOVÁ, K. et al. Kvalitativní výzkum v pedagogických vědách. Praha: Portál, 2014. 384 s. ISBN 978-80-262-0644-6.

[7] BUDINSKÁ, L. MAYEROVÁ, K. WINCZER M. Analýza grafových úloh zo súťaže iBobor – Juniori a Seniori. In *Didinfo 2018*. [online]. Liberec, 2018. s. 237-242. ISBN: 978-80-7494-424-6. Dostupné z: https://didinfo.net/images/DidInfo/files/didinfo_2018.pdf

[8] BUDINSKÁ, L. MAYEROVÁ, K. WINCZER M. Gender differences in graph tasks – Do they exist in high school Bebras categories too?. In *ISSEP 2018*. [online]. St. Petersburg, 2018. s. 295-306. ISBN 978-3-030-02750-6. Dostupné z: https://doi.org/10.1007/978-3-030-02750-6

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Nájdeš chybu v mojom programe?
Can you find a bug in my program?

Iveta Csicsolová
KDFMI, Fakulta matematiky, fyziky a informatiky
Univerzita Komenského v Bratislave
Mlynská dolina
84248 Bratislava
Slovensko
ivetkacs@gmail.com

Zuzana Kubincová
KDFMI, Fakulta matematiky, fyziky a informatiky
Univerzita Komenského v Bratislave
Mlynská dolina
84248 Bratislava
Slovensko
kubincova@fmph.uniba.sk

ABSTRACT
Mutual review of students' work is an educational strategy recently addressed by many researchers. It can be implemented into the programming teaching in the form of code review that is, reviewing the program code mutually by the students. This activity can help significantly improve the program code and increase its efficiency. In addition, it can also bring benefits to students in the form of development of soft skills, streamlining their learning, promotion of social learning, and so on. In this paper, we present partial results from our research on using code review in informatics classes at secondary school.

Keywords
Informatics. Programming. Peer review. Code review. Commenting. High school.

ABSTRAKT
Vzájomné recenzovanie práce študentov je vzdelávacia stratégia, ktorej sa venuje mnoho súčasných výskumov. Do vyučovania programovania môže byť implementovaná formou code review, teda recenzovania programového kódu medzi študentmi navzájom. Okrem toho, že táto aktivita môže pomôcť výrazne zlepšiť a zefektívniť skúmaný programový kód, môže tiež študentom priniesť výhody vo forme rozvoja interpersonálnych zručností, zefektívnenia ich učenia sa, sociálneho učenia sa a pod. V tomto príspevku prezentujeme čiastkové výsledky z nášho výskumu využitia code review v informatike na strednej škole.

Kľúčové slová
Informatika. Programovanie. Vzájomné recenzovanie. Code review. Komentovanie. Stredná škola.

1. ÚVOD
Medzi techniky, na ktoré sa v poslednej dobe obracia pozornosť špecialistov vo vzdelávaní patrí okrem iného aj vzájomné hodnotenie práce študentov, tzv. peer review [1, 2]. Táto aktivita môže priniesť študentom rôzne benefity napomáhajúce k zlepšeniu ich študijných výsledkov, prispieť k rozvoju ich interpersonálnych zručností (z angl. soft skills) a posilniť sociálne učenie sa [3, 4].

Jednou z možností, ako zapojiť vzájomné hodnotenie do vyučovania informatiky, je využiť túto metódu na recenzovanie programového kódu pri výučbe programovania – tzv. code review [5]. Takáto vzdelávacia aktivita je založená na sociálnom konštruktivizme [6] a je motivovaná rozsiahlym využívaním code review v softvérových firmách pri vývoji veľkých programátorských projektov [7]. Ako ukazujú viaceré výskumy, vzájomné recenzovanie programového kódu v tímoch profesionálnych programátorov vedie k významnému zlepšeniu kvality programového kódu a celkovej efektívnosti vývoja projektu [8, 9].

Väčšina publikovaných výskumov zaoberajúcich sa využitím code review vo vzdelávaní, bola z prostredia univerzitných kurzov. Pokiaľ ide o výskum v tejto oblasti na Slovensku, nenašli sme žiadnu relevantnú publikáciu týkajúcu sa strednej školy. Preto sme sa zhruba pred rokom začali takémuto výskumu venovať. V tomto článku prinášame čiastkové výsledky nášho výskumu.

2. CODE REVIEW V PROGRAMOVANÍ NA STREDNEJ ŠKOLE
V snahe vyskúšať zapojenie techniky code review do vyučovania na strednej škole a zistiť jej vplyv na učenie sa žiakov sme navrhli aktivity spojené s touto technikou a otestovali ich v podmienkach bežného vyučovania.

Výskum bol realizovaný počas školského roka 2017/2018 na hodinách informatiky v dvoch triedach tretieho ročníka strednej školy[1], počas výučby informatického celku Algoritmické riešenie problémov, kde sa žiaci učili programovať v jazyku Python. Výučba celku prebiehala od novembra do júna, takže výskum zasahoval do oboch polrokov. Žiaci mali jednu hodinu informatiky do týždňa.

Už v predchádzajúcom ročníku žiaci programovali v Pythone, pričom sa stretli so základnými grafickými príkazmi knižnice tkinter, s náhodnými číslami, premennými a cyklom for. Toto učivo bolo najprv preopakované, a následne sa pokračovalo novými témami: funkcie, udalosti myši, klávesnica, podmienky, časovač a posúvanie objektov canvasu.

Do vyučovania boli zapojené dva typy aktivít s code review:

- *Malé recenzie*, ktoré prebiehali formou krátkej písomky po prebratí nejakého celku. Žiaci tu recenzovali vždy dva krátke programy pripravené učiteľkou. V prvom z nich mali za úlohu zistiť, čo daný programový kód vykoná, v druhom bolo treba v zadanom programe nájsť chyby (syntaktické aj logické) a správne ich opraviť. Podrobnejšie informácie o tejto aktivite boli publikované v iných našich prácach [10,11].
- *Recenzie projektov*. Žiaci počas školského roka dostali naprogramovať dva projekty – prvý v 1.polroku, druhý na konci 2.polroka.

V tomto článku sa zameriame predovšetkým na aktivitu druhého typu – recenzovanie projektov.

[1] Ide o bilingválnu strednú školu s 5-ročným štúdiom.

3. VZÁJOMNÉ RECENZOVANIE PROJEKTOV

Po uskutočnení niekoľkých malých recenzií, ktoré bolo možné chápať ako tréning, sme žiakov nechali recenzovať si navzájom projekty. Pri tejto recenzii teda už išlo o vzájomné komentovanie programového kódu vytvoreného samotnými žiakmi – čiže o aktivitu, ktorá bola podobnejšia skutočnému code review.

Vzájomné recenzovanie projektov sme v školskom roku 2017/2018 využili dvakrát. Prvý projekt dostali žiaci vypracovať počas zimných prázdnin, čo bolo len zopár týždňov po tom, ako sa na informatike začalo preberať programovanie. V tom čase mali okrem preopakovaného učiva z predchádzajúceho ročníka zvládnuté náhodné farby, cyklus for, funkcie, udalosti na klikanie myšou a udalosti s klávesnicou. Na celú projektovú aktivitu vrátanie komentovania mali 5 týždňov. Druhý projekt dostali zadaný ku koncu programátorského celku (v apríli), keď mali prebraté navyše podmienky, časovač a posúvanie objektov canvasu. Keďže v projekte bolo treba využiť všetky získané vedomosti, bol značne obtiažnejší, ako predchádzajúci. Žiaci si vyžiadali dlhšiu dobu na jeho vypracovanie, takže súhrnný čas, počas ktorého sa mu venovali, bol 7 týždňov.

Pri prvom projekte si žiaci vyberali jedno z troch zadaní – preteky, skicár a pohľadnica – takže v každej skupine sa projekt s rovnakým zadaním mohol viackrát opakovať. Každé z týchto zadaní bolo podrobne špecifikované.

Pri druhom projekte sme žiakom ponúkli sedemnásť rôznych zadaní – napr. Kameň-Papier-Nožnice, Športka, Odrážanie loptičky, Sofistikovaný skicár, T-Rex, a pod. – a mali si z nich vybrať jedno tak, aby sa v danej skupine žiadne zadanie neopakovalo. Aj pri týchto projektoch boli presne stanovené špecifikácie. Žiaci si mohli navrhnúť a dať schváliť aj svoju vlastnú tému projektu.

3.1 Účastníci

V prvom polroku sa výskumu zúčastnilo 52 žiakov, v druhom polroku 55 žiakov (pozn. traja žiaci, ktorí prestúpili do tried v 2.polroku, programovali naposledy v druhom ročníku).

Žiaci boli rozdelení do štyroch vyučovacích skupín. V skupine 3A1 bolo 13 žiakov, z toho traja chlapci (v 2.polroku plus 1 žiak), v skupine 3A2 bolo 14 žiakov, z toho päť chlapcov. V skupine 3B1 bolo 12 žiakov (v 2.polroku plus 1 žiačka), z toho štyria chlapci a v skupine 3B2 bolo 13 žiakov (v 2.polroku plus 1 žiačka), z toho šesť chlapcov.

3.2 Metodika vzájomného recenzovania

Vzájomné recenzovanie projektov prinieslo do projektovej aktivity nový aspekt – možnosť opraviť, resp. vylepšiť svoj projekt na základe komentárov získaných od spolužiakov. Celú prácu na projekte bolo možné rozdeliť do troch základných fáz: A. vývoj projektu, B. recenzovanie spolužiakovho projektu, C. úprava vlastného projektu. Jednotlivé fázy zahŕňali nasledovné kroky:

A. Vývoj projektu
   - Výber jednej z ponúkaných tém projektu
   - Naprogramovanie projektu
   - Odovzdanie projektu na recenzovanie

B. Recenzovanie spolužiakovho projektu
   - Pridelenie projektov na recenzovanie
   - Komentovanie prideleného projektu
   - Odovzdanie okomentovaného projektu

C. Úprava projektu
   - Vrátenie okomentovaného projektu jeho autorovi
   - Úprava, dokončenie, resp. vylepšenie projektu
   - Odovzdanie finálnej verzie projektu na hodnotenie učiteľkou

Prvá fáza – vývoj projektu – trvala v prípade prvého projektu tri týždne a v prípade druhého projektu až päť týždňov. Tento predĺžený čas vyplýval nielen z vyššej náročnosti druhého projektu, ale aj z toho, že žiaci ho vyvíjali popri iných školských povinnostiach, zatiaľ čo prvý projekt vypracovávali v čase prázdnin.

Keďže sme nemali k dispozícii žiadny nástroj na distribuovanie programov a recenzovanie programového kódu, ktorý by bol vhodný pre použitie na strednej škole, žiaci odovzdávali svoje projekty prostredníctvom e-mailu učiteľke. Táto pridelila projekty iným žiakom na recenzovanie tak, aby autor a recenzent nepatrili do tej istej skupiny. Týmto sme sa snažili zabezpečiť anonymitu pri komentovaní projektu – autor projektu nevedel, kto recenzuje jeho program a recenzent nevedel, čí program recenzuje.

Úlohou žiaka v roli recenzenta bolo skontrolovať pridelený spolužiakov projekt: zistiť, či dodržal všetky časti zadania, odhaliť chyby a upozorniť na ne, poradiť spolužiakovi, čo je nutné prerobiť alebo doplniť a napísať mu celkový svoj názor na jeho projekt (Obrázok 1). Na komentovanie projektu mali vyhradený jeden týždeň. Potom žiaci opäť e-mailom poslali okomentovaný program naspäť učiteľke, ktorá zabezpečila, že sa vrátil naspäť k jeho autorovi.

```
dx3=0
c.create_text(100,100,text="Vyhralo"+vyhra+"auticko",tags="textik")
#text je spojený, možno by bolo treba vypisať každé slovo na statické súradnice osve
def vyhraj():
    c.after(500)
    global vyhra,kolo,kolo2,kolo3,dx,dx2,dx3
    if kolo==2 and dx==0:
        vyhra="prve"
    if kolo2==2 and dx2==0:
        vyhra="druhe"
    if kolo3==2 and dx3==0:
        vyhra="tretie"
    c.delete("textik")
    c.create_text(600,500,text="Vyhralo"+vyhra+"auticko",tags="textik",font="Arial80")
            #text/výsledok pretekov sa po skončení závodu nevypíše
vyhraj()

#Kliknutim na medzernik sa startuje cela hra
#Kliknutim na "c" sa autícka posunu naspat na zaciatok

#problém s vypísaním textu, by sa možno by dal vyriešiť tým, že by si zmenil/a
    #jednu z dx súradníc a tú následne zmenil/a
#aj v podmienkach pri def vyhraj() a už dopredu presne určil/a ktoré auto vyhrá
#vypísanie textu by sa ptm dalo riešiť podmienkou, v ktorej by si určil/a, že ak
    #toto auto príde do istého bodu v ciely,
#tak na súradniciach pod 'Vyhralo autíčko' sa vypíše text s farbou/názvom toho
    #už vopred určeného autička
#tu by však bolo víťazné auto už dopredu určené

#celkovo sa mi však projekt veľmi páči a je vidieť že si si s tým dal/a námahu a čas
```

Obrázok 1. Ukážka komentovania žiakmi

V poslednej fáze, ktorá trvala opäť jeden týždeň, bolo úlohou žiaka zapracovať opodstatnené pripomienky recenzenta k jeho programu, prípadne si opraviť chyby, ktoré si medzičasom v projekte našiel sám. Takto žiaci získali možnosť opraviť si nedostatky v projekte a vylepšiť ho, aby mohli dosiahnuť aj lepšiu známku.

Po poslednej fáze učiteľka zaslala každému žiakovi individuálne hodnotenie, kde okomentovala žiakovu prácu v jednotlivých fázach výskumu. Sledovala, nakoľko žiak zvládol naprogramovať projekt v prvej fáze, ako komentoval spolužiakov projekt v druhej fáze a napokon zhodnotila žiakov výsledný projekt a sledovala, čo a ako v ňom zapracoval na základe spolužiakových komentárov.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

3.3 Výsledky

3.3.1 Výsledky hodnotenia projektov

Pri oboch projektoch boli už pri zadaní každej témy stanovené konkrétne požiadavky, ktoré bolo nutné dodržať. Spoločné kritériá, ktoré sa brali do úvahy pri výslednom hodnotení boli: úplná funkčnosť, správnosť a pravosť projektu, komentovanie spolužiakovho projektu: nájdenie chýb, upozornenie na ne, návrh na riešenie, názor na projekt, ako aj zapracovanie pripomienok od spolužiaka. Za vylepšenia projektu nad rámec zadania mohli žiaci získať ďalšie bonusové body. Keďže oba projekty boli pre žiakov náročnejšou úlohou, ako ostatné úlohy z tohto predmetu, prispievali k výslednej známke z informatiky väčšou váhou.

Tabuľky 1 a 2 zobrazujú výsledky hodnotenia oboch projektov. Údaje v nich ukazujú, koľko percent študentov v jednotlivých skupinách dosiahlo príslušné percentuálne hodnotenie za daný projekt.

Tabuľka 1. Percento žiakov s príslušným percentuálnym hodnotením za prvý projekt podľa skupín

Skupina	100-90%	89-75%	74-50%	49-30%	29-1%	0%	Výnimočne dobré projekty
A1	17.3%	5.8%	1.9%	-	-	-	7.7%
A2	21.2%	1.9%	1.9%	-	-	1.9%	9.6%
B1	11.5%	7.7%	-	-	-	3.8%	11.5%
B2	23.1%	-	1.9%	-	-	-	5.8%
spolu	73.1%	15.4%	5.8%	-	-	5.8%	34.6%

Tabuľka 2. Percento žiakov s príslušným percentuálnym hodnotením za druhý projekt podľa skupín

Skupina	100-90%	89-75%	74-50%	49-30%	29-1%	0%	Výnimočne dobré projekty
A1	18.2%	3.6%	1.8%	-	1.8%	-	7.3%
A2	10.9%	7.3%	5.5%	1.8%	-	-	9.1%
B1	14.5%	3.6%	-	3.6%	-	1.8%	10.9%
B2	10.9%	7.3%	3.6%	3.6%	-	-	5.5%
spolu	54.5%	21.8%	10.9%	9.1%	1.8%	1.8%	32.7%

Výsledky prvého projektu boli veľmi uspokojivé. Po obdržaní komentárov od spolužiakov žiaci opravili mnoho chýb a dosiahli veľmi dobré hodnotenie od učiteľky. Až 73% projektov dosiahlo percentuálne hodnotenie zodpovedajúce známke „výborný“. Viac ako tretinu zo všetkých projektov učiteľka označila za výnimočne dobré, prekračujúce jej očakávania. Len traja žiaci získali za projekt 0% – dvaja z nich projekt vôbec neodovzdali a v treťom prípade išlo o odhalené plagiátorstvo.

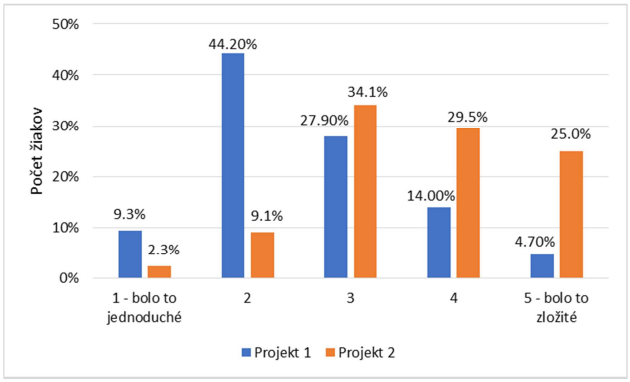
Ako bolo už vyššie spomenuté, druhý projekt bol pre žiakov značne náročnejší, ako prvý. Zodpovedajú tomu aj výsledky, ako možno vidieť v Tabuľke 2. Percentuálne rozdelenie žiakov do jednotlivých kategórií hodnotenia je rovnomernejšie ako v prípade prvého projektu a zahŕňa aj kategórie nižšie ako 50%. Napriek tomu takmer 55% žiakov dosiahlo najlepšie hodnotenie a opäť takmer tretina projektov bola učiteľkou označená za mimoriadne dobré projekty. Hodnotením 0% bol ohodnotený iba jeden žiak, ktorý projekt neodovzdal.

3.3.2 Analýza výsledkov dotazníka

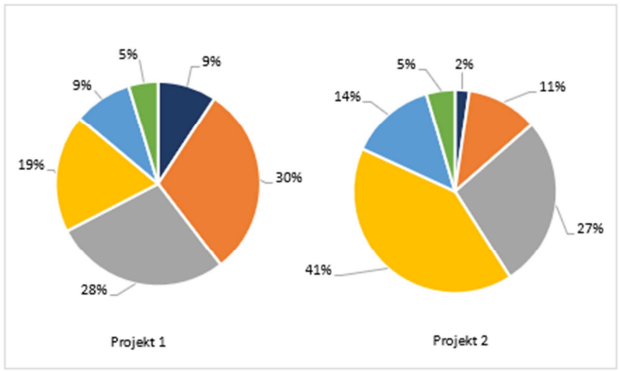
Po ukončení oboch projektových aktivít a ohodnotení všetkých projektov učiteľkou boli žiaci požiadaní o anonymné vyplnenie dotazníkov. Ich cieľom bolo zistiť ich názory na vzájomné recenzovanie projektov a ich skúsenosti z tejto aktivity.

Prvý dotazník vyplnilo 43 žiakov a druhý 44 žiakov, z toho v oboch prípadoch zhruba 61% dievčat a 39% chlapcov (± 0,5%).

V úvode dotazníka sme zisťovali nakoľko programátorsky náročný bol pre žiakov projekt. Prvý projekt nebol náročný (hodnoty 1-3) pre 81,4% žiakov, pri druhom projekte takto odpovedalo 45,5% žiakov (viď graf na obrázku 2). Zatiaľ čo pri prvom projekte sme neočakávali až taký vysoký počet pozitívnych odpovedí, ich pokles v druhom dotazníku podľa nás zodpovedal zvýšenej náročnosti projektu.



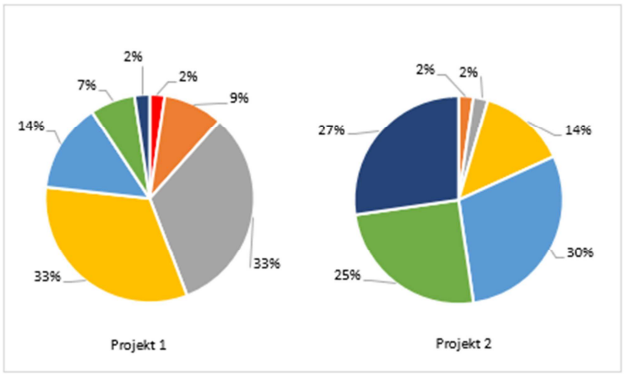
Obrázok 2. Odpovede žiakov na otázku „Bolo pre Teba programátorsky náročné vypracovať projekt?“



Obrázok 3. Odpovede žiakov na otázku „Bol pre Teba projekt časovo náročný?“ Nie, netrvalo mi to dlho, Bolo to pre mňa primerane časovo náročné, Zabralo mi to trochu viac času, ako som predpokladal/a, Áno, trvalo mi to omnoho dlhšie, ako som pôvodne predpokladal/a, Bol to časový zabijak, Iné

Už počas práce na prvom projekte sa žiaci v škole sťažovali na jeho časovú náročnosť, pri druhom projekte si dokonca vyžiadali predĺžený čas na jeho naprogramovanie. Preto sme časovú náročnosť projektov skúmali aj v dotazníku.

Ako vidno na obrázku 3, percento žiakov, pre ktorých projekt nebol časovo veľmi náročný kleslo pri druhom projekte na tretinu v porovnaní s prvým projektom (z 39% na 13%), zatiaľ čo percento žiakov, ktorí považovali projekt za veľmi časovo náročný pri druhom dotazníku stúplo na takmer dvojnásobok oproti prvému dotazníku (z 28% na 55%). Podrobnejšie informácie o čase, ktorý žiaci strávili programovaním projektu sú zachytené v grafoch na obrázku 4.

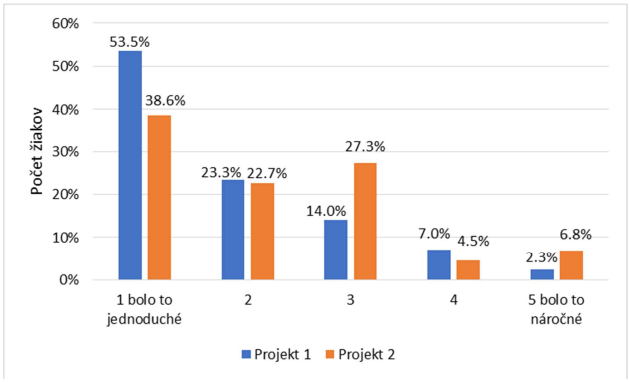


Obrázok 4. Odpovede žiakov na otázku „Ako dlho Ti trvalo naprogramovať projekt?“ Do 30 min, Viac ako 30 min, ale maximálne 1 hod, Viac ako 1 hod, ale maximálne 2 hod, Viac ako 2 hod, ale maximálne 3 hod, Viac ako 3 hod, ale maximálne 4 hod, Viac ako 4 hod, ale menej ako 8 hod, 8hod a viac

Podľa učiteľkinho stanoviska, bolo prijateľné, ak žiakom trvalo prvý projekt naprogramovať do 3 hodín a druhý za 4-8 hodín. Pritom brala do úvahy množstvo podobných už naprogramovaných príkladov a programátorské skúsenosti, ktoré nadobudli v škole a na domácich úlohách. Ak to žiakovi trvalo viac, mal by venovať viac času príprave na vyučovanie. Podľa prvého dotazníka až 77% žiakov zvládlo naprogramovať prvý projekt za menej ako 3 hodiny. V prípade druhého projektu toto číslo kleslo len na 18%, ale pod hranicou 8 hodín zvládlo projekt naprogramovať 73% žiakov (Obrázok 4).

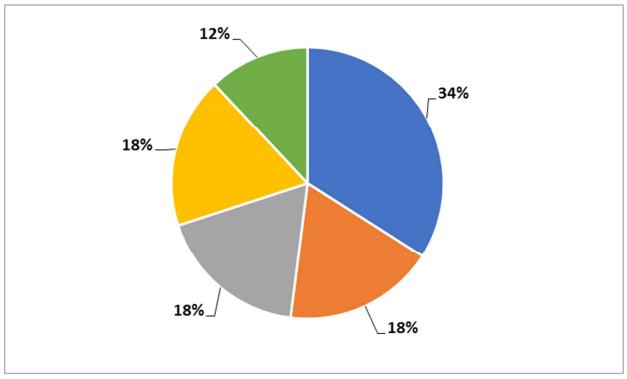
Keďže pred prvým projektom žiaci nemali skúsenosti s recenzovaním programového kódu dlhšieho ako zopár riadkov (malé recenzie), zaujímal nás ich názor na náročnosť komentovania projektu. Otázku sme v dotazníku ponechali aj pri druhom projekte. Aj keď rozloženie odpovedí v druhom dotazníku nebolo také jednoznačne výpovedné ako v prvom (Obrázok 5), percentuálne zastúpenie žiakov, ktorí komentovanie spolužiakovho projektu nepovažovali za náročné (hodnoty 1-3), bolo v oboch prípadoch veľmi vysoké a zhruba rovnako veľké (90,8% a 88,6%).

Učiteľka nám taktiež potvrdila, že žiaci pri oboch projektoch dokázali pochopiť cudzí kód, dokázali v ňom nájsť chyby a okomentovať ho. Dokonca sa stalo aj to, že našli chybu, ktorú si ani učiteľka nevšimla. V prípade prvého projektu bolo medzi žiakmi len veľmi málo takých, ktorí nedokázali odhaliť väčšinu chýb v spolužiakovom projekte. Viacerí dokázali nájsť všetky chyby. Pri druhom projekte, vďaka jeho väčšej náročnosti niektorí žiaci neboli schopní dobre poradiť ako projekt opraviť, avšak mnohí to dokázali, čo následne učiteľka ocenila, keď žiakom poskytovala spätnú väzbu na ich aktivitu.



Obrázok 5. Odpovede žiakov na otázku „Bolo náročné okomentovať spolužiakov projekt?“ Škála od 1 (bolo to jednoduché) do 5 (bolo to náročné)

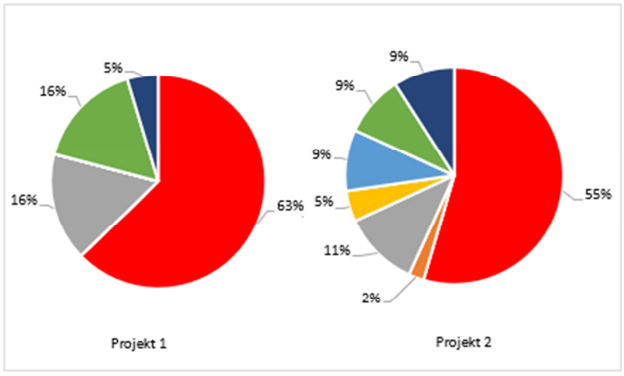
V druhom dotazníku sme od žiakov chceli, aby porovnali náročnosť komentovania prvého a druhého projektu (Obrázok 6). Pre 52% žiakov to bolo jednoduchšie, či už vďaka tomu, že v prvom polroku už komentovali projekt (34%) alebo preto, že usúdili, že už vedia lepšie programovať a lepšie rozumejú kódu, než pri prvom projekte (18%). Podiel žiakov, pre ktorých to bolo ťažšie, bol 18% a dôvodom pre nich bolo, že aj projekt bol ťažší. 12% žiakov pridalo vlastnú odpoveď – pričom asi pre polovicu z nich bolo komentovanie tohto projektu z rôznych dôvodov ľahšie a pre druhú polovicu ťažšie. Celkovo tak komentovanie ťažšieho projektu bolo pre takmer 60% žiakov ľahšie a len pre menej ako štvrtinu žiakov bolo ťažšie.



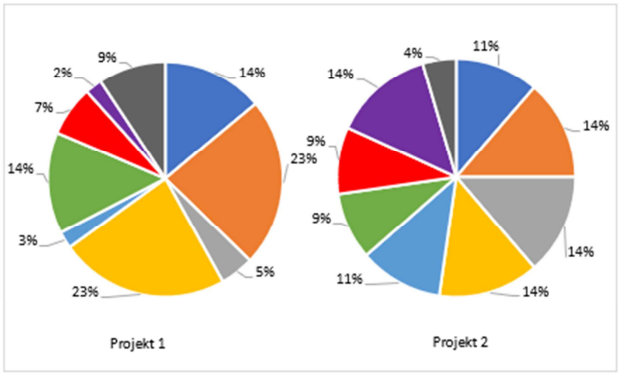
Obrázok 6. Odpovede žiakov na otázku Aké ťažké bolo komentovať tento projekt oproti komentovaniu minulého projektu? Bolo to jednoduchšie práve vďaka tomu, že sme už minule komentovali projekt, Bolo to jednoduchšie, pretože viem lepšie programovať a lepšie rozumiem kódu, než pri prvom projekte, Bolo to rovnako ťažké ako minule, Bolo to ťažšie, lebo projekt bol ťažší, Iné

Zisťovali sme tiež, s akými problémami sa žiaci pri komentovaní projektu stretli (Obrázok 7). Pri prvom projekte až 63% z nich uviedlo, že nemali žiadny problém s komentovaním projektu, 16% žiakov sa nevedelo dobre zorientovať v spolužiakom kóde, 16% nemalo čo komentovať, pretože nenašli žiadne chyby. Pri druhom projekte boli odpovede žiakov distribuované do viacerých kategórií, objavili sa aj odpovede, ktoré sa v predchádzajúcom

dotazníku nevyskytli (napr. „Vôbec som projektu nerozumel/a“, alebo „Nedokázal/a som dobre popísať chybu, ktorú som našiel/a“). Avšak aj napriek tomu, že žiaci označovali druhý projekt za veľmi náročný, 55% z nich nemalo žiadny problém s komentovaním spolužiakovho projektu.



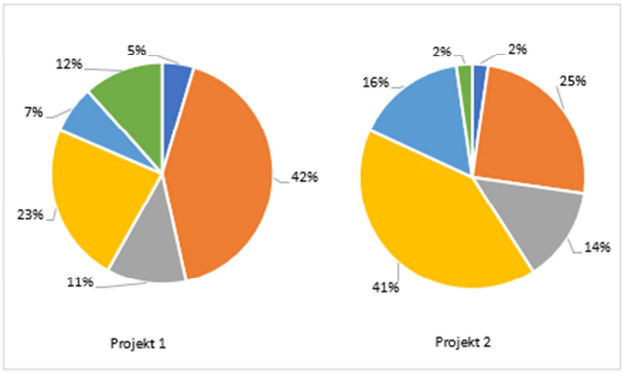
Obrázok 7. Odpovede žiakov na otázku „Čo Ti robilo problémy pri komentovaní spolužiakovho projektu?“ Nemal/a som žiadne problémy pri komentovaní projektu, Vôbec som projektu nerozumel/a, Nevedel/a som sa dobre orientovať v jeho projekte, Nevedel/a som zistiť, prečo projekt nefunguje, Nevedel/a som dobre popísať chybu, ktorú som v projekte našiel/a, Nemal/a som čo komentovať, lebo som nenašiel/a žiadne chyby, Iné



Obrázok 8. Odpovede žiakov na otázku „Pomohli Ti komentáre od spolužiaka pri dokončovaní projektu? Áno, dostal/a som dobré komentáre, ktoré mi veľmi pomohli. Áno, dostal/a som dobré komentáre, ktoré mi pomohli a dokonca ma potešilo, že mi projekt pochválil. Dostal/a som dobré komentáre, ale nevedel/a som ich zapracovať do projektu. Niektoré komentáre mi pomohli, niektoré nie. Projekt mi fungoval, spolužiak nenašiel žiadne chyby, takže som nedostal/a žiadne komentáre, ktoré by som mohol/a zapracovať. Spolužiak nenašiel žiadne chyby, ale potešilo ma, že mi projekt pochválil. Nie, nepomohli mi, lebo som im nerozumel/a a nevedel/a som ich zapracovať do projektu. Nie, dostal/a som nezmyselné komentáre, ktoré mi vôbec nepomohli. Iné

Dôležité boli pre nás aj odpovede na otázku, či žiakovi pomohli komentáre od spolužiaka pri dokončovaní projektu. Odpovede žiakov boli dosť odlišné aj v rámci toho istého dotazníka (Obrázok 8). Pri prvom projekte viac ako polovica žiakov deklarovala pozitívnu skúsenosť s komentármi od spolužiaka a iba 9% žiakov odpovedalo, že im komentáre nepomohli. Pri druhom projekte boli odpovede žiakov takmer úplne rovnomerne rozdistribuované do všetkých kategórií. Zrejme vzhľadom k náročnosti projektu stúplo percento žiakov, ktorí nedokázali spolužiakov projekt zrozumiteľne a zmysluplne okomentovať na 23%. Avšak komentárov ohodnotených ako „dobré“ bolo viac ako tých negatívnych – 39%.

Pri oboch projektoch žiaci videli prínos komentárov (Obrázok 9) od spolužiakov hlavne v tom, že im pomohli opraviť chyby, ktoré si nevšimli alebo nevedeli opraviť pri prvom odovzdaní, prípadne im pomohli inak (58% pri prvom a 41% pri druhom projekte). Náročnosť druhého projektu zrejme ovplyvnila odpovede aj pri tejto otázke – podiel žiakov, ktorí uviedli, že im komentáre nijako nepomohli, stúpol z 23% na 41%.

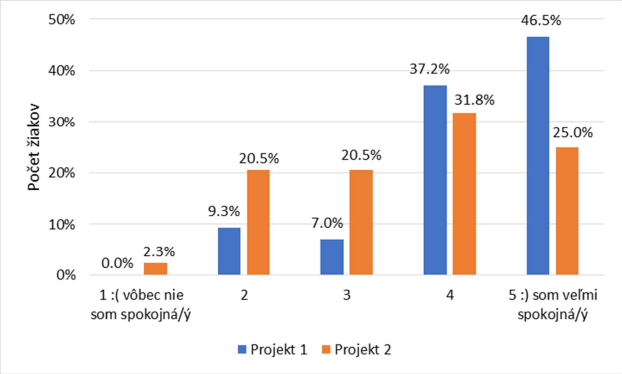


Obrázok 9. Odpovede žiakov na otázku „Čím boli podľa teba prínosné komentáre, ktoré si dostal/a od spolužiaka?“ Pomohli mi lepšie pochopiť zadanie projektu. Pomohli mi opraviť nedostatky, ktoré som si nevšimol/a pred prvým odovzdaním. Pomohli mi opraviť chyby, ktoré som nevedel/a vyriešiť pred prvým odovzdaním. Nijako mi nepomohli. Žiadne relevantné som nedostal/a. Iné

Žiakov sme sa pýtali aj na to, či podľa nich niečo získali tým, že komentovali spolužiakov projekt. V oboch dotazníkoch bola najčastejšia odpoveď (37,2% a 27,3%), že si lepšie uvedomili, aké rôzne chyby sa môžu vyskytnúť v projekte. Ďalej žiaci často uvádzali, že zistili, ako zadanie pochopil spolužiak, získali skúsenosti s recenzovaním programátorských projektov a naučili sa poskytovať konštruktívnu kritiku. Pri oboch dotazníkoch zhruba rovnaké percento žiakov (16,9% a 15,9%) uviedlo, že im komentovanie projektu nedalo nič.

V závere dotazníka sme zisťovali nakoľko sú žiaci spokojní s tým, ako vypracovali projekt. Ako vidno na obrázku 10, s prvým projektom boli žiaci vyslovene spokojní, zatiaľ čo pri druhom projekte boli ich odpovede rozdelené rovnomernejšie do viacerých kategórií. Je možné, že k tomu, aby si lepšie uvedomili nedostatky svojho projektu, im pomohli aj komentáre od spolužiakov.

ISBN 978-80-557-1533-9, ISSN 2454-051X.



Obrázok 10. Odpovede žiakov na otázku „Ako si spokojný/á so svojím vypracovaným projektom?“ Škála od 1 (vôbec nie som spokojný/á) do 5 (som veľmi spokojný/á)

4. ZÁVER

Využívaním code review vo vyučovaní programovania sa pokúšame pomôcť žiakom lepšie vniknúť do tvorby programového kódu, naučiť sa odhaľovať chyby v programe, pochopiť rôzne prístupy k riešeniu toho istého problému.

Code review sme so žiakmi skúšali na dvoch projektoch s odstupom niekoľkých mesiacov. Prvý projekt bol pomerne jednoduchý a žiaci v ňom dosiahli veľmi uspokojivé výsledky. Druhý projekt bol značne náročnejší jednak vďaka novým zložitejším programátorským konceptom, ktoré žiaci museli zmysluplne skombinovať, ale aj kvôli komplexnejšiemu zadaniu. Aj recenzovanie druhého projektu bolo pre žiakov náročnejšie. Výsledky tohto projektu lepšie rozčlenili žiakov do skupín podľa vedomostí z programovania. Celkovo však aj tento projekt dopadol veľmi dobre.

Aktivita code review zo začiatku nebola žiakmi prijatá veľmi pozitívne, keďže ju vnímali ako prácu navyše. Pri komentovaní projektov museli pochopiť spolužiakov program, ktorý mal odlišné zadanie, ako ich vlastný projekt, čo považovali za zložité a časovo náročné. Niekedy uvádzali ako problém veľké množstvo chýb, ktoré našli a museli okomentovať tak, aby autorovi neprezradili správne riešenie, iba ho k nemu nasmerovali.

Viacerí žiaci si však uvedomili užitočnosť takéhoto spôsobu učenia sa programovania a uviedli napr., že ich to prinútilo zamýšľať sa aj nad programátorskými problémami, s ktorými sa vo svojom vlastnom projekte nestretli. Okrem toho, mnohí žiaci ocenili aj spätnú väzbu od učiteľky, ktorá im pomohla presne pochopiť, čo robili v projekte dobre a čo zle, a tým dostali šancu zlepšiť sa nabudúce.

Úlohou žiakov bolo naprogramovať (takmer) finálny projekt už pri prvom odovzdaní. Z pohľadu učiteľky však žiaci odovzdali vo väčšine prípadov omnoho lepší projekt po opravení chýb, ktoré im našiel spolužiak, alebo keď sa s odstupom času na projekt pozrel sám žiak. Žiaci sa skúmaním cudzieho aj vlastného kódu naučili predchádzať chybám, ako aj hľadať a opravovať chyby. Pred použitím code review na vyučovaní žiaci počas hodiny často žiadali učiteľku o pomoc už pri malých chybových hláškach a syntaktických chybách. Tieto aktivity ich naučili istej sebestačnosti pri čítaní kódu.

Na základe výsledkov nášho výskumu si dovolíme tvrdiť, že code review je možné využiť vo vyučovaní už na strednej škole. Žiakov učí čítať kód, orientovať sa v ňom, pochopiť aj cudzí kód, nie len vlastný. Algoritmické myslenie žiakov sa obohacuje aj tým, že žiak hľadá alternatívne riešenia alebo nájde alternatívne riešenie na problém v cudzom kóde. Aktivita code review vo vyučovaní na strednej škole prináša žiakom veľa pozitív, i keď oni ju často môžu chápať ako prácu navyše.

Pre učiteľa code review prináša iný pohľad na žiaka a možnosť lepšie sledovať jeho progres v programovaní a algoritmickom myslení.

Prináša však aj úskalia. Odovzdávanie projektu na niekoľko fáz, prerozdeľovanie projektov a ich preposielanie, je bez pomocného softvéru časovo náročnou prácou pre učiteľa. Taktiež precízna kontrola jednotlivých fáz projektov učiteľovi zlepší obraz o tom, ako sú na tom žiaci s programovaním a na čom treba zapracovať, avšak zaberie omnoho dlhší čas, než keby žiak odovzdával iba finálny projekt. Napriek tomu si myslíme, že úsilie, ktoré musí vynaložiť učiteľ aj žiaci pri zapájaní code review do vyučovania je dostatočne vyvážené benefitmi, ktoré takáto vzdelávacia stratégia prináša.

POĎAKOVANIE

Príspevok je súčasťou výskumu v projekte VEGA 1/0797/18.

BIBLIOGRAFICKÉ ODKAZY

[1] Purchase, H. and Hamer, J. Perspectives on peer-review: eight years of Aropä. *Assessment & Evaluation in Higher Education*, 43(3), pp.473-487, 2018.

[2] Dropčová, V. and Kubincová, Z. Team-based projects and peer assessment. it works!. In International Conference on Interactive Collaborative Learning, pp. 112-127. Springer, 2016

[3] Tseng, S.C. and Tsai, C.C. On-line peer assessment and the role of the peer feedback: A study of high school computer course. *Computers & Education*, 49(4), pp.1161-1174, 2007.

[4] Topping, K.J. Peer assessment between students in colleges and universities, *Review of Educational Research*, 68, pp. 249-276, 1998

[5] KUBINCOVÁ, Z. and M. Homola, “Code review in computer science courses: Take one,” in International Conference on Web-Based Learning, pp. 125–135, Springer, 2017

[6] Falchikov, N., Goldfinch, J. Student peer assessment in higher education: a meta-analysis comparing peer and teacher marks *Review of Educational Research*, 70, pp. 287-322, 2000

[7] POZOR NA CHYBY V KÓDE - CODE REVIEW, blog UI42, https://www.ui42.sk/blog/pozor-na-chyby-v-kode-code-review

[8] BOEHM, B. W. et al., Software engineering economics. Prentice-hall Englewood Cliffs (NJ), 1981.

[9] WIEGERS, K. E., Peer reviews in software: A practical guide. Addison-Wesley Boston, 2002.

[10] Kubincová, Z. and Csicsolová, I. Code Review in High School Programming. In *17th International Conference on Information Technology Based Higher Education and Training (ITHET),* pp. 1-4, IEEE, 2018.

[11] Csicsolová, I., Kubincová, Z. Code review v programovaní na strednej škole. In Sborník konference DidInfo 2018, pp. 71-76, Technická univerzita v Liberci, 2018.

Aktivity s BBC micro:bit podporujúce kolaboratívnu prácu žiakov ZŠ

Activities with BBC micro:bit Supporting the Collaborative Work of Elementary School Pupils

Martin Cápay
Katedra informatiky FPV UKF v Nitre
Trieda A. Hlinku 1
949 01, Nitra
Slovensko
mcapay@ukf.sk

Magdaléna Bellayová
Gymnázium, Golianova 68, Nitra
Golianova 68
949 01, Nitra
Slovensko
magdalena.bellayova@studentstc.sk

ABSTRACT
Programming microcomputers with sensors are in recent years, even with the advent of IoT, very popular. However, many programmable devices are not fully usable in computer science lessons, especially because of the need for complicated connecting of hardware elements. However, with the invention of the BBC micro:bit, microcomputer that was specially developed for educational purposes, the situation has changed significantly. Micro:bit has an integrated range of technologies in the basic set, making it a powerful tool in physical computing and STEM. In the article, we will focus on the use of BBC micro:bit in elementary school using the Makecode, block-oriented graphic language. The activities that we describe were made on the children's computer science club. Activities are designed so that children can first analyze the task, discuss it without computer, then program micro:bit and verify functionality of the program again without computer in some playful form. Cooperation is required during realization or during verification of functionality.

Keywords
micro:bit, programing, after school club, collaborative work

ABSTRAKT
Programovanie mikropočítačov so senzormi je v posledných rokoch, aj s príchodom IOT, veľmi populárne. Mnohé programovateľné zariadenia nie sú však úplne použiteľné na hodinách informatiky na ZŠ, a to najmä kvôli nutnosti komplikovaného zapojenia hardvérových prvkov. S príchodom mikropočítača BBC micro:bit, ktorý bol špeciálne vyvinutý pre edukačné účely, sa však situácia výrazne zmenila. Micro:bit má v základnej výbave integrované množstvo technológií, ktoré z neho robia silný nástroj v oblasti „physical computing“ a STEM. V článku sa zameriame na jeho využitie na základnej škole s použitím blokovo orientovaného grafického jazyka MakeCode. Aktivity, ktoré budeme popisovať boli realizované na detskom informatickom krúžku. Sú navrhnuté tak, aby deti mohli úlohu najskôr analyzovať, diskutovať o nej mimo počítača, následne na počítači naprogramovať správanie sa micro:bita a funkčnosť programu overiť opäť mimo počítača v nejakej hravej forme. Počas realizácie, sa od nich vyžaduje aj spolupráca.

Kľúčové slová
micro:bit, programovanie, detský krúžok, kolaboratívna práca

1 ÚVOD
Programovanie je oblasť informatiky, kde sa môžeme naučiť klásť otázky, hľadať vzory, či rôzne riešenia pre konkrétny problém a uvažovať o ich vhodnosti a efektívnosti. Skúsenosti z programovania môžu v budúcnosti poslúžiť aj pri neinformatických činnostiach, v ktorých je potrebné správne nastaviť rôzne podmienené, či opakujúce sa procesy.

Programovaním môžeme naučiť počítač chápať nové doposiaľ nepoznané príkazy, môžeme tvoriť. Programovanie môže byť praktické, väčšinou je to však oblasť abstraktná, málo hmatateľná a nanešťastie aj žiakmi považovaná za veľmi náročnú oblasť v informatike. Za programovaním často vidia len prácu s textovým jazykom s konzolovým výstupom. Žiaci sa veľmi ťažko motivujú k práci, a to najmä tí, ktorí svoju budúcnosť nevidia v IT sfére. V školskej informatike sa navyše v programovaní často riešia projekty matematického charakteru (hľadanie maxima, minima, priemer, sumy, matice a pod.) a to často len s náhodne generovanými dátami, čo je pre väčšinu žiakov nezáživné a ešte viac narušuje ich už naštrbený vzťah k prírodovedným predmetom. Demotivácia môže viesť až k tomu, že si žiaci memorujú programy, ktorým vôbec nerozumejú. Našou víziou je aby sa na hodinách programovania čo najviac pracovalo s reálnymi a vlastnými dátami získanými cez senzory z okolitého prostredia, aby sa prepájali projekty s vlastnými záujmami a záľubami, a aby programovanie dostalo reálny a hmatateľný význam a zmysel pre väčšinu žiakov. V našom článku by sme radi poukázali na overené aktivity, pomocou ktorých môžeme zmeniť pohľad na školské programovanie, ktoré je často vnímané ako programovanie matematicky orientovaných aplikácií, ktoré spracúvajú len množinu náhodne generovaných dát a zobrazujú informácie na obrazovke. Našim cieľom je priblížiť programovanie reálnemu svetu.

Cennou zručnosťou 21. storočia je nepochybne schopnosť spolupracovať na dosiahnutí nejakého spoločného cieľa. Preto by malo byť samozrejmé, aby aj školské aktivity boli nastavené tak, aby od žiakov vyžadovali istú formu spolupráce a tímovej práce v ľubovoľnej fáze riešenia problému.

2 Physical computing
V posledných rokoch s príchodom oblasti IOT – Internet of Things (Internet vecí), prichádza do životov ľudí veľmi veľa cez internet komunikujúcich zariadení. Chladničku, práčku, kúrenie, klimatizáciu, či žalúzie už môžeme ovládať na diaľku. Je preto viac než vhodné programovať spôsobom, ktorým komunikujeme s podnetmi z okolia. A práve preto tu je tzv. physical programming, kde sa programujú malé mikropočítače s rôznymi senzormi. Žiak, programátor, tam nepracuje iba so vstupmi a výstupmi na obrazovke, ale rieši reálne problémy. Programy už nie sú len texty

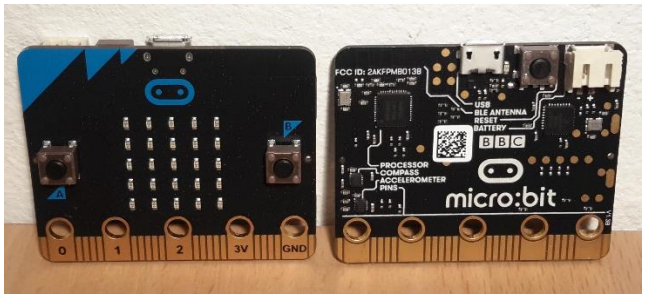
a čísla. Programy zaznamenávajú dáta získané zo svojho vlastného okolia (teplota, vlhkosť, pohyb) a vhodným spôsobom reagujú. Najkrajším aspektom využívania programovateľného hardvéru je možnosť prepájať ho s inými predmetmi a vytvárať medzipredmetové projekty, ktoré informatiku prepájajú napr. s umením, matematikou, či fyzikou. Žiakov tak vo výsledku môžeme motivovať k programovaniu prostredníctvom riešenia hmatateľných projektov s využitím propagovateľného hardvéru so senzormi (meteostanica, sledovacie zariadenie, zavlažovacia stanica, detektor pohybu a pod.). Študent má možnosť sa viac zamerať na realizáciu svojich nápadov ako na sledovanie obmedzení, ktoré často súvisia s tradičným prístupom v programovaní [1, 10].

Programovanie hardvéru prináša rôzne aspekty a výstupy vrátane kreativity [2], kognitívnej práce [3], pochopenia [4] a motivácie [5]. Veľkou výhodou hmatateľného programovania, je ucelený pohľad na počítač od hardvéru až po softvér, čo skutočne povzbudzuje kreativitu, podporuje učenie sa cez praktické činnosti a zapája takpovediac telo aj myseľ [6]. Dokonca je dokázané, že tento prístup pomáha získať sebavedomie v programovaní aj u dievčat [7] alebo u žiakov, ktorí nevidia svoju budúcnosť v IT sfére [8]. Programovanie hardvéru je viac než kódovanie, súvisí s reálnymi programovateľnými interaktívnymi systémami alebo zariadeniami. Je to prepojenie medzi fyzickým svetom a virtuálnym svetom počítača [9].

2.1 BBC micro:bit

V roku 2017 firma BBC zrealizovala projekt vývoja edukačného nástroja vhodného pre všetkých žiakov bez rozdielu a tak vznikol programovateľný hardvér micro:bit. Na Slovensku sa v poslednom čase rozbiehajú projekty so snahou o integrovanie tohto zariadenia do výučby (Učíme s Hardvérom, Programovanie ako ho nepoznáme, Klub informatikov s podporou eTeacher o.z.), čo svedčí o tom, že zameranie sa týmto smerom by mohlo mať budúcnosť.

Micro:bit (obr. 1) je malé, ľahko prenositeľné zariadenie, ktoré sa poľahky zmestí do dlane. Priamo na obvodovej doske je integrovaný procesor podobný tým, čo sa nachádza aj v mobilných zariadeniach, pamäť, displej s 25 LED diódami červenej farby, dve programovateľné tlačidlá, senzor intenzity svetla, akcelerometer, magnetometer, bluetooth anténa, micro USB port, port napájania a vstupno-výstupné piny. V skutočnosti teda ide o programovateľný mikrokontrolér. Pomocou krokosvoriek je možné pripojiť rozmanité prídavné zariadenia, ktoré rozširujú funkcionalitu základnej dosky o nespočetné možnosti. Je programovateľný cez blokový jazyk, MicroPython alebo Javascript.



Obrázok 1: BBC micro:bit

3 Detský krúžok

V článku sa zameriame na využitie zariadenia BBC micro:bit na základnej škole s využitím blokovo orientovaného grafického jazyka MakeCode. Aktivity, ktoré budeme popisovať boli realizované na detskom informatickom krúžku, ktorý je vedený v rámci projektu IT Akadémia – vzdelávanie pre 21. storočie na Katedre informatiky, PFV UKF v Nitre. Cieľom projektu je vytvorenie modelu vzdelávania a prípravy mladých ľudí pre aktuálne a perspektívne potreby vedomostnej spoločnosti a trhu práce so zameraním na informatiku a IKT [11]. Overovanie aktivít prebieha aj v spolupráci s projektom Programovanie ako ho nepoznáte, realizovaného pod záštitou programu Komenského inštitút. Cieľom programu je prostredníctvom praktických riešení realizovaných priamo učiteľmi ukázať možné cesty postupného prekonania problémov v školstve. V Komenského inštitúte majú učitelia možnosť získať vedomosti a zručnosti k realizácii praktickej školskej reformy vo svojej komunite. Pozitívnymi príkladmi budú ukazovať ako je možné školstvo postupne meniť [12]. Podporu na realizáciu aktivít sme získali aj z projektu Učíme s Hardvérom, ktorého misiou je vniesť do výučby informatiky na základných a stredných školách praktické zadania, ktoré žiakom ukážu zaujímavé možnosti využitia informatiky a vďaka ktorým sa žiaci sami začnú zaujímať o rozvíjanie svojich informatických zručností [14]. Krúžok je vedený aj ako jedna z aktivít v rámci programu Medzinárodná cena vojvodu z Edinburghu (skrátene DofE). DofE je rozvojový program, ktorý dáva mladým ľuďom šancu rozvinúť svoje schopnosti a naplniť svoj potenciál [15].

Krúžok je vedený v týždennom intervale v poobedňajších hodinách v priestoroch Katedry informatiky. Zúčastňujú sa ho žiaci 5. triedy ZŠ, pre ktorých bol BBC micro:bit novinkou. Všetci však už mali predchádzajúce skúsenosti s programovaním, konkrétne v jazyku Baltík a Scratch.

4 Aktivity s BBC micro:bit

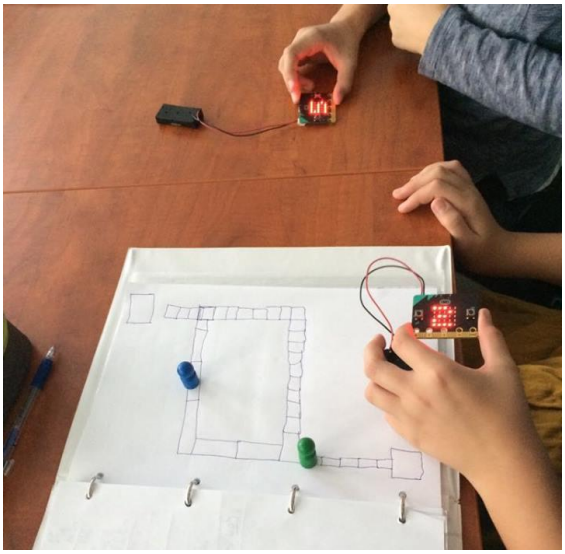
V tejto kapitole popíšeme niektoré aktivity, ktoré sme realizovali v období október 2018 až január 2019. Našim cieľom bolo, aby aktivity neboli len samostatnou prácou ale aby sa striedala činnosť žiakov pri počítači aj mimo neho, aby sa vyžadovala kolaboratívna práca a aby programovanie bolo aj zábavou. Aby ich činnosť bola v istom momente považovaná skôr za hru ako vzdelávaciu aktivitu.

4.1 Animácia

Animácia je úvodnou aktivitou určenou na oboznámenie sa so zobrazovaním obrázkov na LED displeji. Úlohou bolo simulovať signál SOS. Očakávali sme, že žiaci na riešenie použijú rozsvecovanie LED s krátkymi a dlhými pauzami, tak ako je to v morzeovke. Prišli však aj iné, neočakávané riešenia. Jedno z riešení bolo veľmi dlhé a podľa nás neefektívne tým, že žiak nepoužíval očakávané pauzy, ale opakovane zadával príkazy na rozsvietenie a zhasnutie. Program bol tak síce oveľa dlhší, ale v konečnom dôsledku pracoval, tak ako mal. Krúžku sa ako pozorovateľ zúčastnil aj študent VŠ, ktorý má záverečnú prácu na tému micro:bit na základnej škole. Tento študent za ten istý čas svoju verziu nedokázal priviesť do funkčného stavu. Dôvodom bolo hľadanie efektívnejšieho, ale príliš zložitého riešenia, ktoré bolo v konečnom dôsledku nefunkčné. Snažil sa totiž použiť vnárané cykly, ktoré do úvodnej hodiny jednoznačne nepatrili. Ďalšie riešenie je pekným príkladom toho, ako žiaci vedia využiť nepresnosť zadania úlohy. Namiesto použitia morzeovky žiačka len zobrazila na obrazovke písmenká S, O, S.

4.2 Náhodné číslo

Na vysvetlenie náhody, alebo náhodne generovanej hodnoty, sme využili princíp hracej kocky. Deti museli sami prísť na to, že čísla padajúce na kocke sú náhodné a nevedia vopred určiť, čo padne ako ďalšie. Po tom, čo mali možnosť vidieť princíp reálnej hracej kocky, mali za úlohu naprogramovať si vlastnú micro:biťácku verziu. Tu sa museli naučiť používať generátor náhodných čísel a uvedomiť si v akom intervale sa majú čísla generovať. Prišli aj na to, že potrebujú nejaký impulz, spúšťač, na opakované generovanie. V tomto prípade si ako vstupnú udalosť zvolili potrasenie (shake). Keby tak neurobili, museli by čísla generovať stlačením tlačidla reset. Po nahratí programu do micro:bita opustili počítače a nakreslili si vlastnú dráhu, na ktorej sa mohli zahrať hru a využiť tak svoje naprogramované kocky (obr. 2).



Obrázok 2: Stolová hra s micro:biťáckymi hracími kockami

Na náhodné čísla sme nadviazali aktivitou **Salute**, ktorá priniesla aj prepojenie s matematikou. Princípom hry je, že dvaja oproti sebe stojaci žiaci držia nad hlavou svoje micro:bity, na ktorých sa im vygenerovalo náhodné číslo. Tretí žiak, ktorý vidí obe čísla, povie súčet týchto dvoch čísel. Ten žiak, ktorý na základe tohto súčtu dokázal ako prvý uhádnuť (vypočítať) svoje číslo, ktoré vlastne nevidí, vyhráva. V tejto aktivite museli žiaci zmeniť interval náhodne generovaných čísel a vstupnú udalosť.

4.3 Kameň, papier, nožnice (KPN)

Každý pozná hru kameň, papier, nožnice. V tejto aktivite mali deti možnosť naučiť túto hru aj svojich micro:bitov. Po predošlých aktivitách rýchlo zistili, že ide opäť o náhodu, ale nevedeli, ako vygenerovať náhodný obrázok. Doteraz pracovali iba s náhodnými číslami. Žiaci museli prísť na to, že aj v tomto prípade je potrebné vygenerovať náhodné číslo, na základe ktorého sa zobrazí obrázok. Každému obrázku (kameň, papier alebo nožnice) museli najskôr priradiť nejaké číslo. Keďže s vygenerovaným číslom bolo potrebné pracovať niekoľkokrát, museli si ho uložiť do premennej. Túto premennú následne porovnávali. Pri samotnom porovnávaní použili tri podmienky (neúplné vetvenia), pre každú možnosť jednu. Po úspešnom naprogramovaní sa mohli so svojimi micro:bitmi navzájom zahrať. Aby sme čo najlepšie vystihli princíp klasickej verzie hry, opäť sme ako vstupnú udalosť zvolili potrasenie.

Premennú deti použili aj v situácii, kedy potrebovali doprogramovať počítadlo víťazstiev. Tu sa naučili zvyšovať premennú, zobrazovať ju a aj ju v prípade potreby vynulovať.

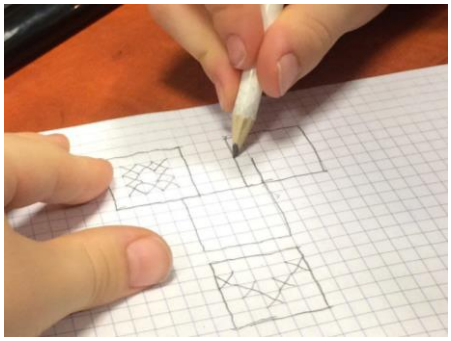
4.4 Rádio YES NO

Táto aktivita bola zameraná na vzájomné posielanie správ a prácu s kanálmi/skupinami. Využité tu boli aj poznatky z predošlých aktivít. Deti sa rozdelili do dvojíc, kde si určili svoj vlastný komunikačný kanál/skupinu, cez ktorý si budú posielať „tajné“ správy. Ich úlohou bolo, aby si po stlačení tlačidla A poslali veselého smajlíka a po stlačení tlačidla B zase smutného smajlíka. Deti už vedeli, že posielať sa dajú len reťazce alebo čísla, tak museli prísť na to, že transformácia odoslaného čísla na obrázok sa bude diať až u príjemcu, nie na strane odosielateľa. Príjemca tak musel porovnať prijaté číslo, a teda znova použiť podmienky, ktoré poznali z hry KPN.

Žiaci si najskôr medzi sebou vo dvojici odpovedali na jednoduché otázky typu Áno/Nie, pri čom veselý smajlík predstavoval odpoveď Áno a smutný smajlík Nie. Takto žiaci poznali odpoveď svojho spolužiaka aj bez toho, aby ju vyslovil nahlas. Keď už si skúsili odpovedanie, mohla začať hra, kde museli hľadať svoju dvojičku, teda micro:bita, ktorý je nastavený na rovnaký kanál/skupinu. Všetky micro:bity sa dali na jednu kopu, zamiešali sa a potom náhodne rozdali žiakom. V tom momente nikto nevedel, kto im vlastne bude odpovedať, teda koho odpoveď sa zobrazí na displeji. Na otázku typu Áno/Nie odpovedali vždy všetci. Počas hry sa žiaci museli pýtať vhodné otázky, vďaka ktorým mohli ľahšie identifikovať spárovaného micro:bita.

4.5 Kolaboratívny smajlík

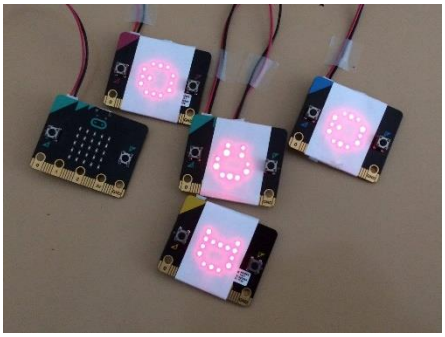
V tejto aktivite mali žiaci navrhnúť vlastného smajíka zloženého zo štyroch micro:bitov – jeden zobrazoval ústa, druhý nos, tretí ľavé oko a štvrtý pravé oko. Žiaci mali najskôr na štvorčekový papier zaznačiť navrhovaného smajlíka (obr. 3).



Obrázok 3: Návrh smajlíka

Následne sa striedali pri štyroch počítačoch, kde postupne prepisovali svoj návrh do programu a to tak, že v každom počítači zapisovali program pre rozdielneho micro:bita. Každý micro:bit fungoval na rozdielnom kanáli/skupine. Smajlíci mali priradené poradové číslo na základe, ktorého sa program potom vetvil. Keď žiaci zapísali všetky svoje návrhy do spoločného programu, spojili všetkých micro:bitov do tvaru smajlíka. V tejto aktivite sme mali pripravený diaľkový ovládač, ktorý rozposielal náhodne vygenerované číslo na všetkých kanáloch/skupinách. Keď micro:bity zaregistrovali, že im bolo zaslané číslo, tak sa prepli na

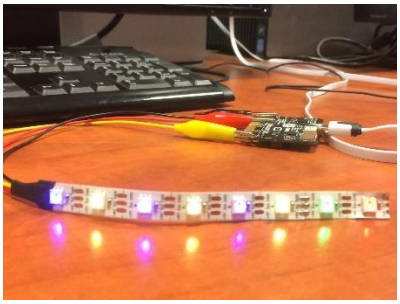
požadovaného kolaboratívneho smajlíka. Žiaci tak celý čas pracovali na jednom spoločnom výstupe (obr. 4).



Obrázok 4: Kolaboratívny smajlík

4.6 Vianočné osvetlenie na diaľkove ovládanie

V tejto aktivite sme k micro:bitom pomocou krokosvoriek pripojili LED pásik NeoPixel s ôsmimi RGB svetielkami (obr. 5). Prácu s LED svetielkami sme spojili s blížiacimi sa Vianocami a deti si mali možnosť naprogramovať vlastné vianočné osvetlenie na stromček (obr. 6), ktorý si spoločne postavili a ozdobili. Tu pracovali v podstate s jednorozmerným polom. Najprv sa učili rozsvecovať všetky svetielka naraz, potom postupne a neskôr každému svetielku priradili nejakú farbu a naučili sa ich rotovať.



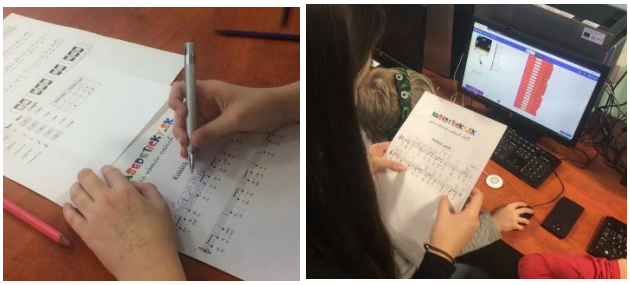
Obrázok 5: Rozsvecovanie LED pásika NeoPixel

Obrázok 6: Vianočné micro:bit svetielka

Pôvodným zámerom bolo rozsvietiť všetky naprogramované LED pásiky od každého žiaka naraz pomocou diaľkového ovládača. Ovládanie sa nám na krúžku nepodarilo dokončiť. Aktivitu sme však so zaujímavosti zaradili aj pre študentov na gymnáziu, kde to fungovalo a vyvolalo veľmi pozitívne reakcie.

4.7 Kódovanie hudby

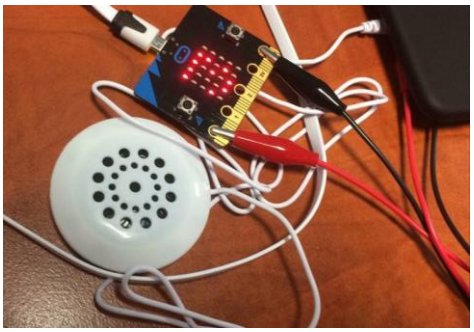
Micro:bit podporuje aj prehrávanie zvukov na rôznych frekvenciách, čím sa dá zložiť pesnička. Potrebujeme však vedieť ich notový zápis. Ale čo ak noty nepoznáte? My sme sa to deťom snažili priblížiť tak, aby si aj bez znalostí nôt vedeli naprogramovať aspoň jednoduchú pesničku. Vo dvojiciach dostali pieseň v notovom zápise. Ich prvou úlohou bolo, ešte bez počítača previesť notový zápis do zápisu, ktorému by rozumel micro:bit. Dostali vysvetlivky vytlačené na papieroch, aby vedeli, ktorá nota je ktorá a koľko má dôb, a mohli začať dekódovať. Keď mali notový zápis dekódovaný, už to stačilo len preniesť do skutočných príkazov. Takto si deti dokázali naprogramovať jednoduché pesničky ako Kohútik Jarabý, Sedemdesiat sukien mala, alebo Po valašky od zeme, a to aj bez toho, aby niekedy predtým videli noty (obr. 7). Nie všetkým sa to podarilo úspešne zvládnuť na prvýkrát. Spoločne sme však mohli sledovať ako sa prehrávajú jednotlivé noty, identifikovať chybné tóny a opraviť ich. Žiaci mali možnosť zmeniť aj tempo piesne.



Obrázok 7: Prekódovanie notového zápisu do počítačového programu

4.8 JukeBox

Vytvoriť vlastnú pesničku z nôt a prehrať ju už žiaci vedeli. Ale čo tak vytvoriť si vlastný JukeBox a mať možnosť si vyberať z viacerých pesničiek, či už vlastných, alebo ponúkaných programovacím prostredím? Aby sme pieseň počuli, museli sme k micro:bitovi cez krokosvorky pripojiť reproduktor (obr. 8).



Obrázok 8: JukeBox s aktuálnou piesňou číslo 3

Najskôr žiaci mimo počítača v diskusii popísali ako funguje skutočný JukeBox, a tak zistili, čo vlastne potrebujeme. Museli mať zoznam pesničiek, ktorým priradili vlastné poradové čísla,

spôsob výberu konkrétnej piesne a spúšťací mechanizmus. Následne museli popísané funkcionality pretransformovať do programu. Zadefinovali si premennú určujúcu poradie piesne, ktorú potom pomocou tlačidiel zmenšovali alebo zväčšovali.

Stlačením oboch tlačidiel súčasne sa spustila pieseň z ich pripraveného zoznamu s číslom zobrazeným na displeji.

5 ZHRNUTIE

Počas realizácie sme so žiakmi precvičili rôzne programátorské konštrukty – sekvencia, vetvenie, definovanie premennej, jednorozmerné pole, cyklus, obslužné udalosti, vysvetlili si princíp fungovania prijímača a vysielača. Naučili sme sa pripájať zariadenie cez microUSB port ako aj sťahovať súbor. Žiaci k aktivitám pristupovali so záujmom. Programovacie prostredie bolo intuitívne. Prostredie sa dá prepnúť do Slovenčiny a tak boli príkazy jednoznačné aj pre 10 ročných žiakov.

6 ZÁVER

Všetky popísané aktivity sme zopakovali aj na Gymnáziu, kde boli transformované do jazyka MicroPython. Aktivity boli po zverejnení výsledkov na školskej sociálnej sieti vyžadované aj od žiakov, ktorý pôvodne neboli súčasťou práve realizovaného projektu. Zaevidovali sme zvýšenie motivácie na hodinách informatiky a to vo všetkých ročníkoch, kde bolo experimentálne použité programovanie s hardvérom. Na základe praktických skúseností môžeme vyhlásiť, že implementácia mikropočítača micro:bit do školského kurikula má veľký význam.

POĎAKOVANIE

Ďakujeme za podporu projektom IT Akadémia, Učíme s Hardvérom, Komenského inštitútu a DofE.

BIBLIOGRAFICKÉ ODKAZY

[1] SENTANCE S., WAITE J., YEOMANS L., and MACLEOD E. *Teaching with physical computing devices: the BBC micro:bit initiative*, In Proceedings of the 12th Workshop on Primary and Secondary Computing Education (WiPSCE '17), Erik Barendsen and Peter Hubwieser (Eds.). ACM, 2017, New York, NY, USA, 87-96.

[2] SENTANCE S., SCHWIDERSKI-GROSCHE S. 2012. *Challenge and creativity: using .NET gadgeteer in schools*. In Proceedings of the 7th Workshop in Primary and Secondary Computing Education. ACM, 90–100. http://dl.acm.org/citation. cfm?id=2481473

[3] DESPORTES K., ANUPAM A., PATHAK N., DISALVO B. 2016. *BitBlox: A Redesign of the Breadboard*. In Proceedings of the The 15th International Conference on Interaction Design and Children. ACM, 255–261.

[4] KAFAI J., LEE E., SEARLE K., FIELDS D., KAPLAN E., and LUI D.. 2014. *A crafts-oriented approach to computing in high school: Introducing computational concepts, practices, and perspectives with electronic textiles*. ACM Transactions on Computing Education (TOCE) 14, 1 (2014), 1.

[5] KALOTI-HALLAK F., ARMONI M., and BEN-ARI, M. M.. 2015. *Students' attitudes and motivation during robotics activities*. In Proceedings of the Workshop in Primary and Secondary Computing Education. ACM, 102–110.

[6] JIN K. H., HAYNIE K. and KEARNS G., *Teaching Elementary Students Programming in a Physical Computing Classroom*. In Proceedings of the 17th Annual Conference on Information Technology Education (SIGITE '16). ACM, 2016, New York, NY, USA, 85-90.

[7] RUBIO M. A., ROMERO-ZALIZ R., MAˆNOSO C., and de MADRID A. P. *Closing the gender gap in an introductory programming course*. Computers and Education, 82(C):409–420, 2015.

[8] CÁPAY M., KLIMOVÁ N. *Engage Your Students via Physical Computing!*, Educon 2019. In Press.

[9] SCHULZ S. and PINKWART N. *Physical Computing in STEM Education*. In Proceedings of the Workshop in Primary and Secondary Computing Education (WiPSCE '15). ACM, 2015 New York, NY, USA, 134-135.

[10] SENTANCE S., WAITE J., HODGES S., MACLEOD E., YEOMANS LE. *Creating Cool Stuff - Pupils' experience of the BBC micro:bit*. In Proceedings of the 48th ACM Technical Symposium on Computer Science Education: SIGCSE 2017. 2017.

[11] IT Akadémia – vzdelávanie pre 21. storočie http://itakademia.sk/zakladne-informacie/

[12] Komenského inštitút http://www.komenskehoinstitut.sk/

[13] Učíme s Hardvérom https://www.ucimeshardverom.sk/ciele/

[14] Medzinárodná cena vojvodu z Edinburghu ttps://www.dofe.sk/

Dokáže programovanie hardvéru zaujať študentov?
Is Physical Computing Able to Engage Students?

Martin Cápay
Univerzita Konštantína Filozofa v Nitre
Tr. A. Hlinku 1
949 74 Nitra
Slovensko
mcapay@ukf.sk

Nika Klimová
Univerzita Konštantína Filozofa v Nitre
Tr. A. Hlinku 1
949 74 Nitra
Slovensko
nika.klimova@ukf.sk

ABSTRACT
Physical computing covers the design and realization of interactive objects, installations and allows students to develop concrete, tangible products of the real world. This paper describes the basics and benefits of integrating physical computing with block-based language MakeCode or Micropython using microcontroller BBC micro:bit. It provides cross-curricular hands-on learning opportunities. We used the microcontroller during the lessons with pupils at different age. After the first use of the programming with hardware concept we noticed the rise of the motivation to learn programming even among the 'non-informatics' pupils and among the girls.. The great benefit of "getting physical" is a holistic view of computer science which encourages creativity, promotes learning by doing and engaging the whole mind and body learning.

Keywords
Physical computing. Programming. Sensors. BBC micro:bit.

ABSTRAKT
Programovanie s hardvérom zahŕňa dizajn a realizáciu interaktívnych objektov a umožňuje študentom vytvárať konkrétne, hmatateľné produkty skutočného sveta. Článok opisuje základy a benefity integrácie programovania mikrokontroléra BBC micro:bit cez blokový programovací jazyk MakeCode alebo textový jazyk Micropython. Takýto prístup umožňuje prepájať medzipredmetové vzťahy. My sme mali možnosť integrovať mikrokontrolér do formálnej výučby so žiakmi rôznych vekových skupín. Po prvom použití konceptu programovania s hardvérom sme si všimli nárast motivácie učiť sa programovanie aj medzi žiakmi, ktorí nemajú v pláne maturovať z predmetu informatika, ale aj medzi dievčatami. Veľkou výhodou tejto metódy učenia sa je holistické nazeranie na informatiku, ktoré podporuje kreativitu, učenie sa robením a angažovaním celej mysle a tela.

Kľúčové slová
Programovanie s hardvérom. Programovanie. Senzory. BBC micro:bit.

1 ÚVOD

Informatika pokrýva rôzne témy ako napríklad počítačová architektúra, siete, softvérové inžinierstvo, programovanie ale aj návrh programu a uchovávanie dát, či manipuláciu s nimi. Programovanie je oblasť informatiky, v ktorej si kladieme otázky, hľadáme logické podstaty a vzory, nachádzame riešenia, alebo učíme počítač novým príkazom. Programovanie môže byť kreatívne a praktické, ale hlavne je to oblasť abstraktná, menej hmatateľná, avšak v rámci súkromného sektora považovaná za veľmi žiadanú oblasť v rámci informatiky. Žiaci sa dnes ťažko namotivujú pracovať, hlavne tí, ktorí nevidia svoju budúcnosť v IT sfére. Navyše v školskej informatike, projekty s matematickým charakterom (hľadaj maximum, minimum, priemer, súčet, matice atď.) sú často výsledkami s náhodne generovanými dátami, ktoré sú na míle vzdialené pre väčšinu žiakov alebo môžu viac narušiť ich už aj tak chladný vzťah k prírodovedným predmetom. Takýto prístup môže viesť k memorovaniu programov, ktorým vôbec nerozumejú.

2 Programovanie s hardvérom

Súhlasíme s ideou Red Burnsa, ktorý vnímal informatiku nie len ako predmet pre technofilov, ale ako predmet pre všetkých [1]. Informatika je viac než len učenie programovania, odkazuje na interaktívne fyzické systémy alebo zariadenia, ktoré môžu byť programované cez softvér. Je to výsledok kreatívneho procesu vytvárania hmatateľných interaktívnych objektov alebo systémov cez programovateľný hardvér [7]. Informatika zahrňuje tvorbu a realizáciu interaktívnych objektov a ich inštaláciu. Umožňuje študentom vyvíjať konkrétne, hmatateľné produkty reálneho sveta, ktorá vzrastá z predstavy učiaceho sa [22]. Zatiaľ čo svetový trh práce pokračuje v masívnej zmene, u informaticky zameraných profesiách je predpoklad, že v budúcnosti budú rásť dvakrát rýchlejšie než hociktoré iné profesie [2]. Je tu mnoho dôvodov, prečo je dôležité integrovať informatiku do medzipredmetových vzťahov [2] a prečo je potrebné budovať dostupné bádateľské a projektovo-zamerané aktivity k vizualizácii dát skrz vedu, technológiu, inžinierstvo a matematické (STEM) kurikulum [3][4]. Informatika vyzerá byť vhodným kandidátom „ako vytvárať konverzáciu medzi fyzickým svetom a virtuálnym svetom počítača"[18].

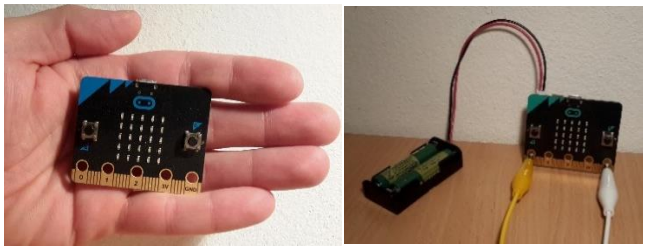
Najhlbšia úroveň učenia, podľa Bloomovej taxonómie, sú aktivity, keď študenti konštruujú, vytvárajú a pracujú s učebnými pomôckami. Veľkou výhodou zapojenia fyzického zariadenia do výučby je holistický pohľad na informatiku (skrz hardvér a softvér), ktorý podporuje kreativitu projektov, vyzdvihuje učenie robením a angažovanie celého učenia sa (mysle a tela) [2]. Programovanie s hardvérom má rôznu škálu aspektov a výsledkov vrátane kreativity [9], kognitívnej väzby [10], študentského vnímania [11] a motivácie [12]. Bolo dokonca dokázane, že prístup, ktorý zahŕňa programovanie s hardvérom, pomáha študentkám získať sebavedomie v programovaní [6]. Hlavne pri používaní blokových programovacích jazykoch môžeme vytvárať mobilné aplikácie jednoduchým spôsobom, čím sa môže zvýšiť podporu záujmu o prírodné vedy [15][16][17]. Programovanie s hardvérom, integrované s blokovým prostredím, sa stalo populárnym hlavne pri uvádzaní programovania mladším študentom [19]. Z pohľadu študenta môže byť prijaté oveľa viac pozitívne, než „tradičné" vyučovanie, pretože sa viac sústredí na podstatu než na obmedzenia [13]; študenti ocenia budovanie reálnych, hmatateľných zariadení a hovoria, že programovanie s hardvérom a jeho platformy

stimulujú ich kreativitu [14]. Na druhej strane, väčšina z praktických projektov sú síce zábavné, umelecké a pútavé, ale nemusia byť veľmi nápomocné pri demonštrovaní základov, ako napríklad premenné, podmienky alebo komplexnejšie riadiace štruktúry [20]. Naším cieľom je identifikovať sadu projektov, ktoré nie sú len pútavé a zábavné, ale aj efektívne pri demonštrovaní abstraktných programovacích konceptov pre začiatočníkov.

3 BBC micro:bit

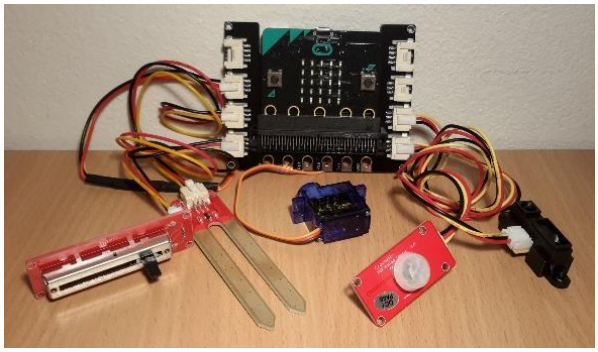
Programovanie s hardvérom sa môže stať populárnym prístupom pre učenie žiakov základných škôl. Dnes máme možnosť veľkého výberu reálnych programovateľných hardvérov, napr. programovateľné roboty (LEGO Mindstorm robots, Bee-Bot), programovateľné vstupné alebo výstupné zariadenia (napr. Makey Makey), elektronické konštrukčné sady (napr. LittleBits), mikrokontroléry (napr. Arduino, micro:bit, Crumble), samostatné minipočítače (napr. Raspberry Pi) alebo iné hmatateľné programovateľné zariadenia [2] [7]. My sa budeme koncentrovať na používanie BBC micro:bit-u vo vyučovaní..

BBC micro:bit (obr. 1) je malé programovateľné zariadenie [7], programovateľný minipočítač, doska plošných spojov poháňaná malými batériami, špeciálne navrhnutá pre učenie mladších žiakov [8]. Je to relatívne nové, nízko-rozpočtové zariadenie, ktoré bolo v roku 2016 distribuované každému 11-12-ročnému žiakovi vo Veľkej Británii [7]. Hlavný mikrokontrolér na micro:bit-e je ARM M0 čip, podobný tým v mobilných zariadeniach.



Obrázok 1: Mikrokontrolér BBC micro:bit

Micro:bit má 16 Kb RAM a 128 Kb flash pamäte používanej na ukladanie sťahovateľných programov do zariadenia, vbudovaný displej s 25 LED diódami, dvoma programovateľnými tlačidlami, akcelerometer, teplomer, senzor na meranie svetla a digitálny kompas, Bluetooth pre bezdrôtovú komunikáciu, USB pripojenie, a toto všetko sa vojde do dlane ruky [7][8][13][21]. Obsahuje aj päť vstupných a výstupných konektorov na priame pripájanie BBC micro:bitu do iných zariadení pomocou krokodílích svoriek a možnosť posielania príkazov do zariadenia ako aj zo zariadenia (obr. 2). Môže byť programovaný cez PC, notebook alebo tablet, cez blokový programovací jazyk, MicroPython alebo JavaScript.



Obrázok 2: BBC micro:bit a senzory

BBC micro:bit sa môže prepojiť s inými minipočítačmi, čím sa zvýši potencionálna aplikácia a rozšíria sa tak jeho možnosti, napr. využitím bluetooth technológie v rámci BBC micro:bit na vytvorenie malého prenosného diaľkového ovládača pre hry alebo vozidlá vybudované na Raspberry Pi, alebo Arduino doske [21].

Bolo zistené, že micro:bit pomáha študentom pracovať kreatívne, novosť fyzickej povahy zariadenia je dobrou motiváciou pre triedy. Navyše sa tvrdí, že hmatateľnosť zariadenia je kľúčový element pri stimulácii záujmu a podpory pochopenia. [13]

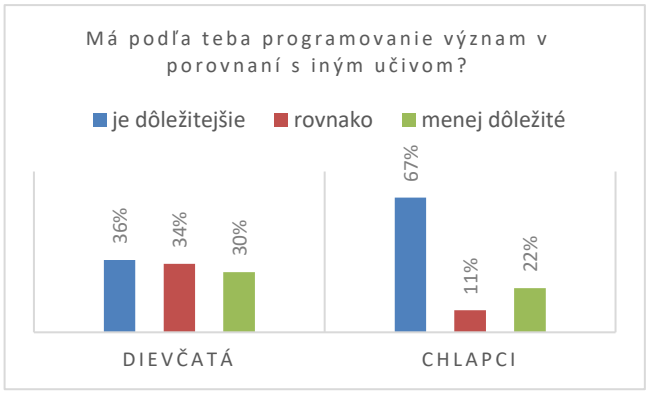
Počas nášho posledného Informatického letného tábora v roku 2018 sme pracovali s mikrokontrolérmi. Záujemci boli nadšení a dokonca sa spytovali na možnosti získania takéhoto programovateľného hardvéru. A tak sme sa rozhodli so začatím používanie mikrokontroléra aj v reálnom vyučovaní. Článok opisuje skúsenosti z realizácie aktivít v rámci kampane Meet and Code (https://www.meet-and-code.org/).

3.1 Metodológia

Náš výskum sa začal v septembri 2018, na začiatku školského roka 2018/2019, a stále trvá. Rozhodli sme sa zahrnúť 2.–5. ročník z Gymnázia, Golianova 68, Nitra, Slovensko. Ako metódu na meranie dát sme použili dotazníky pred začatím a po ukončení realizácie aktivít, kde študenti odpovedali na sadu takmer identických otázok. Výsledky sa zameriavajú na študentov spolu, ale takisto sme ich rozdelili na chlapcov a dievčatá a skúmali, či nájdeme medzi nimi rozdiely.

Dotazník pre realizáciu výskumu bol vyplnený 53 stredoškolskými študentami (2.- 5. ročník). Z nich bolo 9 chlapcov (17%) a 44 dievčat (83%). Na otázku, či niekedy programovali, všetci študenti odpovedali, že áno. Keďže Štátny vzdelávací program obsahuje časť Algoritmické riešenie problémov, nebola odpoveď nijak prekvapivá. Časti priamo sa dotýkajúce programovania sú obsiahnuté nielen v stredoškolskom ŠVP, ale v ŠVP pre informatiku prvého a druhého stupňa základných škôl.

Myslíme si, že študenti vidia programovanie ako opodstatnenú zručnosť, pretože 90% (46 študentov) súhlasilo, že programovanie je dôležité. 2 dievčatá (2%) odpovedali nie, zatiaľ čo žiaden z chlapcov nezvolil túto odpoveď.



Obrázok 3: Ako veľmi je programovanie dôležité?

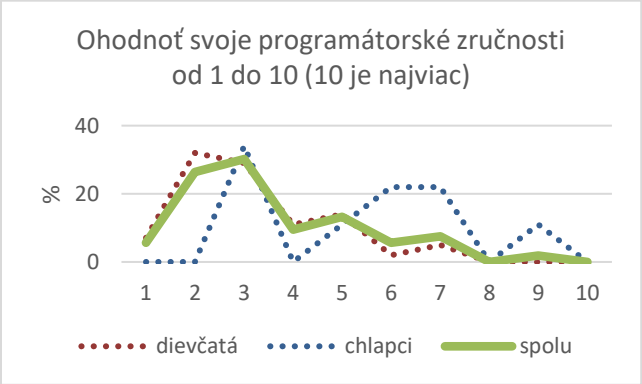
V dotazníku bola možnosť dobrovoľne zodpovedať aj na otázky s voľnou odpoveďou, vlastnými slovami. Niektorí to využili - dievčatá spomenuli slová ako zlepšenie (2x), budúcnosť (4x), pomoc (2x). 8 dievčat odpovedalo, že „programovanie je dôležité pre svet, avšak nie pre ne samotné“. Chlapci taktiež zvolili dobrovoľnú časť a vyjadrili sa. Spomenuli, že „je to dôležité pre moju potenciálnu kariéru“, „v 21. storočí sa všetko riadi

počítačmi“, „uľahčenie práce“ (2x), „hry by neexistovali“, alebo „pre vedecký pokrok“. Na záver môžeme rozdeliť ich odpovede na 2 pohľady - na programovanie, ktoré je dôležité pre svet všeobecne; a na programovanie, ktoré je dôležité pre nich samotných.

V porovnaní s inými predmetmi, 72% študentov si myslí, že programovanie je tak isto alebo viac dôležitejšie než iné učivo. Informatika je v tomto prípade 1 až 2-krát týždenne.

Keď sa hovorí o dievčatách a chlapcoch (obr. 3), väčšie percento chlapcov než dievčat si myslí, že programovanie je viac dôležité než iné učivo.

Požiadali sme ich, nech subjektívne ohodnotia svoje programovacie zručnosti na stupnici od 1 do 10, kde 10 je najviac (obr. 4). Často svoje schopnosti hodnotili na stupnici 2 až 3, čo bolo pre nás prekvapujúce. Najviac dievčat volilo číslo 2, pričom chlapci volili najviac 3. Môžeme badať rozdiel medzi chlapcami a dievčatami, pokiaľ ide o opísanie svojich programátorských zručností. Dievčatá vyberali medzi množinou 1 až 7, zatiaľ čo chlapci množinu 3 až 9. Priemer dievčat je 3,2, zatiaľ čo u chlapcov je 5,4. Môže to byť spôsobené tým, že chlapci trávia voľný čas programovaním viac než dievčatá, alebo veria v sami seba viac ako dievčatá.



Obrázok 4: Ohodnoť svoje programátorské zručnosti

Na otázku „Čo je to BBC micro:bit?“ sa odpovedalo otvorenou vetou. Skúmali sme výskyt slov alebo koreňov slova. *Program* sa vyskytol 18-krát (34 %), *zariadenia/HW/PC* 14-krát (26 %), *neviem* 19-krát (36 %).

V dotazníku sme zahrnuli aj otázku, či niekedy programovali micro:bit – 58 % (31 študentov) odpovedalo áno. Dôvodom, prečo sa toľko študentov takto vyjadrilo, je, že druháci a piataci absolvovali micro:bit workshop a niektorí ho už zažili počas výučby ešte pred realizáciou dotazníkového prieskumu.

Na otázku „Tešíte sa na programovanie micro:bita?“ sa tiež odpovedalo na škále 1 – 10. 20 % odpovedalo 1 – vôbec sa neteším, 28 % označilo 10 – veľmi sa teším. Myslíme si, že ak študenti nevedeli, čo je to micro:bit, nemohli mať ani žiadne veľké očakávania. Priemer bol u dievčat 5,2 a u chlapcov 6,5.

Opýtali sme sa ich, čo by radi robili s micro:bitom a najčastejšou odpoveďou bolo programovať.

Vyučujúci taktiež vypĺňal dotazník. Mal sa zamerať na svoje očakávania: očakával, že motivácia študentov vzrastie, najmä u tých, ktorí sa neplánujú ďalej uberať informatickým smerom.

3.2 Mikrokontrolér v praxi

Počas realizácie sme používali BBC micro:bit pri žiakoch rôznych vekových skupín: základná škola (5. ročník), stredná škola (2. – 5. ročník). Rozhodli sme sa pre 2 odlišné princípy programovania:

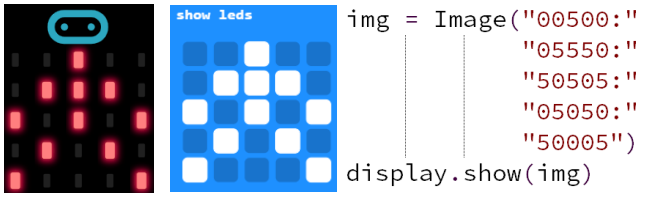
- **Microsoft MakeCode** je robustné vizuálne programovacie prostredie, plne vybavený textový editor s možnosťou prepínania medzi jednotlivými režimami.
- **MicroPython** je efektívnou implementáciou programovacieho jazyka Python 3, ktorý zahŕňa malú podmnožinu štandardnej knižnice Python a je optimalizovaný pre beh na mikrokontroléroch a v obmedzených prostrediach.

Naše aktivity v rámci projektu boli realizované medzi stredoškolskými študentmi a 1 aktivita bola pripravená pre žiakov ZŠ. Všetci stredoško participanti mali predchádzajúce skúsenosti s programovacím jazykom Pascal, takže prezentované koncepty pre nich neboli neznáme. Naším cieľom bolo identifikovať sadu projektov, ktoré nie sú len motivujúce a zábavné, ale takisto efektívne v demonštrovaní abstraktných programovacích konceptov pre začiatočníkov.

3.2.1 *Animácie*

Micro:bit má priamo integrovaný výstupný displej, matricu 5 x 5 LED-iek. Môžeme ho použiť na vykresľovanie jednoduchých obrázkov, ktoré by mohli byť použité ako časti krátkeho animovaného filmu. Dá sa na to vytvoriť príbeh, aktivita, napríklad: *Nakreslíme malú postavu, ktorá bude na telesnej výchove. Poďme sa hýbať!*

Študenti si potrebujú navrhnúť pózu na obmedzenej ploche (obr. 5), usporiadať a nastaviť prechodovú rýchlosť. Musia zvážiť, či všetky pózy (ako skok, skok, kop) majú rovnaký čas.



Obrázok 5: Projekt Animácia - Figúrka zobrazená na displeji; príkaz v MakeCode a v MicroPython

Princíp: Hlavným cieľom je ukázať sekvenciu príkazov. Študent potrebuje poznať a použiť pauzu na „zamrznutie“ danej časti (fázy, záberu) animácie. Musí rozmýšľať o počte opakovaní pózy (raz, n-krát, navždy) ktorá ho vedie k premýšľaniu o cykloch.

3.2.2 *Hod kockou*

Nasledujúci opis projektu je simulácia hodu kocky. Študenti museli prísť na to, že táto úloha sa od ostatných odlišuje v náhodnom elemente.

Obrázok 6: Projekt Hod kockou - číslo zobrazené na displeji; príkaz v MakeCode

Nemôžeme stále ukazovať rovnaké obrazy, napriek tomu vieme, že máme len 6 možností výberu, ale nedokážeme predvídať, ktorá z nich sa zobrazí (obr. 6).

Študenti prichádzajú na to, že ak chcú nové náhodné číslo, musia reštartovať celý program. Hoci micro:bit má resetovacie tlačidlo, jeho umiestnenie vzadu nie je pohodlné. Najvhodnejším spôsobom je uviesť do interakcie predné tlačidlá. V MicroPythone musíme akceptovať, že hardvér potrebuje neustále kontrolovať, či došlo k stlačeniu tlačidla. Potrebujeme teda v nekonečnom cykle zisťovať, či sa vykonala udalosť stlačenia tlačidla. Pri blokovom jazyku nám táto potreba odpadá, lebo udalosť „on button pressed“ je natívne nastavená ako nekonečná slučka.

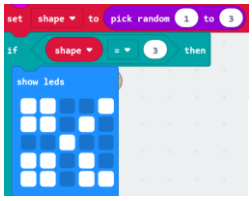
```
while True:
    if button_a.was_pressed():
        display.show(random.randint(0, 5))
```

Obrázok 7: Projekt hod kockou s použitím tlačidla. Kód v MakeCode aj MicroPython

Princíp: Hlavným cieľom je ukázať princíp generovania výberu náhodných čísel (obr. 7). Študenti potrebujú rozhranie (tlačidlo) na komunikáciu s používateľom. Boli vysvetlené princípy akcie-reakcie alebo udalosti-obslužnej metódy udalosti.

3.2.3 *Kameň, papier, nožnice*

Všetky kroky viedli k finálnemu projektu Kameň, papier, nožnice, čo je veľmi známa hra zvyčajne hraná medzi dvoma ľuďmi, v ktorej každý hráč súčasne tvorí jeden z troch tvarov s natiahnutou rukou. Predpokladáme, že každý túto hru pozná. Ako simulovať túto ruku na fyzickom zariadení?



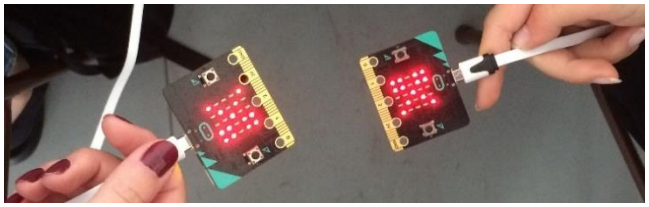

```
if accelerometer.was_gesture('shake'):
    x = random.randint(1, 3)
    if (x == 3):
        display.show(Image.SCISSORS)
```

Obrázok 8: Projekt Kameň, papier, nožnice s použitím podmieneného príkazu

Projekt je takmer rovnaký ako v projekte Hod kockou, ale s menšou množinou zobrazovaných obrazov. Študenti potrebujú zistiť, že nie sme schopní priamo vybrať náhodný tvar, ale môžeme nahradiť tvar špecifickým číslom. Generátor náhodných čísel je už známy z predchádzajúceho projektu. Na základe náhodného čísla sa zobrazí konkrétny tvar. Študenti musia zistiť, že táto úloha sa líši od predchádzajúcej a musíme porovnať hodnotu náhodného čísla, nielen číslo priamo zobraziť. Hodnotu potrebujeme porovnať dokonca až trikrát, musíme ju preto uložiť „na budúce použitie“ v premennej (obr. 8).

Skutočná hra vyžaduje aktivitu – trasenie rukou. Možnosti by sa preto aj na micro:bitovi mali meniť trasením (obr. 9). Nakoniec bol ako interaktívna časť nášho hardvérového projektu použitý akcelerometer (trasenie, shake).

Princíp programovania: Predstavuje sa koncept premennej. Hodnotu premennej nemeníme, len sme ju nastavili na náhodnú hodnotu. Zobrazí sa syntax sekvencie príkazov IF-THEN-ELSE, ktorá riadi vykonanie príkazu.



Obrázok 9: Projekt Kameň, papier, nožnice

3.2.4 *Počítanie víťazstiev*

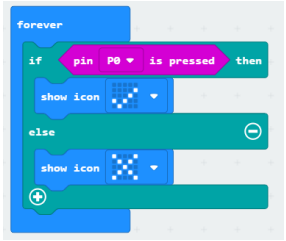
Hra pre dvoch môže byť povýšená na triedny turnaj, v ktorom žiaci hrajú systémom každý s každým, „bojujú“ a snažia sa získať toľko výhier, koľko je to možné. Preto je lepšie počítať výhry pomocou internej premennej. Stlačením tlačidla A sa výhry navýšia o jeden bod. Na konci aktivity sa zobrazí počet všetkých výhier stlačením tlačidla B. Študenti môžu nájsť víťaza triedy.

Princíp programovania: Praktizoval sa koncept premennej. Študenti musia rozhodnúť, kedy je potrebná inicializácia premennej, kedy má byť hodnota zmenená a kedy je zobrazená hodnota.

Urobili sme rovnaký projekt pomocou blokov MakeCode vo všetkých skupinách a pomocou MicroPythonu medzi študentmi 2. ročníka.

3.3 Projekt na Základnej škole

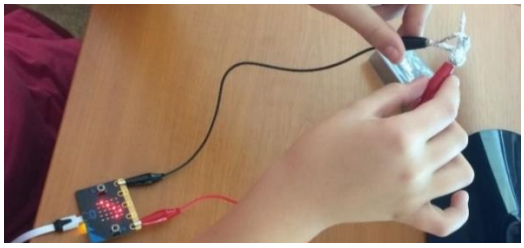
Experimentálne sme sa počas kampane Meet and Code pokúsili použiť micro:bit-y aj na základnej škole. Všetci 10-roční žiaci mali predchádzajúce skúsenosti s programovaním vo vizuálnom jazyku Baltík. Rozhodli sme sa použiť len dva jednoduché projekty, animáciu tlčúceho srdca a simuláciu elektronického kľúča dverí.



Obrázok 10: Projekt Otváranie dvier pomocou kľúča

Na simuláciu uzavretého okruhu, spínača na zámok, sme použili krokodílie svorky a funkčnosť sme zabezpečili programom s jednoduchým vetvením (obr. 10). Okrem toho bola realizovaná aj praktická aktivita (obr. 11, 12). Študenti potrebovali vytvoriť hliníkový kľúč a hliníkovú kľúčovú dierku. Niektorí si však uvedomili, že môžeme použiť aj skutočný železný kľúč (obr. 11).

Obrázok 11: Praktická časť Otvárania dvier pomocou kľúča



Obrázok 12: Praktická časť Otvárania dvier pomocou kľúča - Uzatvorený obvod

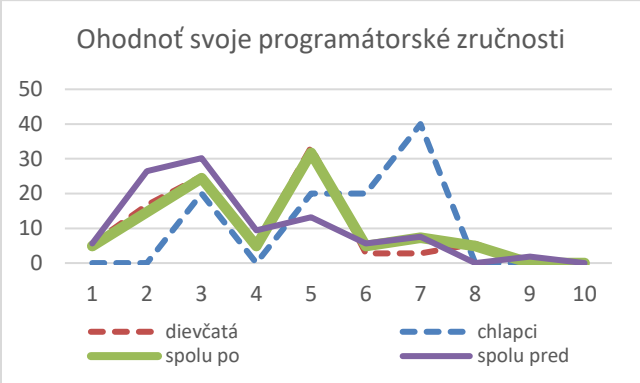
4 Výsledky

V novembri 2018 bolo realizované aj zbieranie údajov prostredníctvom dotazníkov po realizácii týchto aktivít. Spolu odpovedalo len 41 respondentov, z toho 5 chlapcov, zvyšných 36 boli dievčatá.

Porovnali sme oba dotazníky a pri otázke „Čo je to micro:bit?“ sa zvýšila frekvencia slova *program** z 18 na 25 ráz, *hardvér* zo 14 na 21, *edu** z 1 na 8, *micro* z 3 na 15 ráz.

Pri otázke, ako sa im páčili lekcie, sa respondenti vyjadrili, že veľmi – 63 %, viac áno – 24 %, neutrálne – 7 % a len 1 dievča odpovedalo, že sa jej to nepáčilo.

Keď sme sa spýtali, čo sa naučili, odpovedali, že „Informatika môže byť zábavná,“ „myslieť v inej sfére,“ „programovať s iným jazykom,“ „programovanie môže byť zábavné,“ „pracovať viac samostatne.“



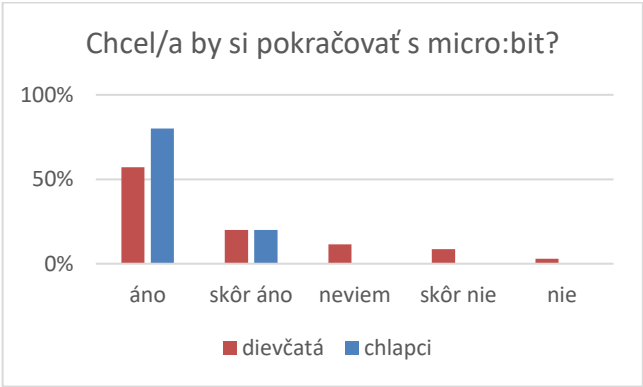
Obrázok 13: Ohodnoť svoje programátorské zručnosti

Nezabudli sme sa spýtať, či je programovanie dôležité. Študenti sa vyjadrili, že „áno, ale verím, že na svete sú inteligentní ľudia, ktorý budú programovať namiesto mňa,“ alebo „Pomáha to zachraňovať životy, robiť umenie, stavať. Ešte chcem dodať, že sa práve učím programovať doma v Jave.“

Objavili sa rozdiely v tom, ako študenti po realizácii vnímajú svoje programátorské zručnosti (obr. 13). Priemer u dievčat sa zvýšil z 3.2 na 3.8 a vyberali spomedzi škálovej množiny 1-8. Priemer pri chlapcoch sa zvýšil z 5.4 na 5.6 a kým v pred dotazníku sa sami hodnotili v rozmedzí 3-9, po realizácii to bolo medzi 3-7.

Pri otázke (obr. 14), či chcú s micro:bit-om pokračovať, 57 % dievčat a 80 % chlapcov odpovedalo určite áno, 20 % dievčat a 20 % chlapcov odpovedalo áno.

Vo všeobecnosti 80 % študentov súhlasilo (obr. 14), 10 % nevedelo a 11 % nesúhlasilo s ďalším pokračovaním.



Obrázok 14: Chcel/a by si pokračovať s micro:bit?

4.1 Pozorovanie

Ako výskumné metódy sme zvolili metódu dotazníkov, ale aj priameho pozorovania, kde úlohu pozorovateľa mal na starosti učiteľ, ktorý predmet učil. Zaznamenané boli len výsledky aktivít. Komplexný videozáznam nebol realizovaný. Cieľom pozorovania bolo sledovať reakcie žiakov a hlavne analyzovať, či sa s pribúdajúcimi hodinami ich motivácia pracovať stupňuje. Mali sme veľmi pozitívnu spätnú väzbu vo všetkých vekových kategóriách účastníkov Meet and Code. Vizuálny jazyk založený na blokovom princípe bol považovaný za perfektnú náhradu za textový jazyk Pascal. Prvý príspevok na školských sociálnych sieťach mal veľký vplyv na ostatné triedy, kde sa aktivita práve nerealizovala. Záujem a angažovanosť študentov sa rýchlo zvýšil. Študenti si chceli vyskúšať ten istý proces, takže nasledujúci deň boli skupiny pripravené okamžite pracovať. Pozorovali sme, že motivácia a názor na programovanie boli o niečo lepšie ako kedykoľvek predtým.

Skupina dievčat, ktoré zvyčajne veľmi šomrú počas programovania, požiadali učiteľa, aby pokračoval v ďalšej lekcii. Dokonca učiteľovi poďakovali za najlepšiu lekciu programovania.

Kolaboratívna časť projektu, „boje“ počas hry Kameň papier nožnice, bola dynamická a študenti preukázali svoju čestnosť. Stlačili tlačidlo víťazstva až potom, čo skutočne porazili niekoho iného.

Najlepšie z pohľadu učiteľa je intuitívne programovacie prostredie. Jazyk založený na blokoch umožňuje študentom rozvíjať programy tak, že spolu vytvárajú vizuálne bloky, čím odstraňujú potrebu naučiť sa zložitú syntax jazyka založeného na texte. Čím menej času potrebujú na písanie kódu, tým viac času majú na premýšľanie o probléme. Študenti by sa radšej mali sústrediť viac na riešenie problémov než na jazykovú syntax.

Po prvom použití programovania s hardvérovým konceptom sme zaznamenali nárast motivácie učiť sa programovanie aj medzi žiakmi, ktorí nemajú v pláne maturovať z predmetu informatika, ale aj medzi dievčatami a budúcimi učiteľmi.

Žiaci v základnej škole vytvárali veľa nápadov o tom, ako urobiť kľúč a kľúčovú dierku (použili sme kus hliníka). Niektorí z nich si uvedomili, že môžu okruh uzavrieť len tým, že spoja priamo krokodílie svorky alebo dokonca len pomocou vlastných prstov. Potom sa jeden zo žiakov spýtal: "Koľko ľudí by mohlo vytvoriť ľudskú reťaz na simuláciu uzavretého okruhu s micro:bit-om?". A tak sme si reťaz úspešne vyskúšali. Naša trieda začínala programovaním, ale končila dôkazom toho, že ľudská reťaz stačí

na zatvorenie elektronického obvodu, čo je viac než počítačová vedomosť.

5 Záver

Vizuálne blokové príkazy sú motiváciou pre tých študentov, ktorí vôbec nemajú radi programovanie. Na druhej strane, možnosť používať podmnožinu jazyka Python otvára priestor pre tých, ktorí chcú používať reálny jazyk. Programovanie mikropočítača BBC micro:bit robí informatiku zábavnou a hmatateľnou. Študenti sú zainteresovaní, pretože program komunikuje s vonkajším svetom prostredníctvom trasenia, nakláňania a blikania LED.

Popisované skúsenosti zachytávajú krátke obdobie, kedy žiaci mali len prvé skúsenosti s programovaním hardvéru. Aj napriek tomu, že sme dosiahli pozitívne výsledky, nie je v tomto momente vhodné zovšeobecňovať vhodnosť použitia mikrokontroléra vo výučbe. Potrebujeme naďalej identifikovať a overiť vhodnosť aktivít, ktoré by pokrývali pokročilejšie koncepty.

POĎAKOVANIE

Táto publikácia je výsledkom výskumu spojeného s Učíme s hardvérom, Komenského inštitút a projektom Innovative Methods in Programming Education in the University Education of Teachers and IT Professionals (KEGA 029UKF-4/2018).

BIBLIOGRAFICKÉ ODKAZY

[1] O'SULLIVAN, D., IGOE, T. *Physical computing: sensing and controling the physical world with computers*, Boston: Course Technology Press, 2004, p. 136.

[2] JIN, K. H. et al., *Teaching Elementary Students Programming in a Physical Computing Classroom.* In Proceedings of the 17th Annual Conference on Information Technology Education (SIGITE '16). ACM, 2016, New York, NY, USA, 85-90.

[3] Physical computing for the non-computer science educator https://education.microsoft.com/courses-and-resources/courses/physical-computing-and-its-role-in-education

[4] Hacking STEM Lessons & Hands-On Activities. https://www.microsoft.com/en-us/education/education-workshop/default.aspx

[5] Training for Teaching STEM.

[6] RUBIO, M. A. et al., Closing the gender gap in an introductory programming course. Computers and Education, 82(C):409–420, 2015.

[7] SENTANCE, S. et al.,*Teaching with physical computing devices: the BBC micro:bit initiative*, In Proceedings of the 12th Workshop on Primary and Secondary Computing Education (WiPSCE '17), Erik Barendsen and Peter Hubwieser (Eds.). ACM, 2017, New York, NY, USA, 87-96.

[8] Centre for computer History. What is Micro:Bit? The FAQ. http://www.computinghistory.org.uk/news/39194/What-is-Micro-Bit-The-FAQ/

[9] SENTANCE, S., SCHWIDERSKI-GROSCHE, S. 2012. *Challenge and creativity: using .NET gadgeteer in schools*. In Proceedings of the 7th Workshop in Primary and Secondary Computing Education. ACM, 90–100. http://dl.acm.org/citation. cfm?id=2481473

[10] DESPORTES, K. et al. 2016. *BitBlox: A Redesign of the Breadboard.* In Proceedings of the The 15th International Conference on Interaction Design and Children. ACM, 255–261.

[11] KAFAI, Y. B. et al. 2014. *A crafts-oriented approach to computing in high school: Introducing computational concepts, practices, and perspectives with electronic textiles*. ACM Transactions on Computing Education (TOCE) 14, 1 (2014), 1.

[12] KALOTI-HALLAK, F. et al. 2015. *Students' attitudes and motivation during robotics activities*. In Proceedings of the Workshop in Primary and Secondary Computing Education. ACM, 102–110.

[13] SENTANCE, S. et al. *"Creating Cool Stuff" - Pupils' experience of the BBC micro:bit.* In Proceedings of the 48th ACM Technical Symposium on Computer Science Education: SIGCSE 2017. 2017.

[14] HODGES, S., *.NET Gadgeteer: a new platform for K-12 computer science education*. In Proceedings of the 44th ACM technical symposium on Computer science education, pages 391–396. ACM, 2013.

[15] VOŠTINÁR, P., HANZEL, P. *Mobile application – a tool for teachers, pupils and their families*, APLIMAT 2016 – 15th Conference on Applied Mathematics, 2016, pp. 1118–1125.

[16] VOŠTINÁR, P. *Creating Mobile Apps For Teaching*, In International Technology, Education and Development onference, 2018, pp. 810 – 816.

[17] VOŠTINÁR, P. *Phytagorean Theorem – Mobile Application*, In International Technology, Education and Development Conference, 2018, pp. 800 – 805.

[18] SCHULZ, S., PINKWART, N. *Physical Computing in STEM Education.* In Proceedings of the Workshop in Primary and Secondary Computing Education (WiPSCE '15). ACM, 2015 New York, NY, USA, 134-135.

[19] JIN, K. H. et al. *Using Physical Computing Projects in Teaching Introductory Programming*, In Proceedings of the 19th Annual SIG Conference on Information Technology Education (SIGITE '18). ACM, 2018, New York, NY, USA, 155-155.

[20] JIN, K. H. et al. *Using Physical Computing Projects in Teaching Introductory Programming*. In Proceedings of the 19th Annual SIG Conference on Information Technology Education (SIGITE '18). ACM, 2018, New York, NY, USA, 155-155.

[21] element14 announces public availability of the BBC micro:Bit.. M2 Presswire Retrieved from https://search.proquest.com/docview/1792594424?accountid=59680

[22] PRZYBYLLA, M., ROMEIKE, R. *Physical computing and its scope - towards a constructionist computer science curriculum with physical computing*, Informatics in Education, 2014 13(2), 225-240.

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9 ISSN 2454-051X

Analýza obsahu výuky informatiky na základních a středních školách

Analysis of the content of informatics education at elementary and secondary schools

Rostislav Fojtík
Ostravská univerzita
Dvořákova 7
701 03 Ostrava
Česká republika
rostislav.fojtik@osu.cz

ABSTRACT

Information and communication technologies are developing very dynamically. It is, therefore, necessary to respond to these changes in computer science. Teaching should not only focus on computer and office software. The article deals with the content of computer science education. The results of the thematic plans of the primary and secondary schools in Czech Republic show the most commonly used topics, the deficiencies in the design and the range of issues, the lack of use of modern ICT trends. The article also describes a survey focused on the use of mobile computers by secondary school pupils. The report also describes a case study that deals with Python programming.

Keywords

Analysis, Informatics, Programming, Thematic Plan, Teaching.

ABSTRAKT

Informační a komunikační technologie se velmi dynamicky vyvíjí. Proto je potřeba, aby na tyto změny reagovala i výuka informatiky. Výuka by se neměla zaměřovat pouze na ovládání počítače a kancelářského software. Článek se zabývá obsahem výuky informatiky a práce s počítačem. Pomocí rozboru vybraných tematických plánů základních a středních škol v České republice článek ukazuje nejčastěji používaná témata, nedostatky v návrhu a rozsahu témat, nedostatečné využití moderních trendů v oblasti ICT. Článek rovněž popisuje průzkum, který byl zaměřen na využívání mobilních počítačů žáky středních škol. Dále článek popisuje případovou studii, která se zabývá výukou programování v jazyce Python.

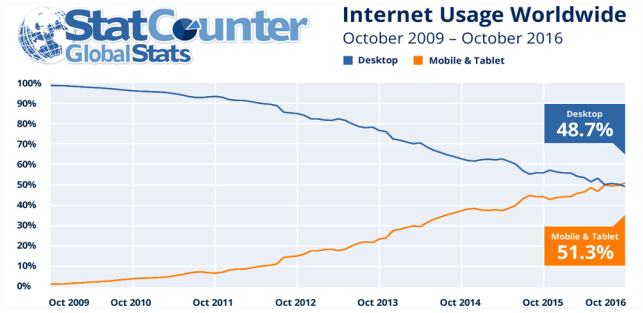
Klíčová slova

Analýza, informatika, programování, tematický plán, výuka.

1. ÚVOD

Výuka informatiky v současné době potřebuje reformovat a upravit obsah i metodickou stránku. V mnoha zemích světa se odborníci zabývají stanovením cílů, obsahů i nových metodických postupů ve výuce informatiky [1], [2], [3]. Přestože se využívání počítačů v posledních letech výrazně změnilo, výuka na tuto skutečnost často dostatečně nereaguje. Na mnohých školách se výuka informatiky stále zaměřuje hlavně na základní ovládání osobních počítačů a využívání kancelářských aplikací. Zpracování informací, multimédia, grafika, moderní styly programování, mobilní zařízení a cloudové služby jsou vyučovány okrajově nebo téměř vůbec ne.

V současné době se mění přístup k využívání počítačových zařízení. Zatímco ještě před několika lety byly využívány především stolní počítače a později přenosné notebooky, jsou to nyní hlavně mobilní zařízení. Již na konci roku 2016 byl počet přístupů na internet z mobilních zařízení větší než ze stolních počítačů (Obrázek 1). Tato skutečnost by se měla objevit rovněž ve výuce informatiky [4], [5], [6].



Obrázek 1. Přístup na internet podle typu zařízení [6]

Snaha o změnu obsahu a zaměření výuky informatiky je patrná nejen mezi některými učiteli, ale rovněž ve snaze změnit koncepci výuky celkově na všech školách.

2. Metodika

Byly stanoveny následující hypotézy:

H1: Obsah výuky informatiky se zaměřuje hlavně na kancelářské aplikace a základní ovládání osobních počítačů.

H2: Výuka informatiky se nedostatečně zaměřuje na moderní prvky v oblasti ICT.

H3: Žáci využívají mobilní zařízení více než 2 hodiny denně.

Pro ověření uvedených hypotéz byla provedena analýza 65 tematických plánů základních a středních škol v České republice. Analýza zkoumala obsah, časovou dotaci témat a zaměření výuky informatiky. Tematické plány byly získány jako veřejně dostupné na webových stránkách škol nebo přímo od vyučujících.

ISBN 978-80-557-1533-9 ISSN 2454-051X

Jedním z dílčích cílů bylo rovněž zjistit, jak žáci využívají mobilní zařízení. Zjišťování bylo prováděno ve dvou fázích. V první respondenti vyplnili dotazník, který obsahoval šest otázek, které se týkaly věku žáků, odhadu času používání mobilních zařízení, typů využívaných mobilních zařízení a mobilních aplikací, používání cloudových služeb a oblíbených mobilních aplikací. Dotazníky byly sice anonymní, ale respondenti je označili svou značkou. V druhé fázi respondenti sledovali reálné využití svých mobilních zařízení. Během sedmi dnů žáci zaznamenávali do připravených formulářů, které mobilní aplikace a jak dlouho během dne používali. Formuláře označili respondenti opět svou značkou. Díky osobním značkám bylo možné porovnat dotazníky a formuláře konkrétních žáků. Na zařízeních s operačním systémem Android ke sledování reálného času využívání mobilních zařízení používali žáci aplikaci QualityTime. Na zařízeních se systémem iOS žáci využívali systémové volby „Čas u obrazovky".

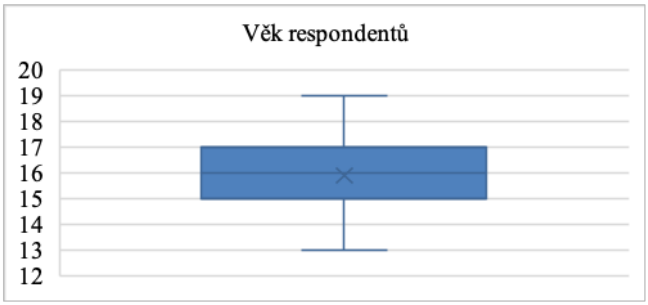
Ve světě se klade velký důraz na zařazení algoritmizace a programování do výuky informatiky. Proto byla provedena případová studie zabývající se výukou programování, konkrétně v jazyku Python.

Získané výsledky byly zpracovávány pomocí statistického software Wizard pro operační systém macOS a software MS Excel.

3. VÝSLEDKY

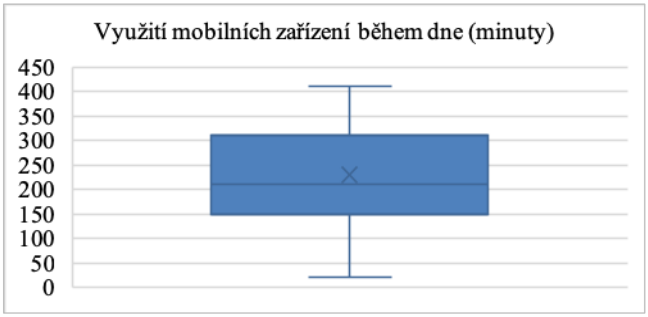
3.1 Výsledky dotazníkového průzkumu

Byl proveden průzkum mezi žáky dvou víceletých gymnázií. Dotazník vyplnilo 123 respondentů a 42 z nich následně sledovalo reálné využívání mobilních zařízení. Věk respondentů se pohyboval mezi 13 až 19 lety. Medián byl 16 let (Obrázek 2).



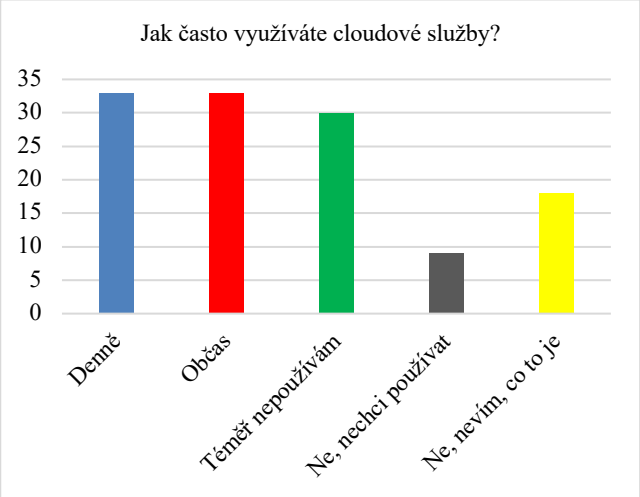
Obrázek 2. Věk respondentů

Prvním úkolem žáků bylo odhadnout čas, který denně věnují používání mobilních zařízení. Rozdíly mezi odhady jednotlivých žáků byly dost výrazné. Medián byl 210 minut (Obrázek 3).



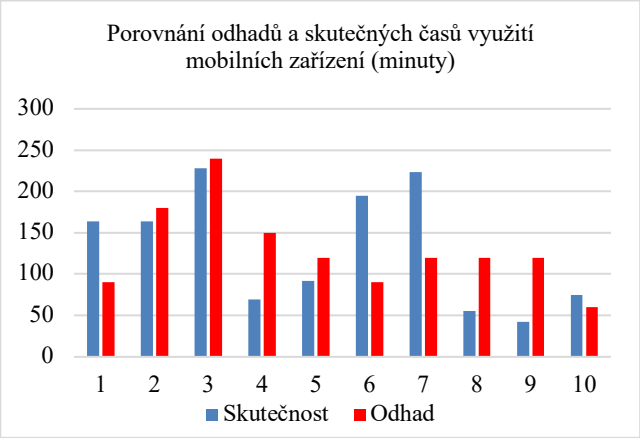
Obrázek 3. Odhad využití mobilních zařízení

Dotazník rovněž zjišťoval, jak respondenti využívají cloudové služby. Pouze 33 respondentů (tj. 27 %) používá cloudové služby a synchronizaci mezi počítači denně. Stejný počet používá tyto služby jen občas. 46 % respondentů cloudové služby téměř nevyužívá nebo nevyužívá vůbec (Obrázek 4). To je alarmující zjištění, neboť současné počítače a operační systémy jsou založeny ve velké míře na synchronizaci mezi zařízeními a cloudových službách. Hlavně mobilní počítače jsou velmi často koncipovány tak, aby využívaly streamovací služby, ukládání dat na cloudová uložiště, synchronizace s ostatními počítači a podobně. Tyto skutečnosti by se měly projevit v obsahu a zaměření výuky informatiky a práce s počítači.



Obrázek 4. Využívání cloudových služeb

Nejčastěji využívanými mobilními aplikacemi mezi respondenty byly sociální sítě. 68 % respondentů uvedlo, že často používají a mezi nejoblíbenější aplikace patří Instagram. 61 % respondentů používalo YouTube, 47 % Facebook, 41 % hry, 36 % webové stránky, 34 % hudební přehrávač. V podobném výzkumu, který jsme provedli mezi mladšími žáky (s věkovým průměrem 13,5 let), byly nejčastěji používanými mobilními aplikacemi hry.



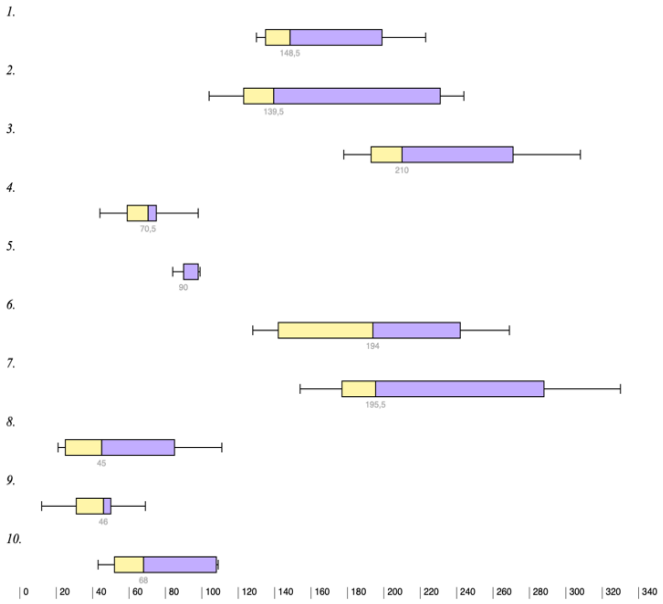
Obrázek 5. Porovnání odhadů a skutečných časů využití mobilních zařízení během dne v minutách

Studenti byli rozděleni na skupiny podle tříd (studijních skupin), do kterých chodili. Na obrázku č. 5 je ukázka srovnání odhadů

ISBN 978-80-557-1533-9 ISSN 2454-051X

a skutečných časů využívání mobilních zařízení. Graf zachycuje srovnání první skupiny respondentů, ve které bylo deset žáků. Průměrný rozdíl porovnaných časů byl u této skupiny pouhé dvě minuty. Podobné výsledky byly rovněž u ostatních sledovaných skupin. Z výzkumu vyplývá, že žáci používají mobilní zařízení během dne mnohdy více než několik hodin.

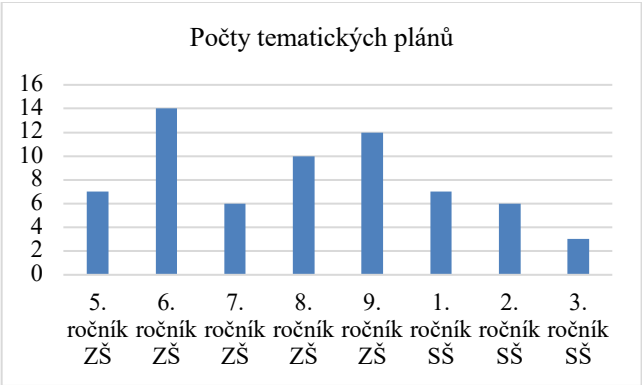
Na obrázku č. 6 je krabicový graf, který znázorňuje reálný čas využití mobilních zařízení v sedmi sledovaných dnech. Na vodorovné ose je uveden čas v minutách a na svislé ose jsou čísla označující sledované respondenty první skupiny. U některých žáků je vidět značný rozptyly mezi časy v různých dnech. Podobné výsledky byly v ostatních sledovaných skupinách žáků.



Obrázek 6. Reálné využívání mobilních zařízení v minutách

3.2 Analýza tematických plánů

Z provedené analýzy šedesáti pěti tematických základních a středních škol vyplývá, že výuka informatiky se stále orientuje hlavně na výuku kancelářských aplikací a neodpovídá současným trendům a potřebám v oblasti výuky informační gramotnosti [7], [8]. Na obrázku 7 jsou uvedeny počty analyzovaných tematických plánů v jednotlivých ročnících.



Obrázek 7. Počty tematických plánů podle ročníků

V tabulce číslo 1 jsou zapsány počty konkrétních témat v jednotlivých tematických plánech. V 60 % zkoumaných tematických plánů se objevuje téma práce s textem. Jako problém se ukazuje název a obsah tématu. Název tématu by měl korespondovat s cíli výuky. Zde se ukazuje, že velká část vyučujících si stanovuje nevhodné cíle.

Tabulka 1. Počet uvedení témat v plánech

Téma	Počet
Práce s textem	39
Grafika	29
Práce s tabulkami	26
Internetové služby	23
Prezentace	21
Bezpečnost, právní aspekty	15
Tvorba webových stránek	12
Práce s informacemi	10
Algoritmy a základy programování	10
Operační systém	9
Databáze	7
Relevantnost informací	7
Práce s videem	3
Vývojové trendy v oblasti informatiky	3
Výukové a motivační hry	2
Cloudové služby	2
Robotika	2

Například u témat zabývající se práci s textem bylo ve 23 % plánech uveden název „MS Word“, v 64 % plánů byl uveden název „Textový editor“ a pouze v 13 % plánů byl název „Práce s textem“. V dílčích tématech se objevuje výčet funkcionalit textového editoru. Výuka se zaměřuje obvykle na nastavování stylů, vlastností písma, vkládání objektů, vytváření tabulek a obrázků, v lepším případě na typografická pravidla. Práce s online editory, sdílení a synchronizace dokumentů se objevila pouze ve dvou ze zkoumaných tematických plánů. Práce s textem je nejen ve většině plánů, ale rovněž je tomuto tématu věnováno až příliš mnoho času. Na jedné základní škole dokonce celý 6. ročník věnují tematickému celku MS Word, celý 7. ročník tématu MS Excel, celý 8. ročník tématu MS PowerPoint. V 9. ročníku pak dva měsíce tyto tři aplikace opakují a pak se teprve dostávají k jiným tématům (grafika, multimédia).

Podobně je tomu u témat zabývající se tabulkami a prezentacemi. 23 % plánů uvádí název MS Excel, 77 % tabulkový procesor, 24 % PowerPoint, 62 % prezentační program a pouze 14 % uvádí název tématu prezentační dovednosti.

Moderní trendy v oblasti informatiky, algoritmizace a programování, robotika, práce s multimédií, mobilní zařízení a podobně se v tematických plánech objevují v menší míře nebo vůbec ne. A to přesto, že například mobilní počítače se dnes používají častěji než počítače stolní. Důvodem malého využívání ve výuce práce s roboty mohou být malé znalosti a zkušenosti

ISBN 978-80-557-1533-9 ISSN 2454-051X

některých vyučujících v této oblasti a rovněž také vyšší cenové náklady na pořízení a případnou údržbu robotů. Během rozhovorů s učiteli informatiky, také vyplynulo, že mnoho z nich má tendenci učit stále stejným způsobem jako v minulých letech. Dalším z důvodů, proč učitele mají problém s vhodným výběrem témat, je stále malá aprobovanost v oboru.

3.3 Případová studie

Byla provedena případová studie, která byla zaměřená na výuku programování v jazyce Python. Do studie se zapojily tři skupiny žáků šestého ročníku osmiletého gymnázia. Cílem případové studie bylo zjistit nakolik jsou žáci schopni pochopit učivo, jakou mají motivaci a jak chápou další uplatnění získaných znalostí a dovedností. Všichni žáci již absolvovali v minulých ročnících krátkou výuku programování pomocí nástroje Scratch. V první sledované skupině bylo 16 chlapců, v druhé skupině bylo 6 chlapců a 5 děvčat a ve třetí skupině bylo 10 děvčat a 3 chlapci. Studovat přírodovědné nebo technické obory po gymnáziu deklarovalo v první skupině 10 žáků, v druhé skupině 7 žáků a ve třetí skupině 3 žáci. Na výuku algoritmizace a programování ve vyšším programovacím jazyku (Python) bylo vyčleněno 18 hodin (2 hodiny týdně). Pro výuku byly připraveny příklady, které byly zaměřeny na co největší praktické využití. Co nejméně příkladů se zaměřovalo na matematické výpočty. Pro žáky byly nejvíce názorné a motivující programy využívající želví grafiku a vykreslování geometrických obrazců. Přes velkou snahu o praktické příklady, část žáků měla problémy s motivací a pochopením praktického využití učiva. Hlavně některá děvčata ve třetí skupině nebyla schopna si moc představit využití v praktickém životě.

Ukazuje se, že výuce vyšších programovacích jazyků je vhodné věnovat větší počet hodin. A to z důvodů, aby vytvářené programy byly reálně využitelné, a nejen jednoduché programy a matematické úkoly. Při vytváření jednoduchých příkladů si část žáků neumí představit praktické využití programovacích jazyků. Jako vhodnější se ukazuje, výuka vyšších programovacích jazyků ve volitelných seminářích. V nich je možné výuce věnovat více času a rovněž cílová skupina žáků je vhodnější. Díky tomu je možné věnovat se moderním principům programování, využívání například objektově orientovaného přístupu, vhodných knihoven a frameworků, vytváření reálně využitelných programů a podobně [9], [10].

4. SHRNUTÍ

Z výsledků lze konstatovat, že všechny tři stanovené hypotézy jsou platné. Z provedené analýzy tematických plánů českých základních a středních škol vyplývá, že stále velká skupina vyučujících neumí správně vybrat vhodnou obsahovou náplň a zaměření výuky informatiky. Výuka se ve velké míře orientuje na kancelářské aplikace a věnuje těmto tématům neúměrně mnoho času na úkor jiných témat. Navíc mnohé plány obsahují témata zaměřená na ovládání konkrétních aplikací, a ne na vytváření dokumentů a práci s informacemi. Mnohem méně času je věnováno moderním oblastem informatiky a výuka nereflektuje změny v oblasti ICT. Mobilní zařízení, cloudové služby, synchronizace dat, robotika, práce s informacemi, relevantnost informací, bezpečnost a podobně se v tematických plánech vyskytují jen v malé míře. Mnohé školy tato témata dokonce zcela vynechávají. Informatika je pak často degradována na základní dovednosti v oblasti ovládání počítače a některých vybraných aplikací. Bylo zjištěno, že žáci využívají své mobilní zařízení i několik hodin denně. Ve vybrané věkové skupině jsou nejčastěji využívanými mobilními aplikacemi ty, které umožňují přístup k sociálním sítím. Výsledky případové studie ukazují, že výuce programování ve vyšším programovacím jazyku je vhodnější věnovat více času, aby obsahem nebylo pouhé přepisování jednoduchých algoritmů do programovacích jazyků, ale aby výsledkem mohly být reálně využitelné programy.

BIBLIOGRAFICKÉ ODKAZY

[1] BARENDSEN, E., GIRGURINA, N., TILBOOM, J. A new Informatics Curriculum for Secondary Education in The Netherlands, *International Conference on Informatics in Schools: Situation, Evolution, and Perspectives,* 2016, DOI: 10.1007/978-3-319-46747-4_9

[2] NEUMAJER, O. Sedm mýtů o informatice a ICT ve vzdělávání. Metodický portál, ISSN: 1802-4785, Výzkumný ústav pedagogický v Praze, 4. 11. 2008

[3] Strategie digitálního vzdělání do roku 2020, http://www.msmt.cz/vzdelavani/skolstvi-v-cr/strategie-digitalniho-vzdelavani-do-roku-2020

[4] CHASSIAKOS, Y. R., RADESKY, J., CHRISTAKIS, D., MORENO, M. A., CROSS, C. Children and Adolescents and Digital Media, in *Pediatrics*, November 2016, Volume 138, Issue 5. ISSN 1073-0397.

[5] RADESKY, J. S., SCHUMACHER**,** J., ZUCKERMAN, B. Mobile and Interactive Media Use by Young Children: The Good, the Bad, and the Unknown, in *Pediatrics*, 2014; 135 (1): 1 DOI: 10.1542/peds.2014-2251.

[6] Worldwide, mobile overtakes desktop Internet usage for the first time, says StatCounter, 2.11.2016, https://9to5google.com/2016/11/02/mobile-versus-desktop-internet-usage-statcounter/

[7] NEUMAJER, O. Jak se bude zavádět informatické myšlení a zvyšovat digitální gramotnost ve školách, 2016, http://ondrej.neumajer.cz/jak-se-bude-zavadet-informaticke-mysleni-a-zvysovat-digitalni-gramotnost-ve-skolach/

[8] Revize RVP v oblasti informatiky a informačních a komunikačních technologií, http://www.nuv.cz/t/revize-rvp-ict

[9] JANKE, E., BRUNE, P., WAGNER, S. Does outside-in teaching improve the learning of object-oriented programming? *Proceedings of the 37th International Conference on Software Engineering*. Volume 2. Pages 408-417. 2015.

[10] LIU, Y., SUN, M. CHAN, Y. Teaching Guidance in Programming Courses from Procedure-oriented to Object-oriented. *Proceedings of the 2016 2nd International Conference on Social Science and Higher Education.* 2016. Volume 53. ISSN 2352-5398.

Nový informatický predmet Riešenie problémov a programovanie

New Informatics Subject Problem Solving and Programming

Ján Guniš
Prírodovedecká fakulta UPJŠ
Jesenná 5
041 54 Košice
Slovenská republika
jan.gunis@upjs.sk

Ľubomír Šnajder
Prírodovedecká fakulta UPJŠ
Jesenná 5
041 54 Košice
Slovenská republika
lubomir.snajder@upjs.sk

Viera Michaličková
Fakulta prírodných vied UKF v Nitre
Tr. A. Hlinku 1
949 74 Nitra
Slovenská republika
vmichalickova@ukf.sk

ABSTRACT

The strategic goal of the National Project IT Academy (NPITA) [1] is to create a model for the education and training of young people for the current and prospective needs of the knowledge and labor market, focusing on IT and ICT. By innovating science and technology education at elementary and secondary schools with a focus on IT and ICT, we want to orient young people to further study in study fields and programs with a prospect of application in the IT sector. One of the specific output of project is the model of a special class at secondary school focusing on IT. Part of the model is the design of eight new subjects for this class. The article deals with design and concept of subject Problem solving and programming. The subject is designed as a superstructure over the basic course of programming. In the subject we put emphasis on problem solving. We use programming language resources to solve problems. For a better idea, we provide a summary of two chapters of the proposed subject.

Keywords

Problem solving. Computational thinking. Programming. Python. New subject. Secondary high school.

ABSTRAKT

Strategickým cieľom Národného projektu IT Akadémia (NPITA) [1] je vytvorenie modelu vzdelávania a prípravy mladých ľudí pre aktuálne a perspektívne potreby vedomostnej spoločnosti a trhu práce so zameraním na informatiku a IKT. Inováciou prírodovedného a technického vzdelávania na základných a stredných školách so zameraním na informatiku a IKT chceme orientovať mladých ľudí na ďalšie štúdium v študijných odboroch a programoch s perspektívou uplatnenia sa v IT sektore. Jedným z konkrétnych výstupov je model špeciálnej triedy na strednej škole so zameraním na informatiku. Súčasťou modelu je návrh ôsmich nových predmetov pre túto triedu. V článku sa zaoberáme návrhom a koncepciou predmetu Riešenie problémov a programovanie. Predmet je navrhovaný ako nadstavba nad základným kurzom programovania na gymnáziách. V predmete kladieme dôraz na riešenie problémov. Pri riešení problémov využívame prostriedky programovacieho jazyka. Pre lepšiu predstavu uvádzame prehľad dvoch kapitol z navrhovaného predmetu.

Kľúčové slová

Riešenie problémov. Informatické myslenie. Programovanie. Python. Nový predmet. Stredná škola.

1. ÚVOD

Tematická oblasť Algoritmické riešenie problémov je súčasťou výučby informatiky od prvého stupňa základných škôl na Slovensku [2]. Sme presvedčení o tom, že pri výučbe tejto oblasti je možné rozvíjať široké spektrum žiackych vedomostí a zručností a že táto oblasť má veľký presah do iných predmetov. Primárnym cieľom výučby tejto oblasti, tak ako jej názov napovedá, by mal byť rozvoj schopnosti riešiť problémy.

Riešenie problémov je súčasťou každej ľudskej činnosti. Uvažovať o riešeniach problémov z pohľadu informatika, abstrahovať z problému podstatné a uvažovať o zovšeobecneniach nájdených riešení, kriticky prijímať ponúkané riešenia a overovať ich správnosť, hľadať a rozpoznávať opakujúce sa vzory, využívať rôzne stratégie pri hľadaní riešení a nájdené riešenia vyjadrovať vo forme algoritmov (bez ohľadu na to, kým budú vykonávané) sú spôsobilosti potrebné pre aktívne a tvorivé zapojenie sa človeka do fungovania reálneho sveta. Uvedené koncepty sú zložkami informatického myslenia [3], rozvoj ktorého považujeme za jeden z kľúčových cieľov vyučovania informatiky.

Schopnosti našich žiakov aktívne využívať vedecké poznatky a myšlienky, vysvetľovať javy vedeckým spôsobom, navrhovať a vyhodnocovať výskumy, interpretovať získané údaje a dôkazy vedeckým spôsobom sú dlhodobo pod priemerom žiakov krajín OECD a ich úroveň dlhodobo klesá [4].

Aj z týchto dôvodov sme sa rozhodli využiť príležitosť, ktorá sa nám ponúkla v Národnom projekte IT Akadémia. Naše predstavy o tom, ako rozvíjať informatické myslenie, čitateľskú, prírodovednú a matematickú gramotnosť sme pretavili do návrhu nového stredoškolského predmetu Riešenie problémov a programovanie.

2. MODEL ŠPECIÁNEJ TRIEDY

Jedným z výstupov Národného projektu IT Akadémia je model špeciálnej triedy na gymnáziách so zameraním na informatiku. Pre

výučbu v tejto triede v rámci projektu pripravujeme 8 nových informatických predmetov: Riešenie problémov a programovanie, Programovanie mobilných zariadení, Počítačové systémy a siete, Informačná bezpečnosť, Databázy, Objektový prístup k riešeniu problémov, Tvorba a prezentácia dát a Programovanie webových stránok. Okrem materiálov pre žiakov (učebné texty, pracovné listy, riešené a neriešené úlohy) sú súčasťou návrhu predmetov aj metodické poznámky pre učiteľov. V ďalšej časti článku sa zaoberáme návrhom a koncepciou predmetu Riešenie problémov a programovanie.

3. KONCEPCIA PREDMETU RIEŠENIE PROBLÉMOV A PROGRAMOVANIE

V predmete Riešenie problémov a programovanie kladieme dôraz na problémy, ich analýzu a návrh ich riešenia. Výber problémov, ktoré sme do predmetu zaradili, pokrýva široké spektrum ľudskej činnosti. Do predmetu sme zaradili množstvo zaujímavých a reálnych problémov, pri riešení ktorých zmysluplne využívame nástroje a silu programovania. Chceme, aby žiaci boli pri hľadaní riešení problémov aktívni, aby využívali a prepájali už raz získané poznatky, postupy, nástroje a techniky [5]. Neobmedzujeme sa teda na „typické“ programátorské zadania úloh.

Programovacím jazykom je jazyk Python, ale samotný predmet nie je koncipovaný ako tradičný kurz pokročilého programovania v jazyku Python.

Predmet Riešenie problémov a programovanie sme navrhli ako dvojhodinový predmet počas jedného školského roka. Predmet nadväzuje na základný kurz programovania (v zmysle Štátneho vzdelávacieho programu) [2]. Nepriamo tak nadväzujeme na [6], kde sme uviedli náš návrh ako základný kurz realizovať. Celková hodinová dotácia predmetu je 66 vyučovacích hodín. Ponúkaný obsah predmetu túto dotáciu prevyšuje a je na učiteľovi, ktoré kapitoly predmetu do vyučovania zaradí. Kapitoly na seba nepriamo nadväzujú a pri ich výbere učiteľovi pomôže prehľadová tabuľka z ktorej sa dá zistiť, v ktorej kapitole sa konkrétny informatický obsah objaví prvý krát (a v tejto kapitole je tento obsah vysvetlený) a v ktorej kapitole sa predpokladá, že uvedený obsah žiaci už zvládli a vedia s nim pracovať.

Predmet je určený žiakom so záujmom o programovanie a žiakom, ktorí chcú na vysokej škole pokračovať v štúdiu informatiky alebo prírodných vied.

Obsah predmetu prekračuje obsah definovaný v Cieľových požiadavkách na vedomosti a zručnosti maturantov z informatiky [7]. Rozumným výberom kapitol je možné tento predmet využiť v príprave žiakov na maturitnú skúšku z informatiky.

3.1 Ciele predmetu

Cieľom predmetu je, aby si žiak rozvíjal a získaval nové znalosti a zručnosti v oblasti riešenia problémov využitím nástrojov programovania. Žiak bude po absolvovaní predmetu schopný:

- využiť pri riešení problémov nástroje programovania,
- analyzovať prírodné deje a pravdepodobnostné javy,
- modelovať a simulovať prírodné deje a pravdepodobnostné javy,
- vyhodnotiť reálne situácie a deje na základe ich modelov a simulácií,
- analyzovať a implementovať rôzne spôsoby kódovania a šifrovania informácií,
- posúdiť vhodnosť a bezpečnosť vybraných kódovacích a šifrovacích algoritmov,
- používať grafy a grafové algoritmy pri riešení problémov,
- pracovať so štruktúrovanými dátami,
- analyzovať štruktúrované dáta,
- navrhovať a implementovať komplexné aplikácie riešiace vybrané problémy.

3.2 Obsah predmetu

Obsah predmetu sme rozdelili do niekoľkých väčších oblastí:

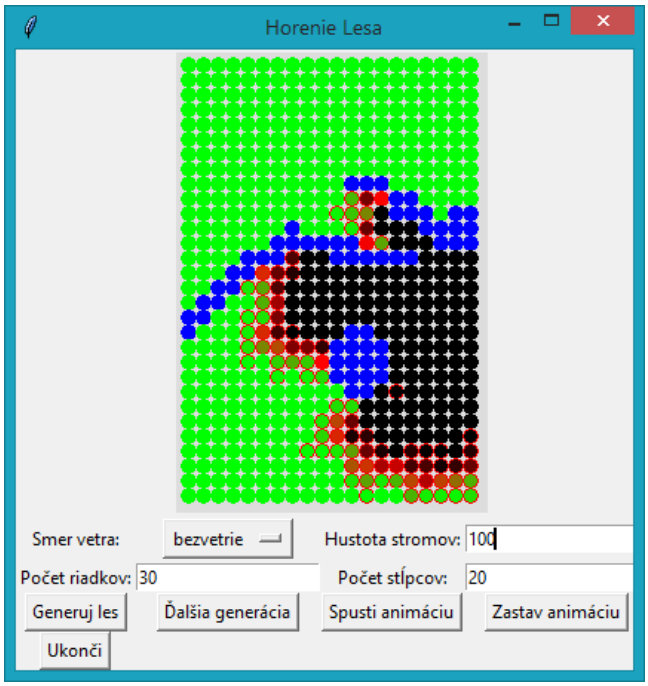
Kapitolu **Programovanie v jazyku Python** je možné využiť ako opakovanie základného kurzu oblasti Algoritmické riešenie problémov. Táto kapitola nie je náhradou za základný kurz a odporúčame, aby ho už žiaci mali absolvovaný. Pre absolvovanie tohto predmetu predpokladáme, že žiaci vedia programovať na úrovni definovanej v inovovanom Štátnom vzdelávacom programe [2]. Ak žiaci absolvovali základný kurz v jazyku Python, môžu túto kapitolu preskočiť alebo ju využiť na rýchle zopakovanie. Ak žiaci absolvovali základný kurz v inom jazyku ako Python, môžu túto kapitolu využiť na to, aby sa oboznámili s tým, aké prostriedky a prístupy pri riešení problémov sa využívajú v jazyku Python. Pre absolvovanie ďalších kapitol predmetu je potrebné, aby žiaci vedeli samostatne riešiť úlohy typovo podobné úlohám z tejto časti.

Oblasť **Modelovanie a simulácie** obsahuje niekoľko kapitol, v ktorých sa zaoberáme simuláciami dejov, ktoré by sme reálne nedokázali zrealizovať. Programovanie využívame na skúmanie fungovania okolitého sveta bez rizika znášania dôsledkov chybných rozhodnutí.

Táto časť obsahuje kapitoly:

- Ako hrať v kasíne a neprehrať - simulácie vybraných pravdepodobnostných dejov.
- Simulácia fyzikálnych pohybov - simulácie fyzikálnych pohybov využitím fyzikálnych zákonov (Newtonov, Coulombov).
- Hra život - simulácia doskovej Conwayovej hry Game of Life. Hľadanie stabilných spoločenstiev.
- Hra život - objektová implementácia doskovej Conwayovej hry Game of Life. Úvod do OOP.
- Ako rýchlo zhorí les - jednoduchý celulárny automat – simulačný model šírenia sa požiaru v lese. Objektová implementácia (pozri Obrázok 1).
- Genetické algoritmy - využitie genetických algoritmov pri jednoduchých optimalizačných problémoch.
- Chcem byť ako Mozart - simulácia hry, ktorú údajne vytvoril sám Mozart. Návrh aplikácie, v ktorej hráč s využitím náhody vytvorí počúvateľné hudobné dielo znejúce „mozartovsky“.

- Tvorba a spracovanie zvukov - generovanie tónov a akordov, prehrávanie zvukových súborov, zvukové efekty.
- Špecifiká floating point aritmetiky - špecifiká číselného dátového typu float, principiálna nepresnosť pri numerickom prístupe k riešeniu problémov. Problémy spôsobené nepresnosťou počítačovej aritmetiky.



Obrázok 1. Simulácia horenia lesa

V oblasti **Kódy a šifry** učíme žiakov porozumieť niektorým šifrovacím algoritmom, poznať ich slabé stránky a implementovať ich. Žiakom približujeme aj problematiku steganografie. Žiaci sú nie len v úlohe pasívneho používateľa kódov a šifier, ale naučia sa ich implementovať, analyzujú ich, odhaľujú ich slabé stránky a navrhujú možné vylepšenia.

Táto časť obsahuje kapitoly:

- Substitučné šifrovanie - Analýza a implementácia šifier: Cézarova šifra, Vigenèrova šifra, Vernamova šifra.
- Pokročilejšie substitučné šifry - Polybiov štvorec, Šifra Playfair, Polyalfabetická šifra s autokľúčom. Transpozičné šifrovanie s kľúčom - šifra Reichelieu, ADFGVX.
- Steganografia - Princíp steganografie. Schovávanie obrázkov v iných obrázkoch.

V oblasti **Rekurzívne algoritmy a grafy** sa zaoberáme rekurziou, ako jednou z metód riešenia problémov a vybranými grafovými algoritmami. Aj keď táto téma je zaraďovaná skôr do vysokoškolského kurikula, žiakom túto tému predstavujeme na jednoduchých a zrozumiteľných príkladoch.

Táto časť obsahuje kapitoly:

- Rekurzia - stratégia rekurzívneho riešenia problémov. Rekurzia pri riešení problémov. Rekurzívne obrázky, fraktály.
- Backtracking - stratégia riešenia problémov prehľadávaním s návratom. Rekurzívne algoritmy s možnosťou návratu: Prehľadávanie dvojrozmerného bludiska. Problém šachových dám. Problém šachového koňa.
- Problém rozdelenia tovarov - stratégia riešenia problémov hrubou silou.
- Vzťahy v sociálnej sieti - reprezentácia sociálnej siete v neorientovanom a orientovanom grafe. Vlastnosti a prehľadávanie grafov.
- Najkratšia cesta - Problém nájdenia najkratšej cesty, implementácia hľadania najkratšej cesty v ohodnotenom grafe.

V časti **Štruktúrované dáta** predstavujeme žiakom rôzne typy štruktúrovaných dát a spôsoby ako s nimi pracovať. Žiaci analyzujú rôzne dostupné otvorené dáta, sami vytvárajú štruktúrované dáta alebo využívajú štruktúrované dáta pre vlastné aplikácie.

Táto časť obsahuje kapitoly:

- Benfordov zákon, verejné dáta. Benfordov zákon pre prirodzene vznikajúce množiny čísel, analýza verejne dostupných dátových súborov, možnosť odhalenia falšovania verejných dát.
- Programovanie obrázkov - programovanie vektorových obrázkov vo formáte SVG.
- Analýza logovacieho súboru - jednoduchý web log mining zo záznamových súborov virtuálneho vzdelávacieho prostredia.
- Index textového dokumentu - vytvorenie indexu textového dokumentu. Tvorba vlastného nástroja na vyhľadávanie v textových dokumentoch.
- Myslíš toto? - automatické zobrazovanie návrhov na dokončenie textu.
- Morphing - plynulá zmena jedného grafického objektu na druhý.

V poslednej časti **Aplikácia** žiaci od základu navrhnú a implementujú aplikáciu pre správu vybraných objektov (napr. správu kníh v domácej knižnici). Súčasťou riešenia je aj návrh grafického rozhrania pre používateľa.

3.3 Štruktúra predmetu

Kapitoly predmetu majú jednotnú štruktúru, aby sa v nich dalo ľahšie orientovať. Kapitola začína motiváciou, uvedením problematickej situácie. V nasledujúcich častiach kapitoly žiakov postupne vedieme cez riešenie jednoduchších úloh k riešeniu problémovej situácie, zavádzame nový informatický obsah a precvičujeme už skôr zavedený obsah. V závere kapitola obsahuje zhrnutie, úlohy na precvičenie, odkazy na ďalšie študijné zdroje a register pojmov. Zaradenie úloh na precvičenie do výučby ponechávame na učiteľovi. Každá kapitola predmetu obsahuje aj metodické poznámky pre učiteľa a vzorové, komentované riešenia

úloh. Riešenia však nechápeme ako jediné správne a možné. Učiteľom však môžu pomôcť rýchlejšie sa v danej problematike zorientovať. Žiakom sú okrem výkladového textu k dispozícii aj pracovné súbory, predpripravené časti programov a pod.

V jednotlivých kapitolách predmetu postupne zavádzame objektový prístup k riešeniu problémov. Žiaci tak prirodzene prechádzajú od procedurálneho programovania s ktorým sa stretli na základnom kurze k objektovému programovaniu. Keďže jazyk Python je multiparadigmový, v situáciách, kde nie je nutné použiť objektový prístup ho násilne nepoužívame.

Pri výbere kapitol a ich implementácii do predmetu sme dodržiavali niekoľko zásad:

- nevytvárajme referenčnú príručku alebo učebnicu jazyka Python,
- primárne chceme žiakov naučiť riešiť problémy (využitím prostriedkov programovania),
- riešenie problému začneme jeho analýzou,
- ak narazíme na „zaujímavý“ problém, ale nebudeme vedieť nájsť „pekné“ riešenie, zvážime jeho zaradenie,
- neučme žiakov všetko, ale len to, čo nevyhnutne potrebujú, učme žiakov pracovať aj s inými zdrojmi informácií,

A nakoniec veríme, že žiakom, ktorí absolvujú tento predmet sprostredkujeme niekoľko dôležitých odkazov:

- Programovať = riešiť problémy.
- Ak vieš programovať, si pre zamestnávateľa zvlášť zaujímavý. A to nie kvôli schopnosti písať programy, ale kvôli vyššej úrovni myslenia.
- Programovanie je zaujímavé a zábavné.

4. Vybrané kapitoly predmetu

Do predmetu sme celkovo zaradili 25 kapitol, ktoré sme rozdelili do vyššie spomenutých oblastí. Pri jednotlivých kapitolách počítame s hodinovou dotáciou od dvoch do piatich vyučovacích hodín. Pre lepšiu predstavu čitateľa uvádzame prehľad dvoch kapitol.

4.1 Ako hrať v kasíne a neprehrať

Kapitola „Ako hrať v kasíne a neprehrať“ nasleduje ihneď za úvodnou, opakovacou kapitolou. Je úvodnou kapitolou do oblasti „Modelovanie a simulácie“. Ciele kapitoly môžeme rozdeliť do viacerých dimenzií.

Z pohľadu programovania si žiaci prehĺbia poznatky o pseudonáhodných číslach a ich generátoroch a naučia sa efektívne používať dátovú štruktúru slovník.

Z pohľadu riešenia problémov sa naučia analyzovať prírodné a spoločenské systémy a naučia sa navrhovať modely týchto systémov.

Z bádateľského pohľadu sa naučia vytvárať a overovať hypotézy simulovaním správania sa systémov pomocou ich modelov.

Z pohľadu kritického myslenia si precvičia kritické prijímanie na prvý pohľad pravdivých tvrdení, naučia sa používať simulácie na overenie pravdivosti týchto tvrdení a overovať správnosť návrhu modelov.

Pre absolvovanie tejto kapitoly predpokladáme aspoň základné vedomosti žiakov z matematiky z oblasti kombinatoriky a pravdepodobnosti.

V úvodnej, motivačnej časti žiakom predstavíme „víťaznú“ stratégiu v hre ruleta. Podľa tejto stratégie máme každý hrací deň zabezpečenú výhru 50 €. Stratégia na prvý pohľad nemá slabé miesto. Úlohou žiakov je overiť, či je táto stratégia správna.

Kým sa žiaci dostanú k overeniu správnosti tejto stratégie, spoločne vyriešia niekoľko jednoduchších úloh. Pri riešení týchto úloh sa žiaci oboznámia so základnou terminológiou oblasti simulácií, naučia sa navrhovať modely jednoduchých reálnych dejov a overovať si správnosť svojich návrhov.

Prvá úloha je pomerné jednoduchá a úlohou žiakov je navrhnúť model systému, pomocou ktorého môžu zistiť aký súčet pri hode dvoma kockami padá najčastejšie. Úloha sa dá bez problémov riešiť aj matematicky (riešenie je v kapitole uvedené), bez použitia simulácie. Cieľom však je, aby sa žiaci naučili navrhovať modely a cez známe výsledky si overili ich správnosť. Zámerne zavedená chyba v návrhu modelu by mala žiakov presvedčiť o dôležitosti testovania nie len modelu ako celku, ale aj jeho jednotlivých častí. V diskusii, pri otázke „Koľko hodov je potrebné simulovať“ sa dotkneme aj zákona veľkých čísiel [8]. Vďaka nemu budú môcť žiaci odhaliť slabé miesto výhernej stratégie spomenutej v úvode. V metodickej poznámke pre učiteľa uvádzame aj iné možné návrhy riešenia: namiesto jedného hodu dvoma kockami použiť dva hody jednou kockou, alebo namiesto simulácie dvoch hodov kocky simulovať ich výsledný súčet. V uvedených prípadoch pre učiteľa zdôvodňujeme aj správnosť, resp. možné problémy takejto implementácie. Pri riešení tejto úlohy žiaci využívajú frekvenčnú analýzu. Vhodnou dátovou štruktúrou, s ktorou sa žiaci v tejto kapitole oboznámia je slovník.

V druhej úlohe žiaci analyzujú riešenie hádanky známej pod názvom Monty Hallova úloha alebo problém troch dverí [9]. V tejto úlohe si súťažiaci vyberá jedny z troch dverí. Za jednými z dverí je hlavná výhra - auto, zvyšné dvoje dverí skrývajú cenu útechy - kozu. Po tom, ako si súťažiaci vyberie jedny dvere, moderátor súťaže otvorí zo zostávajúcich dverí tie, za ktorými je výhra koza. Súťažiacemu ponúkne možnosť zmeniť svoj pôvodný výber dverí. Otázka znie, mal by súťažiaci zmeniť svoje pôvodné rozhodnutie alebo nie? Táto hádanka je zaujímavá aj preto, lebo prípadná zmena pôvodného rozhodnutia súťažiace sa javí ako iracionálna, nemajúca vplyv na celkový výsledok. O to zaujímavejší je výsledok simulácie takejto hry. Aj táto hádanka sa dá vyriešiť bez simulácie (aj toto riešenie je v kapitole uvedené). Vyžaduje však o niečo náročnejšiu štatistickú úvahu než tomu bolo v predchádzajúcej úlohe.

Po vyriešení druhej úlohy sa žiaci vracajú k riešeniu prvého problému. K overeniu víťaznej stratégie v hre ruleta. Stratégia je nasledovná: Stavíme 50 € na červenú. Ak padne červená, berieme výhru (100 €) a odchádzame domov so ziskom 50 €. Ak nepadne červená stavíme v ďalšej hre na červenú toľko, aby sme v prípade výhry mali celkový zisk 50 €. Stavíme teda 100 €. Stávky sú 50 € a 100 €, prípadná výhra je 200 €. Ak vyhráme odchádzame domov zo ziskom 50 €. Ak nie, pokračujeme rovnako až do prvej výhry. Červená raz určite padne a zisk 50 € máme každý deň zaručený. Úlohou žiakov je navrhnúť model tohto systému a simuláciou overiť správnosť stratégie. Pri tejto úlohe je zaujímavé sledovať

„vnútorný“ boj niektorých žiakov. Na jednej strane nevedia odhaliť problém stratégie a na strane druhej odmietajú uveriť tomu, že stratégia funguje (prečo teda nie sú všetci v kasínach?).

Pri riešení problémov je jedným z našich cieľov naučiť žiakov používať rôzne stratégie riešenia problémov. V tomto prípade dekompozíciu problému na podproblémy [10]: simulácia jednej hry → simulácia jedného dňa → simulácia niekoľkých dní. Aj keď model, pomocou ktorého simulujeme jeden deň v kasíne, žiakov presvedča o správnosti stratégie (zatiaľ sa nestalo, aby sme v nejaký deň prehrali) až pri dostatočne veľkom počte dní žiaci odhalia problém tejto stratégie. Veľakrát v kasíne vyhráme, ale vždy len 50 €. Občas je však šnúra prehier taká dlhá, že na ďalšiu stávku už nemáme dostatok zdrojov (alebo narazíme na limit kasína). V tento deň prichádzame o obrovské množstvo prostriedkov. Množstvo malých výhier však nevyrovná ojedinelú, ale zato veľkú stratu.

Riešením uvedených úloh rozvíjame kritické myslenie žiakov a učíme žiakov používať jednu z možností ako overovať pravdivosť tvrdení – navrhovať modely systémov a simuláciami systémov v týchto modeloch overovať pravdivosť tvrdení.

Zbierka úloh obsahuje návrhy a riešenia (pre učiteľa) ďalších zaujímavých úloh: Ako funguje generátor pseudonáhodných čísiel? Ako vypočítať obsah kruhu hádzaním šípok do terča? Ako sa rozdelia súčty hodených kociek, ak týchto kociek bude viac? Ako dopadne duel, ak sa v ňom stretnú rôzne dobrí pištoľníci? Je rozdiel v tom, ktorý začne strieľať ako prvý?

Žiaci sa v tejto kapitole naučia analyzovať reálne systémy, navrhovať a implementovať modely týchto systémov. Naučia sa kriticky prijímať tvrdenia a simuláciou reálnych dejov v modeloch overovať pravdivosť týchto tvrdení. Naučia sa využívať dátovú štruktúru slovník.

4.2 Analýza logovacieho súboru

Z oblasti s názvom „Štruktúrované dáta“ priblížime ciele a obsah kapitoly „Analýza logovacieho súboru“.

V tejto kapitole sa zaoberáme aktuálnym problémom z praxe – spracovaním veľkého objemu automaticky generovaných dát. V našom prípade ide o logovací súbor webového servera (presnejšie webovej aplikácie, ktorou je virtuálne vzdelávacie prostredie). Žiaci získajú praktickú skúsenosť s realizovaním jednoduchého web log miningu. Vytvoria program, ktorý spracuje logovací súbor so skutočnými dátami s cieľom zistiť odpovede na konkrétne otázky. Pri skúmaní formátu a obsahu logovacieho súboru si vytvoria lepšiu predstavu o význame logovacích súborov pre tvorcov a administrátorov webu. V implementácii riešenia sa ako vhodné riešenie ponúka použitie štandardnej údajovej štruktúry množina (*set*), s ktorou sa pravdepodobne v programovacom jazyku Python žiaci zatiaľ nestretli. V kapitole predpokladáme, že žiaci vedia pracovať s reťazcami, poznajú zoznamy a sú schopní písať vlastné funkcie. Zopakujú si tiež čítanie a zapisovanie údajov z a do textového súboru.

V úvode vyučovania je vhodnou motiváciou skúmanie obsahu logovacieho súboru získaného zo skutočného webové servera. Žiaci majú k dispozícii záznamy z jedného dňa prevádzky webového sídla univerzity, môžu si ho otvoriť a prezerať napr. v tabuľkovom kalkulátore. Ich úlohou je odpovedať na rôzne otázky – z akého časového obdobia pochádzajú záznamy o prístupoch používateľov, aké údaje webový server zaznamenáva, koľko riadkov obsahuje, aký veľký by bol logovací súbor s údajmi za 1 mesiac, 1 rok a pod. Zaujímavou úlohou je tiež vyhľadanie prístupu konkrétneho klienta (podľa jeho IP adresy) – žiaci si uvedomia, že požiadavka na zobrazenie jednej stránky v sebe zahŕňa požiadavky na súbory s obrázkami, štýlmi, skriptami atď. Jedno kliknutie používateľa sa tak v logovacom súbore prejaví vo viacerých (obyčajne mnohých) riadkoch. V logovacom súbore žiaci uvidia aj požiadavky robotov vyhľadávacích služieb (*spiders, crawlers, worms*), čo je tiež atraktívny námet na diskusiu.

V ďalších úlohách už žiaci pracujú s logovacím súborom pochádzajúcim z virtuálneho vzdelávacieho prostredia (LMS Moodle), ktorý vzniká z údajov uložených v databáze tejto webovej aplikácie. Webová aplikácia získava potrebné údaje z premenných nastavovaných webovým serverom a má teda k dispozícii rovnaké údaje o požiadavke klienta, aké webový server zapisuje do logu. Logovací súbor z LMS Moodle je však menší a pre žiakov prehľadnejší (očistený od dát, ktoré sú pre účely našej analýzy zbytočné). Žiaci majú z údajov zaznamenaných v logovacom súbore zistiť, z koľkých rôznych IP adries sa v priebehu sledovaného obdobia do vybraného e-learningového kurzu pristupovalo.

Logovací súbor má textový formát, v ktorom jeden riadok zodpovedá záznamu o jednej požiadavke klienta. Žiaci najskôr napíšu krátky skript, ktorým celý logovací súbor zobrazia na obrazovke. Následne sa sústredia na parsovanie jedného riadku (pripomenú si reťazcovú metódu *split()*) a všimnú si, ktorý údaj v riadku je podstatný pre riešenie úlohy.

Ukladanie IP adries čítaných v jednotlivých riadkoch do zoznamu a opakované vyhľadávanie v tomto zozname je jednou z možností, ako úlohu vyriešiť. Existuje však efektívnejšie riešenie. Diskusia o riešení by mala smerovať k záveru, že na filtrovanie duplicít sa hodí údajová štruktúra množina (ako neusporiadaná kolekcia jedinečných prvkov), ktorú žiakom predstavíme. Žiaci experimentujú s príkazmi pre prácu s množinami, odhadujú ich význam a overujú svoje predpoklady v konzole Pythonu.

Po zoznámení sa s novou údajovou štruktúrou žiaci dokončia riešenie problému – napíšu funkciu na analyzovanie logovacieho súboru, ktorá vráti počet rôznych IP adries. V nadväzujúcej úlohe je potrebné zistiť, z ktorého počítača sa do e-learningového kurzu pristúpilo iba jediný raz. Pri rozširovaní riešenia o zistenie unikátnych IP adries pomôže žiakom jednoduchá množinová operácia (rozdiel množín).

V závere kapitoly uvádzame ešte 2 doplňujúce zadania. V prvom z nich je potrebné overiť, či sa v priebehu online testu realizovanom v kurze v LMS Moodle neobjavil pokus o prihlásenie sa mimo priestorov učebne a teda či niektorý zo študentov nepodvádzal. Táto úloha je časovo náročnejšia a odporúčame ju riešiť so žiakmi ako súťaž dvojčlenných tímov.

Druhá doplňujúca úloha súvisí s hlbším uvažovaním o vlastnostiach údajovej štruktúry *set*. Žiaci dokončia a otestujú vlastnú objektovú implementáciu abstraktnej údajovej štruktúry množina pomocou zoznamu. Táto úloha je vhodná pre tých žiakov, ktorí už majú skúsenosť s písaním vlastných tried.

Ku všetkým úlohám má učiteľ k dispozícii vzorové riešenia s komentárom. Pre učiteľa uvádzame v závere kapitoly pomocný študijný text vysvetľujúci niektoré súvislosti logovania požiadaviek na webový server.

Kapitola o analýze logovacieho súboru dáva žiakom príležitosť riešiť praktický problém s využitím programovania ako nástroja na

riešenie problémov. Žiakov zároveň uvádza do oblasti dolovania znalostí z dát, ktorá je pre prax v informačnej spoločnosti objektívne stále dôležitejšia. V diskusii o formáte a obsahu logovacieho súboru žiaci lepšie porozumejú procesu logovania na strane webového servera ako aj významu jednotlivých fáz web log miningu – od výberu a transformácie dát, cez ich analýzu až po interpretáciu výsledkov. Porovnanie dvoch prístupov k riešeniu problému (pomocou zoznamu a pomocou množiny) dobre ilustruje vplyv správnej voľby údajovej štruktúry na efektívnosť riešenia a vedie žiakov ku kritickému mysleniu.

5. Overovanie predmetu

Navrhnúť nový predmet, ktorý by sa dal úspešne implementovať do školských vzdelávacích programov je náročné. Predmet musí byť obsahovo správny a reflektujúci aktuálne trendy v danej oblasti. Rovnako dôležitá je aj otázka metodiky vyučovania daného predmetu a motivácie žiakov.

Pri návrhu predmetu spolupracujeme s odborníkmi z praxe, ktorým vďačíme za pripomienky k odbornosti a aktuálnosti obsahu predmetu. V súčasnosti prebieha pilotné overovanie vybranými učiteľmi v reálnej výučbe na školách. Vďaka spätnej väzbe od overovateľov budeme vedieť upraviť náročnosť a vylepšiť metodiku vyučovania. Boli by sme radi, aby sa do overovania zapojilo čím viac odborníkov a učiteľov.

6. ZÁVER

V predmete Riešenie problémov a programovanie predstavujeme žiakom sériu problémov pri riešení ktorých vyžívame nástroje a silu programovania. Našim cieľom je ukázať žiakom, že programovanie nie je samoúčelná záležitosť, ale naopak. Vedieť programovať znamená vedieť analyzovať a riešiť množstvo reálnych a náročných problémov, z ktorých mnohé by bez programovania boli prakticky neriešiteľné.

V príspevku sme predstavili našu koncepciu predmetu a pre lepšiu ilustráciu sme popísali dve kapitoly z navrhovaného predmetu.

Momentálne prebieha overovanie predmetu učiteľmi v reálnej výučbe a pripomienkovanie odborníkmi z IT oblasti. Po spracovaní spätnej väzby a následnej úprave materiálov bude výsledný návrh predmetu a metodika jeho vyučovania prístupná učiteľom , žiakom a širokej verejnosti.

POĎAKOVANIE

Tento článok bol vytvorený v rámci Národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje a v rámci projektu KEGA 029UKF-4/2018 Inovatívne metódy vo výučbe programovania v príprave učiteľov a IT odborníkov. Ďakujeme učiteľom-overovateľom a aj konzultantovi Ing. Martinovi Fedorovi zo spoločnosti T-systems za ich cenné pripomienky a komentáre k metodickým a učebným materiálom predmetu Riešenie problémov a programovanie.

BIBLIOGRAFICKÉ ODKAZY

[1] *IT Akadémia – vzdelávanie pre 21. storočie* [online]. Bratislava: Centrum vedecko-technických informácií SR, 2018 [cit. 2018-04-02]. Dostupné z: http://itakademia.sk/sk/domov/

[2] Štátny pedagogický ústav. Inovovaný Štátny vzdelávací program. *Štátny pedagogický ústav* [online]. Bratislava, 2015 [cit. 2018-04-02]. Dostupné z: http://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/

[3] *Barefoot Computing* [online]. England, 2014 [cit. 2019-01-16]. Dostupné z: https://barefootcas.org.uk/

[4] MIKLOVIČOVÁ, Júlia, Andrea GALÁDOVÁ, Jakub VALOVIČ a Katarína GONDŽÚROVÁ. *Národná správa PISA 2015*. Bratislava: Národný ústav certifikovaných meraní vzdelávania, 2017, , 87. Dostupné také z: https://www.nucem.sk/dl/3482/NS_PISA_2015.pdf

[5] ŠKRINÁROVÁ, Jarmila a Patrik VOŠTINÁR. Aktivizujúce úlohy vplývajúce na rozvoj výpočtového myslenia. In: DRÁBKOVÁ a Ján BERKI. *Sborník konference Didinfo 2018* [online]. Liberec: Technická univerzita v Liberci, Univerzita Mateja Bela v Banskej Bystrici, 2018, s. 102-109 [cit. 2019-01-15]. ISBN 978 - 80 - 7494 - 424 - 6. ISSN 454 - 051X. Dostupné z: http://www.didinfo.net/images/DidInfo/files/Didinfo_2018.pdf

[6] GUNIŠ, Ján a Ľubomír ŠNAJDER. Návrh koncepcie výučby programovania v jazyku Python na gymnáziách. In: DRÁBKOVÁ a Ján BERKI. *Sborník konference Didinfo 2018* [online]. Liberec: Technická univerzita v Liberci, Univerzita Mateja Bela v Banskej Bystrici, 2018, s. 63-70 [cit. 2019-01-15]. ISBN 978 - 80 - 7494 - 424 - 6. ISSN 454 - 051X. Dostupné z: http://www.didinfo.net/images/DidInfo/files/Didinfo_2018.pdf

[7] *Cieľové požiadavky na vedomosti a zručnosti maturantov z informatiky*. In: . Bratislava: Štátny pedagogický ústav, 2016. Dostupné také z: http://www.statpedu.sk/files/articles/nove_dokumenty/cielove-poziadavky-pre-mat-skusky/informatika.pdf

[8] BUDÍKOVÁ, Marie, Maria KRÁLOVÁ a Bohumil MAROŠ. *Průvodce základními statistickými metodami*. Havlíčkův Brod: Grada Publishing, 2010. ISBN 978-80-247-3243-5.

[9] ELLIS, Keith M. *The Monty Hall Problem* [online]. 1995 [cit. 2019-01-15]. Dostupné z: https://www.montyhallproblem.com/

[10] BEECHER, Karl. *Computational thinking: A beginner's guide to problem-solving and programming*. Dorchester, United Kingdom: Dorset Press, 2018. ISBN 9781780173641.

Rozšírená realita vo vzdelávaní
Augmented reality in education

Ing. Dana Horváthová, PhD.
Katedra informatiky FPV UMB
Tajovského 40
974 01 Banská Bystrica
Slovensko
dana.horvathova@umb.sk

Miroslav Blšák
Katedra informatiky FPV UMB
Tajovského 40
974 01 Banská Bystrica
Slovensko
mblsak@studenti.umb.sk

ABSTRACT
In this paper, we describe the augmented reality, its definition, its future and especially its use in education. Augmented reality has a huge potential in visualizing spatial objects in the real world. We tried to use it in the application we created for mathematics education to display Platonic bodies, more precisely in geometry in primary schools. Our goal was to point out the possibilities of augmented reality to demonstrate these bodies in space with the possibility of their display, rotation and viewing from different angles. The application is usable via OS Android smartphone.

Keywords
Augmented reality. Education. Mathematics

ABSTRAKT
Článok približuje problematiku rozšírenej reality, venuje sa jej definícií, budúcnosti, no najmä jej využitiu vo vzdelávaní. Rozšírená realita má obrovský potenciál pri vizualizácií priestorových objektov v reálnom svete. Snažili sme sa ju využiť v aplikácií, ktorú sme vytvorili pre potreby výučby predmetu matematika, presnejšie geometria na základných školách na zobrazenie Platónových telies. Naším cieľom bolo poukázať na možnosti rozšírenej reality, demonštrovať tieto telesá v priestore s možnosťou ich zobrazenia, otáčania, prehliadania z rôznych uhlov pohľadu. Aplikácia je využiteľná prostredníctvom smartfónu s OS Android.

Kľúčové slová
Rozšírená realita. Vzdelávanie. Matematika

ÚVOD
Prudký vývoj digitálnych technológií zasahuje do všetkých oblastí nášho života. Silne ovplyvňuje aj vzdelávanie a núti učiteľov na všetkých stupňoch škôl neustále sa zaoberať myšlienkou, ako sprístupniť nové poznatky svojim žiakom a ako využiť nové technológie, metódy a postupy pre dosiahnutie vyššej kvality vzdelávania ďalších generácií. Jednou z takých progresívnych a markantne sa rozvíjajúcich technológií je rozšírená realita (augmented reality – AR[1]), ktorá sa nenápadne, v tieni virtuálnej reality (virtual reality – VR), prediera na svetlo sveta a zaujíma tam dôležité miesto. Niektorí však vidia budúcnosť v zlúčení obidvoch týchto realít do jedného celku, do tzv. zmiešanej reality (mixed reality – MR), ktorá predstavuje koláž reálneho a virtuálneho sveta s cieľom vytvoriť nové prostredie a vizualizácie, kde fyzické a digitálne objekty koexistujú a interagujú v reálnom čase.

1 ROZŠÍRENÁ REALITA A JEJ DEFINÍCIA
Rozšírená realita je v posledných rokoch veľmi intenzívne skúmaná, vyvíjaná a implementovaná do viacerých sfér a zdá sa, že ešte ani zďaleka neukázala svoj maximálny potenciál. Je to významný fenomén, ktorý preniká do rôznych oblastí ľudskej činnosti, predovšetkým marketingu, reklamy, či zábavného priemyslu. (1)

Keďže je stále považovaná za špecifickú, inovatívnu a vyvíjajúcu sa technológiu, je zložité vytvoriť jej presnú a jednotnú definíciu. Vzhľadom na uvedenú skutočnosť je zrejmé, že samotná definícia je závislá od kontextu alebo metódy implementácie (2).

Na to, čo AR je, sa možno pozrieť z viacerých uhlov pohľadu. Možno ju chápať ako nadstavbu skutočného sveta doplnenú o vizuálne prvky, zvuk alebo iné senzorické podnety. Digitálna informácia sa potom zobrazuje ako časť reálneho sveta, prinajmenšom v používateľskej perspektíve (3).

AR je možné považovať za technológiami navodené percepčné prostredie založené na kombinácií percipovaného reálneho prostredia a pridaných, počítačom vygenerovaných prvkov. (4)

Teda môžeme AR chápať ako integráciu digitálnych informácií do používateľského prostredia v reálnom čase. Objekty skutočného prostredia sú doplnené o dodatočné informácie v digitálnej podobe prostredníctvom realitu snímajúceho zariadenia (mobilný telefón, tablet, špeciálne okuliare a pod.). Obohatenie reality sa obvykle deje v sémantickom kontexte s časťami prostredia.

2 ROZŠÍRENÁ REALITA A JEJ BUDÚCNOSŤ
Do virtuálnej a rozšírenej reality sa v poslednej dobe investuje viac finančných prostriedkov ako kedykoľvek predtým. Pozrime sa napríklad na niekoľko zaujímavých predikcií, ktoré sa môžu stať realitou už o pár rokov. Prieskumy poukazujú na najsilnejšie odvetvia z hľadiska potenciálneho príjmu v roku 2025. Podľa viacerých expertov bude prvý herný priemysel so sumou približne 11,6 miliárd dolárov. Prvú trojicu dopĺňa zdravotníctvo so sumou 5,1 miliárd dolárov a inžinierstvo so sumou 4,7 miliárd dolárov. Bohužiaľ neprekvapujúca je pozícia vzdelávania, ktorá je so sumou

[1] V našom článku budeme používať anglickú skratku AR, nakoľko je viacej zaužívaná ako jej slovenský ekvivalent RR.

7 miliónov amerických dolárov na poslednom mieste v tomto prieskume (5).

3 ROZŠÍRENÁ REALITA VO VZDELÁVANÍ

Existuje viacero rôznych metód vzdelávania, trénovania, či učenia (sa), ktoré sú prispôsobené potrebám konkrétneho jedinca. Tieto metódy sú rozpracované v učebných materiáloch, počítačových softvéroch, aplikáciách prípadne iných technologických nástrojoch (6).

Moderné školstvo si vynucuje potrebu inovovať tradičné postupy a učebné metódy a podrobiť ich skúmaniu z hľadiska informačnej dimenzie, formy informačného obohatenia vnímaného prostredia, formy interaktivity, či schopnosti rozlišovania a uvedomenia si hranice medzi skutočným svetom a pridanými digitálnymi prvkami. (1)

Vizuálna percepcia je kľúčovým komponentom na pochopenie, prenos a zapamätávanie si informácií. Podľa pyramídy učenia (the Cone of Experience) amerického pedagóga Edgara Dalea sa uvádza, že si pamätáme 10% z toho, čo čítame, 50% z toho, čo vidíme a počujeme a 90% z toho, čo robíme, vykonávame vlastnou činnosťou. Zobrazovaním digitálnych informácií do reálneho prostredia a interakcie s nimi aplikujeme najspodnejšiu zložku Dalovej pyramídy učenia, dôsledkom čoho je väčšia šanca zapamätania si učiva. Z toho vyplýva, že práve školstvo je jedným z odvetví, kde má rozšírená realita obrovský potenciál uplatniť sa.

4 APLIKÁCIA „PLATÓNSKE TELESÁ“

Zámer využiť tento potenciál začal hľadaním medzier v oblasti vzdelávania, kde zatiaľ AR neprenikla. Prieskum v oblasti výučby priestorovej geometrie a geometrických útvarov na základných školách ukázal, že niektoré zložitejšie geometrické štruktúry nie sú zaradené do výučby predmetu Matematika práve pre náročnosť ich vizualizácie. Takýmito útvarmi sú napríklad Platónske telesá, ktorých steny sú tvorené zhodnými pravidelnými mnohouholníkmi a z každého vrcholu vychádza rovnaký počet hrán. Platónskych telies existuje v trojrozmernom euklidovskom priestore práve päť a vzhľadom k ich vysokej symetrii sa bežne objavujú v kryštalografii, molekulárnej fyzike, či chémii.

Obrázok 1 Hlavné menu aplikácie. Zdroj: [autor]

Tento náš prieskum vyústil do vytvorenia aplikácie, ktorá sa venuje téme Platónskych telies. Naším cieľom bolo poukázať na možnosti AR demonštrovať tieto telesá v priestore, s možnosťou ich zobrazenia, otáčania, prehliadania z rôznych uhlov pohľadu.

4.1 Vývoj aplikácie

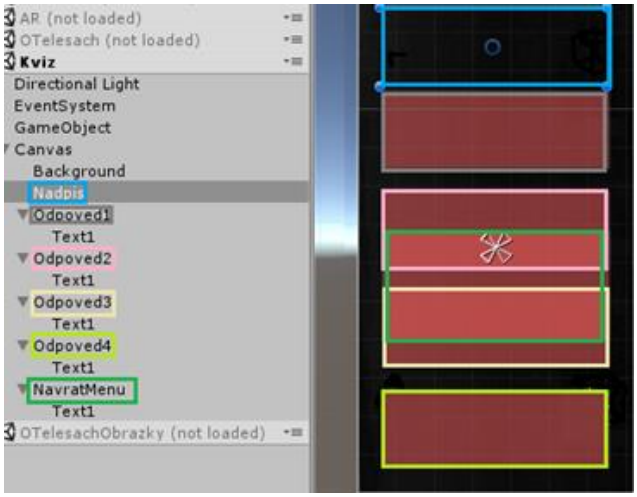
Na tvorbu aplikácie sme použili herný softvér Unity3D. Program je k dispozícii v rôznych verziách (platených alebo neplatených) na stránke www.unity3d.com v jej hlavičke (Get Unity). Verzia softvéru bola v čase tvorby aplikácie 2017.4.0f1 a to pre 64 bitovú architektúru. Pre prácu s rozšírenou realitou sme používali knižnicu Vuforia. Od verzie Unity 2017.2 je Vuforia priamo integrovaná do Unity3D.

Tvorba aplikácie bola komplexná a náročná činnosť a tu sa pokúsime len veľmi stručne opísať postup krokov.

Po nainštalovaní programu Unity 3D sme importovali knižnicu Vuforia. Vytvorili sme nasledovnú hierarchiu scén.

Obrázok 2 Hierarchia scén. Zdroj: [autor]

Scéna s menom AR je scéna, v ktorej prebieha samotné zobrazenie 3D modelov a môžeme ju nájsť v knižici Vuforia**.** Do každej scény vkladáme GameObjecty. Sú to objekty (2D alebo 3D) tvoriace obsah scény. Jedná sa o rôzne grafické prvky, ktoré môžu byť viditeľné a neviditeľné. Ako príklad uvedieme scénu Kvíz.



Obrázok 3 Štruktúra scény Kviz. Zdroj: [autor]

Pod *GameObject-om* typu *canvas* rozumieme ohraničenú plochu, v našom prípade je to zobrazovacia plocha, ktorá bude viditeľná na mobilnom displeji. Pod *Background-om* typu *image* máme označené pozadie, pričom sa jedná o čierny obdĺžnik. Odpoveď1 až Odpoveď4 sú objekty typu *button*, ktoré slúžia na interakciu používateľ - aplikácia. *Nadpis* a *Text1* v každom tlačidle sú už samotné texty. Týmto štýlom fungujú všetky scény, sú vytvorené z podobnej štruktúry.

Aby tlačidlá (ale aj iné objekty) vôbec fungovali, musíme im priradiť nejakú činnosť. Túto činnosť naprogramujeme v jazyku c#. Jedná sa o skripty, ktoré majú rôzne funkcie – zobraziť, aktivovať, vypísať, uložiť, skontrolovať. Každému *GameObjectu* musíme priradiť metódu obsiahnutú v skripte.

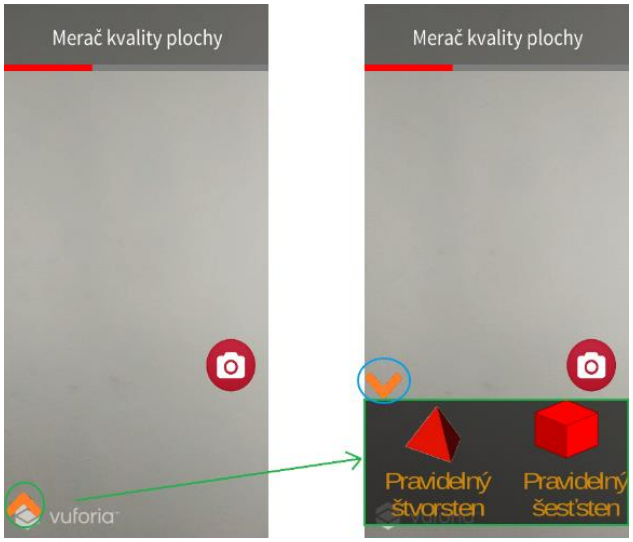
Finálnym krokom bolo vyplnenie údajov potrebných na beh programu, ako napr. názov aplikácie, ikona aplikácie, verzia Androidu a iné špecifikácie a následná kompilácia *(buildovanie).* Celá aplikácia bola vytváraná pre mobilné zariadenia, konkrétne pre smrtfóny s OS Android vo verzií 4.4 a vyššie v čierno-oranžovej farbe. Po spustení aplikácie sa zobrazí načítavacia

obrazovka, ktorá pozostáva z textu „Načítavanie ...%“, pozadia, načítavacej lišty a piatich Platónskych telies na spodnej časti obrazovky. Lišta sa postupom času napĺňa a pri úspešnom načítaní všetkých potrebných zdrojov pre beh aplikácie sa vyplní celá oranžovou farbou.

4.2 Ovládanie aplikácie

Po načítaní sa aplikácia prepne do hlavného menu, ktoré pozostáva zo 4 tlačidiel. Tie sú usporiadané podľa logickej chronológie (Obrázok 1 Hlavné menu aplikácie. Zdroj: [autor]). Používateľské rozhranie je prispôsobené pohodlnému ovládaniu aplikácie. V celej aplikácií funguje mobilné hardvérové, poprípade softvérové tlačidlo späť, ktorým sa dostaneme z akejkoľvek scény do hlavného menu. Výnimku tvorí kvíz, z ktorého sa do ukončenia testu nemožno vrátiť a scéna *Časť obrázky*, ktorou sa dostaneme späť k základným informáciám o telesách.

Po kliknutí na *AR Kamera* sa dostaneme do časti programu, ktorá umožňuje ukotvenie zobrazenia Platónskych telies do priestoru na základe zosnímania kotviacej plochy.

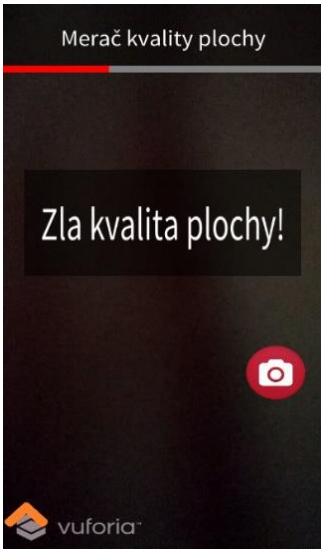


Obrázok 4 Scéna, v ktorej prebieha zobrazovanie 3D modelov. Zdroj: [autor]

V hornej časti obrazovky sa nachádza meradlo kvality plochy pozostávajúce z textu a samotného meradla, ktoré má zakomponované tri farby: červenú, žltú a zelenú. Tieto farby symbolizujú, či je daná plocha dostatočne heterogénna (červená symbolizuje príliš homogénnu plochu). Zameraním kamery (napr. na mobilnom telefóne) na príslušnú plochu a stlačením ikony fotoaparátu v červenom kruhu, sa objekt zobrazí na danú plochu. V prípade, že svieti iba červená farba a chceme zobraziť 3D model, zobrazí sa chybová hláška s textom: „Zlá kvalita plochy!“ (Obrázok 5 Chybová hláška pri homogénnej ploche. Zdroj: [autor]).

V spodnej časti sa nachádza oranžová šípka, pomocou ktorej vieme otvoriť množinu Platónskych telies. Šípkou otočenou smerom nadol ju naopak vieme skryť.
Šípkou smerom nahor otvoríme zoznam telies, v ktorom sa môžeme pohybovať smerom doprava a doľava v horizontálnom smere, pričom po dotyku na určité teleso sa nám toto teleso vyberie na zobrazenie. Samotným stlačením ikony fotoaparátu ho zobrazíme (Obrázok 6 Príklad zobrazenia objektu na plochu. Zdroj: [autor]).

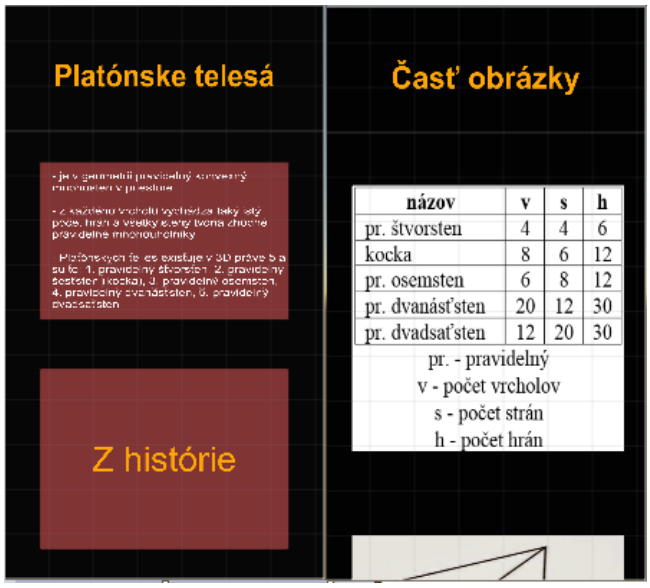
Stlačením tlačidla *Opis telies* v hlavnom menu na úvodnej obrazovke sa presunieme do scény s informáciami. Tu sa nachádza niekoľko tlačidiel. Kliknutím na dané tlačidlá prepíname medzi ich nadpismi a informáciami, ktoré jednotlivé tlačidlá obsahujú. Posúvanie sa vo vertikálnom smere je umožnené priložením prsta k obrazovke a posúvaním smerom nahor alebo nadol.



Obrázok 5 Chybová hláška pri homogénnej ploche. Zdroj: [autor]

Obrázok 6 Príklad zobrazenia objektu na plochu. Zdroj: [autor]

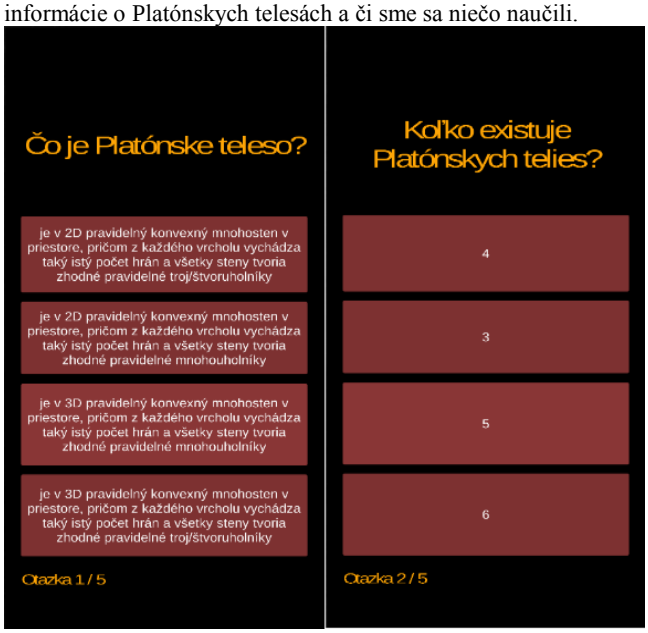
Opis telies obsahuje nadpis a tlačidlá. Na tlačidlách, ktoré sú podfarbené červenou farbou, sa nachádzajú dôležité informácie o Platónskych telesách, napríklad ich definícia, príklady vyskytovania v prírode či dualizmus Platónskych telies. V scéne s obrázkami sa nachádzajú dva obrázky, kde sú informácie o každom jednom telese (počet vrcholov, počet strán a počet hrán) a ukážka dualizmu pre každé jedno teleso.



Obrázok 7 Scéna obsahujúca informácie o Platónskych telesách. Zdroj: [autor]

Obrázok 8 Ukážka zmeny textu po kliknutí. Zdroj: [autor]

Poslednou možnosťou výberu z menu je časť *Otestuj sa*, kde je umiestnený krátky kvíz, v ktorom môžeme nájsť jednotlivé otázky. Pri každej otázke máme možnosť vybrať si zo štyroch odpovedí a zistiť tak rýchlo spätnú väzbu, či sme pozorne sledovali informácie o Platónskych telesách a či sme sa niečo naučili.

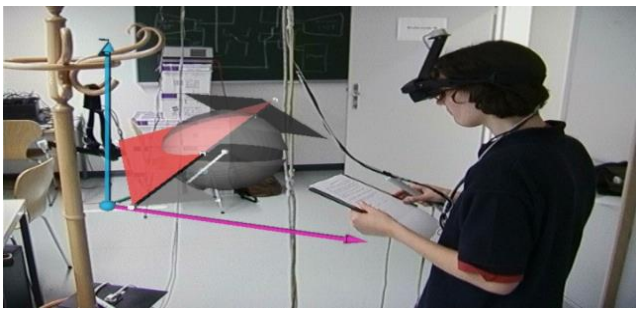


Obrázok 9 Scéna s testovaním. Zdroj: [autor]

Aplikácia zatiaľ nie je dostupná pre verejnosť, jedná sa o testovaciu/demonštračnú aplikáciu, ktorá má ukázať možnosti využitia rozšírenej reality vo výchove a vzdelávaní. Aplikácia bola otestovaná viacerými ľuďmi, medzi ktorými boli žiaci základných škôl, študenti vysokých škôl aj dospelí ľudia. Ich názor je v podstate jednotný - aplikácia sa im páčila, mali pozitívne ohlasy a radi by videli, keby sa vzdelávanie uberalo aj týmto smerom.

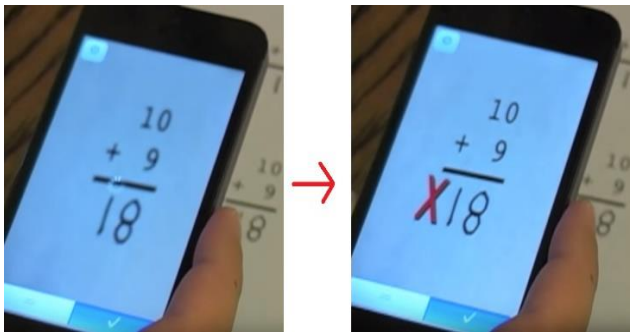
5 INÉ APLIKÁCIE

Aplikácia nazývaná Construct3D, ktorá bola vytvorená pre matematikov, špeciálne pre potreby geometrie s 3D geometrickými modelmi. Aplikácia dovoľuje viacerým užívateľom (učiteľ, žiaci) mať spoločný virtuálny priestor pre konštrukciu geometrických objektov. Všetko je realizované pomocou dátovej prilby, tzv Head Mounted Display – HMD (6).



Obrázok 10 Ukážka fungovania aplikácie Construct3D (7)

Ďalšia aplikácia, ktorá sa nazýva Pocket Tutor - Math Help, nachádza uplatnenie pri riešení domácich úlohách, učení sa na test alebo pochopení matematických súvislostí. Môžeme pomocou nej kontrolovať správnosť výsledkov, určiť výsledok, zobraziť a pozerať edukatívne videá bez internetového pripojenia a množstvo ďalších činností.



Obrázok 11 Ukážka fungovania aplikácie Pocket tutor - Math Help (8)

ZÁVER

Cieľom nášho článku bolo upriamiť pozornosť na možnosti prudko sa rozvíjajúcej technológie rozšírenej reality a priblížiť mobilnú aplikáciu na demonštráciu Platónskych telies. Veríme, že si naša aplikácia nájde uplatnenie v oblasti matematiky, resp. geometrie a snáď poslúži aj ako inšpirácia ku vzniku podobných aplikácií vzdelávacieho charakteru s využitím rozšírenej reality.

POĎAKOVANIE

Tento príspevok vznikol za podpory projektu „E-Talent“ – Nadácia Tatra Banky a bol vytvorený aj v rámci národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje.

BIBLIOGRAFICKÉ ODKAZY

[1] JEŘÁBEK, T. *Rozšířená realita ve vzdelávaní. Didaktická specifika rozšířené reality.* Praha : Univerzita Karlova, Pedagogická fakulta, 2015. ISBN 978-80-7290-829-5.

[2] JUNG, T., DIECK, C**.** *Augmented Reality and Virtual Reality: Empowering Human, Place and Business.* s.l. : Springer, 2018.

[3] SCHMALSTIEG, D., HÖLLERER, T. *Augmented Reality Principles and Practice.* 1. vyd. Boston : Addison Wesley, 2016. p. 528. ISBN 978-0-321-88357-5.

[4] AZUMA, R. T. *A Survey of Augmented Reality.* s.l. : Presence: Teleoperators and Virtual Environments 6 [online], 1997.

[5] Obrázok *Rozšírená realita a priemysel.* [Online] [Dátum: 1. 12. 2018.] https://cdn-images-1.medium.com/max/800/0*n999S19vX_nwVszY.png.

[6] LEE, K. *Augmented Reality in Education and Training.* [Online] 2012. [Dátum: 10. 11. 2018.] https://pdfs.semanticscholar.org/070d/2e3d8d781f2316fceeafabd3d8f059288d28.pdf.

[7] Obrázok *Aplikácie Construct3D.* [Online] [Dátum: 1. 12. 2018.] https://www.cg.tuwien.ac.at/research/vr/studierstube/construct3d/images/lisi-modell5-1.jpg.

[8] YouTube Video *Pocket Tutor - Math Help.* [Online] [Dátum: 1. 12. 2018.] https://www.youtube.com/watch?v=tDDkSYLF3vM.

Blokové programovanie v rámci informatického krúžku
Block programming within the informatics extra-curricular lessons

Dana Horváthová
Katedra informatiky FPV UMB
Tajovského 40
974 01 Banská Bystrica
Slovensko
dana.horvathova@umb.sk

Jaroslav Knežník
Katedra informatiky FPV UMB
Tajovského 40
974 01 Banská Bystrica
Slovensko
jkneznik@studenti.umb.sk

ABSTRACT
This article deals with block programming activities for middle school students. We have prepared a few interesting methods the suitability of which has been verified at the Informatics club (an extracurricular activity of middle school students) at the Informatics Department of Faculty of Natural Sciences at Matej Bel University in Banska Bystrica. Educational methodology was based on activities such as Introduction to Scratch environment, where students tried to work with this editor and learn its basics. They had the possibility to program the Phiro robot in offline Scratch editor and to see if they programmed it correctly. Eventually students worked with microcomputers called BBC Micro:bit and used a Radio function to communicate between the robots. All the methods were successfully verified via observation and the following interview with the students.

Keywords
Scratch. Phiro. BBC Micro:bit. Block programming. Educational methodology.

ABSTRAKT
Článok sa zaoberá ukážkami aktivít s využitím blokového programovania pre žiakov 2. stupňa ZŠ. Pripravili sme niekoľko metodík, ktoré predstavujeme v našom článku a ktorých vhodnosť sme overovali v rámci informatického krúžku na Katedre informatiky Fakulty prírodných vied Univerzity Mateja Bela v Banskej Bystrici. Metodiky boli postavené na aktivitách, kde žiaci mali možnosť vyskúšať si prácu v editore Scratch a naučiť sa základy práce s ním. V offline editore Scratch programovali Phiro robota, kde mali možnosť názorne vidieť, či zostavili program správne. Nakoniec pracovali s mikropočítačmi BBC Micro:bit, kde využívali rádiové spojenie a dorozumievanie sa medzi robotmi navzájom. Všetky metodiky boli úspešne overené prostredníctvom pozorovania a rozhovoru so žiakmi.

Kľúčové slová
Scratch. Phiro. BBC Micro:bit. Blokové programovanie. Vzdelávacie metodiky.

1 ÚVOD
Na katedre informatiky FPV UMB v Banskej Bystrici sa druhý rok realizuje informatický krúžok[1], ktorý má ambíciu podchytiť talentované deti a poskytnúť im príležitosť zábavnou formou pracovať s modernými digitálnymi technológiami. Na krúžku sa v súčasnosti pracuje so softvérovými aplikáciami typu Scratch, Kodu Game Lab, Microsfot Makecode, so softvérom pre tvorbu mobilných aplikácií, ale aj s využitím hardvéru a robotov ako Phiro roboty, Lego roboty, dron, mikropočítače BBC Micro:bit a iné. V rámci krúžku si viacerí študenti učiteľského štúdia informatiky, ale aj študenti aplikovanej informatiky skúšajú svoje pedagogické zručnosti a venujú sa individuálne (jeden študent na max. dvoch až troch žiakov) žiakom z banskobystrických základných škôl, ktorí sa prihlásili na tento krúžok. (Horváthová, 2018), (Jacková, 2018) Rastúci záujem o spomínaný krúžok poukazuje na jeho kvalitu a úspešnosť.

Počas nášho pozorovania a následného rozhovoru so žiakmi navštevujúcimi informatický krúžok sme zistili, že na viacerých základných školách nie je programovanie bežnou súčasťou informatiky, a preto sa nám zdá, že na viacerých základných školách na Slovensku je vyučovanie programovania ako súčasť predmetu informatika, ešte stále na nedostatočnej úrovni. Následkom toho sú rozdiely v programátorských skúsenostiach medzi žiakmi veľmi výrazné. Práve preto sme sa rozhodli vytvoriť niekoľko metodík, ktoré využívajú blokové programovanie. Naším cieľom bolo vytvoriť úlohy, ktoré budú jednak rozvíjať algoritmické myslenie prostredníctvom blokových editorov, ako aj programátorské schopnosti premietajúce sa do samotnej práce s robotom alebo iným podobným zariadením.

2 BLOKOVÉ PROGRAMOVANIE
Blokové programovanie patrí medzi najvyužívanejšie formy programovania detských programovacích jazykov, je vytvorené predovšetkým k ľahšiemu pochopeniu problematiky programovania a jej jednoduchšiemu zvládnutiu. Podporuje počiatočné kroky vo výučbe programovania, algoritmické myslenie a obsahuje jednoduché prostredie pre vstup do programátorského sveta. Široké spektrum úloh, ktoré sa dajú vytvoriť pomocou blokových editorov je vhodné pre žiakov viacerých vekových kategórii. Blokové programovanie (BÉLIK, 2018) pozostáva z dielikov navrhnutých tak, aby každý zapadol na určené miesto. Dielik obsahuje iba jeden príkaz a pospájaním viacerých dielikov dostaneme postupnosť alebo viacero postupností príkazov, ktoré spolu vytvárajú program. Tieto dieliky sú rozdelené do skupín ako pohyb, vzhľad, zvuk atď., okrem toho sú farebne rozlíšené, čo zvyšuje prehľadnosť a vyberanie dielikov sa časom stáva intuitívne. Pomocou blokového programovania vieme vytvárať nielen hry, ale aj animácie a interaktívne príbehy.

[1] Autorkou myšlienky informatického krúžku na KI FPV UMB bola absolventka KI FPV UMB Mgr. Nika Klimová

V súčasnosti už jestvuje množstvo online aj offline editorov dostupných na internete. Medzi najvýznamnejšie softvérové nástroje blokového programovania patrí napríklad Scratch, ktorý je celosvetovo rozšírený a obrovský úspech má aj na Slovensku. Okrem softvérových nástrojov je tu aj široká podpora prepojenia na hardvér, ktorý je možné programovať tiež cez blokové editory ako Microsoft Makecode.

V blokovo orientovaných programovacích prostrediach akým je napríklad aj Scratch, (4) môže žiak vidieť všetky bloky hneď na začiatku jeho práce. Preto ak je žiak úplný začiatočník, tak už pri zdanlivo jednoduchej úlohe sa v prostredí nedokáže zorientovať a tak netuší, ktoré bloky má z tohto množstva blokov vybrať. Žiak tu zbytočne stráca čas hľadaním správnych blokov a následným skúšaním. Z tejto situácie ponúkajú východisko rôzne tutoriály alebo hry obsahujúce blokovo orientované prostredia, ktoré sú dostupné na internete.

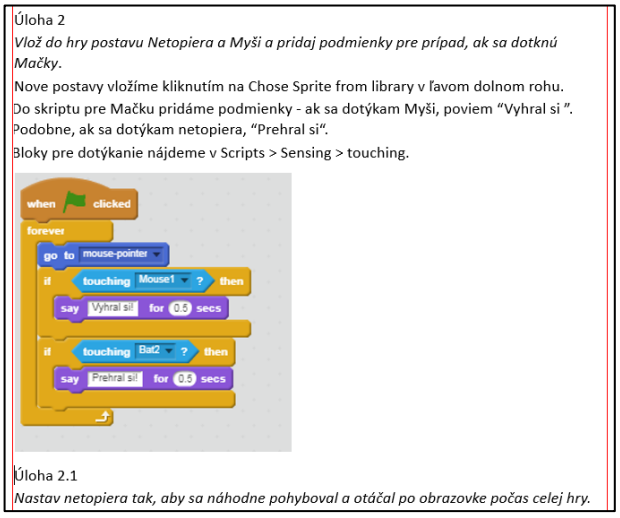
Tieto tutoriály ponúkajú úlohy v ktorých žiak dostane k dispozícii len niekoľko blokov, tie musí použiť v správnom poradí, pričom má možnosť ich kopírovať. Úlohy sú zoradené podľa náročnosti od tých najjednoduchších a v každej ďalšej úlohe pribúdajú zložitejšie bloky. Takéto úlohy vedia byť žiakovi nápomocné pri rýchlejšom zoznámení sa s prostredím a pri lepšom pochopení funkcií blokov.

3 METODIKY

Metodiky, ktoré sme overovali v rámci informatického krúžku, sú založené na blokovom programovaní a využívajú, napríklad Phiro robota, mikropočítače, ale taktiež softvérové programy ako Scratch offline editor alebo editor Microsoft Makecode. V našom článku sme predstavili len niektoré zaujímavé ukážky úloh, ktoré sa nám najviac osvedčili a ktoré sa deťom najviac páčili. Do úloh sa zapájali prevažne žiaci druhého stupňa, ktorí už majú s programovaním skúsenosti a vedia algoritmicky myslieť. Na začiatku aktivity mali žiaci možnosť vidieť ukážku hry, ktorú majú vytvoriť, ako sa má správať robot alebo ako majú vyzerať komunikujúce mikropočítače. Vždy s komentárom týkajúcim sa požadovanej funkčnosti. Ako formu spätnej väzby sme pri overovaní metodík využívali predovšetkým metódu pozorovania žiakov počas toho ako pracovali, zodpovedali sme na prípadné otázky a metódou rozhovoru sme zisťovali, či sa žiakom úloha páčila, či bola zvládnuteľná alebo či sa niečo nové naučili. Až po úspešnom overení navrhnutých metodík môžu byť rozdiferencované na učiteľské metodické a žiacke pracovné listy. V metodikách sme využívali anglické verzie softvérových editorov a prostredí, nakoľko žiakom práca s nimi nerobila žiadny problém a navyše vďaka pôvodným verziám sa u žiakov rozvíjali aj anglické jazykové zručnosti.

3.1 Metodika 1 – Úvod do prostredia Scratch

Metodika bola vytvorená pre žiakov 6. až 9. ročníka. Je založená na naprogramovaní hry v programe Scratch. V tejto hre *Mačka* naháňa *Myš* a pritom sa vyhýba *Netopierovi*, ktorý ju prenasleduje. Hra sa ovláda šípkami, využíva 3 objekty, z ktorých každý má svoj Skript a využíva cykly a podmienky.



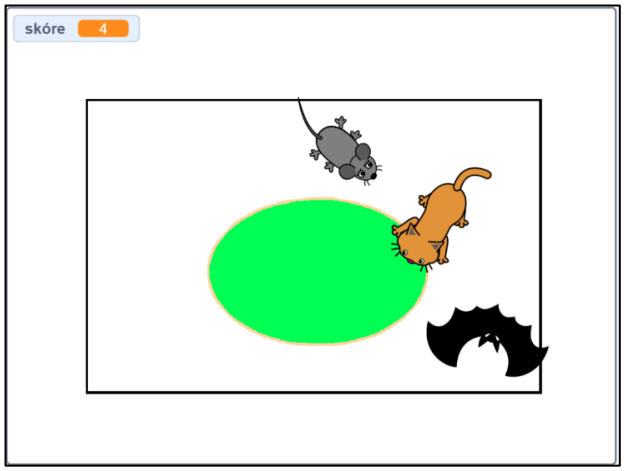
Obrázok 1 Ukážka metodiky 1 - Úvod do prostredia Scratch

3.1.1 Overenie metodiky 1 – Úvod do prostredia Scratch

Žiaci pracovali jednotlivo a každý žiak mal k dispozícii počítač s internetom. Boli to prevažne žiaci 8. ročníka ZŠ, ktorí ešte nepracovali v programe Scratch alebo len veľmi okrajovo. Jednotlivé časti zadania boli žiakom dôkladne vysvetlené. Podmienkam a vetveniu žiaci spočiatku nerozumeli, no po vysvetlení na väčšiu časť úloh prišli aj sami. Zadanie sa im páčilo a vytváranie vlastnej hry ich veľmi bavilo.

V tejto metodike sme sa zamerali predovšetkým na vysvetlenie základných informácii a ich vyskúšanie, lebo nie každý už pracoval v programe Scratch, a preto bolo potrebné postaviť pevné základy. Žiakom sa podarilo splniť všetky zadané úlohy okrem bonusovej v stanovenom čase 2 vyučovacie hodiny.

Obrázok 2 Overenie metodiky 1



Obrázok 3 Ukážka hry

3.2 Metodika 2 – Alarm

Pri tejto metodike sme pracovali s mikropočítačom BBC Micro:bit. Podstatou je prepojenie dvoch Micro:bitov pomocou funkcie Radio, kde by mal jeden zareagovať rozsvietením a zabzučaním, ak by sa niekto pokúsil pohnúť druhým. Jeden z mikropočítačov využíva periférne zariadenia ako Led dióda a malý reproduktor.

Úloha 1

Vytvor poplašný systém pre svoj Micro:bit. Pomocou funkcie Rádio prepoj dva Micro:bity tak, aby prvý poslal správu druhému, ak sa ho bude snažiť niekto ukradnúť. Pripoj si svoj prvý Micro:bit k externému zdroju alebo počítaču. Na svoj druhý Microbit pripoj LED diódu a reproduktor, aby po prijatí správy zahral nejaký tón a rozsvietil diódu. Micro:bity rozmiestni od seba vo vzdialenosti do 10m a sleduj ako funguje tvoj poplašný systém. (Pre odoslanie správy môžeš použiť funkciu zatrasenia).

Žiaci si prečítajú zadanie a Učiteľ si overí či zadaniu porozumeli. Vysvetlí prípadné otázky, ako funguje funkcia Rádio. Pomôže žiakom správne pripojiť diódu, reproduktor, poprípade externý zdroj. Ďalej Učiteľ kladie žiakom otázky a slovne ich navádza na správne riešenie. Učiteľ pomáha žiakom prísť na riešenie na obrázkoch.

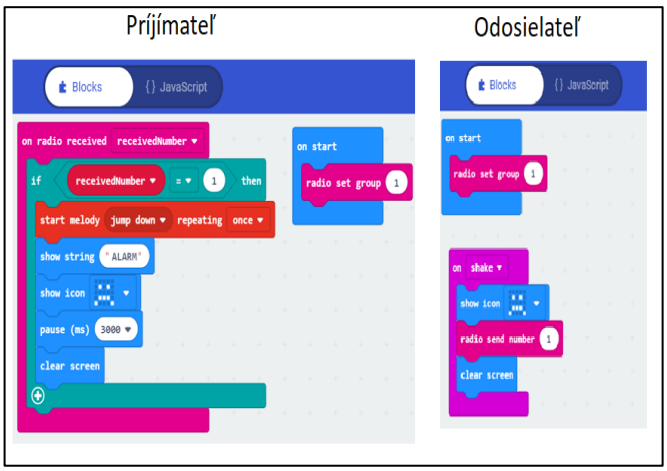
Obrázok 4 Ukážka Metodiky 2 - Alarm

3.2.1 Overenie metodiky 2 – Alarm

Žiakov zadanie veľmi zaujalo, používali dva Micro:bity, ktoré sa medzi sebou dorozumievali. Žiaci sami vraveli, že ak môžu využívať ďalšie pripájacie zariadenia, teda keď to bliká alebo hrá, tak je to pre nich oveľa zaujímavejšie.

Obrázok 5 Overenie Metodiky 2

Menšie problémy nastali pri pochopení toho, že jeden Micro:bit je vysielač a druhý prijímač, to znamená, že potrebujeme dva rôzne kódy. Ďalší problém bol pri použití červenej Led diódy, pretože Micro:bit nemá funkciu, ktorá by rozsvietila diódu, tak ako je to pri reproduktore a preto k nej potrebujeme pristupovať cez „piny“.



Obrázok 6 Ukážka programu

3.3 Metodika 3 – Ovládanie Phiro robota

Táto metodika je zameraná na ovládanie robota pomocou klávesnice cez offline editor Scratch 2.0. Úlohou robota je, aby sa po stlačení kláves pohyboval daným smerom, prípadne využil svetelné Led diódy. Pri programovaní sa využíva dodatok Phiro, ktorý je vložený do tohto editora. Keďže dodatok nemá predvolené pohyby, je pre prehľadnosť kódu dobré využiť vlastné bloky príkazov, kde sa tieto pohyby zadefinujú.

3.3.1 Overenie metodiky 3 – Ovládanie Phiro robota

Na overovanie tejto metodiky sa nám pre nedostatok počítačov s Bluetooth pripojením prihlásili len dvaja žiaci 7. ročníka.

Úloha 4

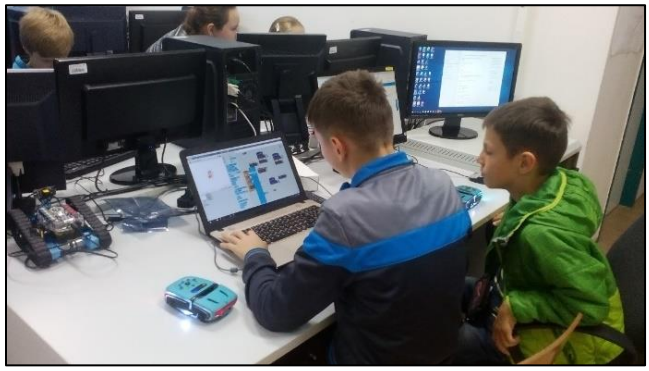
Pre prehľadnosť tvojho kódu si zadefinuj vlastný blok príkazov, do ktorého si vložíš príkazy pre daný pohyb. Choď na Scripts > Make a block > „turn right“. Celý tento blok príkazov potom použi ako jeden príkaz v tvojom hlavnom kóde s podmienkami. Toto zopakuj pre všetky ostatné pohyby.

Žiak si má podľa obrázka zadefinovať vlastné bloky príkazov, ktoré potom použije ako jeden príkaz.

Obrázok 7 Ukážka Metodiky 3 – Ovládanie Phiro robota

Obidvaja už mali skúsenosti s blokovým programovaním aj s vývojovým editorom programu Scratch, v ktorom pracovali aj tentokrát. Žiaci rýchlo porozumeli zadaniu úlohy aj novým príkazom implementovaným do programu Scratch.

Problém nastal pri vytváraní vlastných blokov príkazov. Žiaci nevedeli sami prísť na to, ako majú bloky vytvárať a na čo vlastne tieto bloky slúžia. Preto sme im museli pomôcť s vytvorením prvého bloku. Ostatné bloky už ďalej hravo zvládali sami a úlohu sa im podarilo dokončiť v stanovenom čase.



Obrázok 8 Overenie metodiky 3

4 SPÄTNÁ VÄZBA

Pri všetkých metodikách sme od žiakov dostávali spätnú väzbu jednak formou pozorovania, ako aj prostredníctvom rozhovoru. Zameriavali sme sa hlavne na otázky typu :

- Páčila sa ti dnešná úloha ?
- Čo sa ti na nej najviac páčilo ?
- Čo bolo pre teba najťažšie ?
- Chcel by si niečo podobné robiť aj nabudúce ?

Žiaci hodnotili prácu pozitívne a niektorí boli nadšení, že sa im končene podarilo vytvoriť svoju prvú hru, alebo vyriešiť zaujímavý problém.

ZÁVER

Týmto článkom sme chceli poukázať na pozitíva blokového programovania, ako na jednu z vhodných foriem pre začiatok v programovaní u žiakov 2. stupňa základných škôl. Výhody blokového programovania sa ukazujú aj v prípade rôznych možností softvérového, či hardvérového využitia. Na základe rozhovoru a pozorovania sme dospeli k názoru, že nami vytvorené úlohy sú primerané pre danú vekovú kategóriu žiakov. Vďaka vhodne pripraveným úlohám sa žiaci dokážu namotivovať na zdolávanie problémov, veľa sa naučia a zároveň takúto prácu považujú za zábavnú, zaujímavú a zmysluplnú.

POĎAKOVANIE

Príspevok bol čiastočne finančne podporený projektom KEGA 009KU-4/2017 Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní a vznikol aj za podpory projektu „e-Školy pre budúcnosť 2018/2019“ – Nadácia Orange a v rámci národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje.

BIBLIOGRAFICKÉ ODKAZY

[1.] **BÉLIK, M.** *Malé programovacie jazyky.* [Online] [Dátum: 17. 1. 2018.] https://sopusik.wordpress.com/2011/04/18/19-male-programovacie-jazyky/

[2.] **HORVÁTHOVÁ, D., KLIMOVÁ, N.** *Robot Phiro ožíva v rukách detí.* Liberec : FP TUL (CZ), 2018. DIDINFO 2018, sborník konference. ISBN 978-80-7494-424-6.

[3.] **JACKOVÁ, J., RADOBICKÁ, M., KLIMOVÁ, N., IVANČÁKOVÁ, S., MICHALČÍKOVÁ, K.** *Možnosti zvyšovania pedagogickej praxe u študentov.* Liberec : FP TUL (CZ), 2018. ISBN: 978-80-7494-424-6.

[4.] **VANÍČEK, J**. (2018). Učebnice programování pro 2. stupeň ZŠ ve Scratchi. In: . Liberec: FP TUL (CZ),

Robotika v neformálnom vyučovaní informatiky
Robotics in non-formal CS education

Jana Jacková
Katedra informatiky, Pedagogická fakulta KU v Ružomberku
Hrabovská cesta 1
034 01 Ružomberok
Slovensko
jana.jackova@ku.sk

Janka Majherová
CVTI SR, Školské výpočtové stredisko
Hurbanova 6
031 01 Liptovský Mikuláš
Slovensko
janka.majherova@svslm.sk

Patrik Sitiarik, Roman Stančo, Veronika Motýľová
Katedra informatiky, Pedagogická fakulta KU v Ružomberku
Hrabovská cesta 1
034 01 Ružomberok
Slovensko
sitiaricek@gmail.com, roman.stanco19@gmail.com, v.motylova@gmail.com

ABSTRACT
The paper informs about robotic activities in non-formal education in the subject of informatics. Students of Computer Science at the Catholic University in Ruzomberok are involved in the preparation of various popularization activities for primary and secondary school pupils. Some of them are also engaged in extra-curricular activities at schools at their place of residence. The preparation of methodical materials for these activities contributes to the solution of the KEGA Project Innovative methods in teaching informatics in secondary education. Verification of the proposed materials takes place during extra-curricular activities of students, future teachers of informatics, at workshops within the Children University and within robotic extra-curricular activity at the Grammar school of M.M. Hodza in Liptovsky Mikulas.

Keywords
Robot. LEGO Mindstorms. Algorithmic problem solving. Extra-curricular activities. Future teachers of Informatics. Secondary education.

ABSTRAKT
Príspevok informuje o robotických aktivitách v neformálnom vzdelávaní v predmete informatika. Študenti učiteľstva informatiky na Katolíckej univerzite v Ružomberku sa podieľajú na príprave rôznych popularizačných aktivít pre žiakov základných a stredných škôl a niektorí sa venujú aj krúžkovej činnosti na školách v mieste ich bydliska. Príprava metodických materiálov na tieto aktivity prispieva k riešeniu projektu KEGA Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní. Overovanie navrhnutých materiálov prebieha počas mimoškolskej činnosti študentov učiteľstva informatiky na workshopoch v rámci Detskej univerzity a na krúžku robotiky na Gymnáziu M.M. Hodžu v Liptovskom Mikuláši.

Kľúčové slová
Robot. LEGO Mindstorms. Algoritmické riešenie problémov. Mimoškolská činnosť. Študenti učiteľstva informatiky. Sekundárne vzdelávanie.

1 ÚVOD
V súčasnosti sú už počítače, tablety a smartfóny viac-menej bežnou súčasťou takmer každej rodiny na Slovensku. Preto v rámci mimoškolskej činnosti žiakov základných a stredných škôl často badáme vyšší záujem detí a študentov o informatické krúžky a programovanie najmä v takých prípadoch, keď sa softvérová zložka prepája s nejakým hardvérom. Napr. na informatickom krúžku na Univerzite Mateja Bela v Banskej Bystrici, ktorí viedli študenti učiteľstva informatiky, sa žiakom veľmi páčilo programovanie s robotom Phiro, dronom Airblock alebo s LEGO robotmi [1]. My sa v našom príspevku zameriavame na opis niektorých robotických aktivít so stavebnicou LEGO Mindstorms pre žiakov základných a stredných škôl pod vedením študentov učiteľstva informatiky na Katolíckej univerzite v Ružomberku.

V prílohách metodickej príručky Zavádzanie inovovaných štátnych vzdelávacích programov pre vzdelávaciu oblasť Matematika a práca s informáciami v základnej škole [2] je možné nájsť pre okruh algoritmické riešenie problémov viacero tipov na robotické aktivity. Na predchádzajúcich konferenciách DidInfo odzneli viaceré príspevky k problematike algoritmického riešenia problémov prostredníctvom robotických aktivít so stavebnicou LEGO Mindstorms: (na DidInfe 2014) na 2. stupni ZŠ [3] a v rámci neformálneho vzdelávania [4], (na DidInfe 2016) v príprave budúcich učiteľov informatiky [5], (na DidInfe 2018), tvorba metodických listov [6], krúžková činnosť univerzitných študentov pre žiakov základných škôl [1], [7].

Stavebnica LEGO Mindstorms sa na katedre informatiky na Katolíckej univerzite v Ružomberku používa v príprave budúcich učiteľov informatiky pri výučbe programovania [5], programovania automatov [8], programovania robotov [9] a iných technických zariadení. V rámci mimoškolskej činnosti sa potom niektorí zo študentov podieľajú na príprave rôznych aktivít popularizujúcich informatiku pre žiakov základných a stredných škôl. Pri príprave edukačných aktivít s LEGO robotmi používajú rôzne materiály vydávané výrobcami týchto stavebníc. Zároveň sú usmerňovaní pedagógmi katedry informatiky navrhovať si aj vlastné metodické materiály k tejto problematike, ktoré by boli v súlade s inovovaným štátnym vzdelávacím programom pre predmet informatika [10]. Robotické stavebnice zatiaľ nie sú bežne dostupné na cvičných školách, na ktoré chodia študenti praxovať v rámci povinných predmetov pedagogickej praxe. Svoje materiály k edukačnej robotike si majú možnosť overiť pri práci s deťmi v neformálnom vzdelávaní, a to v rámci

informatických workshopov na Detskej univerzite alebo na krúžku robotiky na Gymnáziu M.M. Hodžu v Liptovskom Mikuláši.

2 POPULARIZAČNÉ AKTIVITY ŠTUDENTOV NA KATOLÍCKEJ UNIVERZITE V RUŽOMBERKU

Pedagógovia katedry informatiky Pedagogickej fakulty na Katolíckej univerzite v Ružomberku sa snažia zapájať študentov bakalárskeho i magisterského štúdia do rôznych popularizačných aktivít, ako napr. Detská univerzita, Týždeň vedy a techniky, Deň otvorených dverí. V poslednom období študenti s úspechom prezentujú edukačné aktivity s LEGO robotmi. Zúčastnení žiaci základných a stredných škôl so záujmom sledujú napr. ukážky inteligentného správania robotov, ktoré reagujú na prekážky v priestore alebo dvojkolesového vyvažovacieho robota, ktorý sa automaticky vyvažuje s jeho postupným zaťažovaním.

2.1 Workshopy počas Detskej univerzity

Na Detskej univerzite na Katolíckej univerzite v Ružomberku počas letných prázdnin v roku 2018 [11] realizovali študenti učiteľstva informatiky Bc. Sitiarik a Bc. Stančo 3 informatické workshopy pre 3 rôzne skupiny detí v stredu a vo štvrtok 4. – 5. júla. Každý z týchto workshopov trval približne 3 vyučovacie hodiny. V rámci nich mali deti možnosť pracovať najmä s LEGO robotmi. Okrem toho videli aj ukážku práce 3D tlačiarne. Na záverečnej Detskej konferencii v piatok 6. júla študenti prezentovali video, ktoré približuje a dokumentuje priebeh niektorých aktivít na týchto workshopoch [12].

Užitočným zdrojom námetov na niektoré aktivity, ktoré pripravili naši študenti pre deti, je webová stránka RoboWiki, ktorá „obsahuje návody, texty, príklady, tutoriály, námety, linky a všetko ostatné, čo sa môže hodiť pri využívaní stavebníc LEGO MINDSTORMS NXT/EV3 vo vyučovaní na základných a stredných školách“ [13].

V ďalšom texte uvádzame prehľad aktivít, ktoré si študenti pripravili na workshopy konané v rámci Detskej univerzity [11]. Tieto aktivity pri práci s deťmi aj úspešne overili. Cieľovou skupinou sú deti vo veku 9 až 14 rokov. Predpokladaný čas trvania sú 3 vyučovacie hodiny.

Zameranie: NXT/EV3 robotika
Tematická oblasť podľa iŠVP [10]: algoritmické riešenie problémov
Cieľová skupina: 9 – 14 rokov.
Výchovno – vzdelávacie ciele: Aktivovať tvorivosť pri stavbe lega a následného programovania robota. Zoznámiť dieťa s algoritmickým myslením.
Predpokladaný čas trvania: 3 vyučovacie hodiny.
Príprava, učebné pomôcky: Potrebujeme si pripraviť základné modely robotov na doskladanie, program NXT-G, nabiť kocky, manuály k NXT, EV3 tlačiareň, postaviť spirograf, perá alebo fixky, malé blokové papiere s lepkami k spirografu, dataprojektor, čierne pásy na zem pre svetelné senzory (bludisko), kruh pre súboj robotov (bude stačiť čierna páska).

1. **Úvod** (5 minút)
V úvode začneme motivačným videom a prezentáciou, aby sa žiaci zoznámili s tým, čo budú na workshope vykonávať. Na oživenie hodiny a uvoľnenie atmosféry budú mať lektori na sebe kostýmy z Hviezdnych vojen – dvaja roboti.

2. **Zoznámenie sa s Legom NXT** (15 minút)
Ukážka komunikácie dvoch NXT robotov na základe programových kódov obsiahnutých priamo v kocke mikrokontrolera robota (základné príkazy pomocou kocky, krátka demonštrácia).

3. **Práca s pripravenými modelmi** (30 minút)
Deti dokončia čiastočne pripravené modely robotov pre aplikáciu vytvorených programov. Po ukážke učiteľ skontroluje všetkých žiakov, prípadne im pomôže (základný model podľa príručky a dva modely zamerané na nasledujúce úlohy).

4. **Práca na kocke – základné algoritmy** a algoritmy v NXT-G (30 minút)
Aktivity v tejto časti je možné realizovať skupinovou formou vyučovania. Deti sú rozdelené do skupín začiatočníci (úloha 3.1) a pokročilí (úloha 3.2). Vzhľadom na rôznu obtiažnosť je vhodné realizovať tieto aktivity pri dvoch učiteľoch.

Začiatočníci budú pracovať v programe LEGO NXT priamo na kocke (úloha 3.1). Vytvoria krátke programy, na základe ktorých budú roboty vykonávať určité funkcie.
Úloha 3.1: Práca na kocke podľa manuálu Lego NXT – základné algoritmy. Učiteľ ukazuje na tabuli elementárne algoritmy a deti sa ich snažia naprogramovať. Potom po skupinách chodia prezentovať funkčné/nefunkčné algoritmy, ktoré sú zamerané na všetky senzory.

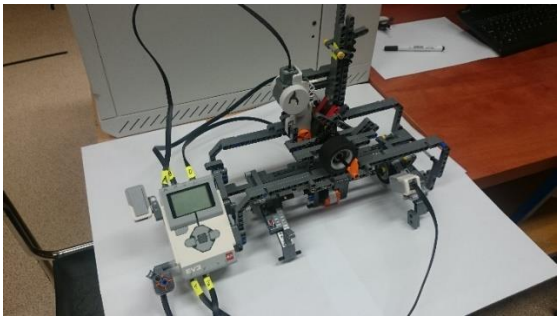
Pokročilejší môžu programovať v NXT-G, ktoré bude realizované na počítači (úloha 3.2; námet na túto úlohu sme našli na stránke RoboWiki [13].
Úloha 3.2 [14]: Poskladajte robot, ktorý sa dokáže otáčať na mieste a má pripojený zvukový senzor. Napíšte program tak, aby:
- sa robot po spustení programu otáčal veľmi pomaly,
- robot začal zrýchľovať ak začneme tlieskať,
- sa robot otáčal tým rýchlejšie, čím rýchlejšie budeme tlieskať,
- robot spomaľoval po spomalení tlieskania (na hlasitosti úderov nezáleží),
- (bonus) robot postupne zastal ak nejaký čas vôbec netlieskame.

Deti následne vyskúšajú program s robotmi a taktiež rôzne kombinácie týchto programov.

5. **Komunikácia robotov** (20 minút)
Deti si vyskúšajú prácu s robotmi na základe komunikácie dvoch kociek prostredníctvom BlueTooth rozhrania alebo wifi rozhrania. Aktivita nie je náročná, ale má veľmi pekný efekt. Námet na nasledujúcu úlohu sme našli na stránke RoboWiki [13].
Úloha [15]: Poskladajte dva podobné roboty, ktoré sa dokážu pohybovať vpred a zatáčať:
- roboty majú namontované ultrazvukové senzory smerom vpred,
- roboty štartujú oproti sebe a začnú sa pohybovať vpred,
- keď sa priblížia na malú vzdialenosť (asi 15-20 cm), zastanú,
- jeden z nich si hodí kockou – vygeneruje náhodné číslo 1 alebo 2 a pošle ho cez BlueTooth druhému robotovi (číslo 1 určuje, že sa roboty navzájom obídu zľava a číslo 2 že sa obídu sprava)
- obidva roboty sa navzájom vyhnú tesnou výhybkou zľava (číslo 1) alebo sprava (číslo 2) a pokračujú po priamke svojho pôvodného pohybu.

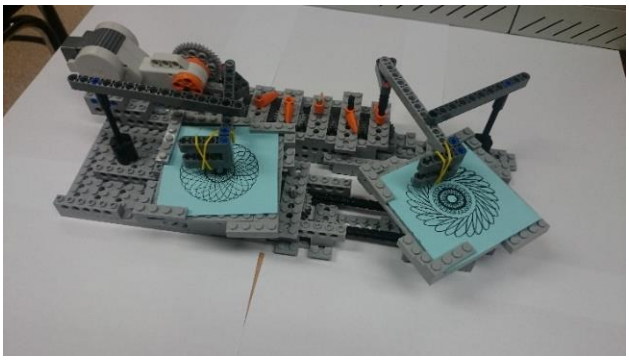
6. **Ukážka práce tlačiarne a spirografu** (20 minút)
Tlačiareň (Obrázok 1) bola zostrojená pomocou novšej kocky Lego EV3 a dokáže vytlačiť Lego nápis, špirálu alebo napr. Hilbertové krivky, čo sú do seba vnárajúce obdĺžniky fungujúce na rekurzívnom princípe, ako je možné vidieť aj vo videu z workshopu [12, začína na 1:36 min.].

Deti si vyskúšajú prácu s novou programovateľnou kockou EV3 a zoznámia sa s menu, ktoré je odlišné od kociek NXT 2.0, na ktorých pracovali doteraz. Za pomoci učiteľa vytlačia niektorý z preddefinovaných vzorov.



Obrázok 1: Tlačiareň s kockou EV3

Po ukážke tlačiarne nasleduje prezentovanie LEGO modelu nazvaného spirograf (Obrázok 2). Model poskladaný z LEGO NXT vykresľuje pomocou kružníc a dvoch pohyblivých osí rôzne obrazce a v závislosti od rozmiestnenia ramena sa obrazce menia. Deti si môžu vyskúšať prácu s týmto Lego modelom a sami môžu nastaviť ramená modelu na kreslenie. Výsledný obrázok si môžu zobrať domov.



Obrázok 2: Spirograf

7. Boj robotov
(15 minút)
Poslednou aktivitou je ukážka boja dvoch robotov. Cieľom tejto aktivity je prostredníctvom hry si zopakovať základné pojmy súvisiace s LEGO robotmi a algoritmickým myslením.

Deti si môžu postaviť v skupinách roboty a naprogramovať ich tak, aby vydržali v súboji robotov. Deti využívajú znalosti algoritmov, ktoré sa doteraz naučili – kombinujú senzory a s nimi súvisiace algoritmy, s cieľom efektívne vyradiť súpera z kruhu. Deti si vymyslia mená robotov a začne sa súboj. Roboty sú pustené oproti sebe v kruhu z čiernej pásky. Kto prvý vypadne z kruhu, prehráva. Ukážka tejto aktivity je vo videu [12, začína na 0:25 min.].

Počas realizácie týchto aktivít by mal učiteľ nepretržite reflektovať ciele, ktoré sa snaží danými aktivitami naplniť: (kognitívny cieľ) dieťa rozumie základným pojmom a je zoznámené s elementárnym algoritmickým myslením, (afektívny cieľ) dieťa má vzťah k algoritmickému a tvorivému mysleniu. Dieťa si hravou formou fixuje poznatky a učí sa samostatne myslieť. Cieľom aktivity je aj psychická regenerácia, ktorá súvisí so spontánnou činnosťou stavby a programovania robota.

3 KRÚŽOK ROBOTIKY NA GYMNÁZIU M.M.HODŽU V LIPTOVSKOM MIKULÁŠI

Na Gymnáziu Michala Miloslava Hodžu v Liptovskom Mikuláši funguje od roku 2010 záujmový krúžok robotiky, na ktorý sa môžu hlásiť žiaci všetkých ročníkov 8-ročného i 4-ročného štúdia. Krúžok začal v roku 2010 pracovať pod vedením vyučujúceho Ing. Václava Králíka, PhD. V roku 2017 prevzala vedenie krúžku bývalá členka tohto krúžku Veronika Motýľová, ktorá v súčasnosti študuje učiteľstvo informatiky na Katolíckej univerzite v Ružomberku, v 1. ročníku magisterského štúdia.

Prvá generácia krúžku bola zmesou mladších žiakov z tried sekundy a kvarty 8-ročného štúdia a študentiek 2. ročníka 4-ročného štúdia. Spolu bolo týchto mladých robotikov približne desať a spoločne sa zúčastnili niekoľkých súťaži na Slovensku aj v zahraničí. Na súťaži FIRST LEGO League, ktorej účastníkmi boli viackrát, riešili vopred zadané tematické úlohy s projektom, ktoré sa týkali napr. ochrany životného prostredia či separácii odpadu. Ďalšia súťaž, na ktorej si zmerali sily so súpermi, sa konala v Bratislave na Slovenskej technickej univerzite pod názvom Istrobot v kategórii Myš v bludisku. Úspechom tejto partie žiakov bola účasť na medzinárodnej súťaži Robot Challenge vo Viedni, keď v rokoch 2011 a 2012 sa im podarilo umiestniť na všetkých troch miestach v kategórii Lego Sumo [16].

V súčasnosti navštevujú záujmový krúžok štyria žiaci vo veku 8 –12 rokov. Tentokrát sa o krúžok zaujímali aj žiaci Základnej školy Márie Rázusovej-Martákovej, čím sa vek robotikov omladil, čo je pozitívne pre rozvoj nielen kognitívnych zručností. Žiaci sa stretávajú jedenkrát do týždňa po vyučovaní v priestoroch informatických učební gymnázia. Počas 1,5-hodinového stretnutia žiaci konštruujú z robotickej stavebnice LEGO Mindstorms NXT a LEGO Mindstorms EV3 rôzne roboty a modely. Využívajú pri tom buď vypracované návody a postupy, ktoré sú súčasťou stavebnicovej sady alebo sú dostupné aj na oficialnej stránke firmy LEGO, alebo zapájajú vlastnú kreativitu a tvorivosť. Výsledky činnosti záujmového krúžku mohli v rokoch 2017 a 2018 vidieť návštevníci Dňa otvorených dverí na Gymnáziu M. M. Hodžu, kde v rámci rôznych oblastí (prirodovednej, technickej, jazykovej, spoločenskej apod.) sa prezentoval aj krúžok robotiky. V júni 2018 sa dvaja žiaci zúčastnili robotickej súťaže Robotický Battle na Alejovej v Košiciach [17] v kategórii Robotická výzva (Obrázok 3). Úloha bola pre všetkých súťažiach rovnaká. Robot mal sledovať čiernu čiaru, na križovatke pokračovať zadaným smerom, prejsť pomedzi steny, presunúť kocky na farebné políčka a na záver vyjsť po schodíkoch, ktorých výška s každým schodom narastala o 1 cm.

Účasť na súťaži pôsobila na žiakov veľmi motivujúco, o čom svedčí ich opätovný záujem o robotiku aj v tomto školskom roku. Cieľom krúžku je rozvíjať u žiakov algoritmické a logické myslenie, pri programovaní využívať matematické modely, podporovať ich tvorivosť, vynaliezavosť, kreativitu atď.

Víziou Gymnázia M. M. Hodžu je nadviazať spoluprácu s organizáciami a školami, ktoré sa tejto oblasti tiež venujú. Najbližším cieľom krúžku je opätovná účasť na súťaži Robotický Battle.

Obrázok 3: Robotická výzva na súťaži RBA

4 PRÍPRAVA METODICKÝCH MATERIÁLOV

Na základe svojich skúsenosti, ktoré získali študenti učiteľstva informatiky Katolíckej univerzity v Ružomberku v rámci ich pomoci pri organizovaní aktivít popularizujúcich informatiku a robotiku na Detskej univerzite, počas Týždňa vedy a techniky alebo počas Dňa otvorených dverí, sú títo študenti zapojení do prípravy metodických materiálov, ktoré budú slúžiť ich nástupcom (ďalším organizátorom z radov študentov) v novom behu týchto podujatí.

V kapitole 2.1 sme uviedli prehľad aktivít (1 – 7), ktoré si študenti pripravili a overili na 3 informatických workshopoch v rámci Detskej univerzity na Katolíckej univerzite v Ružomberku počas letných prázdnin v roku 2018 [11].

Skúsenosti s realizáciou robotického krúžku opísaného v kapitole 3 boli motiváciou pre obsah bakalárskej práce Programátorská príručka prostredia EV3 [16]. Autorka sa zamerala na popis softvéru LEGO Mindstorms EV3 Home Edition. Cieľom bolo vytvorenie didaktickej pomôcky v slovenskej verzii. Táto práca obsahuje tipy a príklady na niekoľko programovateľných robotických modelov, ktoré umožňujú rozvoj algoritmického myslenia zábavnou formou. Ťažiskom je popis programovacieho prostredia a jednotlivých programovacích blokov. V tretej kapitole tejto bakalárskej práce nájdeme námety edukačných aktivít s EV3 vhodných pre vyučovanie, či už v škole alebo na krúžkoch. Ukážky piatich jednoduchých programov, ktorými je možné otestovať funkčnosť robotického modelu sú na s. 45 – 48 [16]. Na s. 50 – 53 sú uvedené bližšie informácie o vybraných robotických súťažiach: Robotický Battle na Alejovej (s.50), IST Robot (s. 52), FIRST LEGO League (s.51) a Robot Challenge (s. 53) [16].

V tomto akademickom roku pripravuje autorka vyššie uvedenej bakalárskej práce ďalšie metodické materiály (v rámci zadaní niektorých predmetov 1.ročníka magisterského štúdia učiteľstva informatiky), ktoré má taktiež možnosť overovať v rámci robotického krúžku, o ktorom sme sa zmienili vyššie. Zistili sme, že keď sa v tomto školskom roku ako vedúca krúžku začala viac venovať podrobnejšej metodickej príprave na jednotlivé vyučovania krúžku, tak (v porovnaní s minulým školským rokom) v súčasnosti viac detí úspešne dokončuje začaté aktivity.

Výsledky vyššie uvedených študentských aktivít v rámci tvorby metodických materiálov môžu poslúžiť záujemcom o robotiku na využitie v neformálnom, ale domnievame sa, že ako inšpirácia aj vo formálnom vzdelávaní informatiky na základných a stredných školách.

Metodike vyučovania robotiky sa venovali v rámci svojich záverečných prác aj niektorí učitelia, ktorí študujú informatiku ako ďalší aprobačný predmet v rozširujúcom štúdiu. Na katedre informatiky Pedagogickej fakulty Katolíckej univerzity v Ružomberku boli vytvorené napr. pracovné listy k programovaniu mikrokontrolerov, pracovné listy k cvičeniam predmetu programovanie - Lego roboty alebo metodika pre predmet robotika na strednej elektrotechnickej škole.

Zaujímavým príspevkom k ilustrácii využitia stavebnice LEGO Mindstorms je bakalárska práca Deterministické automaty [18]. Cieľom tejto práce bolo vytvorenie didaktickej pomôcky deterministického konečného automatu. Autor využíva stavebnicu Lego Mindstorms za účelom vytvorenia funkčného modelu. V druhej kapitole svojej práce je navrhnutý model deterministického konečného automatu pomocou stavebnice Lego Mindstorms a obsahuje aj videonávod na zostrojenie mechanickej časti automatu. Tretia kapitola opisuje činnosť modelu ako aj softvérových prostriedkov. Model tohto automatu zo stavebnice LEGO je možné vidieť aj v [8, s. 94] .

Našim cieľom je všetky overené materiály postupne pridávať na webovú stránku [19]. Viaceré z týchto prác môžu byť podkladom pre ďalší pedagogický výskum spojený s vypracovaním cielených úloh so stanovenými problémami.

5 ZÁVER

Ako sme ukázali v predchádzajúcich kapitolách, k pritiahnutiu žiakov o informatiku prispieva aj možnosť pracovať s LEGO robotmi v neformálnom vzdelávaní, v rámci krúžkov robotiky, robotických súťaží a rôznych podujatí vysokých škôl popularizujúcich informatiku a robotiku.

Pri absolvovaní krúžkov a iných neformálnych aktivít zameraných na robotiku podporíme u žiakov základných a stredných škôl záujem o štúdium v oblasti informačných technológií, k čomu smerujú mnohé súčasné popularizačné podujatia v rôznych projektoch [20, 21].

V súlade s inovovaným štátnym vzdelávacím programom pre predmet informatika [10] a jeho zavádzaním do škôl [2], je možné aktivity opísané vyššie využiť vo formálnom vyučovaní informatiky, najmä pri vyučovaní tematickej oblasti algoritmické riešenie problémov. Napr. v prílohách publikácie [2] s ukážkami vzorových aktivít pre jednotlivé stupne vzdelávania a oblasť algoritmické riešenie problémov sa slovo „robot“ nachádza 16 krát:
Primárne vzdelávanie (Príloha 1, Kubincová) – interaktívne zostavovanie riešenia (3 razy), – pomocou postupnosti príkazov (4 razy), – interpretácia zápisu riešenia (2 razy).
Sekundárne vzdelávanie (Príloha 2, Salanci) – analýza problému (1 raz), – pomocou postupnosti príkazov (1 raz), – pomocou nástrojov na interakciu (1 raz), interpretácia zápisu riešenia (2 razy).
V rámci ukážkového zaradenia témy algoritmické riešenie problémov do ročníkov základnej školy uvádza Kučera pre ročníky 5 – 8 (prílohy 5 – 8) aj Lego robotické stavebnice

(prílohy 5, 6, 8). Časovú dotáciu na tému algoritmické riešenie problémov podľa ročníkov uvádza 6 hodín v 5. ročníku (Príloha 5), 12 hodín v 6. ročníku (Príloha 6) a 16 hodín na 8. ročník.

Naše skúsenosti pri realizácii robotických aktivít v neformálnom vzdelávaní pre žiakov základných a stredných škôl potvrdzujú, že obsah aktivít s danou tematikou by mohol byť po istej úprave zaradený ako učivo aj do formálneho vyučovania predmetu informatika. Nie všetky školy sú ale vybavené robotickými stavebnicami. Preto s algoritmickým riešením problémov formou programovania robotických stavebníc sa žiaci často stretnú až na niektorých stredných odborných školách. Aj táto skutočnosť pridáva na význame neformálneho vzdelávania žiakov základných a stredných škôl v tejto oblasti.

POĎAKOVANIE

Príspevok bol vytvorený v rámci riešenia grantového projektu KEGA 009KU-4/2017 *Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní*.

BIBLIOGRAFICKÉ ODKAZY

[1] KLIMOVÁ, N., JACKOVÁ, J., RADOBICKÁ, M. Čo ak vám katedra informatiky umožní realizovať vaše sny? In *DidInfo 2018* [online]. Liberec : Technická univerzita, 2018, s. 227-236. ISBN 978-80-7494-424-6. [cit. 2019-01-14]. http://www.didinfo.net/images/DidInfo/files/Didinfo_2018.pdf

[2] Štátny pedagogický ústav. *Metodická príručka Zavádzanie inovovaných štátnych vzdelávacích programov pre vzdelávaciu oblasť Matematika a práca s informáciami v základnej škole* [online]. Bratislava : Štátny pedagogický ústav, september 2015. [cit. 2019-01-14]. http://www.statpedu.sk/files/sk/metodicky-portal/metodicke-podnety/matematika_a_praca_s_informaciami.pdf

[3] GUJBEROVÁ, M., MAYEROVÁ, K., VESELOVSKÁ, M. Edukačná robotika na 2. stupni ZŠ a zručnosti pre 21. storočie. In *DidInfo 2014* [online]. Banská Bystrica : Univerzita Mateja Bela, 2014, s. 45-52. ISBN 978-80-557-0698-6. [cit. 2019-01-14]. http://didinfo.net/images/DidInfo/files/didinfo_2014.pdf

[4] CÁPAY, M., LOVÁSZOVÁ, G., MICHALIČKOVÁ, V. Neformálne vzdelávanie detí formou voľnočasových informatických aktivít. In *DidInfo 2014* [online]. Banská Bystrica : Univerzita Mateja Bela, 2014, s. 26-31. ISBN 978-80-557-0698-6. [cit. 2019-01-14]. http://didinfo.net/images/DidInfo/files/didinfo_2014.pdf

[5] KRÁLÍK, V., ZIMENOVÁ, V. Virtuálne robotické laboratórium v príprave budúcich učiteľov informatiky. In *DidInfo 2016* [online]. Banská Bystrica : Univerzita Mateja Bela, 2016, s. 43-45. ISBN 978-80-557-1082-2. [cit. 2019-01-14]. http://didinfo.net/images/DidInfo/files/didinfo_2016.pdf

[6] MICHALÍKOVÁ, A., JOCHMANOVÁ, R. Stavebnica LEGO MINDSTORMS Education EV3 vo vyučovacom procese na základných školách. In *DidInfo 2018* [online]. Liberec : Technická univerzita, 2018, s. 38-46. ISBN 978-80-7494-424-6. [cit. 2019-01-14]. http://www.didinfo.net/images/DidInfo/files/Didinfo_2018.pdf

[7] CSICSOLOVÁ, I., MAYEROVÁ, K., VESELOVSKÁ, M. Ako univerzitní študenti pripravovali obsah robotického krúžku pre žiakov základnej školy. In *DidInfo 2018* [online]. Liberec : Technická univerzita, 2018, s. 47-54. ISBN 978-80-7494-424-6. [cit. 2019-01-14]. http://www.didinfo.net/images/DidInfo/files/Didinfo_2018.pdf

[8] KRÁLÍK, V., SITIARIK, P. Laboratórny model deterministického konečného automatu. In *DidInfo&DidactIG 2017* [online]. Banská Bystrica : Univerzita Mateja Bela, 2017, s. 93-95. ISBN 978-80-557-1216-1. [cit. 2019-01-14]. http://didinfo.net/images/DidInfo/files/didinfo_2017.pdf

[9] MAJHEROVÁ, J., KRÁLÍK, V. Innovative Methods in Teaching Programming for Future Informatics Teachers. In *European Journal of Contemporary Education.* ISSN 2304-9650. Vol. 6, issue 3 (2017), p. 390-400.

[10] Štátny pedagogický ústav. *Inovovaný štátny vzdelávací program* [online]. Bratislava : Štátny pedagogický ústav, 2017. [cit. 2019-01-14]. http://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/

[11] Detská univerzita KU 2018. *Program* [online]. http://du.ku.sk/?page_id=14

[12] STANČO, R., SITIARIK, P. *Katedra informatiky – workshop* [online]. 2018. [cit. 2019-02-04]. https://www.youtube.com/watch?v=bRex3OaOVag&t=1s

[13] RoboWiki. *Stavebnice LEGO MINDSTORMS NXT vo vyučovaní* [online]. [cit. 2019-01-14]. https://wiki.robotika.sk/robowiki/index.php?title=Stavebnice_LEGO_MINDSTORMS_NXT_vo_vyu%C4%8Dovan%C3%AD

[14] *NXT Projekt: Toč sa tak rýchlo, ako rýchlo tlieskam* [online]. [cit. 2019-01-14]. https://wiki.robotika.sk/robowiki/index.php?title=Stavebnice_LEGO_MINDSTORMS_NXT_vo_vyu%C4%8Dovan%C3%AD

[15] *NXT Projekt: Vyhýbanie dvoch robotov* [online]. [cit. 2019-01-14]. https://wiki.robotika.sk/robowiki/index.php?title=NXT_Projekt:_Vyh%C3%BDbanie_dvoch_robotov

[16] MOTÝĽOVÁ, V. *Programátorská príručka prostredia EV3* : bakalárska práca. Ružomberok: Pedagogická fakulta KU v Ružomberku, 2018. 58 s. Dostupné na: //opac.crzp.sk/?fn=detailBiblioForm&sid=3A3A256D9744036C72C498881410

[17] RBA. *Robotický Battle na Alejovej* [online]. RBA, 2018. [cit. 2019-01-14]. https://robotickybattle.sk/

[18] SITIARIK, P. *Deterministické automaty* : bakalárska práca. Ružomberok: Pedagogická fakulta KU v Ružomberku, 2017. 52 s. Dostupné na: //opac.crzp.sk/?fn=detailBiblioForm&sid=5DE0BFAB99270C535F12BBD72595

[19] Katedra informatiky PF KU v Ružomberku. *Metodiky a materiály* [online]. [cit. 2019-01-14]. http://evyucba.ku.sk/eduki/projekt-kega/metodiky-a-materialy/

[20] CVTI SR. *IT akadémia* [online]. CVTI SR, 2018. [cit. 2019-01-14]. http://itakademia.sk/pre-ziakov-new/

[21] ROBOTIKA.SK [online]. [cit. 2019-01-14]. http://www.robotika.sk/mains.php

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Dva programovacie jazyky priateľské k nevidiacim žiakom nižšieho sekundárneho vzdelávania

Two programming languages friendly to lower secondary blind students

Natália Kováčová
CZŠ Narnia
Beňadická 38
85106 Bratislava
Slovenská republika
natalia.kovacova@narniaba.sk

Ľudmila Jašková
KDMFI FMFI UK Bratislava
Mlynská dolina
84248 Bratislava
Slovenská republika
jaskova@fmph.uniba.sk

Mária Karasová
SŠI pre žiakov so zrakovým postihnutím
Svrčia 6
842 11 Bratislava
Slovenská republika
maria.karasova@bee.sk

ABSTRACT

In this paper, we describe our research in the field of teaching programming blind lower secondary school students. We were observing misconceptions that these students commit in process of acquiring basic programming concepts, such as commands, sequencing commands, and loops with a fixed number of repetitions. We used two different programming languages enabling to program simple melodies, stories or rhymes. Our intention was therefore to compare the Alan text-based programming language and the Torino physical programming language in terms of their suitability for teaching the blind learners of this age category. We have found that both languages are suitable and extremely engaging for these pupils.

Keywords

Blind, programming, audio programming language, physical programming language.

ABSTRAKT

V tomto článku opíšeme náš výskum v oblasti výučby programovania nevidiacich žiakov nižšieho sekundárneho vzdelávania. Zaujímalo nás, akých miskoncepcií sa títo žiaci dopúšťajú pri osvojovaní si základných programátorských konceptov, akými sú príkaz, postupnosť príkazov a cyklus s pevným počtom opakovaní. Použili sme dva rôzne programovacie jazyky, ktoré umožňovali naprogramovať jednoduché melódie, príbehy alebo riekanky. Našim zámerom bolo teda aj porovnať textový programovací jazyk Alan a fyzický programovací jazyk Torino z hľadiska ich vhodnosti pre výučbu programovania nevidiacich žiakov uvedenej vekovej kategórie. Zistili sme, že obidva jazyky sú pre týchto žiakov vhodné a mimoriadne pútavé.

Kľúčové slová

Nevidiaci, programovanie, zvukový programovací jazyk, fyzický programovací jazyk.

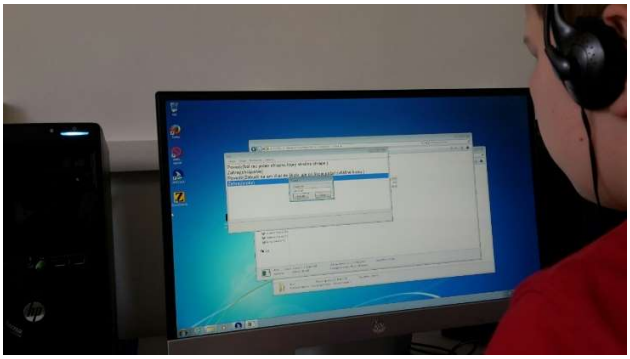
1 ÚVOD

Od roku 2011 prebieha na Katedre základov a vyučovania informatiky FMFI UK v Bratislave výskum zameraný na hľadanie vhodných metód vyučovania informatiky pre nevidiacich žiakov nižšieho sekundárneho vzdelávania [1]. V rámci neho vznikli pilotné verzie programátorských prostredí, ktoré sú určené pre nevidiacich žiakov a edukačné scenáre zamerané na výučbu programovania v týchto prostrediach [2, 3].

Rozvíjanie algoritmického myslenia u nevidiacich žiakov považujeme za veľmi dôležité, nakoľko prax ukazuje, že v softvérových firmách úspešne pracuje nezanedbateľné percento nevidiacich programátorov. Uvedomujeme si, že čím skôr sa vhodne podchytí záujem žiakov o túto profesiu, tým ľahšie sa budú vedieť rozhodnúť pre ďalšie štúdium potrebné pre získanie potrebnej kvalifikácie.

Rozvoj algoritmického myslenia je dôležitý aj pre nevidiacich žiakov, ktorí sa rozhodnú pre kariéru mimo oblasti informačných technológií. Schopnosť vykonávať činnosť podľa návodu, prípadne rozčleniť problém na čiastkové problémy a vedieť popísať postup na ich riešenie, to sú ďalšie dôležité schopnosti potrebné pre štúdium a úspešný život v informačnej spoločnosti.

Kľúčové kompetencie súvisiace s algoritmizáciou a programovaním sú stanovené aj v inovovanom štátnom vzdelávacom programe[1], ktorý je záväzný tiež pre žiakov so zrakovým postihnutím.



Obrázok 1: Nevidiaci žiak programuje v prostredí Alan

Výučba programovania si však vyžaduje vhodné programátorské prostredie. Stručnú analýzu niektorých programátorských prostredí z hľadiska ich vhodnosti pre nevidiacich žiakov uvádzame v nasledujúcej kapitole. Podrobnejšie popíšeme textový zvukový

[1] http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_nsv_2014.pdf

ISBN 978-80-557-1533-9, ISSN 2454-051X.

programovací jazyk Alan a fyzický programovací jazyk Torino, ktoré sme použili počas výskumného overovania s nevidiacimi žiakmi šiesteho a siedmeho ročníka. Bližšie sa o tomto výskumnom overovaní zmienime v kapitole 3. Závery nášho výskumu zhrnieme a prediskutujeme v kapitole 4.

2 PROGRAMOVACIE JAZYKY PRE NEVIDIACICH

Výučba tematickej oblasti **Algoritmické riešenie problémov** v bežnej triede s intaktnými žiakmi (bez akéhokoľvek postihnutia) sa obyčajne realizuje pomocou softvéru bohato využívajúceho vizuálnu reprezentáciu – jazyky typu Logo [4], Scratch [5] a podobe. Tieto prostredia sa však nedajú použiť s nevidiacimi žiakmi, – títo totiž pracujú s počítačom pomocou čítača obrazovky, ktorý im dokáže sprostredkovať iba textové a zvukové informácie.

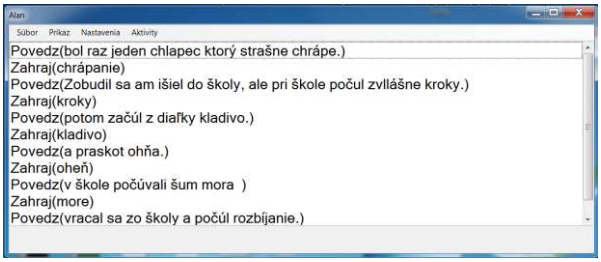
V rámci prieskumu dostupného softvéru sme zvažovali použitie zvukového programovacieho jazyka (audio programming language – APL) pre nevidiacich [6], ktorý vyvinuli v Chille, ale pilotné testy s nevidiacimi programátorskými nováčikmi vo veku 17 až 20 rokov ukázali, že aj pre nich bolo programovanie v tomto jazyku príliš abstraktné. Naši žiaci spadali do nižšej vekovej kategórie 10 až 15 rokov.

Horn [7] popisuje dva hmatateľné programovacie jazyky Tern a Quetzal, ktoré vyvíjal so svojimi kolegami na Tufts Univerzite v Medforde (Oregon, USA). Jazyk Tern umožňuje vytvoriť program pre robota Karla a jazyk Quetzal zase program pre robota skonštruovaného zo stavebnice Lego Mindstorms NXT. V oboch prípadoch program vytvorí dieťa z hmatateľných prvkov mimo počítača. Hotový program sa zosníma kamerou a pretransformuje do ikonického jazyka, ktorý je v počítači. Tento potom vykoná robot, pre ktorého je určený. Autori vytvárali tieto hmatateľné jazyky pre vidiace deti, aby im uľahčili kolaboratívnu tvorbu programov v skupine. Jazyk Quetzal považujeme za vhodný aj pre našich žiakov (nevidiacich aj slabozrakých), nakoľko by mohli hmatom vytvoriť program a aj hmatom kontrolovať vykonávanie programu robotom. Je však otázne nakoľko program spolupracuje s čítačom obrazovky.

Rozhodli sme sa vyvinúť vlastný programovací nástroj. Postupne sme vyvíjali a overovali prostredia **Lienka** [8] a **Svet zvukov** [9]. Tieto prostredia mali, žiaľ, nedostatky, ktoré neumožnili ich použitie vo vzdelávacom procese. Počas ich iteratívneho vývoja sme si však uvedomili, aké vlastnosti má mať programátorské prostredie vhodné pre nevidiacich žiakov ZŠ a sformulovali sme nasledujúce odporúčania [10].

- Prostredie má byť plne ozvučené a plne ovládateľné pomocou klávesnice.
- Prostredie má byť zaujímavé po vizuálnej stránke, má sa dať ovládať aj pomocou myši, aby ho mohli používať aj slabozrakí a intaktní žiaci – spolužiaci, prípadne priatelia nevidiacich.
- Žiak má mať možnosť voliť si príkazy zo zoznamu, aby sa eliminoval výskyt zbytočných syntaktických chýb.
- Základné príkazy programovacieho jazyka majú slúžiť buď na prehrávanie rôznych zvukov, alebo na vyslovovanie textových reťazcov (ľubovoľných alebo zvolených z istej množiny). Jazyk má okrem toho obsahovať aj komplikovanejšie príkazy, ktoré žiakov vedú k pochopeniu pojmov cyklus, podmienený príkaz, premenná a podprogram.
- Výsledkom programu, ktorý žiaci vytvoria, má byť postupnosť zvukov (zvukový súbor alebo vyslovovanie zadaného textu), ktoré sa prehrávajú v požadovanom poradí.

Následne sme navrhli jazyk, ktorý spĺňa tieto kritériá. Podľa tohto návrhu vytvoril študent aplikovanej informatiky v rámci svojej bakalárskej práce programovacie prostredie Alan [11].



Obrázok 2: Program nevidiaceho žiaka vytvorený v prostredí Alan

Prostredie Alan umožňuje programovať zvukové príbehy alebo jednoduché melódie. Prostredie používa jednoduchý programovací jazyk s dvomi základnými príkazmi: ***Zahraj*** a ***Povedz***. Príkaz ***Zahraj*** prehrá zvuk, ktorý si žiak môže zvoliť z viacerých kategórií – Hudba, Noty, Príbeh, Zvieratá. Učiteľ však môže do ponuky pridať ľubovoľné ďalšie zvuky aj kategórie. Príkaz ***Povedz*** zabezpečí, že program prečíta zadaný text. Okrem toho program ponúka možnosť použiť cyklus s pevným počtom opakovaní (príkaz Opakuj). Ďalej môžeme v programe využívať premenné, príkaz vetvenia a vytvárať podprogramy. Prostredie vzniklo primárne pre nevidiacich žiakov, takže sa dá plne ovládať pomocou klávesnice. Z didaktického hľadiska prostredie umožňuje nevidiacim žiakom osvojiť si všetky vedomosti a zručnosti požadované Štátnym vzdelávacím programom, a to zábavnou formou.

Obrázok 3: Fyzický programovací jazyk Torino

Novým prínosom pre programovanie nevidiacich žiakov je fyzický programovací jazyk Torino [12], ktorý vyvinuli v Microsoft Research Lab – Cambridge. Obsahuje hmatovo odlíšiteľné diely predstavujúce jednotlivé príkazy, z ktorých môžeme jednoduchým spájaním vytvárať sekvencie príkazov. Je určený na rozvoj základných programovacích zručností žiakov vo veku 7 až 11 rokov.

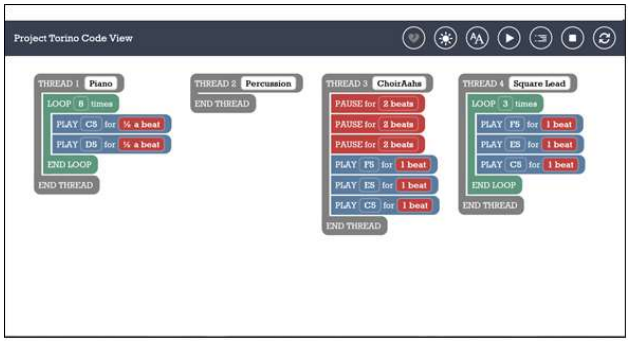
Jednotlivé príkazy sú reprezentované pomocou hmatateľných častí – modulov. Postupnosť týchto modulov vytvára vlákno.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

Tieto vlákna možno pripájať do štyroch vstupov **hlavného modulu** (z anglického *hub*), pričom na každom vstupe si môžeme zvoliť inú množinu zvukov. Torino využíva jazyk Sonic Pi[2]. V súčasnosti Torino obsahuje štyri typy modulov, pomocou ktorých môžu žiaci vytvárať programy:

- Základný **príkazový modul** umožňuje nastaviť zvuk, ktorý sa má prehrávať. Zvuk sa volí výberom z množiny ôsmich zvukov určitej kategórie (napríklad tóny na klavíri), pričom sa dá zvoliť aj dĺžka zvuku.
- **Pauza** – modul, pomocou ktorého žiak nastavuje dĺžku čakania, resp. nečinnosti.
- **Cyklus** – umožňuje opakovanie sekvencie príkazov, ktorá je pripojená k tomuto modulu.
- **Podmienený príkaz** – umožňuje vetvenie programu.

Hlavný modul komunikuje prostredníctvom spojenia bluetooth s obslužným programom v počítači. Tento program umožňuje učiteľovi nastaviť jednotlivé kategórie zvukov, ktoré žiaci využívajú pri tvorbe programov. Obslužný program je určený pre učiteľa, žiaci ho nepoužívajú.



Obrázok 4: Obslužný program jazyka Torino s textovou verziou programu

Okrem príkazových modulov obsahuje sada Torino aj **premenné** a **konštanty**. Napríklad konštanta s označením nekonečno môže byť použitá na nastavenie počtu opakovaní pre cyklus. Žiaci teda môžu pracovať aj s nekonečným cyklom. Tiež je možné použiť generátor náhodných čísel pri výbere zvuku, čím sa umožní prehrať náhodný zvuk z množiny ôsmich zvukov.

3 REALIZÁCIA VÝSKUMU

3.1 Ciele a metódy

V rámci nášho výskumu sme sa zamerali na osvojovanie **konceptov príkaz, sekvencia príkazov a cyklus** v rámci výučby programovania nevidiacich žiakov nižšieho sekundárneho vzdelávania, ktorí nemali predchádzajúce skúsenosti s programovaním.

Zamerali sme sa na nasledujúce dva ciele.

- Zistiť, aké miskoncepcie sa u nevidiacich žiakov vyskytujú pri riešení úloh zameraných na použitie sekvencie príkazov a cyklu s pevným počtom opakovaní.
- Preskúmať výhody, respektíve nevýhody programovacích prostredí Alan a Torino pri riešení úloh zameraných na použitie sekvencie príkazov a cyklu s pevným počtom opakovaní.

Vzhľadom na to, že naša výskumná vzorka bola z hľadiska početnosti veľmi malá, použili sme kvalitatívnu výskumnú stratégiu prípadovej štúdie. Na zber a analýzu dát sme zvolili nasledujúce výskumné metódy [13, 14].

- Priame pozorovanie žiakov (na vybraných hodinách informatiky).
- Pološtruktúrované priame interview (so žiakmi, rodičmi a učiteľkou informatiky).
- Metóda terénnych zápiskov (na vybraných hodinách informatiky).
- Analýza výstupov vzdelávacej činnosti žiakov (analýza žiackych prác pri jednotlivých edukačných aktivitách).
- Analýza videozáznamov a audiozáznamov (realizované počas hodín informatiky a počas rozhovorov).
- Dotazník (pre žiakov, pre učiteľov).

3.2 Výskumná vzorka

Náš výskum sme realizovali s piatimi žiakmi druhého stupňa špeciálnej základnej školy, ktorú kvôli zachovaniu anonymity žiakov nebudeme bližšie špecifikovať. Skúmaní žiaci používali pri práci s počítačom čítač obrazovky. Niektorí žiaci so zvyškami zraku používali súčasne aj zväčšovací softvér.

O skúmaných žiakoch sme sa snažili získať čo najviac relevantných informácií. Realizovali sme rozhovory s triednymi učiteľmi, výchovným poradcom, učiteľkou informatiky. Oslovili sme aj rodičov žiakov, ale nepodarilo sa nám získať informácie od všetkých rodičov.

Medzi skúmanými žiakmi boli zástupcovia troch typov nevidiacich žiakov – žiaci so zvyškami zraku, nevidiaci žiaci so schopnosťou vytvárania vizuálnych predstáv a kongenitálne nevidiaci žiaci (nevidiaci od narodenia).

Z hľadiska veku išlo o troch žiakov siedmeho ročníka a dvoch žiakov šiesteho ročníka. V tabuľke 1 uvádzame najpodstatnejšie charakteristiky žiakov, ktorých sme kvôli zachovaniu anonymity označili písmenami A až E.

Tabuľka 1: Charakteristiky skúmaných žiakov

Žiak	Ročník	Pohlavie	Postihnutie	Informatické zručnosti
A	6	Dievča	Praktická nevidomosť	Nadpriemer
B	6	Dievča	Nevidomosť, svalová dystrofia	Mierny podpriemer
C	7	Chlapec	Nevidomosť, nedoslýchavosť	Mierny nadpriemer
D	7	Chlapec	Kongenitálna nevidomosť	Mierny nadpriemer
E	7	Dievča	Praktická nevidomosť	Mierny nadpriemer

3.3 Edukačné aktivity v prostredí Alan

Pre prostredie Alan sme iteratívnym spôsobom vyvinuli metodický materiál na päť vyučovacích hodín [15]. Počas prvej vyučovacej hodiny bolo cieľom oboznámiť žiakov s možnosťami programovacieho prostredia a zoznámiť ich s dvomi **základnými**

[2] http://sonic-pi.net/

ISBN 978-80-557-1533-9, ISSN 2454-051X.

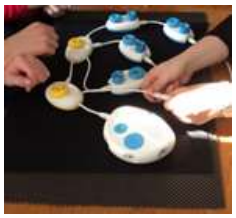
príkazmi jazyka ***Povedz*** a ***Zahraj***. Počas druhej vyučovacej hodiny si žiaci precvičili ich použitie pri vytváraní dlhších **postupností príkazov**. Ich úlohou bolo vytvárať zvukové príbehy kombináciou hovoreného slova a povolenej množiny zvukov. Keď žiaci získali dostatočné zručnosti potrebné na používanie základných príkazov a ovládanie prostredia, zaradili sme počas tretej vyučovacej hodiny úlohy zamerané na **cyklus** (opakovanie sekvencie príkazov). Cieľom štvrtej vyučovacej hodiny bolo naučiť žiakov **hľadať a odstraňovať chyby** v programe (debuggovanie). Piata hodina bola zameraná na opakovanie a **prehĺbenie práce s cyklom**. Žiaci nemali počas tejto hodiny vytvárať svoje programy, ale analyzovať hotové programy. Niektoré úlohy boli zamerané opäť na odhaľovanie a opravovanie chýb, iné si vyžadovali posúdenie efektívnosti z hľadiska dĺžky vytvoreného programu. V rámci poslednej úlohy mali žiaci posúdiť, či je výsledkom dvoch rôznych programov rovnaká sekvencia zvukov.

Popísaných päť vyučovacích hodín sme realizovali s dvomi skupinami žiakov, v jednej skupine boli žiačky šiesteho ročníka (A a B), v druhej skupine žiaci siedmeho ročníka (C, D, E). Každý žiak mal k dispozícii počítač (so slúchadlami, čítačom obrazovky a prostredím Alan) a úlohy riešil samostatne. O niektorých úlohách sme so žiakmi diskutovali buď v skupine alebo individuálne. Cieľom diskusií nebolo prezradiť žiakom riešenie, ale doviesť ich k riešeniu kladením vhodných otázok.

3.4 Edukačné aktivity v prostredí Torino

Pre prostredie Torino sme sa pri tvorbe metodického materiálu inšpirovali internou príručkou pre učiteľov, ktorú vytvorili autori jazyka. Väčšinu úloh sme sa však rozhodli upraviť tak, aby žiaci mohli vytvárať príbehy a piesne v rodnom jazyku a nie v angličtine. Mohli sa tak viac sústrediť na podstatu riešenia problému. Vytvorili sme edukačný scenár pre päť vyučovacích hodín. Počas prvej hodiny sme sa zamerali na získanie **základných zručností pri práci s fyzickým programovacím jazykom** (ďalej ho budeme volať len Torino). Žiaci sa oboznámili s hlavným modulom a procesom jeho spúšťania, ako aj s postupom pri zapájaní **základných príkazových modulov**. Počas druhej hodiny žiaci viac experimentovali s **nastavovaním parametrov zvukov** na príkazových moduloch. V rámci prvých dvoch hodín sme so žiakmi postupne realizovali niekoľko aktivít zameraných na programovanie príbehov, či krátkych úsekov známych melódií a pesničiek. Počas tretej vyučovacej hodiny žiaci **hľadali a opravovali chyby** v nami vytvorených sekvenciách príkazov. Boli to chyby nasledujúcich typov: nesprávny príkaz v sekvencii, nesprávne poradie príkazov, chýbajúci alebo nadbytočný príkaz v sekvencii. Štvrtá vyučovacia hodina bola zameraná na **použitie cyklu**, pričom nebolo potrebné umiestňovať žiadne príkazy pred cyklus alebo po ňom. Úlohy, v ktorých bolo potrebné zapájať **sekvencie príkazov aj mimo cyklu** boli náplňou piatej vyučovacej hodiny. Tiež sme pokusne zaradili úlohy na použitie vnorených cyklov.

Týchto päť hodín sme opäť realizovali s dvomi skupinami žiakov, v jednej boli šiestačky (A, B) a v druhej siedmaci (C, D, E). Žiaci už mali skúsenosti s programovaním v prostredí Alan, ktoré používali niekoľko mesiacov predtým. Pre každú skupinu sme mali k dispozícii dve programovacie sady, takže niektorí žiaci mohli pracovať samostatne a iní pracovali vo dvojici. Podobne, ako pri riešení úloh s prostredím Alan, nechávali sme žiakov riešiť úlohy samostatne a v diskusiách sme sa snažili ich pomocou vhodných otázok navádzať na riešenie.



Obrázok 5: Žiaci vytvárajú program s vnoreným cyklom

Obrázok 6: Textová podoba programu z obrázku 5 (začiatok piesne Jedna druhej riekla)

3.5 Výsledky výskumu

Počas realizácie nášho výskumu sme sa zamerali na miskoncepcie súvisiace s pojmami, príkaz, postupnosť príkazov cyklus s pevným počtom opakovaní a na vhodnosť prostredí Alan a Torino pri zavádzaní týchto pojmov.

Obidve prostredia sa u žiakov stretli s nadšením a prácu v nich zvládli bez väčších problémov. V prostredí Alan žiaci aktívne využívali rôzne klávesové skratky. Tvorba programu a orientácia v ňom im nerobila väčšie problémy.

Všimli sme si, že fyzický programovací jazyk je vhodnejší na prácu vo dvojici ako textový jazyk. Žiaci môžu na tvorbe programu lepšie spolupracovať. Väčšia skupina ako dvojica však nie je vhodná, pretože si každý žiak potrebuje ohmatať zapojenú sekvenciu modulov (príkazov). Ideálne je, ak môže každý žiak sledovať hmatom, ktorý príkaz sa práve vykonáva. Aj Thieme a kolektív autorov [16] overovali jazyk Torino postupne s piatimi pármi žiakov s rôznymi stupňami zrakového postihnutia a tiež zistili, že je vhodným programovacím nástrojom podporujúcim kolaboratívne učenie vo dvojici.

Z hľadiska motivácie pri riešení úloh sme si všimli, že pre žiakov, ktorí neboli hudobne nadaní, bolo programovanie melódií pomerne náročné a menej obľúbené. Títo žiaci uprednostňovali programovanie riekaniek a krátkych príbehov.

Na druhej strane, žiaci s hudobným nadaním si viac obľúbili jazyk Torino, ktorý na rozdiel od jazyka Alan, umožňoval nastaviť presné dĺžky tónov.

Z hľadiska miskoncepcií, nezaznamenali sme problémy pri riešení úloh, v ktorých bolo potrebné vytvárať sekvencie príkazov bez cyklov. Problémy sme pozorovali až pri riešení úloh, ktoré si

ISBN 978-80-557-1533-9, ISSN 2454-051X.

vyžadovali použitie cyklu. O pozorovaných problémoch sa teraz zmienime pre každé programovacie prostredie zvlášť.

V prostredí ALAN sme si všimli nasledujúce miskoncepcie.

- **Doplnenie príkazu, ktorý má nasledovať za cyklom, na koniec tela cyklu.** V tabuľke 2 je porovnanie žiackeho riešenia (vľavo) a správneho riešenia (vpravo). V úlohe mali žiaci opraviť časť programu tak, aby sa zvuk gongu zahral štyrikrát. Doplnený príkaz je znázornený kurzívou.

Tabuľka 2: Doplnenie chýbajúceho príkazu

Žiacke riešenie	Správne riešenie
Opakuj 3 krát Zahraj(Gong) *Zahraj(Gong)* Koniec Opakuj	Opakuj 3 krát Zahraj(Gong) Koniec Opakuj *Zahraj(Gong)*

- **Nesprávne pochopenie opakovaného vykonávania príkazov nachádzajúcich sa v tele cyklu.** Často sme sa stretli s tým, že žiaci chápu opakovanie viacerých príkazov tak, že sa najprv opakuje prvý príkaz v tele cyklu, potom ďalší príkaz, pričom cyklus skončí vtedy, keď sa posledný príkaz tela cyklu vykoná v požadovanom počte. Podobné skutočnosti pozorovali aj Grover a Basu [17], ktorí sa zamerali na miskoncepcie v porozumení cyklov v blokovom programovacom jazyku. Na vzorke 100 študentov strednej školy zistili, že niektorí z nich chápu opakovanie viacerých príkazov v tele cyklu tak, ako sme to spozorovali aj u našich žiakov.
- Podobná miskoncepcia sa vyskytla pri riešení úlohy, v ktorej mali žiaci zistiť, či je výsledkom dvoch rôznych programov rovnaká sekvencia zvukov. Viacerým žiakom sa javili nasledujúce dva programy uvedené v tabuľke 3 ako rovnaké.

Tabuľka 3: Porovnanie dvoch rôznych programov

Program 1	Program 2
Opakuj 5 krát Zahraj(prasa) Koniec Opakuj Opakuj 5 krát Zahraj(sliepka) Koniec Opakuj	Opakuj 5 krát Zahraj(prasa) Zahraj(sliepka) Koniec Opakuj

Tieto miskoncepcie sme už potom počas realizácie vyučovacích hodín s programovacím prostredím Torino nezaznamenali. Bolo to pravdepodobne z toho dôvodu, že žiaci pochopili mechanizmus opakovania príkazov v tele cyklu. Pozorovali sme však nasledujúce skutočnosti.

- Pri používaní cyklov sme si všimli, že niektorým žiakom spočiatku robilo problémy uvedomiť si, kedy zapojiť sekvenciu príkazov pred cyklus.
- Limitovaný počet príkazových modulov v jazyku malo motivujúci vplyv na používanie cyklov. Žiaci zvažovali, kedy je výhodné cyklus použiť a kedy nie. Uvedomovali si napríklad, že ak treba opakovať dva príkazy alebo menej, tak sa neoplatí použiť modul cyklu.
- Používanie vnorených cyklov bolo náročné. Ak to bolo možné, žiaci uprednostňovali riešenia bez nich.

Žiačke A viac vyhovovala práca s fyzickým programovacím jazykom Torino. Nakoľko je hudobne nadaná, programovanie zvukov bolo pre ňu vhodnou motiváciou na rozvoj programátorských zručnosti. Táto žiačka však nemala výraznejšie problémy ani pri práci s prostredím ALAN.

Pre žiačku B považujeme za primerané striedanie viacerých foriem výučby programovania, nakoľko ju nebolo možné zaťažovať monotónnou činnosťou. Práca s prostredím Torino bola pre žiačku pomerne náročná. Kvôli svojmu fyzickému hendikepu mala problémy so zapájaním príkazových modulov a musela jej pomáhať učiteľka.

Žiak C uviedol, že mu viac vyhovovala práca v prostredí ALAN. Zapájanie modulov fyzického programovacieho jazyka považoval za náročné. Tento žiak však rád pracuje v skupine, preto je pre neho vhodnou formou aj využitie fyzického programovacieho jazyka.

Žiak D je všestranne zameraný, rovnako ako žiačka A, preto nemal výrazné problémy so žiadnou z uvedených foriem výučby programovania.

Žiačka E si najviac obľúbila prácu s fyzickým programovacím jazykom. Je zvyknutá pracovať v skupine a je aj veľmi komunikatívna. Nakoľko má záujem o hudbu, programovanie s fyzickým programovacím jazykom jej zaručilo dostatočné možnosti na programovanie hudobných sekvencií.

4 ZÁVER A DISKUSIA

Počas nášho niekoľkoročného výskumu zameraného na rozvoj algoritmického myslenia nevidiacich žiakov nižšieho sekundárneho vzdelávania sme overovali rôzne programovacie prostredia.

Zistili sme, že **úlohy, ktoré boli zamerané na ovládanie robota a využívajú absolútny alebo relatívny pohyb** (napr. v prostredí Lienka, viď kapitola 2)**, sú pre nevidiacich žiakov natoľko abstraktné, že ich nepovažujeme za vhodné** pri zavádzaní základných programátorských konceptov (príkaz, sekvencia príkazov, cyklus). Ako sme zistili, tento typ úloh je náročný aj pre intaktných žiakov. Svedčia o tom aj naše skúsenosti z úvodného kurzu programovania pre intaktných žiakov prvého ročníka gymnázia, ktorí boli považovaní za začiatočníkov v programovaní [18]. Žiaci používali korytnačiu grafiku v jazyku Python. Pozorovali sme, že pre mnohých žiakov bolo náročné predstaviť si polohu a otáčanie korytnačky pri kreslení rôznych geometrických tvarov s využitím cyklu. Za vhodnejšiu alternatívu pre nevidiacich žiakov považujeme programovanie sekvencií zvukov.

Vo všeobecnosti vnímame nami overené programátorské prostredia nasledovne.

- Fyzický programovací jazyk **Torino** sa nám javí ako vhodná forma na výučbu programovania žiakov už v nižšom veku. Ovládanie modulov jazyka je pomerne intuitívne a konštrukcia cyklu je v ňom relatívne jednoduchá na pochopenie. Treba však poznamenať, že limitujúci počet modulov môže, ale aj nemusí motivovať žiakov k aktívnemu využívaniu konštrukcie cyklu v programoch. Clarissa Correa de Oliveira [19] vo svojej práci zmapovala niekoľko rôznych programátorských prostredí, z ktorých fyzický programovací jazyk Torino považuje za zatiaľ najsľubnejší vzdelávací nástroj pre výučbu programovania nevidiacich žiakov.
- Prostredie **Alan** tiež považujeme za vhodné pre žiakov, ktorí sa ešte s programovaním nestretli. Je však potrebné, aby sa vedeli orientovať na klávesnici a aby ovládali základnú prácu so súbormi a priečinkami.

Programovacie prostredia ALAN a Torino považujeme za mimoriadne vhodné pre nevidiacich žiakov nižšieho sekundárneho vzdelávania aj z toho dôvodu, že sú príbuzné blokovým (kartičkovým) programovacím jazykom. Žiaci sú tak uchránení

ISBN 978-80-557-1533-9, ISSN 2454-051X.

pred zbytočnými syntaktickými chybami a môžu sa sústrediť na algoritmizáciu. Mladenović a kolektív autorov [20] porovnali programovacie jazyky Scratch (blokový jazyk), Logo a Python (textové jazyky) z hľadiska miskoncepcií týkajúcich sa konceptu cyklu. Zistili, že najmenej chýb robia žiaci v blokovom programovacom jazyku (Scratch). V neposlednom rade, programovacie prostredia Alan a Torino umožňujú žiakom, aby sa učili programovať konštrukcionistickým spôsobom v zmysle Paperta [21]. Podľa neho sa "*študenti učia efektívnejšie, ak môžu riešiť vlastné úlohy, čím si budujú svoje znalosti sami a vyjadrujú myšlienky cez médium, ktoré umožňuje získať priame skúsenosti*".

Počas nášho výskumu sa nám vynárali stále nové otázky, na ktoré budeme hľadať odpovede v budúcnosti. Radi by sme sledovali skúmaných žiakov aj vo vyšších ročníkoch, aby sme zistili, ako využijú získané vedomosti na osvojovanie ďalších programátorských konceptov a zručností. V úlohách, ktoré žiaci riešili, nebolo potrebné v tele cyklu narábať so žiadnymi premennými. V minulom akademickom roku vznikla novšia verzia prostredia Alan [22], ktorá umožňuje zaviesť koncept premennej a zmysluplne s ňou v cykle manipulovať. Taktiež fyzický programovací jazyk Torino podporuje tento koncept. Podobne je to aj s konceptom vetvenia programu. Vidíme teda veľký priestor pre ďalšie skúmanie miskoncepcií, ktorých sa nevidiaci žiaci môžu dopúšťať pri kombinovaní cyklu s inými štruktúrami v programovacích prostrediach zameraných na programovanie zvukov.

POĎAKOVANIE

Ďakujeme učiteľom, pedagogickým pracovníkom a žiakom špeciálnej školy pre žiakov s poruchami zraku, na ktorej sme mali možnosť realizovať náš výskum. Ďakujeme pracovníkom spoločnosti Microsoft Research Cambridge za zapožičanie fyzického programovacieho jazyka Torino a za ich odbornú pomoc počas jeho testovania.

BIBLIOGRAFICKÉ ODKAZY

[1] JAŠKOVÁ, Ľ.: Ako začať s informatikou na ZŠ pre nevidiacich žiakov? In: *Didinfo 2013*, apríl 2013, Banská Bystrica: FPV UMB, ISBN 978-80-557-0527-9

[2] JAŠKOVÁ, Ľ., KALIAKOVÁ, M. Programovacie prostredia pre deti s poruchami zraku In *Didinfo 2014*, apríl 2014, Banská Bystrica: FPV UMB, s. 79-88, ISBN 978-80-557-0698-6

[3] KOVÁČOVÁ, N. Koncept opakovania vo vyučovaní programovania nevidiacich žiakov základnej školy. In *DidInfo and DidactIG 2017*. Banská Bystrica : Univerzita Mateja Bela, 2017. - S. 88-92. ISBN 978-80-557-1216-1

[4] BLAHO, A., KALAŠ, I.: *Imagine Logo Primary workbook*. Logotron, Cambridge, 2004

[5] RESNICK, M. et al. (2009). Scratch: Programming for All. *Communications of the ACM*, vol. 52, no. 11, pp. 60-67

[6] SÁNCHEZ, J. et. al.: Blind learners programming through audio. In *CHI EA '05*, extended abstracts on Human factors in computing systems, pp. 1769—1772. ACM, New York

[7] HORN, M. et al.: Designing Tangible Programming Languages for Classroom Use. In: 1st international conference on tangible and embedded interaction, pp. 159—162. ACM, New York (2007), http://www.cs.tufts.edu/~jacob/papers/tei07.horn.pdf

[8] BEČVAROVÁ, Ľ.: *Programovacie prostredie pre deti s poruchami zraku – Programovanie robota*, bakalárska práca, Bratislava: FMFI UK, 2013

[9] SUCHÁNEKOVÁ, J.: Programovacie prostredie pre deti s poruchami zraku – Svet zvukov, bakalárska práca, Bratislava: FMFI UK, 2013

[10] JAŠKOVÁ, Ľ.: Blind Pupils Begin to Solve Algorithmic Problems. In: Diethelm, I., Mittermeir, R.T. (Eds.): *ISSEP 2013*, LNCS 7780, pp. 68—79. Springer, Heidelberg (2013). ISBN 978-3-642-36616-1...

[11] KOVÁČ, M. Programovacie prostredie pre zrakovo postihnutých žiakov ZŠ, Bratislava: FMFI UK, bakalárska práca, 2016

[12] MORRISON, C. et al. Torino: A Tangible Programming Language Inclusive of Children with Visual Disabilities, *Human–Computer Interaction*, 2018, DOI: 10.1080/07370024.2018.1512413

[13] ŠVAŘÍČEK, R. et al. (2007) *Kvalitativní výzkum v pedagogických vědách*. Praha : Portál, 2007. ISBN 978-80-7367-313-0

[14] CRESWELL, J. (2008). *Educational Research: Planning, Conducting, and Evaluating Quantitative and Qualitative Research*. 3 vyd. New Jersey: Pearson Education, 2008.ISBN13:978-0-13-207208-0

[15] KOVÁČOVÁ N. *Koncept cyklu vo vyučovaní programovania nevidiacich žiakov nižšieho sekundárneho vzdelávania*, Bratislava: FMFI UK, dizertačná práca, 2018

[16] THIEME, A. et al. Enabling Collaboration in Learning Computer Programing Inclusive of Children with Vision Impairments. In: *Conference on Designing Interactive Systems 2017*,739–752. https://doi.org/10.1145/3064663.3064689

[17] GROVER, S. & Basu, S. (2017). Measuring student learning in introductory block-based programming: Examining misconceptions of loops, variables, and boolean logic. In: Proceedings of the 48thACM Technical Symposium on Computer Science Education (SIGCSE '17). Seattle, WA. ACM.

[18] JAŠKOVÁ, Ľ., MESZÁROSOVÁ, E., WINCZER, M. (2017) Skúsenosti z úvodného kurzu programovania v Pythone na gymnáziu. In: *DidInfo and DidactIG 2017*. Banská Bystrica : Univerzita Mateja Bela, 2017. - S. 88-92. ISBN 978-80-557-1216-1.

[19] DE OLIVEIRA, C. C. Designing educational programming tools for the blind: mitigating the inequality of coding in schools. 2017.

[20] MLADENOVIĆ, M.; BOLJAT, I.; ŽANKO, Ž. Comparing loops misconceptions in block-based and text-based programming languages at the K-12 level. In *Education and Information Technologies, 2017*, 1-18

[21] PAPERT, S.: What is logo? And who needs it?, *Logo Philosophy and Implementation*. Logo Computer Systems Inc., 1999

[22] KOVÁČ, M. *Programovacie prostredie prístupné pre nevidiacich žiakov nižšieho sekundárneho vzdelávania*, Bratislava: FMFI UK, diplomová práca, 2018

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9 ISSN 2454-051X

Řešení úloh v blokovém programování jako součást projektového dne založeného na deskoherních principech

Solving of Tasks in Block-Based Programming as Part of the Project Day Based on Board Game Principles

Zdeněk Lomička
Katedra výpočetní a didaktické techniky (KVD)
Fakulta pedagogická Západočeské univerzity v Plzni
Klatovská tř. 51
306 14 Plzeň
Česká republika
lomi@kvd.zcu.cz

Zbyněk Filipi
Katedra výpočetní a didaktické techniky (KVD)
Fakulta pedagogická Západočeské univerzity v Plzni
Klatovská tř. 51
306 14 Plzeň
Česká republika
filipiz@kvd.zcu.cz

ABSTRACT
The article introduces the project day, based on the board game. The author participated in development both of them, and intends to use the project day also for his further work and research as part of doctoral study programme at the Faculty of Education, University of West Bohemia in Pilsen. The introductory chapter describes the project day itself. The reader learns the main reasons and milestones of the way from the original concept of the board game to the final form of the project day as an experiential game. The second chapter allows to look closer at the project day from the point of view of programming activities for the role of developer (programmer). Introduces block-based programming and its using to solve tasks and describes the developer's activity, which will be used for the upcoming research. The last chapter describes the planned research focusing on the influence of the project day ambience on solving the programming tasks. Research is being prepared and will take place within the described project day in close connection with the role of the developer.

Keywords
Game based project day, Experiential concept, Our companies, Computational thinking, Block-based programming, MBot programming tasks.

ABSTRAKT
Článek seznamuje čtenáře s projektovým dnem, na jehož přípravě se autor podílel a hodlá jej využít i pro svou další práci v rámci doktorského studia. Úvodní kapitola představuje projektový den, který vznikl na základě autorské deskové hry. Následně jsou popisovány zásadní důvody a milníky vedoucí od původního konceptu deskové hry do finální podoby projektového dne jako hry zážitkové, využívající vybrané principy a komponenty hry deskové. Druhá kapitola nechává nahlédnout na projektový den z pohledu programovacích aktivit role vývojáře (programátora). Blíže seznamuje čtenáře s využitím blokového programování k řešení úloh a popisuje činnost vývojáře, na kterou bude zaměřen připravovaný výzkum. Poslední kapitola popisuje zamýšlený výzkumu zaměřený na vliv prostředí projektového dne na řešení úloh z programování. Výzkum autor připravuje a bude probíhat v rámci projektového dne právě ve spojení s rolí vývojáře.

Klíčová slova
Projektový den s herními principy, Zážitkový koncept, Naše firmy, Informatické myšlení, Blokové programování, Programovací úlohy pro MBoty.

1 PROJEKTOVÝ DEN
V poslední době je často zdůrazňována potřeba vést nastupující generace více k technickým oborům. Zájem o ně podporují samotné firmy, které nemají dostatek kvalitních zaměstnanců, ale i územně-správní celky. Např. Plzeňský kraj vypisuje posledních 5 let dotační programy „Motivace pro technické vzdělávání mládeže Plzeňského kraje“ [1]. Ke zvýšení zájmu by mělo sloužit i zlepšení motivace žáků pro technické zaměření – nejlépe již od začátku studia, tedy od základní školy. Tyto úvahy ve spojení se snahou o posílení celé škály dovedností, včetně sociálních, vedly ke vzniku konceptu projektového dne využitelného pro poslední ročníky základních škol a pro střední školy. U obou stupňů vzdělávání se jedná též o spojení s volbou povolání a následného rozhodnutí o svém dalším směřování v profesním světě.

1.1 Koncept deskové hry
Projektový den vznikl na bázi původní deskové hry od týchž autorů. Motivací pro její vytvoření bylo naplnění uvedených cílů a úvaha o jejím využití pro projektové dny nebo kroužky na školách. Základním tématem hry byl vznik a vývoj firmy, která vstupuje na trh s novým výrobkem a snaží se na něm co nejlépe uplatnit. To bylo dáno aktuálností takového tématu a zároveň předpokladem, že pro žáky by hravá forma mohla být vhodným způsobem předání informací. Firma má pět oblastí, do kterých může investovat: výrobu, obchod, vývoj, marketing a zákaznický servis.

Pokud je uvažován pouze tento základní princip, pak se hráči věnují sledování trhu, investování do určitých oblastí, plnění krátkodobých a dlouhodobých cílů, eliminaci rizik atd. Během hry následně zjistí, že kromě oddělení výroby a obchodu, které jsou samozřejmě nutné k tomu, aby se výrobek vůbec mohl dostat na trh, jsou důležitá i nevýrobní oddělení: vývoj, marketing a zákaznický servis, která přináší produktu výraznou přidanou hodnotu.

Z uvedeného vyplývá, že ve své podstatě se jedná o finanční hru, v níž dochází k nakupování karet (investicím), řešení toku peněz

ISBN 978-80-557-1533-9 ISSN 2454-051X

a sledování, jak na tom jsou spoluhráči. Pokud by se hra obsahující čistě tento princip hrála ve třídě, ať už by byli hráči rozděleni na týmy nebo hráli samostatně, byl by navzdory myšlence o hravosti její průběh pro všechny zúčastněné velice nezáživný. Zahájení hry ve třídě by znamenalo výrazné časové náklady pro učitele a stejně tak příprava každého nového kola (roku „života“ firmy). Zaujmout celou třídu něčím, kde se neustále přepočítávají fiktivní peníze, splněné úkoly, náklady a zisky po každém odehraném kole, a navíc se sledují takové počiny i u ostatních, by bylo téměř nemožné.

1.2 Transformace do zážitkového konceptu

Po zkušenostech z pilotního testování deskové hry se pojetí projektového dne založené čistě na hře deskové postupně přesunulo více do týmově kooperativní a kreativní roviny. Z deskové hry zůstal zachován základní motiv budování firmy, herní principy, vybrané komponenty a některé zajímavé prvky.

Převzato bylo zejména:

- řešení investic, situace na trhu:
  - příležitosti a rizika objevující se před každým kolem a vyhodnocovaná na jeho konci,
  - výzvy motivující hráče během celé hry, u nichž vždy ten jediný nejlepší přebírá finanční ocenění,
  - a hrozby, kde pokaždé, když je někdo v něčem jediný nejhorší, vykazuje každé kolo určitou ztrátu,
- oceňování vstupů rolí (marketing, vývoj),
- finální hodnocení prezentace firmy a designu výrobku.

Podstatné pro fungování projektového dne je i to, že jeden hráč z původní hry byl nahrazen celým týmem. Členové týmu fungují v určité roli: ředitel, výroba, obchod, vývoj a marketing. Každý z nich se pak snaží spolupracovat pro úspěch své firmy. Takovéto rozdělení umožňuje využít projektový den pro celou třídu a vytvořit v ní zároveň i konkurenční prostředí.

Úvahy o uživatelské přívětivosti a jednoduchosti pro žáky i mentora vyústily do vývoje vlastní aplikace, kterou průvodci (mentoři) řídí průběh hry. Ta zajišťuje celý finanční management během hry od počátečního nastavení přes strhávání peněz za investice a provozní náklady až po vyhodnocování odměn či ztrát za výzvy, příležitosti a rizika a hrozby. Také umožňuje případné doplňující vstupy ze strany mentora (speciální odměny či pokuty).

Během hry jsou využity zmíněné koncepty a komponenty z deskových her – zejména v podobě karet investic, výzev a zadání pro oddělení vývoje. Dalšími nosnými prvky ve hře jsou roboti reprezentující výrobek firmy, tablety pro plnění úkolů pro oddělení vývoje a výtvarné potřeby pro řešení designu výrobku. Tyto komponenty naleznou uplatnění již u samotných aktérů – žáků ve spojení s jejich kreativitou. Zejména robot má zásadní roli ve hře, neboť je jakýmsi zástupným prvkem za uvažovaný výrobek. Pro projekt jsou využívány mBoti. Důvodů pro jejich volbu je několik:

- jsou rozebíratelní – je nutné je nejprve sestavit, což představuje nutnost porozumění návodu sestavení a také zručnostní aktivitu, která je při hře žádoucí; zároveň je ale samotné složení poměrně rychlé, v několika krocích,
- programování v jejich nativní aplikaci Makeblock je velice podobné nejrozšířenějším prostředím pro blokové programování, jako je Scratch apod.,
- díky stavebnicové konstrukci je lze rozšiřovat o další komponenty a skládat tak i jiné typy vozidel a strojů,
- jsou k dostání na našem trhu a cenově dostupní.

Řídící aplikace pomohla zautomatizovat nejméně záživné a zároveň pro obsluhu nejpracnější části hry. To umožnilo výrazně zkrátit vyhodnocovací procesy a vedlo k mnohem lepšímu zážitku pro zúčastněné. Volba mBotů jako zástupného produktu pak vnesla do hry zručnostní i technický prvek. Po dalším testování začal být tento projektový den nabízen školám pod názvem Naše firmy [2]. Při pohledu zpět na desítky odehraných dnů je ze zkušenosti i ze zpětné vazby zřejmé, že projektový den v podobě hry zaujme žáky i učitele a vnímají jej jako přínosný.

1.3 Průběh projektového dne

Po úvodní prezentační fázi představující hru a popisující, jak (zejména výrobní) firmy a jejich oddělení fungují, se žáci losem rozdělí do týmů (firem) a rolí, ve kterých budou po dobu hry fungovat. Role mají své specifické úkoly, jejichž plnění přináší týmu úspěch.

Projektový den probíhá v 10 kolech po cca 5–10 minutách. Délku kola určuje mentor na základě průběhu hry a zbývajícího času. Každé kolo symbolizuje 1 rok vývoje firmy.

Týmy, podobně jako běžné firmy, řeší během té doby produkt, se kterým přichází na trh (zpravidla nějaký konkrétní výrobek, ale může jím být i SW řešení), své financování, situaci na trhu a vztahy s konkurencí. Sestavují robota, programují jej, vymýšlejí design v souladu se zamýšleným výrobkem a vytváří svou značku, logo. Na konci každého kola firmy prezentují informace o svém produktu a cílové skupině zákazníků. Zároveň připravují vizuální a mediální prezentaci své firmy na závěrečné hodnocení. Členové týmů spolupracují s cílem být nejlepší firmou na trhu.

Na konci hry shrnou týmy vývoj a postavení své firmy, představí vizuální prezentaci a design robota, který připodobňují k uvažovanému výrobku. Po následném vzájemném ohodnocení, které firmám rovněž přináší nemalý příjem, vítězí ten tým, jehož firma má nejvyšší kumulovaný zisk (hodnotu).

Obrázek 1: Tým při práci; zdroj [2]

Samotný projektový den není ovšem pouhou hrou. Jak již bylo zmíněno, umožňuje žákům zažít pocit sounáležitosti a spolupráce za účelem úspěchu svého týmu, vyzkoušet si fungování ve firmě a často si i uvědomit, co by je bavilo či zajímalo do budoucna. Zároveň umožňuje učitelům, kteří jsou vždy přizváni ke sledování průběhu dne, jiný (zpravidla i zcela nový) pohled na svoji třídu.

ISBN 978-80-557-1533-9 ISSN 2454-051X

Často se žáci i učitelé poprvé setkávají s některými v současnosti hojně využívanými metodami či pojmy z obchodního světa. Příkladem může být „value proposition canvas“ (popis hodnoty výrobku pro zákazníka). Tento nástroj je nesmírně důležitý pro firmy, které hodlají uvést na trh nový výrobek, tj. zejména pro různé start-up projekty, ale v praxi jej využívá velká řada i renomovaných firem [3].

V neposlední řadě poskytuje mentor účastníkům po každém vyhodnocení kola zpětnou vazbu v podobě vysvětlení situace ve hře – na trhu či v některé z firem – a její uvedení do kontextu s reálnými situacemi v běžném životě.

Shrnutí informací o průběhu projektového dne provázené slovem i obrazem může poskytnout reportáž ZAK [4] nebo ČT [5].

1.4 Firmy a role v nich

Role jsou jednou z velmi důležitých součástí projektového dne. V jednotlivých týmech (firmách) je využito těchto pět základních:

- výroba: sestavuje výrobek reprezentovaný mBotem; jeho sestavení je mimo jiné jednou ze tří podmínek (společně s investicí do výroby a obchodu), aby firma vůbec mohla vstoupit na trh a mít stálý příjem; následně výroba pracuje na jeho designu, aby odpovídal zvolenému produktu a přinesl zisk i při závěrečném hodnocení,
- obchod: věnuje se investicím a sledování situace na trhu – aby firma investovala v souladu s okamžitou situací i dlouhodobě zvolenou strategií; počáteční investice do výroby zároveň přinese týmu stavebnici robota, aby výroba mohla vůbec pracovat,
- vývoj: inovuje a zlepšuje produkt; při hře je toto řešeno programováním postaveného robota, přičemž vývojář nemůže otestovat předem svůj vytvořený program; to lze až při předložení výsledku při tzv. certifikaci, kdy úspěšné splnění úkolu přináší firmě zisk; každý pokus o certifikaci je ovšem zpoplatněn, proto se vývojáři snaží sestavit program co nejlépe a uspět pokud možno napoprvé,
- marketing: pracuje na prezentaci firmy – logu a reklamním plakátu; na konci každého kola informuje o dění ve firmě, dosažených výsledcích a splněných metách; využívá pracovní sešit obsahující mimo jiné value proposition canvas, pomocí něhož při hře (často ve spolupráci s celým týmem) definuje svoji cílovou zákaznickou skupinu, mapuje její potřeby a bolesti a následně pracuje na definici konkrétních vlastností svého produktu, které zajistí právě onu významnou hodnotu pro zákazníka,
- ředitel: zastřešuje a koordinuje chod firmy a poskytuje určitou formu servisu; jeho zodpovědností je kontrola investic, vybízení ke komunikaci i spolupráci v rámci týmu a sledování konkurenčního prostředí – úzce tedy spolupracuje s obchodem a marketingem; v případě potřeby může pomoci ostatním.

Proto, aby tým fungoval a firma prosperovala, je samozřejmě důležitá spolupráce jednotlivých oddělení. Během hry se pak v rámci týmů projevují zpravidla pozitivní jevy kooperace, jako je pozitivní vzájemná závislost, osobní odpovědnost a skládání účtů, rozvoj interpersonálních a skupinových dovedností, snaha uspět přinášející vyšší výkon a větší produktivitu u žáků, příznivý vliv na chování a sebedůvěru [6].

V některých případech se mohou objevit i některé negativní jevy, např. když motivace nepomáhá a složení týmu neumožní fungovat všem stejně. Může se stát, že se společné práce neúčastní všichni rovnoměrně (někdo se může až stranit), někdo se může cítit odstrčen, jiný může dominovat a výrazně prosazovat své názory, nebo se členové skupiny mohou hádat. Na tyto jevy upozorňuje Kasíková při popisu kooperativní výuky [7]. Pokud se objeví, je dobré se je naučit rozeznat, vysvětlit si příčiny a důsledky a zkusit je v rámci týmu vyřešit.

Tematicky lze Naše firmy přirovnat k projektu autentického učení „Moja firma“, který byl uskutečněn v letech 2013, 2014 a 2016 na Slovensku [8]. V obou případech se jedná o motivaci žáků k založení fiktivní firmy, navržení fiktivního produktu a následná prezentace výsledků. Motivujícím faktorem pro zúčastněné je vytváření něčeho prospěšného pro sebe a své okolí. Rozdíl mezi projekty je pak zejména ve způsobu provedení, délce trvání, cílové skupině, obsahu, využitých technologiích, pomůckách a principech, zaměření, výstupu a potenciálním dalším využití.

Tabulka 1: Srovnání projektů Naše firmy a Moja firma

NAŠE FIRMY	MOJA FIRMA
intenzivní projektový den (dopoledne) odehrávající se ve škole.	několikadenní projekt (8 hodin výuky), který žáci řeší ve škole i doma
zážitek vzniku a vývoje firmy v čase, simulace trhu, cílení na zákazníka a jeho potřeby, specifikace produktu, propojení s reálnými firmami	zážitek založení firmy, specifikace a prezentace nového produktu
pro jakoukoli třídu, rozdělení do 3–5členných skupin (firem) náhodně nebo výběrem	pro omezený počet účastníků rozdělených na týmy o 1–2 žácích, spolupráce vidomých a slabozrakých žáků
žáci pracují ve škole v časovém omezení s poskytnutými prostředky a pomůckami – rychlé kreativní řešení; výtvarná složka je bez technologií	žáci pracují doma i ve škole, mohou využívat volný čas a libovolné aplikace na tvorbu (hyper)médií
zaměření na rozvoj informatického myšlení a sociálních dovedností, simulaci tržního prostředí, týmovou práci, výběr povolání, technické vzdělávání	zaměření na využití informatických dovedností při řešení reálných problémů z praxe
využití technologií, spolupráce, zručnostních aktivit, kreativity, programování, komunikace, financí, plánování, prezentace	využití technologií, kreativity, spolupráce, prezentace
spolupráce v týmu nezbytná; tým funguje jako celek, firma	spolupráce může váznout při omezeném kontaktu týmu
realizace s vizí dalšího rozvoje a šíření – dlouhodobě udržitelný projekt	realizace zejména pro výzkum

2 PROGRAMOVACÍ ÚLOHY V RÁMCI PROJEKTOVÉHO DNE

Programování sestaveného robota zajišťuje žák v roli vývojáře. Náplní jeho role je zejména plnění zadaných úkolů. Realizuje je na tabletu s využitím aplikace, v níž pomocí blokového programování vytváří programy pro robota svého týmu.

ISBN 978-80-557-1533-9 ISSN 2454-051X

2.1 Blokové programování

Blokové programování patří mezi možnosti, jak co nejjednodušším způsobem uvést účastníky vzdělávacího procesu do základů programování i bez předchozích znalostí jakýchkoli programovacích jazyků a programování obecně. Jednotlivé bloky představují příkazy v nesmírně jednoduché, popisné a uživatelsky maximálně srozumitelné formě.

Myšlenka strukturovaného programování v blocích byla poprvé popsána ve zprávě ALGOL 58 z roku 1958. Následující úprava ALGOL 60 z roku 1960 a další modifikace pak již zavádí pojem blok a popisuje jeho syntaxi a sémantiku. Jako zásadní je v definici zmíněna jasná ohraničenost jednotlivých bloků, tj. že proměnné jsou deklarované a využívané vždy v rámci jednoho bloku, stejně jako potřebné výpočty a další instrukce, a nelze je využívat napříč bloky. Takovéto zapouzdření umožňuje skládat bloky libovolně do struktur, bez nutnosti řešit nutnost nějaké návaznosti či vztahu. Tedy samozřejmě kromě návaznosti bloků nutné pro správnou funkci připravovaného programu [9].

Samotný pojem a popis na širší využití nestačí. Nesmírně důležitou roli v masovém rozšíření sehrálo grafické programovací rozhraní umožňující zcela intuitivně skládat jednotlivé bloky (příkazy) do jednotlivých částí a ty pak třeba do širšího programového celku. Zejména rozhraní umožňující princip „drag and drop“ (tj. „přetažení a puštění“) jednotlivého bloku či celé skupiny bloků výrazně usnadňuje jejich sestavování do programu.

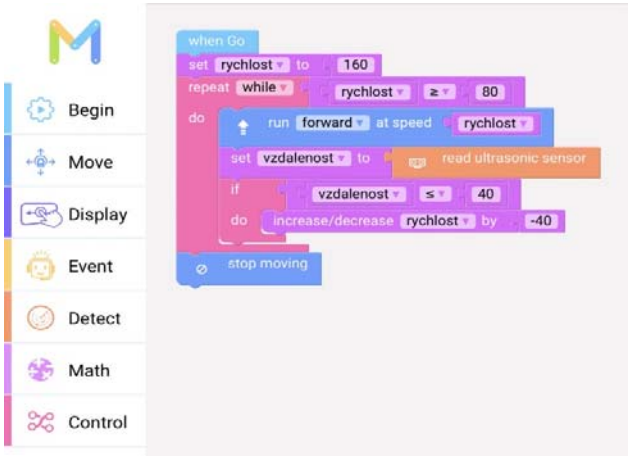
Dvěma nejznámějšími a nejčastěji využívanými prostředími pro blokové programování jsou Hour of Code neziskové organizace Code.org [10] a program Scratch vyvíjený výzkumným oddělením MIT Media Lab při Massachusetts Institut of Technologiy (MIT) [11]. Oba koncepty jsou koncipovány pro využívání dětmi a jednoduché, intuitivní rozhraní na bázi skládání bloků programu. I to je důvodem, proč se úspěchem používají ve výuce. Další výhodou je, že je není nutné instalovat. Stačí je pouze spustit v internetovém prohlížeči.

Díky svým vlastnostem je blokové programování často zařazováno do úvodu výuky programování. V našem školství je kladně přijímáno a doporučováno ze strany MŠMT [12] a nachází uplatnění zpravidla na druhém stupni základních škol. Ve světě se nezřídka objevuje již ve stupni vzdělávání odpovídajícího našemu předškolnímu či prvnímu stupni [13].

Na základě rozšíření obou uvedených prostředí a jejich dalšího využití právě pro programování robotů mBot, bylo takové spojení využito i v rámci uvedeného projektu.

2.2 Řešení úloh v roli vývojáře

Vývojář plní programovací úkoly na přiděleném tabletu. Pro projektový den se z uvedených důvodů využívají roboti mBot, tedy vývojáři pracují v prostředí pro ně určeném – Makeblock (původně mBlockly). Makeblock využívá blokové programování na bázi Scratche a umožňuje rychlé osvojení i naprostým začátečníkům [14].

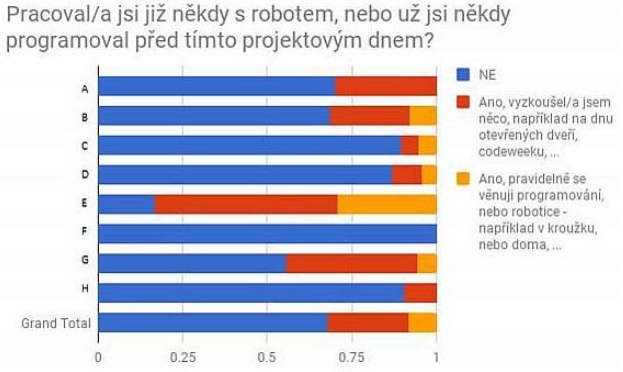


Obrázek 2: Ukázka prostředí Makeblock a programu na postupné zastavení vozítka s využitím cyklu, proměnných a měřením vzdálenosti ultrazvukovým čidlem; zdroj vlastní

V průběhu hry dostane tým tablet a zadání prvního úkolu teprve s první investicí do vývoje. Jednotlivé přidělované úkoly jsou pro všechny týmy stejné a mají vzrůstající obtížnost:

- ujetí určité vzdálenosti za určenou dobu,
- zatáčení a změna rychlosti,
- objetí obrysu obdélníku se zadaným poměrem stran,
- jízda ve tvaru kružnice (mnohoúhelníku) s určitým průměrem,
- jízda ve tvaru spirály,
- pro nejschopnější jsou pak připraveny úlohy s cykly, proměnnými a vyhodnocováním vzdáleností.

Cílem je, aby žáci postupně poznávali programové prostředí a získané zkušeností mohli využít v dalších úkolech a ještě je rozšířit. Důraz na vzrůstající obtížnost od úloh pro úplné začátečníky byl dán úvodními informacemi ze strany škol, kde měl projekt probíhat. Následné zkušenosti ve školách potvrdily (a vyplynulo to i ze zpětné vazby od 156 žáků, kteří vyplnili dotazníky), že v průměru více než 70 % z nich před realizací projektového dne nepracovalo s robotem, ani neprogramovalo.



Obrázek 3: Zkušenosti žáků s roboty a jejich programováním (odpovědi od 156 respondentů z 8 anonymizovaných ZŠ (A–H) a jejich 8. a 9. tříd, v každé cca 20 žáků); zdroj [2]

Zajímavý aspekt při hře je tzv. certifikace. Tým totiž nemá možnost zkoušet připravený program se svým robotem při vývoji. Toho je docíleno jednoduchým způsobem: žádný z týmů nemá

ISBN 978-80-557-1533-9 ISSN 2454-051X

u robota modul s bateriemi. Ty má v držení pouze „certifikační autorita“, tj. mentor. Poté, co vývojář usoudí, že program je hotov, ohlásí se k certifikaci a přinese tablet a robota k otestování. Společně s mentorem ověří funkčnost programu – splnění zadání.

Každá certifikace je zpoplatněna tak, že v zisku je tým při splnění zadání do dvou pokusů. Pokusů může absolvovat, kolik chce. Někdy má smysl absolvovat více pokusů a dostat se i do ztráty, pokud tým chce plnit další úkol v řadě, u kterého se domnívá, že jej zvládne na základě získaných poznatků snáze. Po splnění zadání dostává tým kromě odměny i nový úkol, při jehož plnění může využít poznatky získané při plnění úkolu předchozího.

3 VYUŽITÍ PROJEKTOVÉHO DNE PRO VÝZKUM

Programovací aktivity v rámci projektového dne slouží nejen pro rozšíření povědomí o programování, ale budou využity i pro účely výzkumu v rámci disertační práce autora. Jejím cílem je popsat a prozkoumat vliv zážitkového herního projektu na způsob, jakým žáci základní školy řeší úlohy z programování.

3.1 Popis a záměr chystaného výzkumu

Plánovaný výzkum se odvíjí od problému definovaného na základě průběhu prvních projektových dnů ve škole: Nakolik ovlivňuje kontext hry a zážitkového herního projektu žákovský způsob řešení vybraných úloh z oblasti programování? Jaké aspekty informatického myšlení využívá žák při řešení těchto programovacích úloh?

Ve světě již proběhla řada výzkumů na téma pozitivního vlivu deskových her na rozvoj žáků a studentů. Bylo zkoumáno např. zlepšení koncentrace a matematického myšlení pomocí šachů (Scholz a kolektiv [15]). Následně využívání informatického myšlení a jeho konkrétních aspektů (Berland a Lee [16]) podle Wingové (podmínkové logiky, distribuovaného zpracování, analýzy, simulace a sestavování algoritmů) [17]. Stejně tak byl popsán a publikován příznivý dopad využití deskových her na rozvoj kritického myšlení a komunikačních dovedností (Thatcher [18]).

Lze předpokládat, že i návazné projekty využívající herní principy mohou mít na účastníky obdobně příznivý vliv. Projekt Naše firmy je pak koncipován s vizí, aby jako celek přispíval k lepší připravenosti účastníků pro věk digitálních technologií a jejich využitím v každodenním životě. Nejsou v něm řešeny jednotlivé školní předměty, ale jejich propojení do projektového dne. Je kladen důraz na rozvoj informatického myšlení a jeho propojení s reálným světem kolem, jak jej uvádí Wingová [19]. Zároveň je součástí projektu část spojená s informatikou jako takovou, konkrétně programováním. Plánovaný výzkum bude zaměřen právě na tuto aktivitu v rámci širšího kontextu projektového dne a roli s ní spojenou – vývoj.

Na základě analýzy projektových dnů byla zvolena metoda výzkumu vývojem (design based research), jejíž využití pro pedagogický výzkum popisuje např. Kalaš [20]. Na základě přípravy prostředí pro roli vývojáře a intervencí bude zkoumán způsob, jak žáci v kontextu projektového dne řeší sled programovacích úloh s rostoucí obtížností.

V současné době probíhá pilotáž výzkumu (dalo by se říci iterace 0). Na jejím základě – a dat z ní získaných – budou následně uskutečněny dílčí úpravy herního materiálu a proběhnou iterace návazné. Postup výzkumu v rámci projektového dne je uvažován takto:

- V rámci projektového dne proběhne pozorování 3–5členného týmu tvořícího jednu firmu. Pozorování bude zaměřeno zejména na roli vývoje a řešení přidělených úkolů. Bude-li to možné, bude proveden i audiovizuální záznam.
- Jelikož v rámci týmu během projektového dne dochází ke komunikaci a spolupráci, je práce v týmu považována za platnou a pro roli vývoje bude sledována i jeho aktivita a spolupráce v rámci týmu.
- Po skončení projektového dne proběhne ještě slovní dotazování řešitele programovacích úloh (případně více, pokud spolupracovali) formou polostrukturovaného rozhovoru.
- Následně bude provedeno porovnávání zápisků z pozorování (případně i ze sledování videozáznamu). Pro analýzu bude podstatné identifikovat v řešení využívání aspektů informatického myšlení podle Wingové [17]. Sledovaný bude zejména způsob žákovského řešení problému: jakou strategii řešitel volil, co a jak se mu podařilo, jak vyřešil případné problémy apod. Doplňující informací pak může být subjektivní dojem z prostředí kolem: vliv spolupráce v týmu, ale i případný stres a zodpovědnost vůči týmu
- Po analýze dat z několika projektových dní budou provedeny potřebné změny konceptu a bude připravena další iterace.

Cílem je zajistit pro studenty takové podklady a prostředí, aby si dokázali rychle osvojit svou roli a využít poskytnuté materiály k efektivnímu plnění zadávaných úkolů. Lze předpokládat, že postupným zlepšováním intervencí toho bude možné dosáhnout. Výzkum vychází z předpokladu, že vliv na řešení úloh v rámci projektového dne je významný a řešitel dosahuje řešení v kratším čase než v běžné výuce. Zároveň lze předpokládat, že vlivem práce v týmu bude přikládat úspěšnému dokončení větší důležitost žák na projektovém dnu a bude mít také mnohem lepší pocit z dokončeného úkolu.

BIBLIOGRAFICKÉ ODKAZY

[1] *Motivace pro technické vzdělávání.* Plzeňský kraj. [online]. [cit. 2019-01-20]. Dostupné z: http://plzensky-kraj.cz/cs/kategorie/motivace-pro-technicke-vzdelavani

[2] *NAŠE FIRMY. Program pro základní školy.* [online]. nvias, z. s. a SmartEdu Plzeň. [cit. 2019-01-18]. Dostupné z: https://nasefirmy.eu/

[3] OSTERWALDER, A., PIGNEUR, Y., BERNARDA, G., SMITH, A. *Value Proposition Design: How to Create Products and Services Customers Want*, John Wiley & Sons. 2014. 320 s. ISBN: 978-1-118-96805-5

[4] Naše firmy. YouTube - Naše firmy v Centru Robotiky v Plzni [online]. [cit. 2019-03-08]. Dostupné z: https://www.youtube.com/watch?v=y13UdqHgXeM

[5] Naše firmy na ZŠ Neštěmická v Ústí nad Labem. Události v regionech (Praha) — iVysílání. Česká televize [online]. [cit. 2019-03-08]. Dostupné z: https://www.ceskatelevize.cz/ivysilani/10118379000-udalosti-v-regionech-praha/218411000140424-udalosti-v-

ISBN 978-80-557-1533-9 ISSN 2454-051X

regionech?x=65&y=15&index%5B%5D=615707&index%5B%5D=615708

[6] KASÍKOVÁ, H. *Kooperativní učení ve výuce: teorie – výzkum – realita*. In *Pedagogika*. Roč. 67, čís. 2, s. 106-125. Dostupné z: http://pages.pedf.cuni.cz/pedagogika/?p=11733&lang=cs

[7] KASÍKOVÁ, H. *Kooperativní učení, kooperativní škola.* Praha. Portál, 2010. 151 s. ISBN 978-80-7367-712-1

[8] JAŠKOVÁ, Ľ., KALIAKOVÁ, M. Záverečný projekt z informatiky pre zrakovo postihnutých žiakov ZŠ – skúsenosti s autentickým učením In: HORVÁTHOVÁ, D. et al.(Eds.) Didinfo & DidaktIG 2017, Banská Bystrica : Univerzita Mateja Bela, 2017. ISBN 978-80-557-1216-1

[9] BACKUS, J. W., BAUER, F. L., GREEN, J., KATZ, C., MCCARTHY, J., PERLIS, A. J., RUTISHAUSER, H., SAMELSON, K., VAUQUOIS, B., WEGSTEIN, J. H., WIJNGAARDEN, A. van, WOODGER, M. *Revised Report on the Algorithmic Language Algol 60.* [online]. [cit. 2019-01-18]. Dostupné z: https://www.masswerk.at/algol60/report.htm

[10] CODE.ORG [online]. Code.org, 2019. [cit. 2019-01-21]. Dostupné z: https://code.org/

[11] SCRATCH. *Imagine, Program, Share.* [online]. Dostupné z: https://scratch.mit.edu/

[12] MŠMT ČR. *Stovky českých škol zařadily do své výuky programování.* [online]. MŠMT ČR. 2013. [cit. 2019-01-30]. Dostupné z: http://www.msmt.cz/ministerstvo/novinar/stovky-ceskych-skol-zaradily-do-vyuky-programovani

[13] CODE.ORG. *CS Fundamentals for grades K-5*. [online]. Code.org. 2019. [cit. 2019-01-30]. Dostupné z: https://code.org/educate/curriculum/elementary-school

[14] MAKEBLOCK: Global STEAM Education Solution Provider [online]. 2013 [cit. 2019-01-29]. Dostupné z: https://www.makeblock.com/

[15] SCHOLZ, M., NIESCH, H., STEFFEN, O., ERNST, B., LOEFFLER, M., WITRUK, E., et al. *Impact of chess training on mathematics performance and concentration ability of children with learning disabilities.* 2008. Int. J. Spec. Educ. 23, 138–156

[16] BERLAND, M., LEE, V. R. *Collaborative strategic board games as a site for distributed computational thinking. International Journal of Game-Based Learning.* 2011. 1(2), pp. 65–81. doi: 10.4018/ijgbl.2011040105

[17] WING, J. M. *Computational Thinking.* Communications of the ACM. 2006. Vol. 49, p. 33-35. doi: 10.1145/1118178.1118215

[18] THATCHER, G. *Using Boardgames to Foster Critical Thinking and Communication Skills.* BOOST Best of Out-Of-School Time Conference. [online]. [cit. 2019-01-11]. Dostupné z: http://www.boostconference.org/PDF/2017/Using%20Boardgames%20-%20Greg%20Thatcher.pdf

[19] WING, J. M. Computational Thinking: What and Why? Carnegie Mellon School of Computer Science. [online]. [cit. 2019-01-18]. Dostupné z: https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

[20] KALAŠ, I. *Pedagogický výskum v informatike a informatizácii (2. časť)*. In: DidInfo 2009. 15. ročník medzinárodnej konferencie, B. Rovan (ed.), Univerzita Mateja Bela, Fakulta prírodných vied, Banská Bystrica, 2009. pp. 15–24. ISBN: 978-80-8083-720-4.

Nový informatický predmet Programovanie mobilných zariadení: Tvorba projektov

New Informatics Subject Programming of Mobile Devices: Creation of Projects

Gabriela Lovászová
Fakulta prírodných vied UKF
Tr. A. Hlinku 1
949 74 Nitra
Slovenská republika
glovaszova@ukf.sk

Viera Michaličková
Fakulta prírodných vied UKF
Tr. A. Hlinku 1
949 74 Nitra
Slovenská republika
vmichalickova@ukf.sk

Ján Guniš
Prírodovedecká fakulta UPJŠ
Jesenná 5
041 54 Košicce
Slovenská republika
jan.gunis@upjs.sk

Ľubomír Šnajder
Prírodovedecká fakulta UPJŠ
Jesenná 5
041 54 Košicce
Slovenská republika
lubomir.snajder@upjs.sk

ABSTRACT

Conception of the new informatics subject Programming of mobile devices was developed within the National Project IT Academy – Education for the 21st century. It draws on constructivist principles and agile approaches in education. The curriculum is flexible and project-oriented from the very beginning and guides pupils from simple programming etudes toward more demanding individual and team projects. Pupils' learning is active, experienced-based, motivated by working on meaningful products (mobile applications for the Android platform using MIT App Inventor as a tool) and enriched by various social interactions. At the end of the school year, pupils are challenged to create an original product (games, utilities, learning tools etc.) as a team of developers. In this paper, we give an overview and classification of complex projects included in the curriculum and describe two examples selected from different categories in more detail. The learning materials for pupils (texts, working sheets, sample source codes, media files) are complemented by methodological materials for teachers. After finishing the ongoing pilot verification (during the school year 2018/2019|, these materials will be updated and published for using in secondary schools.

Keywords

Programming. Mobile devices. MIT App Inventor 2. Teaching methodology. Project-based learning.

ABSTRAKT

Koncepciu nového informatického predmetu Programovane mobilných zariadení vyvíjame v rámci Národného projektu IT Akadémia – vzdelávanie pre 21. storočie. Je založená na konštrukcionistických princípoch učenia sa a agilných prístupoch vo vzdelávaní. Plán výučby je flexibilný, už od začiatku orientovaný na projektové vyučovanie, vedie žiakov od jednoduchších programátorských etúd k náročnejším individuálnym a tímovým projektom. Učenie sa žiakov je založené na získavaní skúsenosti, motivované tvorbou zmysluplných produktov (mobilných aplikácií pre platformu Android s využitím nástroja MIT App Inventor 2), obohatené rôznymi sociálnymi interakciami. V závere školského roka žiaci ako tím spolupracujúcich vývojárov vytvára vlastnú, originálnu aplikáciu (napr. hru, užitočný nástroj, aplikáciu na podporu vzdelávania). V príspevku uvádzame prehľad a klasifikáciu projektov zaradených do predmetu a podrobnejšie opisujeme dva projekty vybrané z odlišných kategórií. Učebné materiály pre žiakov (text, pracovné listy, vzorové zdrojové kódy, mediálne súbory a pod.) dopĺňajú metodické materiály pre učiteľa. Po ukončení pilotného overovania prebiehajúceho v školskom roku 2018/2019 budú aktualizované a poskytnuté pre vyučovanie novej témy na strednej škole.

Kľúčové slová

Programovanie. Mobilné zariadenia. MIT App Inventor 2. Metodika výučby. Projektové vyučovanie.

1 ÚVOD

V príspevku [1] o novom informatickom predmete Programovanie mobilných zariadení (PMZ) pre žiakov stredných škôl, ktorý sa koncepčne pripravuje a pilotne overuje v rámci Národného projektu IT Akadémia – vzdelávanie pre 21. storočie [2], sme predstavili základnú koncepciu predmetu: ciele, obsah, rozsah, metodiku. Metodika je založená na agilných prístupoch vo vzdelávaní, ktorých hlavnou myšlienkou je namiesto dôkladného plánovania na začiatku vzdelávacieho procesu inkrementálny a iteratívny prístup otvorený zmenám [3].

Predmet nemá pevné plány, dôraz sa kladie na *flexibilitu,* pri ktorej sa berú do úvahy záujmy žiakov, ich rôzne schopnosti, problémy, skúsenosti. Dôležitejšie ako plán výučby je reagovanie učiteľa na spätnú väzbu od žiakov a vytváranie konštruktívnych vzťahov založených na spolupráci medzi žiakmi a učiteľom.

Výstupy práce žiakov v podobe vytvorených mobilných aplikácií sú najhmatateľnejším meradlom ich progresu. Preto je v predmete od začiatku podporované *projektovo orientované vyučovanie*. Žiaci tvoria zmysluplné aplikácie, pričom sa pozornosť kladie na technickú vyspelosť a dobrý dizajn, ako aj na jednoduchosť a elegantnosť riešení.

Autori predmetu pripravili námety na projekty, podrobné učebné materiály pre žiakov a metodické materiály pre učiteľa, z ktorých si učiteľ môže flexibilne zostaviť vyučovanie podľa záujmov, schopností a potrieb svojich žiakov. Jednoduché programátorské projekty (etudy) sú v predmete koncipované ako série praktických tematických úloh na rôznych úrovniach náročnosti vedúcich ku konvergentnému riešeniu. Ich cieľom je získanie prvých skúseností s programovaním v App Inventore. Komplexnejšie projekty sa vyznačujú väčším rozsahom, vyššou náročnosťou a vyššou mierou samostatnosti a tvorivosti žiakov pri špecifikácii zadaní a postupov riešenia.

V tomto príspevku podrobnejšie predstavíme koncepciu projektového vyučovania v predmete Programovanie mobilných zariadení [4] a prácu na komplexnejších programátorských projektoch.

2 PROJEKTOVÉ VYUČOVANIE V PREDMETE PMZ

Projektové vyučovanie je koncepcia vyučovania založená na zapojení žiakov do riešenia komplexných problémov s požadovaným výstupom. Základným pojmom je projekt, ktorého podstatnými charakteristikami podľa [5] sú:

- vyžaduje sa zmysluplný autentický problém, ktorého riešenie je riadiacim motívom aktivít žiaka,
- výsledkom práce žiaka je konkrétny produkt,
- žiak sa podieľa na špecifikácii zadania, je zodpovedný za postup riešenia a výsledný produkt.

Projektové vyučovanie simuluje prácu na projektoch v praxi. Učiteľ vystupuje v role zadávateľa projektu, ktorý je zvyčajne laik, a žiaci v role profesionálov, ktorí na základe požiadaviek zadávateľa a po dohode s ním bližšie špecifikujú parametre výsledného produktu, zvolia postup riešenia a v dohodnutom termíne odovzdajú výsledok.

V skutočnosti sú však tieto roly len simulované a existujú rozpory medzi simulovanou a reálnou pedagogickou situáciou. Učiteľov záujem sa v skutočnosti neorientuje na výsledný produkt, ale na dosiahnutie vytýčených vzdelávacích cieľov. Žiaci v skutočnosti nie sú experti s profesionálnymi schopnosťami, ale práca na projekte ich má posunúť bližšie k získaniu takýchto schopností. Učiteľ v skutočnosti nie je laik, ktorý prenecháva celú zodpovednosť za prácu na projekte na profesionáloch, ale stále je zodpovedný za priebeh vyučovania.

Úlohou učiteľa v projektovom vyučovaní je vytvárať pre žiakov také podmienky, aby si žiaci tento rozpor uvedomovali čo najmenej a práca na projekte bola pre nich čo najviac autentická. Zároveň si však musí plne uvedomovať svoju zodpovednosť vo vyučovaní, poskytovať pomoc a vedenie žiakom, keď ich pri riešení problémov potrebujú. Úlohou učiteľa je tiež podnietiť v žiakoch reflexiu o prežitej skúsenosti, aby žiak nezotrval vo svojej simulovanej role sústredený len na tvorbu produktu.

2.1 Programátorské projekty

Vo vyučovaní programovania sa prirodzene ponúka využitie projektového prístupu. Programovanie je činnosť, výsledkom ktorej je vždy reálny produkt (program). Jeho vytvorenie si vyžaduje isté profesionálne kompetencie a tvorivý vklad. Prácu na programátorskom projekte môžeme rozdeliť do troch fáz:

Fáza prípravy

V tejto fáze učiteľ pripraví námet na projekt, ktorým sleduje určité vzdelávacie ciele. Dostatok času venuje tomu, aby sa žiaci oboznámili s témou projektu. V spoločnej diskusii sa dohodnú na špecifikácii požiadaviek, ktoré je potrebné do aplikácie implementovať, a na termíne dokončenia. Splnenie požiadaviek a termínu bude základom hodnotenia projektu.

Fáza riešenia

Žiaci pracujú na projekte samostatne, vo dvojiciach alebo v tímoch. Úloha učiteľa je flexibilne reagovať na vzniknuté pedagogické situácie. Môže aj aktívnejšie vstupovať do samostatnej práce žiakov vysvetľovaním nových pojmov a techník. Žiaci okamžite aplikujú nové poznatky do riešenia svojho projektu.

Fáza hodnotenia

Po dokončení práce na programátorskom projekte žiaci prezentujú svoje aplikácie učiteľovi a spolužiakom, testujú si ich navzájom, používajú v praxi. Prezentácia výsledkov práce na projekte je dôležitá pre rozvoj kritického myslenia, prezentačných a argumentačných schopností žiakov. Učiteľ moderuje a aktívne vstupuje do diskusie žiakov k prezentovanému projektu. Podnecuje reflexiu žiakov (spätné skúmanie a hodnotenie prežitej skúsenosti). Reflexia sa má týkať výsledného produktu (čo sme vytvorili), aj cieľov vyučovania (čo sme sa naučili).

Reflexia týkajúca sa výsledného produktu je dôležitá na formovanie kritických postojov založených na argumentácii, ktoré sú potrebné pri objektívnom hodnotení projektov. Pri programátorských projektoch skúmame funkčnosť a využiteľnosť v praxi, efektívnosť v používaní, odolnosť voči chybám používateľa a príjemnosť aplikácie otázkami:

- *Sú služby, ktoré aplikácia poskytuje, užitočné pre účel, na ktorý je určená? Mohla by mať nejaké ďalšie užitočné funkcie?*
- *Vie s aplikáciou jednoducho pracovať človek pri prvom kontakte? Je ovládanie aplikácie efektívne aj pre skúseného používateľa? Mohlo by sa ovládanie aplikácie nejako zjednodušiť, vylepšiť?*
- *Je aplikácia odolná voči chybám používateľa? Rieši problémy s chybnými vstupmi?*
- *Má aplikácia príjemný dizajn (farby, grafiku, zvuky)?*

Reflexiu, ktorá presmeruje pozornosť žiaka z výsledného produktu na proces jeho tvorby a objavenie vzdelávacieho a výchovného zmyslu projektu, podnietime otázkami:

- *Aké problémy ste museli pri práci na projekte vyriešiť?* Otázka podnieti sformulovanie problémov a vzájomnú výmenu zážitkov a skúseností žiakov s ich riešením.
- *Čo ste si pri práci na projekte precvičili a čo nové ste sa naučili?* Otázka podporí kategorizáciu a systemizáciu poznatkov.

2.2 Zoznam a kategórie projektov

Námety na projekty sú v obsahu predmetu rozdelené do 4 tematických celkov podľa prevažujúceho zamerania: *vývoj aplikácií s využitím multimédií* (fotografie, animácie, video, zvuky, analýza a syntéza reči, QR kódy), *vývoj aplikácií s využitím sietí* (webový prehliadač, lokálna a webová databáza, bluetooth), *vývoj aplikácií s využitím geolokácie* (senzor polohy), *vývoj aplikácií s využitím senzorov a aktuátorov* (senzory zrýchlenia, priblíženia, orientácie). Žiaci programujú *hry, aplikácie určené na podporu*

vzdelávania alebo *nástroje užitočné v rôznych oblastiach*. Žiaci pracujú na projekte *individuálne*, *v pároch* alebo *v tímoch*.

Tabuľka 1: Prehľad projektov

Téma	Názov projektu	Typ aplikácie	Typ projektu
Multimédiá	Multimediálny zápisník pre mladého reportéra	užitočný pomocník	individuálny
	Dychový tréner	užitočný pomocník	individuálny
	Asistent prvej pomoci	užitočný pomocník	párový
Siete	Záznamník terénnych dát	užitočný pomocník	individuálny
	Hlasovací systém	podpora vzdelávania	tímový
	Pomocník pri učení sa cudzieho jazyka	podpora vzdelávania	individuálny
	Spoločenská hra pre tablet	hra	tímový
	Kockový poker	hra	párový
Geolokácia	Reverse caching	hra	individuálny
	Geolokačná hra	hra	párový
Senzory a aktuátory	Tréner cvikov pre pacientov a športovcov	užitočný pomocník	individuálny
	Hra ovládaná dotykovými gestami	hra	individuálny

V predmete nepredpisujeme, ktoré projekty by žiaci mali riešiť, učiteľ pri výbere zadaní zohľadňuje záujmy a predchádzajúce vedomosti a zručnosti svojich žiakov. Odporúčame však zaradenie niektorého *z tímových projektov*. Dôvodom na rozdelenie úloh v tíme môže byť komplexnosť a časová náročnosť zadania, programovanie dvoch spolupracujúcich aplikácií alebo možnosť rozložiť implementáciu riešenia na viac nezávislých podproblémov.

Párové programovanie [3] je príkladom agilnej metodiky vývoja softvéru, ktorá podporuje žiakov pri oboznamovaní sa s novými konceptami a urýchľuje a skvalitňuje vývoj aplikácie (žiaci sa radia, hodnotia svoje nápady, kontrolujú sa, sú produktívnejší).

Uvedené námety na projekty sú rôznorodé aj z pohľadu metodiky práce a činností žiakov:

- postupné dokončovanie alebo rozširovanie základného riešenia (remixovanie),
- inkrementálny vývoj usmerňovaný pridávaním požiadaviek na funkcionality,
- štúdium pomocných projektov demonštrujúcich použitie nových komponentov a techník,
- skúmanie vlastností senzorov plánovaním a realizovaním experimentov,
- dôkladné porozumenie riešenému problému pomocou aktívneho experimentovania s pracovným listom,
- tvorba konceptuálneho návrhu aplikácie a jej vizualizácia,
- príprava multimediálnych dát pre aplikáciu,
- testovanie aplikácie.

3 UKÁŽKY PROJEKTOV

Na ukážku uvedieme dva príklady projektov. V prvej ukážke predstavíme projekt z kategórie užitočných pomôcok. Obsahovo patrí medzi projekty, v ktorých sa tvorí aplikácia s využitím multimédií. Pre podporu tvorivého myslenia je vhodná realizácia formou párového programovania – žiaci pracujú vo dvojici pri jednom počítači a všetko, čo navrhnú a naprogramujú, je zároveň okamžite konzultované a revidované v rámci spolupracujúcej dvojice.

Druhý projekt je ukážkou tímového projektu, v ktorom sa uplatní deľba práce a žiaci použijú špeciálny nástroj na zlúčenie častí programu vytvorených členmi tímu. Výsledkom projektu je mobilná aplikácia z kategórie hier a obsahovo je projekt zameraný na využitie sietí.

3.1 Projekt Prvá pomoc

Témou projektu je poskytovanie prvej pomoci pri ohrození života bez špeciálnych pomôcok a zdravotníckej kvalifikácie. Povinnosť poskytnúť prvú pomoc podľa svojich možností a schopností má každý, ak tým neohrozí vlastný život. Naprogramovaním vlastnej mobilnej aplikácie urobia žiaci veľa pre svoju prípravu aj pre reálne poskytnutie pomoci v prípade potreby. Vytvorená mobilná aplikácia poslúži v krízovej situácii ako zdroj okamžitých, rýchlo dostupných informácií.

Opis výsledného produktu

Aplikácia má poskytovať dôležité základné informácie a asistenciu v krízovej situácii pri záchrane života. Aby sme sa k dôležitým informáciám dostali čo najrýchlejšie, aplikácia nesmie byť príliš rozsiahla, má byť zameraná len na život ohrozujúce stavy a postupy, ktoré vedú k záchrane života, a na vysokú efektívnosť podávania informácií a pomoci. Po spoločnej diskusii žiakov a učiteľa sa špecifikujú minimálne požiadavky na funkcionalitu výslednej aplikácie, ktoré budú neskôr slúžiť ako podklad pre hodnotenie projektu.

Výsledkom má byť mobilná aplikácia na platforme Android vytvorená v programovacom prostredí App Inventor, ktorá obsahuje:

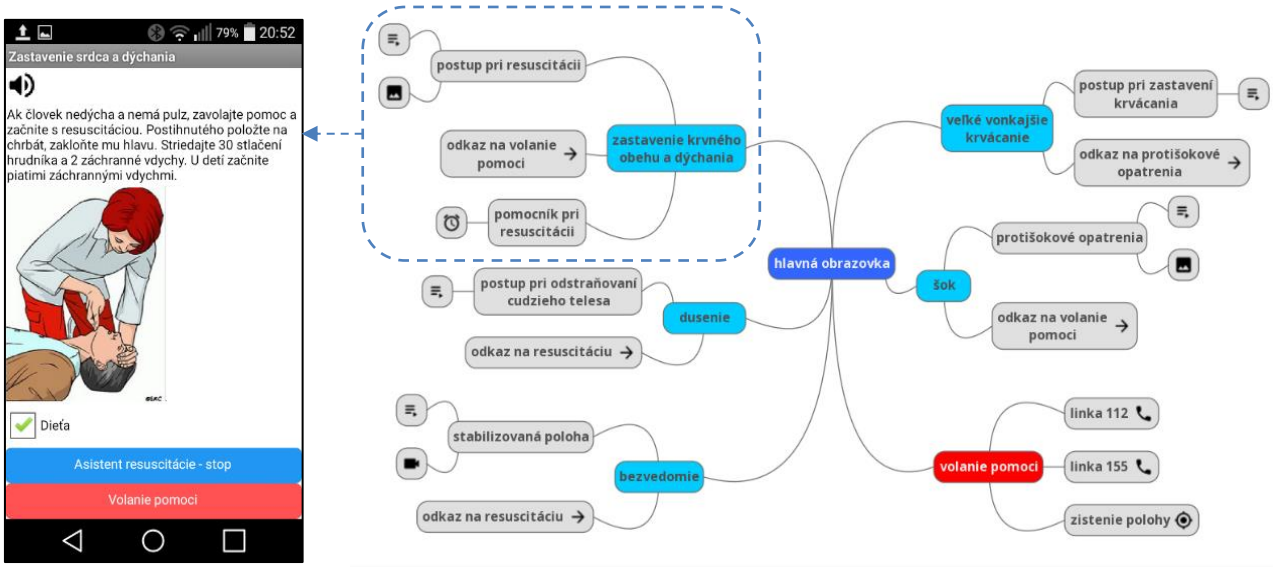
- návody na prvú pomoc pri život ohrozujúcich stavoch (zástava srdca a dýchania, krvácanie, dusenie, bezvedomie a šok) vo forme:
  - krátke texty,
  - obrázky alebo fotografie,
  - krátke videá,
- asistenciu pri čítaní návodov – prevod textu na reč,
- asistenciu pri resuscitácii – časované zvukové signály určujúce frekvenciu masáže srdca a umelého dýchania pre dospelého a dieťa,
- asistenciu pri volaní pomoci – vytočenie tiesňovej telefónnej linky.

Vzdelávacie ciele

Projekt nie je zameraný primárne na osvojenie si principiálne nových programovacích techník, ale skôr na schopnosť naštudovať problematiku témy projektu a vyriešiť zadanie od konceptuálneho

návrhu aplikácie, cez prípravu dát, až po softvérovú realizáciu. Ciele sú:

- Žiak dokáže vytvoriť konceptuálny návrh aplikácie: štruktúru poskytovaných informácií a asistenčných služieb v mobilnej aplikácii a vzájomné prepojenia medzi nimi formou grafickej myšlienkovej mapy,
- vie navrhnúť graficky príťažlivé používateľské rozhranie aplikácie s viacerými obrazovkami zodpovedajúce konceptuálnemu návrhu aplikácie,
- rozumie princípu programovania aplikácie s viacerými obrazovkami, pozná a vie použiť vlastnosti a metódy komponentu `Screen`,
- pozná a vie používať komponenty na realizáciu multimediálnych prvkov v aplikácii: na prehrávanie videa (komponent `VideoPlayer`), na generovanie hlasových a zvukových výstupov (komponenty `TextToSpeech`, `Sound`)
- pozná a vie použiť komponent `PhoneCall` na realizáciu telefónneho hovoru z aplikácie,
- vie vytvoriť užitočnú, funkčnú, efektívnu, používateľsky príjemnú aplikáciu odolnú voči chybám používateľa.



Obrázok 1: Konceptuálny návrh aplikácie Prvá pomoc s ukážkou implementácie jednej časti vo vzorovej aplikácii

Práca na tomto projekte si vyžaduje veľa neprogramátorskej tvorivej práce na konceptuálnom návrhu aplikácie a pri výbere a príprave obsahu. Odporúčame preto pracovať vo dvojiciach alebo v trojiciach. Komunikácia vo dvojici alebo v malej skupine podporuje tvorivosť, generovanie nápadov. Kvalita obsahu vkladaného do aplikácie sa zvýši vnútornou oponentúrou v spolupracujúcej dvojici. Práca vo dvojici alebo v malej skupine umožňuje pripraviť vlastné fotografie a videá. Efektívna deľba práce na príprave materiálov zrýchli prácu na projekte.

Úlohou učiteľa je motivovať žiakov k dôkladnému naštudovaniu problematiky poskytovania prvej pomoci. Poskytne žiakom na preštudovanie kvalitné texty recenzované alebo autorizované lekárom, videá, webové stránky, voľne dostupné mobilné aplikácie a vedie žiakov k analyzovaniu a kritickému hodnoteniu obsahu.

Podstatnou fázou riešenia projektu je vytvorenie konceptuálneho návrhu aplikácie. Je dôležité, aby návrh bol fyzicky zdokumentovaný a existoval nielen ako predstava v mysliach žiakov. Odporúčame na tento účel použiť myšlienkovú mapu (Obr. 1).

Na naplnenie aplikácie obsahom je potrebné pripraviť údaje rôzneho typu (texty, obrázky alebo fotografie, videá, zvuky). V učebných materiáloch pre žiaka pripravených k tomuto projektu poskytujeme pomôcku („lešenie“) vo forme tabuľky na prehľadnú organizáciu pripravených dát. Pomoc poskytuje žiakom učiteľ alebo učebný materiál aj predstavením a vysvetlením funkcií nových komponentov, keď je to potrebné.

Pri záverečnej prezentácii a hodnotení projektov je úlohou učiteľa iniciovať a moderovať diskusiu o výslednej aplikácii, ako aj o procese jej vývoja. Diskusia a reflexívne rozmýšľanie o prežitej skúsenosti má umožniť žiakom objaviť zmysel práce na projekte. Tým je okrem vytvorenia užitočnej aplikácie predovšetkým rozvoj poznania a poznávacích schopností v oblasti programovania v prostredí App Inventor a v oblasti metodológie vývoja softvéru (vyjadrené sú vo vzdelávacích cieľoch), rozvoj poznania v oblasti témy projektu (poskytovanie prvej pomoci v život ohrozujúcich stavoch), ako aj rozvoj osobnosti (žiaci sa učia byť kreatívni a precízni v obsahovom a technickom spracovaní programátorského projektu, zaujať etické postoje k poskytovaniu prvej pomoci a k zneužívaniu tiesňových liniek).

3.2 Projekt Spoločenská hra pre tablet

Spoločenskou hrou rozumieme hru pre dvoch a viac hráčov. Hráči sú pri hre fyzicky prítomní na jednom mieste, súperia proti sebe individuálne alebo v tímoch (hrajú sa spolu, relaxujú, zabávajú sa). Pri spoločenskej hre sa nám môže zísť aj tablet (napr. na cestách). Dokáže ľahko zastúpiť hraciu kocku, kresliace plátno, zápisník, stopky a pod. Niektoré spoločenské hry môžu byť však na použití tabletu a jeho špecifických vlastnostiach aj založené. V takomto prípade hru riadi mobilná aplikácia: automaticky zobrazuje úlohy a vyhodnocuje úspešnosť. Zmysluplná interakcia s tabletom prispieva k dynamike a atraktívnosti hry. Spestrením býva tiež netradičné ovládanie aplikácie s využitím senzorov či multimediálny výstup (zvukové a grafické efekty).

Opis výsledného produktu:

Od žiakov sa požaduje vytvorenie tabletovej verzie slovnej spoločenskej hry *Kartičky,* v ktorej jeden z dvojice hráčov háda slová na základe opisu druhého hráča. Hra má časový limit, cieľom oboch hráčov je, aby hádajúci hráč uhádol čo najviac slov. Hra je zaujímavejšia, keď proti sebe súperia viaceré dvojice hráčov. Pravidlá pre opisovanie slov môžu byť rôzne a ovplyvňujú obťažnosť a zábavnosť hry: hráč nesmie použiť v slovnom opise základ slova, musí hovoriť v cudzom jazyku, musí pri opisovaní spievať, veršovať, imitovať zvuky a pod.

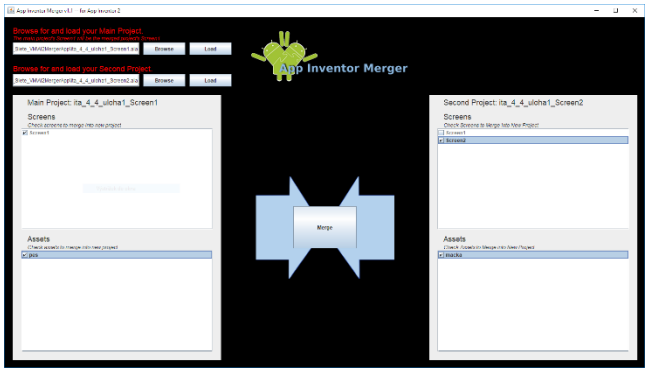
Použitie tabletu *zefektívni organizačnú stránku hry*: nebudeme musieť vopred pripravovať žiadne papierové kartičky ani s nimi pri hre manipulovať, nemusíme merať čas ani počítať skóre. Na výber bude viacero kategórií slov, databázu slov bude možné upravovať a rozširovať. Aplikácia má byť určená pre plaftormu Android a naprogramovaná v prostredí App Inventor.

Vhodnou motiváciou v úvode prvej projektovej hodiny je odohranie jedného kola tradičným spôsobom s použitím papierových kartičiek a tabule na počítanie skóre. Bezprostredný zážitok pripomenie žiakom pravidlá a typický priebeh hry. Žiaci by mali sami navrhnúť, ktoré prvky v hre je možné automatizovať naprogramovaním aplikácie. Zaujímavé nápady môžu vzniknúť aj pri uvažovaní o spôsobe zaznamenávania úspešných a neúspešných pokusov. Bezdotykové ovládanie s využitím senzoru zrýchlenia (náklon tabletu smerom vpred pre „áno“ a vzad pre „nie“) je praktické a atraktívne riešenie, ktoré žiaci pravdepodobne zatiaľ v iných aplikáciách nepoužili.

Diskusia o funkcionalitách aplikácie by mala smerovať k identifikovaniu dvoch podproblémov, ktoré je možné v projekte riešiť nezávisle (komunikácia s databázou, riadenie hry náklonom tabletu) a nápadu realizovať vývoj aplikácie ako tímový projekt.

Prvý člen tímu bude zodpovedný za to, aby si aplikácia po každom spustení z webovej databázy stiahla aktuálne dostupné kategórie slov a z kategórie, ktorú si hráč vyberie, uložila všetky slová na hádanie do lokálnej databázy. Pripravené dáta použije druhý člen tímu vo svojej časti riešenia. Jeho úlohou bude zabezpečiť riadenie priebehu hry, jej riadne ukončenie a vyhodnotenie. Čiastkové projekty členov tímu sa v záverečnej fáze spoja do jedného spoločného projektu pomocou nástroja *App Inventor Merger*.

Učiteľ ako sprievodca a konzultant pri riešení projektu uvedie žiakov do problematiky zlučovania projektov a umožní žiakom vyskúšať si nový nástroj pri tvorbe jednoduchej aplikácie s dvomi obrazovkami.



Obrázok 2: Zlučovanie projektov v okne nástroja App Inventor Merger

Ďalej už žiaci pracujú samostatne na svojich úlohách, bez frontálneho výkladu, s použitím pomocného študijného materiálu a s možnosťou konzultovať s učiteľom. Učiteľ však na začiatku dozrie na to, aby sa žiaci v tíme najprv dohodli na pomenovaní druhej obrazovky a lokálnej databázy, ktorú budú v riešení používať a dôsledne prediskutovali plán práce. Žiak, ktorý implementuje časť riešenia s riadením hry, nemá zatiaľ k dispozícii dáta v lokálnej databáze. Pri testovaní svojej obrazovky tak musí použiť pomocný zoznam naplnený slovami napevno (pripraviť si testovacie dáta).

Záverečnú prezentáciu projektov je možné realizovať ako zábavný turnaj tímov, v ktorom dvojice súťažia proti sebe s využitím vlastných aplikácií. Riešenie možno akceptovať ako úspešné, ak aplikácia v priebehu hry nezlyhá. Aj keď aplikácie žiaci vyvíjajú podľa konkrétnej špecifikácie, v detailoch sa ich produkty budú líšiť (napr. v grafickom dizajne, v spôsobe poskytovania spätnej väzby, v spôsobe vyhodnotenia a pod.).

Pre záverečnou súťažou je potrebné prispôsobiť aplikácie žiakov pre podmienky turnaja. V aplikácii musí byť nastavený rovnaký časový limit a musí byť schopná stiahnuť si kategórie slov na hádanie z webového úložiska. Učiteľ usmerní žiakov, aby v prípade potreby vykonali úpravu v zdrojovom kóde (uložili príslušné číselné a reťazcové konštanty do globálnych premenných). Zorganizuje tiež spoločné naplnenie vzdialenej databázy kategóriami slov na hádanie tak, aby každý z tímov naplnil jednu z určených kategórií.

Vzdelávacie ciele:

- žiak vie v zadaní identifikovať podproblémy, ktoré je možné riešiť nezávisle rôznymi členmi tímu vývojárov,
- získa praktickú skúsenosť s prácou v tíme a s nástrojom *App Inventor Merger* na zlučovanie projektov,
- uvedomuje si význam dohody o konvencii tvorby identifikátorov pri používaní zdieľaného komponentu alebo ukladaní a čítaní dát do/zo zdieľanej databázy,
- pozná a vie používať komponent `TinyWebDB` pre prácu s webovým databázovými úložiskom,
- dokáže realizovať experimenty so senzormi mobilného zariadenia,
- rozumie významu parametrov získavaných v udalostnom bloku komponentu `AccelerometerSensor` a vie ich použiť pri implementovaní riadenia priebehu hry,
- vie dokončiť funkčný prototyp aplikácie, ktorý je možný otestovať v praxi a neskôr rozširovať.

V priebehu práce na projekte žiaci spoznajú výhody a úskalia rozdeľovania úloh medzi viacerých členov tímu. Tímová práca je charakteristická pre vývoj komplexných riešení v praxi. Komunikácia o koncepcii riešenia, spôsobe výmeny údajov či pomenúvaní používaných komponentov a finálna integrácia čiastkových riešení rozvíja programátorské myslenie žiakov a približuje žiakom inžiniersky prístup k riešeniu problémov.

4 METODICKÉ A UČEBNÉ MATERIÁLY K PROJEKTOM

Námety na projekty v predmete PMZ sme spracovali v dvoch verziách ako učebný materiál pre žiaka a metodický materiál pre učiteľa. Učebný text pre žiaka obsahuje:

- krátky úvod k téme projektu,

- zoznam kľúčových slov,
- vzdelávacie ciele (*Čo sa naučíme a čo si precvičíme*),
- motivačný a informačný text o téme projektu (*Čo zaujímavé sa môžeme dozvedieť*), ktorý môže obsahovať aj úlohy a otázky na zamyslenie, ktorými aktivizujeme žiakov a podnecujeme ich záujem o tému projektu,
- požiadavky na výsledný produkt (*Akú zaujímavú aplikáciu môžeme vytvoriť*),
- náčrt riešenia vo forme úloh a textov vysvetľujúcich nové prvky učiva (*Ako budeme postupovať pri tvorbe aplikácie*),
- reflexiu týkajúcu sa výsledného produktu (*Ako vylepšiť či rozšíriť našu aplikáciu*),
- reflexiu týkajúcu sa vzdelávacích cieľov (*Čo sme sa naučili*),
- zoznam literatúry.

Okrem učebného textu sú v závislosti od konkrétneho projektu pripravené pre žiaka ďalšie doplnkové materiály – informačné materiály k téme projektu, pracovné listy, pracovné dátové súbory, vzorové miniaplikácie na preštudovanie, návody, pomocné softvérové nástroje a podobne.

Metodický text pre učiteľa obsahuje celý učebný text pre žiaka doplnený metodickými poznámkami a návodmi na riešenie podstatných častí projektu. V úvodnej časti textu je učiteľ oboznámený s potrebnou prípravou na výučbu, odporučená je forma výučby. Priebežne alebo v závere učebného textu sú zaradené odporúčania k priebehu výučby.

Ďalšie metodické materiály pre učiteľa sú: kompletné vzorové riešenia, ktoré môže použiť na svoje štúdium alebo ako ukážku hotovej aplikácie pre žiakov, vzorovo vyplnené pracovné listy, návrhy na štruktúrované hodnotenie projektov a podobne.

5 ZÁVER

Koncepcia projektového vyučovania v predmete Programovanie mobilných zariadení je založená na konštrukcionistických princípoch učenia sa z vlastnej skúsenosti prostredníctvom tvorby zmysluplných produktov (mobilných aplikácií). Plán výučby je flexibilný, projektovo-orientovaný hneď od začiatku výučby, zostavuje sa výberom projektov – jednoduchších etúd alebo komplexnejších projektov. Učenie sa žiakov je založené na aktívnej činnosti, na získavaní skúseností pri riešení problémov, je motivované prácou na produkte, ktorý má praktické využitie. Takýto spôsob vyučovania je zdrojom bohatých sociálnych interakcií medzi žiakmi navzájom, ako aj medzi žiakmi a učiteľom.

V príspevku sme uviedli prehľad projektov, ktoré sme spracovali vo forme učebných materiálov pre žiaka a metodických materiálov pre učiteľa, a ukážku dvoch projektov z rôznych kategórií.

Materiály sú v súčasnosti v procese overovania vo vyučovaní na stredných školách. Po jeho ukončení a získaní spätnej väzby od učiteľov-overovateľov budú aktualizované a poskytnuté pre vyučovanie novej témy v stredoškolskej informatike v rámci špeciálneho predmetu Programovanie mobilných zariadení alebo ako súčasť vyučovania informatiky.

POĎAKOVANIE

Tento článok bol vytvorený v rámci Národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje a v rámci projektu KEGA 029UKF-4/2018 Inovatívne metódy vo výučbe programovania v príprave učiteľov a IT odborníkov. Ďakujeme učiteľom-overovateľom a aj konzultantovi Ing. Martinovi Fedorovi zo spoločnosti T-systems za ich cenné pripomienky a komentáre k metodickým a učebným materiálom predmetu Programovanie mobilných zariadení.

BIBLIOGRAFICKÉ ODKAZY

[1] ŠNAJDER, Ľ., GUNIŠ, J., LOVÁSZOVÁ, G., MICHALIČKOVÁ, V.: Nový informatický predmet Programovanie mobilných zariadení – koncepcia jeho výučby a ukážky etúd. In *DidInfo 2019.*

[2] Národný projekt IT Akadémia – vzdelávanie pre 21. storočie. http://itakademia.sk/

[3] SALZA, P., MUSMARRA, P., FERRUCCI, F.: Agile Methodologies in Education: A Review. In Parsons, D., McCallum, K. (Eds) *Agile and Lean Concepts for Teaching and Learning*. Singapore : Springer, 2019. ISBN 978-981-13-2750-6

[4] ŠNAJDER, Ľ., GUNIŠ, J., LOVÁSZOVÁ, G., MICHALIČKOVÁ, V.: Programovanie mobilných zariadení – Moodle kurz pre učiteľov – overovateľov a používateľov. https://vzdelavanie.itakademia.sk/course/view.php?id=16

[5] BLUMENFELD, P. C. et al.: Motivating Project-Based Learning: Sustaining the Doing, Supporting the Learning. In *Educational Psychologist*, vol. 26, no. 3-4, 1991, pp. 369-39

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Using a text adventure game approach to train GNU/Linux command line skills

Marek Nagy
Comenius University in Bratislava
Mlynská dolina F1
842 48 Bratislava
Slovakia
mnagy@ii.fmph.uniba.sk

ABSTRACT

In this paper the new method of teaching GNU/Linux command line is presented. It is based on checking every result of commands which users enter into a shell interpreter. The checking is realized automatically as a shell script. Whole scenario of exercises is written in a form of a text adventure game. The scenario can varied according to situations. The suitability of the proposed method is tested on students in a course at the university. The actual results show that the new method overcomes a standard method significantly. Thanks the method students can proceed individually. Because the described process checks students automatically teachers need not intervene so much and they have time to help students with more difficult problems.

Keywords

GNU/Linux. teaching. command line.

1. INTRODUCTION

GNU/Linux is a part of many devices. It can be found, except computers, in mobile phones, smart TVs, routers, digital radios and so on. Thanks to its license it can be freely used anywhere. For example the project Raspberry Pi [1] brings people a cheap possibility to experiment with computer automation. Basis of GNU/Linux is a kernel with many GNU utilities which are used through shell commands. This approach is very old and came from previous types of UNIXes many years before. The simplest method to manage GNU/Linux is a textual terminal where the shell is executed in. Of course, nowaday GNU/Linux distributions have a graphical environment with clickable alternatives to shell commands. But to solve complex jobs the command line approach overcomes the graphical one. It can be considered that it is like a programming. Existence of shell scripts confirms this idea. Also, the textual approach has not high requirements on resources. It is important in a case of embedded applications like Raspberry Pi [1].

This contribution will concentrate on a problem how to teach students to use shell commands to solve typical situations that they can meet with in real life. Several methods with students from the university were tried on a course which is called Linux for users. The course sessions consist of two parts standardly. At the beginning the shell commands are presented with some examples and after it a practical part with computers follows. LMS Moodle [2] environment is being used. There are placed all course materials there. Also texts of exercises are in it. Typical scenario of the exercise is a task which students must solve in a terminal. The command that did the task was copied into the Moodle. After the exercise the task answers were checked carefully and students were given an appropriate score. The problem is how to check the answers. The simplest way is a visual test. But it is too slow. Some automated tests can be found in Moodle [2]. But all tests are based on a pattern matching approach that can't find out if the command is doing what is needed. On the other side, why must students wait until their answers will be corrected? An approach that checks students' answers fully automatically in real time was introduced. GTA application was made [3]. It was developed by students according to requirements which came from real using.

2. PROPOSED METHOD

The new approach was inspired by Bandit game [4] where an user has to read a task from the web page and solve it in a terminal which is connected to Bandit server. If users are successful they can read the next task. The GTA application [3] had to manage the problem to merge these two things (text of tasks and checking of answers) together. To solve it the bash command cycle [5] was explored (see Figure 1). Of course, the state of command processing also includes parsing, expansion and redirection stages.

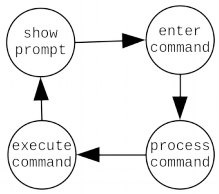


Figure 1. The command line interpreter (shell) is processing commands in a cycle.

Content of a shell environment variable PROMPT_COMMAND is executed right before showing the prompt [5]. It is a function and it is an ideal moment where to insert the checking of students' answers (see Figure 2). The checking can be made by bash commands too. It can be written as a straightforward solution of the task. Of course, in this case, students can't have possibility to see the checking commands. Or it can be written as a test of presence of some system properties.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

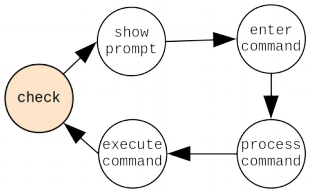


Figure 2. Modified shell command cycle.

Every exercise consists of several tasks which students have to do. The tasks are called **levels** and are labeled by arbitrary names which are mostly constructed as a small letter "L" followed by number. A collection of levels is written into a text file which is called **challenge** and is solved on one exercise. Example of the level is on Figure 3. Levels are separated by lines with minus signs.

```
name: l00
test: test "$(pwd)" = "/tmp"
next: [l01, l03, l06]

Go into the directory /tmp.

----------------------------
```

Figure 3. Example of an easy level.

There are three things in the level description (see Figure 3). The name of level (**name**), the checking command (**test**) and the sequence of levels (**next**) which the next level is randomly selected from. Below it, text of the level follows.

If the exercise is being solved by more students it is good to have more variants of same task. To make it easily the challenge can contain templates (see Figure 4) which are filled from the data file (see Figure 5). The variability is introduced by a random choice of the next level (see Figure 6).

```
{{range $index, $letter := .letters}}
name: l05-{{add $index 1}}
test: test "$(grep -o '{{$letter.name}}'
/etc/passwd | wc -l)" = "$(cat
/tmp/result.txt)"
next: [l13, l14]

Count number of letters {{$letter.name}} in
file /etc/passwd and redirect it into the
file /tmp/result.txt.

--------------------
{{end}}
```

Figure 4. Three levels which will be generated from this template and the data file (from Figure 3).

```
letters:
  - name: "a"
  - name: "b"
  - name: "c"
```

Figure 5. The file with data in the yaml format.

```
name: l04
test:
next: [{{generate_levels "l05" .letters "%s-
%d"}}]

it is same as:

next: [l05-1, l05-2, l05-3]
```

Figure 6. Generating of levels sequence.

Special level is the last level. There is not sequence of next levels and a test command is always false. See Figure 7.

```
name: l09
test: false

Type command exit to finish exercise.
```

Figure 7. The last level.

While using the application and preparing exercises three types of levels has been identified. The **first type** is based on that the answer command must produce a suitable output that can be checked easily. The output can be checked through a control sum (see Figure 8). In many cases it is better than using direct commands. We must realize that a sequence of commands spends more time than one equation with the control sum. Another fact is that the checking commands are executed every time when students enter a command on the prompt.

```
test: |
 test -f /tmp/sort.txt \
 -a "$(cat /tmp/sort.txt | md5sum)" = \
    "cf8148dab9afd0086935e17a37253f2c  -"
```

Figure 8. Checking an output of student solutions.

The **second type** of levels is based on that after the entered command is completed a system must be in a suitable state. Is is more difficult to write such test commands and it requires more thinking about many situations which can appear. A simple example on Figure 9 only tests a presence of an installed package. Every output of test commands is fully ignored.

```
test: dnf list installed | grep "httpd"
```

Figure 9. Testing a presence of an installed package.

The **third type** of levels is based on the pattern matching of an entered command which is stored in LAST_COMMAND variable. This variant is suitable to bring students to write commands in an appropriate fashion (see Figure 10). The regular expressions can be used quite successfully.

```
test: |
 echo "$LAST_COMMAND" | grep "^ *df \+-h *$"
```

Figure 10. Matching of the correct command.

Of course, these three types of tests can be used in combination manner (see Figure 11). Test commands can be joined by && and

ISBN 978-80-557-1533-9, ISSN 2454-051X.

|| symbols that the bash uses to compute the final exit code. It is used instead of AND and OR logical operators [5].

```
test: |
 test "$(pwd)" = "/home/john" \
 -a -f r.txt \
 -a "$(cat r.txt) | md5sum" = \
    "cf8148dab9afd0086935e17a37253f2c  -"
 && echo "$LAST_COMMAND" | grep "tail"
```

Figure 11. Combination of the testing types

In generally, the test section is executed as a script. The variables and other features can be used in it. According to the final exit code it is decided if student has to stay in the level or proceed to next one.

As an improvement, the content of a variable LAST_COMMAND is being sent on a server every time when students enter commands (see Figure 12). The command wget is used for this purpose straightly. At the server a simple web application is running and presents actual work progress of students in real time. Also the server is saving histories for every student.

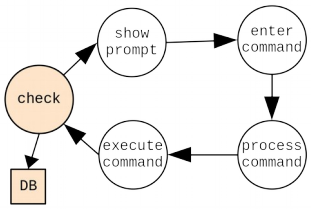


Figure 12. The saving of commands on a server is added into bash command cycle.

3. COURSE CURRICULUM

The proposed method was used in course which is named Linux for users. The whole course takes 13 weeks in summer term. But only 12 weeks are relevant. Every week includes a presentation, an exercise and homework. Following thematic sections are assigned to weeks straightforwardly:

1. GNU/Linux and text console

Introduction and basic commands to navigate in file system. Commands: pwd, ls, cd, less, cat, man, w, who, exit, date, ssh, uname.

2. Directories and files

Closer description of file system and manipulating with files and directories. Commands: mount, umount, df, mkdir, rmdir, rm, cp, mv.

3. Users, groups, redirection and searching

Basic philosophy about users and groups. Principles of redirection, pipes and regular expressions. Command: id, su, passwd, chsh, groups, newgrp, grep.

4. Attributes of files and directories

Detailed description of attributes in connection with users and groups. Linking of files. Commands: chown, chgrp, chmod, umask, stat, touch, ln, echo, printf, wc.

5. Text editor vim

Basic principles and commands. Regular expressions again.

6. Sorting and selecting

Commands: sort, head, tail, nl, uniq, tac, shuf.

7. Finding

Detailed description of command find. Commands: find, xargs, cut, tr.

8. Processes

Philosophy and manipulation with processes. Jobs in the terminal. Commands: ps, top, kill. Commands oriented on a compression: tar, gzip, bzip2, xz.

9. sed - stream editor

Complete look on a command sed. Regular expressions again. Other commands: diff, vimdiff.

10. awk

Complete look on command awk. Other commands: join, paste, split.

11. bash scripts 1

Basic principles of typing bash scripts. Basic syntax.

12. bash scripts 2

Extended principles of typing bash scripts.

Only first 10 exercises are realized by the proposed method. Exercise no. 5 is quite specific. Of course, it is based on the GTA challenge but after students read text of a task they started up editor vim. They had limited keystrokes to solve the task. It was inspired by VimGolf [6]. This procedure was also used before the automation takes place. The last two sessions are about bash scripts. Students are taught how to assemble known commands into a script. The results of these exercises are bash scripts with desired properties.

The presentation with exercise takes one and half hours. There is an effort that the presentation is short but it takes about half an hour. Students are in a computer room whole time. At the beginning the presentation was separated in a lecture room without computers. But it has turned out that it is better to join these two parts. An missed time at the start of exercises is saved.

The presentation is made through a data projector. The pdf slides consist of descriptions and examples how to use commands. These slides can be consider as a quick tutorial and students can use it when they are solving exercises. It was found out that it is better to demonstrate examples online in console. And therefore the slides are used as a side document.

After the lecture (with exercises) students can study materials at home. But it was happening very often that students came on the next lecture and they did not see it at all. For this reason homework were introduced. It is a questionnaire with simple questions about an actual lecture. Moodle [2] provides tools to manage it easily. Questions are of choice type mainly but some of embedded answers type or drag and drop type are used. It is not aim to force students to study home long time but to bring students to repeat and refresh information at least. Therefore the questionnaires are short and the time limit to solve it is about 10 minutes. The score for homework is one point only. (The exercises have 5 points. Overall 100 points can be obtained.) Small score does not force students to cheats as it is assumed.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

4. RESULTS

From the beginning it is used a feedback in a form of a questionnaire. Questions were filled out by students at home and mirrored skills obtained on exercises. Students have limited time to solve it, typically 10 minutes. The questions for a student are chosen from sets randomly. These sets of questions have not been changed significantly for 5 years.

On Figure 13 total score of students' answers of the questions can be seen. Columns mean weeks and rows are in percentages. The course consists of 12 exercises and homework. In average, the number of students is ca 50 per course.

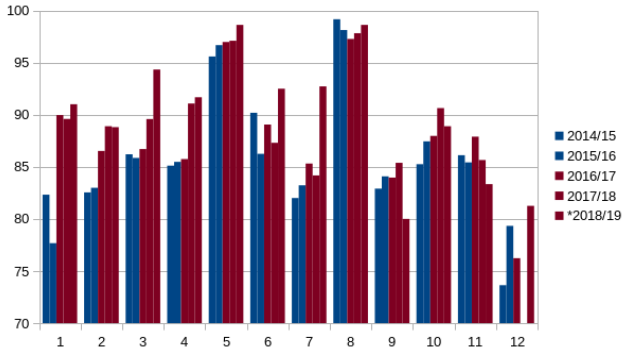


Figure 13. The result score of questions which students make at home. Columns mean weeks and rows are in percentages.

The first two school years the standard form of exercises was used. Students read text of a task and tried to find solution. The command which fulfilled the task students pasted into Moodle [2]. The answers were checked offline later. Next, in following three years the proposed approach was used at the course. The original tasks were incorporated into the challenge files. Students obtained the challenge file and started it. Then they went forward according to instructions.

The last school year 2018/19 is only realized with 6 students. It is an other course which is timed in the winter term. It is intended for students who do not study informatics.

On Figure 13 the columns correspond to thematic section straightforwardly. Therefore columns 11 and 12 are not significant for discussion because students make and send bash scripts through Moodle [2] on exercises. Also column 5 has not significant changes in score. This exercise has had same process for 5 years. Column 8 is a week with a thematic orientation on Linux processes. It means listing, killing, stopping, ... There are not differences in the score.

In Table 1 results summaries can be seen. Concretely, significance levels. It has been selected the minimal score from the standard teaching method and the maximal score from the proposed new teaching method. Then significance levels have been computed. Typical boundary is 10%. Every level below 10% can be considered as a reliable. It means the score from the proposed method overcome the standard method in more cases.

Table 1. A significance level α between minimum and maximum value.

Week	1	2	3	4	5	6	7	8	9	10	11	12
α [%]	1	1	0	3	4	16	0	41	13	8	31	35

Overall it can be seen that the thematic sections oriented on a filesystem manipulation has better scores at significance level α=0.1. But a stream manipulation (week 6, 9, i.e. sort, sed, ...) has comparable scores.

5. CONCLUSION

The new method to teach GNU/Linux command line skills was presented. As results show this method outperforms standard approach of teaching. Students can solve exercises with own speed individually and they have feedback in real time. The method can be presented as automatic tutoring system. This method also helps teacher to manage a bigger group of students at once.

In this paper the results are shown about using the method in course Linux for users which is oriented for beginners. Most of the work is solely in the terminal. The next stage is a course which is named Linux - principles and means. Remake of exercises has started and it is used partially. This course is oriented on setting up some properties of Linux and therefore tests whether a result state is good are more complicated. But first experiences show that students go forward more quickly. Atmosphere in class is also more peaceful because they need not wait to encourage by a teacher so often.

ACKNOWLEDGMENTS

I would like to thanks students for helping with GTA development and preparing exercises in a novel form.

REFERENCES

[1] Raspberry Pi, https://www.raspberrypi.org/
[2] Moodle, https://moodle.org/
[3] GTA, Go Term Adventures, https://github.com/NaiveNeuron/go-term-adventures
[4] The Bandit wargame, http://overthewire.org/wargames/bandit
[5] GNU Bash, https://www.gnu.org/software/bash/
[6] VimGolf, https://www.vimgolf.com/

Automatická kontrola programových kódů ve výuce programování

Automatic checking of program codes in teaching programming

Ingrid Nagyová
Slovenská technická univerzita
Ilkovičova 2
842 16 Bratislava
Slovensko
ingrid.nagyova@stuba.sk

ABSTRACT

In this paper, we describe the possibilities of automatic checking of program codes created by students in the programming teaching, which is performed by special software systems. On particular examples it demonstrates the possibilities of these systems to find faults in the student's solution and give him appropriate feedback. If feedback provides the student with appropriate and relevant information, it has significant didactic potential and helps him to do better on similar tasks in the future. However, examples also point to the opposite side of teaching with the use systems for automatic checking, which can manifest by formalism in teaching.

Keywords

Program. Feedback. System for automatic control of code. Syntax and semantics. Optimization.

ABSTRAKT

Příspěvek se zabývá možnostmi automatické kontroly programových kódu vytvářených studenty při výuce programování, která je prováděna speciálními softwarovými systémy. Na konkrétních příkladech demonstruje možnosti těchto systémů vyhledat chyby v řešení studenta a poskytnout mu přiměřenou zpětnou vazbu. Pokud zpětná vazba dává studentovi vhodné a potřebné informace, má významný didaktický potenciál při řešení podobných úloh do budoucnosti. Příklady však ukazují i na odvrácenou stranu výuky s využitím systémů automatické kontroly, která se může projevit formalizmem ve výuce.

Klíčová slova

Program. Zpětná vazba. Systém pro automatickou kontrolu kódu. Syntax a sémantika. Optimalizace.

1. ÚVOD

Automatická kontrola programů, obvykle podle výstupů z programu, se při výuce programování používá poměrně často. Žák vypracuje program a spustí kontrolu. Ta se spouští opakovaně pro různé vstupy s úmyslem postupně vést studenta ke správnému řešení zadané úlohy. Každý z testů odhalí některou z obvyklých chyb začátečníků, student chybu vyhledá a opraví, až se postupně dopracuje k správnému řešení. Systém umožňující automatickou kontrolu programových kódů tak může výrazně ulehčit práci učitele, který by jinak musel všechny programy kontrolovat ručně. Příspěvek se zabývá vzdělávacími aspekty systému pro automatickou kontrolu programových kódů. Mapuje možnosti, které podobný systém může výuce programování nabídnout. Současně se ale snaží upozornit na úskalí a problémy výuky v takovémto prostředí, na hranice a omezení, která podobná prostředí nutně musí mít.

2. SYTÉMY PRO AUTOMATICKOU KONTROLU KÓDU

Myšlenka využít možnost automatické kontroly programů je s výukovou programování spjata od nepaměti. První systémy a jejich popis jsme zaznamenali již na konci sedmdesátých let 20. století [1, 2]. Od té doby bylo publikováno mnoho článků a provedeno mnoho výzkumů zejména v oblasti vývoje těchto systémů. Články se zaměřují zejména na technickou stránku tvorby inteligentního systému pro revizi úloh. Využívají vztahy mezi jednotlivými prvky programovacího jazyka a ty implementují do systému ve snaze o maximální úspěšnost a přesnost při vyhledání a identifikaci chyb v programech studentů.

Crow a kol. [3] se pokouší z různých zdrojů mapovat známé inteligentní systémy pro podporu výuky programování. Nahlíží na tyto systémy z různých pohledů a z hlediska podpory výuky je dělí na dvě skupiny:

- *Systémy poskytující zpětnou vazbu* – hodnotí práci studenta, obvykle bodově. Existují ale i systémy, které se snaží o co nejpřesnější identifikací chyb, a to jak syntaktických, tak sémantických, ale také chyb ve struktuře a efektivnosti (rychlost, využití paměti) kódu apod. – viz klasifikace zpětné vazby [4].
- *Systémy s automatickou navigací* – automaticky vedou studenta od úlohy k úloze, obvykle v posloupnosti dané učebnicí. Existují ale i systémy, které reflektují předchozí práci a úspěchy studenta a snaží se o inteligentní navigaci podle aktuálních znalostí a potřeb studenta.

Se systémem automatické revize programů se v nejrůznějších podobách setkáme na většině škol, kde se programování vyučuje pro větší skupiny studentů. Například na Matematicko-fyzikální fakultě Univerzity Karlovy v Praze se využívá systém ReCodEx [5] (náhrada systému CodEx), který slouží nejenom k odevzdávání úloh, ale dokáže také úlohy spouštět a kontrolovat výstupy

z programů. Vhodně zvolenými testy tak dokáže navést studenty k přemýšlení nad problémem z různých pohledů a následně k nutné verifikaci algoritmu a úpravě postupu řešení. Na ČVUT v Praze je využíván systém BRUTE [6], který rovněž dokáže úlohy nejenom archivovat, ale také automaticky hodnotit a přidělovat body podle nastavení. Příkladem používání úlohového serveru na Slovensku je systém L.I.S.T. [7] provozovaný Fakultou matematiky, fyziky a informatiky Univerzity Komenského v Bratislavě.

Všechny tyto systémy pracují s osobními údaji studentů a s jejich pracemi, proto jsou chráněny heslem a je obtížné získat o nich bližší informace. Dále se proto zaměříme na konkrétní systém pro odevzdávání úloh z programování, a to na systém Turing [8], který je využíván na Fakultě informatiky a informačních technologií na Slovenské technické univerzitě v Bratislavě.

2.1 Systém Turing

Systém Turing je poměrně komplexní systém pro odevzdávání a automatické hodnocení úloh z programování vytvořených v jazyce C. Systém sestává z několik vzájemně propojených částí.

Turing obsahuje jednoduchý editor, který mohou studenti využít k tvorbě a editaci zdrojového kódu. Pro přehlednost editor barevně zvýrazňuje některé části kódu – příkazy jazyka C, komentáře, textové řetězce apod. Editor nesignalizuje ani neopravuje chyby v kódu, dokáže pouze informovat o chybách, které odhalí překladač (využíván je překladač GCC). Součástí systému nejsou žádné nástroje pro ladění programu. Přesto je editor studenty poměrně často využíván (viz kapitola 3.6).

Důležitou části systému jsou možnosti evidence a kontroly programátorských úloh. Studenti v Turingu najdou přesně specifikované zadání úlohy včetně ukázky vstupů a výstupů programu. V prostředí jsou připraveny testovací scénáře (obvykle 2-3 k dané úloze, u složitějších úloh i více), na kterých má student možnost ověřit si správnost svého řešení. V případě správného řešení student získává body, které se automaticky přičtou k celkovému bodovému zisku. Částečně správná řešení nejsou bodově hodnocena. Učitel má jenom omezené možnosti zasahovat do automatického hodnocení provedeného systémem.

Systém Turing je využíván ve výuce čtyř předmětů zaměřených na výuku programování. Jednotlivé předměty navštěvuje cca 200 studentů, proto je kontrola úloh bez elektronické podpory ve formě systému Turing nemyslitelná. Příspěvek zpracovává data z výuky v zimním semestru akademického roku 2018/2019. Prezentované úlohy byly zadány studentům ve dvou různých předmětech:

- Procedurální programování – pro studenty 1. ročníku; předmět absolvovalo 220 studentů.
- Datové struktury a algoritmy pro studenty druhých ročníků.

3. PŘÍKLADY POUŽITÍ

Dále uvádíme příklady vybraných úloh, které jsou přes systém Turing zadávány studentům. Na těchto příkladech a postupu jejich automatického vyhodnocení se pokusíme demonstrovat možnosti a výhody automatické revize úloh z programování, ale také problémy, které mohou podobný projekt provázet.

Navigace v systému Turing je dána posloupností úloh dané učitelem. Systém automatickou navigaci neumožňuje. Zaměříme se proto na poskytovanou zpětnou vazbu. Budeme postupovat podle Nguyen-Thinh [4], který definuje jednotlivé aspekty a formy adaptivní zpětné vazby ve vzdělávacích systémech pro výuku programování následovně:

- zpětná vazba typu špatně/dobře;
- syntaktické chyby v programu;
- zpětná vazba související se sémantikou programu – je prováděna na dvou úrovních, na úrovni záměrů nebo zvolené strategie řešení úlohy a na úrovni chyb v řešení úlohy s ohledem na její zadání;
- rozvržení (struktura) programového kódu;
- optimalizace řešení, jeho časová i prostorová efektivnost.

3.1 Formátování textů

Na začátku kurzu Procedurální programování studenti řeší úlohu na formátování textového řetězce.

Úloha 1:

Napište program, který vypíše následující řetězec:

```
\*/Toto je "backslash": '\'\*/
```

Před prvním znakem by měl být tabulátor, za posledním znakem by měl být přechod na nový řádek. □

Úloha byla automaticky vyhodnocena systémem Turing. Systém vyhodnotil jako správné 198 (90%) řešení. Z nich pouze 35 (18%) bylo zcela správných podle zadání. Pouze 61 (31%) řešení mělo podle zadání na začátku znak tabulátoru a 122 (62%) mělo na konci znak nového řádku. Většina řešení obsahovala nejrůznější chyby, které systém nedokázal odhalit.

Příčinou neúspěšnosti systému odhalit chyby při řešení podobných úloh je zaměření na testování výstupů. Při řešení této úlohy by automatická revize měla zkontrolovat řetězec znak po znaku, přesně podle zadání. To je ale neefektivní v mnoha jiných případech, například pokud je výstup správný bez ohledu na znak konec řádku na konci výpisu nebo bez ohledu na přebytečné mezery na začátku řetězce. V řetězcích se také často zaměňují lomítko a opačné lomítko nebo uvozovky s apostrofem.

V úloze 1 se vyžaduje přesně zadaný výstup a ten by měl být přesně znak po znaku zkontrolován. Automatická kontrola by měla mít možnost nastavení tak, aby bylo možné zkontrolovat daný výstup přesně nebo aby bylo možné abstrahovat od drobných nepřesností (například ve zpracování řetězce), kterým ani přesným zadáním nelze vždy předejít.

3.2 Formátování vstupu a výstupu

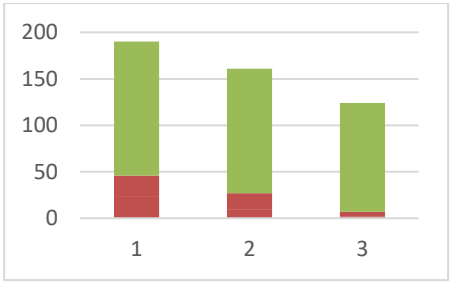
Když se studenti seznámí s proměnnými a jejich základními typy, je nutné, aby uměli načíst hodnoty proměnných a také je vhodně vypsat. Úvodní úlohy při výuce programování v jazyce C jsou obvykle zaměřeny na formátování vstupu a výstupu [9].

Úloha 2:

Napište program, který načte tři reálná čísla a vypíše jejich aritmetický průměr. Vstupem je jeden řádek obsahující tři reálná čísla oddělená mezerou. Výstupem je jeden řádek obsahující zprávu `Prumer cisel x y z je: p` následovaný znakem konce řádku. Čísla `x`, `y` a `z` jsou načtená čísla a `p` je vypočtený průměr. Čísla na výstupu vypisujte jenom na platný počet desetinných míst (použijte formát `%g`). □

Studenti postupně řeší tři podobné úlohy zaměřené na vstup a výstup programu. Kromě úlohy na výpočet průměru (viz úloha 2) řeší úlohu na výpočet ceny s daní či indexu tělesné hmotnosti (BMI). Postup řešení úloh, jak je zaznamenaný v prostředí Turing, poukazuje na jeho výukové možnosti.

Vybrali jsme 83 náhodných studentů, jejichž řešení označil Turing za správné. Obrázek 1 srovnává pro tyto studenty počty pokusů o odevzdání v jednotlivých úlohách (první, druhé a třetí), a to včetně závěrečného správného řešení. Klesající tendence grafu, tj. klesající počet pokusů o odevzdání, poukazuje na to, že studenti postupně nabývali zkušenosti v práci. Pokud v první úloze nacházíme nejrůznější překlepy, chybějící písmena nebo znaky, tyto chyby jsou postupně v řešeních dalších úloh eliminovány. Rovněž je patrné, že pokud v první úloze mnoho studentů zkoušelo formátovací značky pro celá a reálná čísla, ve třetí úloze většina studentů naformátovala výstup napoprvé.



Obrázek 1. Počet pokusů o odevzdání v úlohách

V dolní části grafu jsou barevně zvýrazněny chyby ve formátování výstupů. I zde je patrna významná klesající tendence – z původních 46 chyb ve formátování výstupu v první úloze klesl počet na 7 chyb v poslední třetí úloze.

Výsledky poukazují na skutečnost, že při vhodných úlohách a jejich skladbě má systém automatického hodnocení potenciál formovat výukový proces studentů a naučit je přesněji a pozorněji pracovat.

3.3 Funkce

V programech je žádoucí strukturovat kód do funkcí. Ty se studenti učí nejprve vytvářet přesně podle zadání – jméno funkce, její parametry a popis funkčnosti.

Úloha 3:

Napište program, který bude vypisovat tabulku mocnin čísla `x` od 1 do `n`. Výpočet bude realizovat funkce `void mocnina(double x, int n)`. Na vstupu budou dvě čísla oddělená mezerou. První bude reálné číslo `x`, které představuje základ mocniny, druhé bude celé číslo představující maximální exponent (číslo `n`). Výstupem je `n` řádků, přičemž `i`-tý řádek má formát `x ^ i = y`, kde `x` je základ mocniny, `i` je aktuální exponent a `y` výsledek umocnění. □

Úloha jednoznačně popisuje, jak má funkce vypadat. Ze 197 řešení, které systém Turing vyhodnotil jako správné, ve 4 případech nebylo použito správné jméno funkce. Ve třech dalších případech nebyla funkce v programu správně deklarována (na správném místě v programu) a ve čtyřech případech dokonce nebyla funkce použita vůbec. To znamená, že 11 (6%) řešení nebylo systémem správně ohodnoceno, bez ohledu na správnost funkce podle zadání. Nízký počet neodhalených chyb je dán spíše obecně malou chybovostí – pokud je úloha takto definována, málokoho napadne funkci nedeklarovat a v programu nepoužít.

Automatické hodnocení podobných úloh nelze provádět kontrolou výstupů z programu. Ty v tomto případě od správné implementace funkce, jejich parametrů apod. nutně nezávisí. Problém nelze vyřešit ani jednoduchým vyhledáním řetězce s názvem funkce, resp. hlavičkou funkce v řešení úlohy. Hlavička funkce může být ukryta v komentářích nebo na jiném nevhodném místě v programu. Navíc kontrola na pouhou přítomnost požadovaného řetězce (například názvu funkce) může být až matoucí.

> **Příhoda:** V Turingu byl zadán úkol zjistit, zda zadané číslo *n* je prvočíslo metodou ověřování dělitelů. Pro optimalizaci algoritmu byla vyžadována kontrola v rozmezí 2 až $\sqrt{n}$. Protože výpočet byl verifikován pro malá čísla *n*, časová kontrola nebyla možná. V kódu byl proto pouze vyhledáván řetězec `sqrt` (odmocnina), bez ohledu na jeho souvislost s výpočtem. Student nedokázal pochopit, jak je možné, že jeho řešení je správné již v okamžiku, kdy se tento řetězec v jeho kódu vyskytuje – bez souvislosti s počtem iterací v cyklu.

Přestože systém Turing v těchto jednoduchých úlohách neprovede správnou kontrolu implementace funkce, s automatickým testováním funkcí v složitých projektech si systém poradit umí.

Úloha 4:

Ve standardní knihovně jazyka C jsou pro alokaci a uvolnění paměti k dispozici funkce `malloc` a `free`. Implementujte vlastní verzi alokace paměti pomocí čtyř funkcí:

```
void *memory_alloc(unsigned int size);
int memory_free(void *valid_ptr);
int memory_check(void *ptr);
void memory_init(void *ptr, unsigned int size); □
```

Testování této úlohy proběhlo odlišným způsobem. Z projektů studentů byl odstraněn hlavní program, který byl nahrazen testovacími scénáři připravenými v Turingu. Scénáře využívaly volání jednotlivých funkcí (včetně jejich parametrů) a systém kontroloval změny provedené na výstupu (v paměti). O výsledcích kontroly systém studenty průběžně informoval (viz obrázek 2), čímž jim pomohl odhalit chyby v řešení a přispěl tak k vyšší efektivnosti řešení.

```
Scenar #1 -- 100 bytov, rovnake bloky 8
alloc(8) -> 94282147164552
alloc(8) -> 94282147164568
alloc(8) -> 94282147164584
alloc(8) -> 94282147164600
alloc(8) -> 94282147164616
alloc(8) -> 94282147164632
alloc(8) -> NULL (FAILED)
free(94282147164552) OK (length 8).
free(94282147164616) OK (length 8).
free(94282147164568) OK (length 8).
free(94282147164584) OK (length 8).
alloc(8) -> 94282147164552
alloc(8) -> 94282147164568
alloc(8) -> 94282147164584
alloc(8) -> 94282147164616
alloc(8) -> NULL (FAILED)
free(94282147164632) OK (length 8).
free(94282147164600) OK (length 8).
free(94282147164616) OK (length 8).
Nezvladli ste alokovat vsetky pozadovane urovne pamate -- cim lepsi
Statistika pridelovania (vyhradeny blok: 100 bytes)
1: poziadavka na uroven spotrebovanej pamate: 50 / dosiahnute: 48
2: poziadavka na uroven spotrebovanej pamate: 20 / dosiahnute: 16
3: poziadavka na uroven spotrebovanej pamate: 70 / dosiahnute: 48
4: poziadavka na uroven spotrebovanej pamate: 30 / dosiahnute: 24
Pocet uspesnych alokacii: 10
Uspesnost: 83.33%
```

Obrázek 2. Informace o kontrole projektu ze sytému Turing

Tento způsob automatické kontroly kódu lze využít u složitějších projektů, kdy je předpoklad, že práce s funkcemi nečiní studentům problém. Automatická kontrola může navíc poskytnout užitečnou zpětnou vazbu na úrovni sémantiky programu a jeho optimalizace – viz obrázek 2, informace o úspěšné alokaci a uvolnění požadované paměti a informace o celkové úspěšnosti projektu.

U jednodušších projektů (viz úloha 3) je problém revize programového kódu složitější. Zde je důležitý hlavní program i způsob volání funkce. Pokud kontrola není řádně provedena, studenti opakují chyby v dalších úlohách. To se projevilo ve všech 11 případech chybné deklarace funkce, kdy studenti opakovali stejnou chybu v dalších 2-3 úlohách. Studenti si při prvním setkání s funkcí vytváří představu toho, jak má program s funkcí vypadat. Pokud tato představa zůstává studentem nepochopena a navíc je dokonce chybná, stává se tento poznatek pro studenta formálním [10] a dále jenom stěží použitelným.

3.4 „Chytáky“

Automatická kontrola úloh pomocí počítače může vyvolat mnohá nedorozumění a některé úlohy se mohou studentům jevit jako „chytáky“ – zdánlivě velice jednoduché, avšak najít správné řešení nemusí být vždy snadné.

Úloha 5:

Napište program, který ze standardního vstupu načte dvě celá čísla a vypíše všechna prvočísla z daného intervalu. V programu použijte funkci `int zisti_prvocislo(int cislo)`, jejímž parametrem je dané číslo a návratovou hodnotou je číslo 1, je-li dané číslo prvočíslem, anebo -1 v opačném případě. □

Tato zdánlivě jednoduchá a nekontroverzní úloha se může ukázat problematickou, pokud je automaticky testován výstup z programu. Podle definice je prvočíslo každé číslo větší než 1, které má jenom dva dělitele – jedničku a sebe sama. Ověřit podmínku dělitelnosti v cyklu nečiní studentům problém. Pokud se však do testovacích vstupů dostanou neočekávané hodnoty – jednička, nebo dokonce záporná čísla, přijít na chybu v kódu a opravit ji není triviální.

V této úloze se výrazně zvýšil neúspěch studentů – úlohu dokázalo uspokojivě vyřešit pouze 156 (71%) studentů. Z nich 101 (46%) vyřešilo úlohu napoprvé. Zbývajících 55 studentů využilo testovací možnosti systému Turing a k řešení se postupně dopracovali. Většina z nich zapomněla ošetřit případ záporných čísel a s úlohou měli hodně problémů.

Při analýze řešení úlohy 5 se ukázali dva možné přístupy. První nejprve předpokládá, že zadané číslo `cislo` je prvočíslo a pokud najde dělitele čísla (mimo 1 a sebe sama), dané číslo prvočíslem není – viz zápis v programovém kódu. Po skončení výpočtu nabývá proměnná `je` hodnotu 1 nejenom pro prvočísla, ale také pro jedničku a záporná čísla. Při tomto postupu řešení je nutné tyto případy speciálně ošetřit.

```
int je = 1;
for(int i = 2; i < cislo; i++)
  if (cislo % i == 0)
    je = 0;
```

Ve druhém případě spočteme přímo počet dělitelů zadaného čísla – viz programový kód níže. Počet dělitelů se pro prvočísla rovná 2, pro složená čísla je větší než dvě a pro jedničku a záporná čísla je nulový. To, zda dané číslo je či není prvočíslo lze zjistit jednoduchým dotazem na hodnotu proměnné `pocet`.

```
int pocet = 0;
for(int i = 1; i <= cislo; i++)
  if (cislo % i == 0)
    pocet += 1;
```

Tento zjednodušený postup použilo 31 (15%) studentů, kteří také úlohu snadno napoprvé vyřešili.

Úloha 6:

Napište funkci `int nasobky(int *x, int pocetx, int *y, int k)`, která z pole `x` překopíruje do pole `y` všechny násobky čísla `k` ($k \geq 0$), v pořadí, v jakém se v poli `x` nachází a vrátí počet prvků pole `y`. Argument `pocetx` určuje počet prvků pole `x`. □

Úloha je zaměřena hlavně na kopírování prvků pole a na realokaci pole `y`. Přesto se nakonec ukázalo, že nejvíce času zabrala studentům ne práce s poli, ale práce a odhalení chyby v řešení přesně podle zadání – případ $k = 0$. Když proměnná `k` nabývá hodnotu nula, nemůže být použita ve výrazech v pozici dělitele a tento případ je v programu nutné speciálně ošetřit.

Úloha 7:

Napište program, který ze vstupu načte počet `n` prvků posloupnosti, dynamicky alokuje pole `n` celých čísel a do tohoto pole načte `n` prvků posloupnosti. Dále do proměnné `suma` načte celé číslo a vypíše nejkratší posloupnost, jejíž součet je roven hodnotě proměnné `suma`. Pokud taková posloupnost neexistuje, vypíše se zpráva: `Hledana posloupnost neexistuje`. □

Při hledání nejkratší posloupnosti je situace přehledná – postupně procházíme prvky posloupnosti a od daného prvku hledáme posloupnost, jejíž součet je roven hodnotě `suma`. Při překročení hodnoty `suma`, není nutné v hledání posloupnosti pokračovat. Z možných posloupností vyhledáme nejkratší.

Situace se značně zkomplikuje, pokud chceme vyhledat nejdelší posloupnost. V posloupnosti mohou být obecně i záporná čísla, která hledání nejkratší posloupnosti neovlivní, při hledání nejdelší posloupnosti však sehrávají důležitou úlohu. Sledujme následující posloupnost, v níž vyhledáme nejdelší souvislou posloupnost se součtem 15:

```
10 9 8 1 2 3 4 13 6 7 10 9 8 -29 2 3 5 5 6 7
```

Výsledkem je posloupnost `7 10 9 8 -29 2 3 5`. Při jejím hledání se již součtem prvních dvou čísel 7 a 10 dostáváme nad hodnotu `suma`. Odhalit tento problém činil studentům velké potíže a úlohu správně vyřešilo pouze 66 (30%) studentů.

Úloha 8:

Napište funkci `int NSD(int a, int b)`, která zjistí největší společný dělitel dvou čísel. Předpokládejme, že argumenty funkce mohou být i záporná čísla. □

Hledání největšího společného dělitele (dále NSD) je klasická úloha, kterou studenti znají z nižších stupňů škol. V programování sestavují pro výpočet NSD funkci, jejímiž parametry nejsou pouze kladná čísla, ale celá čísla obecně. Pro efektivní výpočet NSD se používá Euklidův algoritmus. Ten, stejně jako vyhledání největšího čísla dělícího obě čísla beze zbytku, nemusí pro záporná čísla správně fungovat. V tomto případě studenti většinou před výpočtem správně převedli záporný vstupní argument na kladné číslo, upravit znaménko v konečném výsledku zapomněli napoprvé podle záznamu v systému Turing všichni.

Prezentované „chytáky“ jsou úlohy, na kterých si studenti mají ujasnit obecně známé pojmy a rozšířit jejich chápání v pro ně netypických situacích – v oblasti tvorby programů. Tyto situace přímo vyplývají z reprezentace dat v paměti počítače a jejich zvládnutí je pro studeny nezbytné. Navíc jsou dané pojmy obvykle přesně dány i definitoricky. Protože však studenti nebyli zvyklí nahlížet na předkládané situace z různých pohledů, na jejich jednotlivé detaily, působí na ně tyto úlohy často neobvyklým až zavádějícím dojmem.

Využívání úloh vyžadujících transfer a využití poznatků v netypických situacích je nezbytné pro důsledné osvojení učiva – viz zákony učení podle Ďuriče, zákon transferu[11]. Řešením těchto úloh studenti poznávají způsob myšlení v informatice, jeho specifičnost, jsou vedeni k správným otázkám a problémovým situacím, s nimiž se při programování mohou setkat.

Problémem je automatická kontrola těchto úloh a chybějící nebo nedostatečná zpětná vazba na úrovni specifikace chyb v řešení. Řešení není systémem akceptováno a student nedokáže odhalit proč. Systém se z pohledu studenta chová nepřirozeně – vždyť řešení spouštěné ve vývojovém prostředí a pro vzorové vstupy funguje. To vzbuzuje nedůvěru studentů v systém a automatické hodnocení a daná úloha se jeví jako „chyták“, který je bez pomoci zvenčí stěží řešitelný.

Úloha 9:

Napište program, který bude ze vstupu číst znaky do proměnné `c1`, dokud nenačte znak `'*'`. Každý načtený znak se překopíruje do proměnné `c2`, přičemž nebude použit příkaz `c2=c1`. Nepoužívejte žádné jiné proměnné než `c1`, `c2` a `p_c` – ukazatel na typ `char`. Kolika způsoby se to dá udělat? Výstupem programu je jeden řádek pro každý načtený znak ve formátu:

```
c1: x (y), c2: z (p), p_c: q (r)
```

kde `x` je obsah proměnné `c1` a `y` adresa této proměnné. Podobně, `z` je obsah proměnné `c2` a `p` její adresa, `q` představuje hodnotu proměnné, na kterou ukazuje `p_c` a `r` je hodnota `p_c`. □

Poměrně složitě popsaná úloha vyžaduje jednoduchou práci s ukazateli. Její hodnocení systémem Turing ukazuje hranice možností automatické kontroly. Hodnocení systémem Turing bylo totiž velice kontroverzní. Stejný programový kód byl v některých případech hodnocen kladně, v jiných případech byl označen jako chybný. Systém označil pouze 108 (49%) řešení za správné, dalších 95 (43%) řešení bylo označeno jako chybné, i přesto, že tato řešení chybu neobsahovala. Přesnou příčinu chyby systému se nepovedlo odhalit, lze se jenom domnívat, že to souvisí s prací s ukazateli.

3.5 Pokročilá kontrola úloh

V předmětu Datové struktury a algoritmy byly použity pokročilejší testovací nástroje systému Turing. Jedním z nich je možnost testování velkých projektů nahrazením hlavního programu testovacím scénářem. Ten je vytvářen tak, aby nejenom testoval funkčnost požadovaných části kódu, které student vytvořil, ale současně poskytuje studentovi zpětnou vazbu o optimálnosti a efektivitě jeho řešení (viz úloha 4). Následující úloha demonstruje další nástroje, které lze pro automatickou kontrolu programového kódu použít.

Úloha 10:

Napište funkci, která jako parametr dostane počet `n` všech hraček v obchodě, pole cen `n` hraček a číslo `k` udávající kolik hraček si chceme koupit. Chceme nakupovat jenom dražší hračky, funkce proto vrátí součet cen `k` nejdražších hraček v obchodě. □

Studenti se na přednášce seznámili s algoritmem na vyhledání k-tého největšího prvku posloupnosti, který měli v úloze implementovat. Po krátké době jsme z řešení studentů pochopili, že pole nejprve utřídí a až následně vyhledají k-tý prvek. Proto do zadání přibyl následující požadavek:

> Algoritmus, který řadí ceny třídícím algoritmem v čase $O(n \log n)$, je pomalý. Řešení musí běžet v čase $O(n + k)$.

Současně bylo potřeba přehodnotit a nastavit testovací požadavky systému – časově omezit dobu řešení pro danou posloupnost cen hraček. Studenti reagovali velice rychle a požadovanou lineární složitost se pokusili dosáhnout změnou řadícího algoritmu – využili countingsort, jehož časová složitost odpovídá zadání. Algoritmus však klade neobvyklé požadavky na velikost použité paměti, a proto bylo nutné omezit v systému Turing také velikost paměti pro řešení úlohy.

Nutno konstatovat, že testy na čas a paměť výrazně omezili počet řešení, která systém označil za správná. Tím se úloha stala nejhůře hodnocenou v průběhu celého semestru. Systém označil za správné pouze 78 (35%) řešení (celkový počet studentů byl 222).

Pokud se zaměříme na kvalitu testů přidaných do systému Turing, pak lze při detailnějším rozboru pozorovat, že ze 78 správných řešení bylo pouze 21 (27%) doopravdy správných – bylo v nich využito třídění quicksort nebo řazení haldou. Zbylá řešení vycházela z algoritmu countingsort. Testy na čas a prostor potřebný pro výpočet, které souvisí s optimalizací úlohy, jsou v systému automatického hodnocení použitelné jenom do jisté míry. Studentům stačily drobné úpravy programu, aby systém vyhodnotil jejich řešení jako správné. Mezi „drobné úpravy“ programu patřily například omezení pomocného pole algoritmu countingsort zdola (nejvyšší ceny jsou v horní části pole) anebo velice sofistikovaný countingsort provedený postupně na jednotlivých osminách rozsahu cen (od 0 do maximální ceny).

3.6 Editor programového kódu

Jak již bylo naznačeno, obsahuje systém Turing editor pro editaci a drobné úpravy programového kódu. Editor barevně označuje části kódu – klíčová slova jazyka, konstanty a komentáře. Umožňuje překládat a spouštět programy, zadávat vstupy a zobrazovat výstupy z programu. Tyto vlastnosti jsou ohromnou výhodou editoru integrovaného v testovacím systému.

Velkou nevýhodou editoru v systému Turing je slabá možnost identifikace chyb při překladu (je omezena pouze na schopnost překladače najít a označit chybu) a absence ladících nástrojů. Tyto funkce, které jsou standardem v profesionálních vývojových prostředích, chybí ve výuce hlavně proto, že se s nimi studenti nenaučí zacházet. Editor neoznačuje chyby v programu, mnohé chyby „přehlíží“ a student je tak ochuzen o adekvátní zpětnou vazbu. Navíc nemá možnost zkoušet a objevovat proces ladění programu, pozorovat změny hodnot proměnných za běhu programu či sledovat průběh algoritmu krok po kroku.

Studenti na začátku vytváří programy v profesionálních prostředích (Visual Studio apod.). Editor v systému Turing je však velice rychle svede k dokončení řešení v systému, kde pak často začínají řešit i další úlohy. Obvykle až problémy v kompilaci programu, nemožnost odhalit chyby apod. přivedou studenta zpět k práci v profesionálním prostředí. Z tohoto pohledu nelze implementaci editoru do prostředí automatické revize kódu doporučit.

4. DISKUZE

Systémy pro automatickou revizi programových kódů jsou nesporným přínosem ve výuce programování. Dokážou evidovat a vyhodnocovat v krátkém čase velké množství úkolů, které by jinak nebylo možné odpovídajícím způsobem vyhodnotit. S myšlenkou těchto systémů se setkáváme ve výuce na mnohých vysokých školách, systémy jsou využívány napříč programovacími jazyky, včetně tvorby webových stránek.

Kromě pomoci při opravování velkého množství úloh mohou mít systémy při vhodné prezentaci a seřazení úloh nezanedbatelný vzdělávací efekt. Úlohy musí na sebe navazovat, každá další by měla zopakovat a dále prohloubit problematiku předchozí úlohy. Takové úlohy vedou studenta k postupnému odhalování chybných konceptů, které při práci používá, a tím přispívají k výuce (viz kapitola 3.3).

Systém Turing a možnosti kontroly programů jsme demonstrovali na řadě příkladů. Dále se pokusíme shrnout výsledky s ohledem na možnosti a formy adaptivní zpětné vazby podle Nguyen-Thinh [4].

Systémy pro automatickou kontrolu programového kódu obvykle předpokládají tvorbu programů mimo systém, v profesionálním vývojovém prostředí, které při překladu zkontroluje syntaktické chyby studenta. Prací v tomto prostředí student navíc získá prvotní zpětnou vazbu o správnosti vytvořeného programu jeho spouštěním podle podmínek zadání úlohy. Systémy pro automatickou revizi obvykle neprovádí syntaktickou kontrolu kódu i s ohledem na náročnost této operace, která vyžaduje implementaci překladače daného jazyka.

Rozvržení a celková struktura kódu je důležitým prvkem, který by studenti měli ve výuce programování zvládnout. Konvence struktury kódu jsou pro jednotlivé programovací jazyky obvykle přesně specifikovány. Kontrola struktury kódu není pro svoji obtížnost v systémech automatické kontroly implementována.

Hlavním požadavkem všech systémů automatické kontroly je zpětná vazba typu špatně/dobře. Efektivnost tohoto základního hodnocení programu závisí na několika faktorech:

- na zvoleném překladači a jeho možnostech identifikovat chyby:
- na testovacích podmínkách a omezeních, na testovacích datech (vstupech a výstupech) a na jejich pořadí;
- na typu úlohy – požadavky na přítomnost funkce nebo typ použitých dat se vyhodnocují obtížně (viz úloha 3), naopak komplexnější úlohy lze často snáze vyhodnotit v závislosti na vstupech a výstupech programu.

Jak bylo ukázáno na příkladech, představuje problematika zpětné vazby a správného hodnocení programových kódů komplexní problém, který nemá jednoduché a jednoznačné řešení. Systémy automatické kontroly se snaží o maximální úspěšnost v hodnocení. Chyby systémů, které jsou přímo navázány na jednotlivé faktory hodnocení (viz výše), přesto nelze vyloučit. Jistá omezení jsou dána použitým překladačem – viz úloha 9. Snížit úroveň hodnocení mohou nesprávně stanovené testovací podmínky – viz úloha 1. Testovací scénáře se zaměřují na předpokládané a opakující se chyby studentů (úloha 2), na „chytáky“ v zadání apod. Chyby či dokonce chybné stereotypy studentů často nedokážou odhalit (viz úloha 3 – chyby v definici funkce).

Velkým problémem automatického hodnocení je zpětná vazba na úrovní sémantiky programu. Mnohé ze sémantických chyb mohou být systémem automatické revize jednoznačně a pro studenta srozumitelně odhaleny. Studenti v rámci předmětu Procedurální prokazování řešili v systému Turing 42 úloh, které byly obvykle adekvátně hodnoceny. Na druhou stranu pokud má student pouze strohou informaci, že jeho program nedokázal správně zpracovat testovací vstupy a neví, jak tyto vstupy vypadají, neví, jakou chybu v programu udělal, pak klesá jeho důvěra v evaluační systém a často i schopnost nebo ochota chybu vyhledat a opravit. Automatická kontrola podobných „chytáků“ je proto vždy na důsledné zvážení, i přesto, že o nezbytnosti podobných úloh ve výuce programování nelze pochybovat (viz kapitola 3.4).

Velkou předností automatických revizí může být zpětná vazba na úrovni optimalizace řešení, na úrovni časové a prostorové složitosti. Tyto kontroly lze provádět velice jednoduše náhradou hlavního programu testovacím kódem, který nejenom prověří funkčnost jednotlivých částí programu, ale změří čas potřebný pro výpočet nad danou strukturou dat a výsledky ve vhodné formě studentovi prezentuje – viz úlohy 4 a 10.

ZÁVĚR

Systémy pro automatické hodnocení programů jsou v dnešní době ve výuce programování téměř nezbytností. Studenti řeší v rámci jednoho předmětu 30-50 úloh za semestr a jejich ruční vyhodnocení je při větším počtu studentů téměř nepředstavitelné. V příspěvku jsme se pokusili na konkrétních úlohách a jejich automatickém hodnocení systémem Turing ukázat možnosti automatické revize úloh z programování se zaměřením na vzdělávací efekt a působení takové revize na studenta.

Z výsledků vyplývá, že systémy pro automatickou revizi programových kódů se uplatní zejména při zpětné vazbě typu špatně/dobře. Ve speciálních případech mohou poskytnout zpětnou vazbu i na úrovni sémantiky, tj. odhalit chyby v řešení úlohy s ohledem na její zadání. Zde však mohou nastat problémy, pokud je řešení systémem označené jako chybné, ale původ chyby zůstává pro studenta skrytý. Při vhodné implementaci mohou mít systémy automatické kontroly nástroje pro zpětnou vazbu na úrovni optimalizace kódu, tj. v oblasti časové a prostorové složitosti.

BIBLIOGRAFICKÉ ODKAZY

[1] RUTH, Cl. R. Intelligent program analysis. In *Artificial Intelligence*, 1976, vol. 7, no. 1, p. 65-85. ISSN: 0004-3702.

[2] ADAM, A., LAURENT, J. P. LAURA, a system to debug student programs. In *Artificial Intelligence*, 1980, vol. 15, no. 1–2, p. 75-122. ISSN: 0004-3702.

[3] CROW, T., LUXTON-REILLY, A., WUENSCHE, B. Intelligent Tutoring Systems for Programming Education: A Systematic Review. In *ACM International Conference Proceeding Series*, 2018, p. 53-62. ISBN: 978-145036340-2.

[4] NGUYEN-THINH LE. A classification of adaptive feedback in educational systems for programming. In *Systems.* vol. 4 (2016), no. 2, 22: 1-17.

[5] ReCodEx – ReCodEx Code Examiner. https://recodex.mff.cuni.cz/

[6] BRUTE. https://cw.felk.cvut.cz/brute

[7] L.I.S.T. https://list.fmph.uniba.sk/

[8] Turing. https://www.turing.sk/fiit/

[9] KERINGHAM, B. W., RITCHIE, D. M. *Programovací jazyk C*. Bratislava: Alfa, 1988. 250 s.

[10] HEJNÝ, M., NOVOTNÁ, J., STEHLÍKOVÁ, N. *25 kapitol z didaktiky matematiky*. Praha: PdF UK, 2004. 470 s. ISBN 80-7290-189-3.

[11] ĎURIČ, L., GRÁC, J., a ŠTEFANOVIČ, J. *Pedagogická psychológia.* Bratislava: Jaspis, 1991. ISBN: 80-900477-6-9.

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Použití Lego Mindstorms jako platformy pro výuku robotiky za pomoci Linuxu a Pythonu

Using Lego Mindstorms as the teaching platform for the educations of the robotic by the using Linux and Python

Jiří Pech
ÚAI PřF Jihočeská univerzita
Branišovská 1760
370 06 České Budějovice
ČR
pechj@prf.jcu.cz

Milan Novák
ÚAI PřF Jihočeská univerzita
Branišovská 1760
370 06 České Budějovice
ČR
novis@prf.jcu.cz

ABSTRACT
This paper is focused on the possibility of running Linux on Lego brick from Lego Mindstorms construction set. Linux distribution Ev3dev was used and the description of its installation is also included in the text. The list of programming languages that can be used in this distribution follows further in the article. The most attention is focused on programming language EV3 Python, the example of Explor3r robot illustrates its potential and application. Commented example of program for robot's autonomous movement among obstacles is presented further. In the end, the experience with usage of this software is summarized.

Keywords
Lego Mindstorms, Ev3dev Linux. EV3 Python.

ABSTRAKT
V tomto příspěvku se zabýváme možností spouštění Linuxu na počítači – cihličce ze stavebnice Lego Mindstorms. Diskutována je distribuce Linuxu Ev3dev, u které je vysvětlena její instalace. Dále jsou uvedeny programovací jazyky, které je možné v této distribuci použít. Pozornost je soustředěna zejména na programovací jazyk EV3 Python, jeho možnosti a použití je předvedeno na příkladu robota Explor3r. Je uvedena okomentovaná ukázka programu pro autonomní jízdu tohoto robota v členitém prostředí. V závěru pak jsou shrnuty zkušenosti s používáním tohoto software.

Klíčové slova
Lego Mindstorms, Ev3dev Linux. EV3 Python.

1 ÚVOD
Lego Mindstorms má nezastupitelnou roli ve výuce informatiky, programování a robotiky od základních škol po univerzity.

Bohužel je mezi vyučujícími povětšinou známa, pouze možnost programování pomocí grafického nástroje ROBOLAB založeném na LabVIEW, spočívajícím v přetahování a umísťování bloků na obrazovce. Touto metodou mnozí pedagogové na vyšších stupních škol zejména technických ať právem, či neprávem pohrdají. Málokdo z nich však alespoň tuší, že na cihličce (Lego Brick), která skrývá řídící počítač Lego Mindstorms lze použít plnohodnotný Linux, který je vlastně upravený Debian. A navíc jeho použitím není nijak dotčen původní firmware, takže je možné stále pracovat s Lego cihličkou jako dříve.

2 EV3DEV Linux
Ev3dev Linux [1] je komunitou vyvíjená distribuce Linuxu určená k běhu na cihličkách Lego a dále na počítačích Raspberry Pi nebo BeagleBone propojených s Legem. Primárně je určená k ovládání modelů sestavených ze stavebnice Lego Mindstorms. Aktuální verze je založena na poslední verzi Debianu, tedy Debianu 9 Stretch.

Ev3dev se instaluje na micro SD kartu a z ní také startuje. Původní software tak není nijak dotčen a můžeme jej stále používat. Pokud například potřebujeme ve výuce, aby jedna třída pracovala s Linuxem a jiná s původním softwarem, není to tak žádný problém.

Pro Ev3dev je k dispozici několik programovacích jazyků, které obsahují podporu pro práci s periferiemi ze stavebnice Mindstorms. Jsou to:

- Python [2]
- Java
- Go
- C++
- C
- Jazyky Vala a Genie založené na knihovně GObject

Pro příznivce grafického programování pak je k dispozici prostředí Open Roberta Lab [3] [4] s programovacím jazykem NEPO, ve kterém se programuje podobně jako ve Scratchi.

2.1 Příprava na instalaci
Co potřebujeme k instalaci? Počítač, na kterém Ev3dev poběží. Máme na výběr buď originální cihličku Lego nebo Raspberry Pi či BeagleBone. Máme-li Raspberry Pi potřebujeme dále rozšíření BrickPi a PiStorms a k počítači BeagleBone potřebujeme rozšíření FatCatLab a QuestCape (které však už není v prodeji).

Tento článek se bude nadále zabývat pouze instalací a používáním na cihličce Lego.

Pro instalaci potřebujeme Micro SD kartu o kapacitě minimálně 4 GB. Pokud budeme chtít uchovávat větší množství programů, pak je lepší volbou 8 GB karta. Tato kapacita je naprosto dostatečná.

Dále potřebujeme WiFi dongle. Ačkoliv je instalace možná i bez něj, práce s Ev3dev bez vzdáleného přístupu přes SSH je obtížná a nepohodlná. Lze se připojit pomocí USB kabelu, ale nemůžeme

ISBN 978-80-557-1533-9, ISSN 2454-051X.

pak vzdáleně ovládat cihličku mimo dosah kabelu. Na rozdíl od originálního firmware spolupracuje Ev3dev s většinou WiFi dongle, které můžeme zakoupit. Abychom upřesnili – autoři se v praxi dosud nesetkali s nekompatibilním dongle. Je to dáno tím, že linuxová jádra obsahují podporu pro většinu běžně dostupného hardware.

2.2 Instalace

Nyní již můžeme přistoupit k samotné instalaci. Ze stránek Ev3dev Linuxu si stáhneme image požadované verze systému. Je třeba si dát pozor, abychom stáhli verzi pro Lego Mindstorms (a ne pro Raspberry nebo BeagleBone). Stahujte co nejnovější verzi, aby obsahovala co nejnovější knihovny programovacích jazyků.

Image systému nyní nafleshujte na micro SD kartu. Autoři Ev3dev Linuxu doporučují program Etcher [5], ale můžete samozřejmě použít libovolný jiný program se stejnou funkcí. V Linuxu a MacOS třeba obyčejné `dd`.

Po úspěšném nafleshování vložte micro SD kartu do zdířky na levé straně Lego Mindstorms. Současně zasuňte hned vedle do USB portu WiFi dongle. Je vhodné, aby byl zasunut již při startu systému, linuxové jádro jej rozpozná a samo použije odpovídající ovladač.

Máte-li rovněž vloženy dostatečně nabité baterie nebo akumulátor, zapněte Lego cihličku. LED na čelní straně se zbarví do červena a po malé chvilce začne oranžově blikat. To nám oznamuje, že systém startuje. Nová verze Ev3dev Linuxu již vypisuje na displej cihličky informace o průběhu startu, takže víme, co se děje a zda se systém z nějakého důvodu nezasekl. První start trvá poměrně dlouho – v řádu minut, další starty již jsou poměrně rychlé. Pokud vše proběhne v pořádku na displeji se objeví základní menu.

Máte-li vložen WiFi dongle a k dispozici vhodnou WiFi síť (pozor nelze použít Eduroam – alespoň autorům se jej nepodařilo rozchodit), je čas nastavit Wifi. Z nabídky Wireless and Networks vyberte Connect, zvolte vhodnou síť a zadejte heslo, pomocí ovládacího kříže cihličky. Není to příliš pohodlné a možná se vám to ani napoprvé nepovede, ale systém si toto nastavení pamatuje a příště tuto činnost nemusíte opakovat.

Pokud vše dopadne dobře, uvidíte na displeji nahoře IP adresu, která byla cihličce přidělena. Nyní nastal čas se k systému připojit pomocí SSH.

Pokud nemáte k dispozici WiFi dongle, můžete připojit cihličku pomocí USB kabelu a postupovat dle návodu na stránkách [1].

2.3 Nastavení

Nyní máme připravený a nabootovaný systém a známe jeho IP adresu. Je čas se k němu připojit, dokončit konfiguraci a začít jej používat. Pro připojení ze systému Windows použijte program PuTTY, z MacOS nebo Linuxu běžný příkazový řádek a SSH.

Přihlašovací defaultní jméno je `robot` a heslo `maker`. Tento uživatel také může implicitně získat právo roota pomocí programu `sudo`.

Nyní jsme v příkazovém řádku. Ti z vás, kteří se vyznají v Linuxu se nyní cítí jistě jako doma. Můžete nyní zjistit některé informace o systému, např., že je použit dvaatřicetibitový ARM procesor a systém má pouze 64 MB RAM!

Pokud se v Linuxu nevyznáte, doporučuji nejprve absolvovat nějaký linuxový kurz pro úplné začátečníky. Potřebujete alespoň tyto znalosti:

- Editace souborů
- Správa souborů
- Instalace a spouštění programů

Nyní rovněž můžete pomocí příkazu `passwd` změnit defaultní heslo pro účet robot a popřípadě i přidat dalšího uživatele, kterému nezapomeňte nastavit povolení pro `sudo`. Pokud nevíte, jak se to dělá, nepouštějte se do toho. Pokud máte více cihliček (pro třídu), tak nedoporučuji měnit ani uživatele, ani hesla, přes prázdniny to zapomenete nebo ztratíte papírek, kde jsou napsaná.

Doporučuji provést aktualizaci zdrojů software a upgradovat software pomocí známé dvojice příkazů:

```
apt update
apt upgrade
```

Pozor může to nějakou dobu trvat, klidně i hodinu nebo více, systém je opravdu pomalý. Na druhou stranu získáte nejnovější verze programovacích jazyků a podpůrných knihoven.

Další své oblíbené programy můžete vyhledat a nainstalovat pomocí apt, např. pro správce souborů Midnight Commander:

```
apt search mc
apt install mc
```

Případné další nastavení a konfiguraci zde nebudeme popisovat, to ostatně ani není cílem tohoto článku. Pokud vás zajímá více, opět vás odkazuji na stránky [1].

Ještě je třeba se zmínit o vypínání systému. Ačkoliv (zejména uživatele znalé Linuxu) bude svádět použít pro vypnutí systému příkaz `sudo halt` anebo `sudo shutdown now`, nedělejte to, systém se zasekne a jediná možnost, jak jej znovu oživit je vyndat z něj baterie. Pro vypnutí anebo restart používejte pouze menu a klávesy cihličky.

Pokud chcete nastartovat Lego cihličku do originálního systému, postačí vyjmout micro SD kartu a případně i odstranit WiFi dongle. Pro bezproblémové vyjmutí micro SD karty doporučujeme pinzetu.

3 PROGRAMOVÁNÍ v EV3DEV

Přehled použitelných programovacích jazyků byl podán již v kapitole 2. My si nyní pouze něco krátce řekneme o kompilovaných jazycích a pak se budeme věnovat Pythonu.

3.1 Kompilované jazyky

Pokud pracujete s jazyky typu C, C++ anebo Go, pak je pro vás dobrá zpráva, že je máte k dispozici i zde včetně všech knihoven. Autoři mají zkušenosti pouze s použitím jazyka Go, proto se nebudeme hlouběji těmito jazyky zabývat.

Obecně platí, že vzhledem k výše popsanému slabému hardware cihličky není dobrý nápad kompilovat programy přímo na cihličce. Ostatně zde ani nemusí být vše nainstalováno. Mohly by vám dojít systémové zdroje, a navíc by kompilace trvala opravdu dlouho. Daleko lepší nápad je provést takzvanou cross kompilaci. Ta spočívá v tom, že na svůj počítač si nahrajete potřebný překladač a knihovny pro práci s ev3dev. Kompilaci pak provedete na svém počítači a přeložený program si zkopírujete pomocí scp (MacOS, Linux) nebo WinSCP (Windows) na cihličku a tam jej spustíte. Vše je poměrně zdlouhavé, ale rozhodně rychlejší než se snažit kompilovat přímo na cihličce. Návody k jednotlivým programovacím jazykům najdete v odkazech uvedených na stránce [1] v sekci programing languages.

3.2 Open Roberta Lab

Poměrně zajímavá možnost je grafické programování pomocí nástroje Open Roberta Lab. Na webové stránce [4] si vytvoříte účet a přihlásíte se k němu. Vyberete možnost EV3 a ev3dev. Pomocí

ISBN 978-80-557-1533-9, ISSN 2454-051X.

grafického prostředí sestavíte program. Prostředí trochu připomíná grafické prostředí pro programování Micro:bitu.

Před použitím je nutné z příkazového řádku Ev3dev Linuxu, tuto službu odemknout, spustit a povolit:

```
sudo systemctl unmask openrobertalab.service
sudo systemctl start openrobertalab.service
sudo systemctl enable openrobertalab.service
```

Nyní v menu Ev3dev spustíte možnost Open Roberta Lab a obdržíte kód. Tento kód zadáte na stránce [4] a program se vám podle pokynů buď přímo spustí nebo stáhne do cihličky. Je rovněž třeba na web stránce nejprve nastavit připojení motorů a senzorů k jednotlivým portům cihličky. Když si jej prohlédnete, ať v cihličce nebo na stránkách [4], zjistíte, že se vlastně jedná o kód v Pythonu.

Zůstává trochu otázka použitelnosti tohoto systému. Spuštění je poměrně zdlouhavé a pracné a je nutné nainstalovat ev3dev Linux atd. Proto pokud chcete programovat v grafickém režimu, myslím je lepší zůstat u původního programovacího nástroje, který má i více možností.

Snad jediné důvody, které nás napadají jsou:

- Stránky Open Roberta Lab jsou kompletně lokalizovány do češtiny, a tak zejména mladším studentům bude programování za použití tohoto nástroje přístupnější.
- Programování v Open Roberta Lab se nám zdá jednodušší a přehlednější než u originálního software. Na druhou stranu originální software je podobný LabView.
- Open Roberta Lab může sloužit jako generátor pythonovského kódu, který pak můžeme ladit, zjednodušovat a upravovat i mimo tento systém.

3.3 EV3dev Python

EV3dev Python se v nedávné době dočkal aktualizaci na verzi 2. Tato verze je obsažena i v poslední verzi Ev3dev Linuxu stretch. Dejte si, proto jak jsme již říkali výše pozor, zda instalujete poslední verzi Ev3dev Linuxu.

Pokud používáte Windows, pak si můžete nainstalovat Microsoft Visual Code s rozšířením EV3 (návod je na [2]). Pokud používáte MacOS nebo Linux, pak je nejlepší způsob přihlásit se k cihličce pomocí SSH, editovat kód pomocí nějakého obyčejného linuxového editoru například `nano program.py`, nastavit ho jako spustitelný `chmod a+x program.py` a pak ho spustit `./program.py`. Samozřejmě tento postup můžete použít i ve Windows a přihlásit se pomocí programu PuTTY.

Vše podstatné k programování pomocí EV3 Pythonu naleznete na stránkách [2]. Je zde mnoho ukázek a podrobný popis jednotlivých knihoven.

Mnoho zejména starších programátorů Pythonem pohrdá, neboť se nepodobá syntaxí oblíbenému C, C++ či Javě. Na druhou stranu kód v Pythonu je přehledný a dobře čitelný (v Pythonu nelze de facto napsat „ošklivý kód“). Python je proto oblíben nejen mezi studenty, ale i mezi uživateli, kteří začínají s programováním v pokročilém věku. Python je navíc jedním ze základních programovacích jazyků v robotice, zejména na platformě Robot Operating System (ROS) [6].

3.4 Robot Explor3r

Pro demonstraci možností EV3 Pythonu na platformě Ev3dev Linux jsme si vybrali robota Explor3r [7].

Jedná se o poměrně jednoduchého robota s dvěma koly poháněnými motory a dvěma senzory. Jednak dotykovým pro detekci překážek a pak volitelně buď detektor vzdálenosti anebo

Obrázek 1: Robot Explor3r

detektor světla pro sledování čáry na podlaze. Na obrázku 1 je s detektorem vzdálenosti.

Jedná se o jednoduchého, snadno ovládatelného robůtka, na kterém se dají dobře demonstrovat různé úlohy pro autonomní chování a současně i různé problémy autonomního chování.

Následující program umožňuje autonomní chování robota, který sám jezdí prostorem a pokud narazí na překážku (ať dotekovým senzorem anebo senzorem vzdálenosti) otočí se doprava pokračuje. Počet kolizí je omezen, aby robot nejezdil do nekonečna. Jedná se o podobný algoritmus, který používají například robotické vysavače v autonomním režimu.

```
!/usr/bin/env python3
#Zavedení potřebných funkcí z knihovny ev3dev
from ev3dev2.motor import LargeMotor, OUTPUT_B, OUTPUT_C, SpeedPercent, MoveTank
from ev3dev2.sensor import INPUT_1, INPUT_4
from ev3dev2.sensor.lego import TouchSensor, InfraredSensor
from ev3dev2.led import Leds
from time import sleep
# Nastavení senzorů
ir = InfraredSensor(INPUT_4)
ts = TouchSensor()
# Nastavení Led panelu
leds = Leds()
# Nastavení rychlosti motorů
tank_pair = MoveTank(OUTPUT_B, OUTPUT_C)
tank_pair.on(left_speed=30, right_speed=30)
#Počáteční nastavení Led panelu
leds.set_color('LEFT', 'GREEN')
leds.set_color('RIGHT', 'GREEN')
# Proměnná pro informaci zda byla detekována
# překážka
stop = False
```

ISBN 978-80-557-1533-9, ISSN 2454-051X.

```
# Počet povolených kolizí
pocet = 20
#Hlavní cyklus programu
while pocet:
  # Narazili jsme do překážky?
  if ts.is_pressed:
    stop = True
  # Je před námi méně než 70 cm volného místa
  if ((ir.proximity * 0.7) < 40):
    stop = True
  # Překážka byla detkována
  if (stop):
    leds.set_color('LEFT',  'RED')
    leds.set_color('RIGHT', 'RED')
    # Couvnout
    tank_pair.off()
    tank_pair.on(left_speed=-10,
right_speed=-10)
    sleep(2)
    # Pootočit
    tank_pair.off()
    tank_pair.on_for_degrees( left_speed=10,
right_speed=30, degrees=180)
    tank_pair.off()
    tank_pair.on(left_speed=30,
right_speed=30)
    # Zrušení detekce překážky
    stop = False
    # Zbývající počet kolizí
    pocet = pocet - 1
  else:
    leds.set_color('LEFT',  'GREEN')
    leds.set_color('RIGHT', 'GREEN')
  sleep (0.01)
# Konec – vše vypnout
tank_pair.off()
leds.all_off()
```

Jedná se o jednoduchý prográmek pro demonstraci autonomního chování a na kterém lze například pěkně demonstrovat nebezpečí uvedení robota do stavu, ze kterého se nedokáže sám dostat.

4 ZKUŠENOSTI S POUŽÍVÁNÍM

EV3 Python a Ev3dev Linux používáme při výuce robotiky na naší fakultě. S jejich nasazením na nižším stupni škol bohužel dosud nemáme zkušenosti. Na základě našich zkušeností s výukou na nižších stupních škol si však dovolujeme tvrdit, že by zde bylo minimálně vhodné je vyzkoušet.

Napadají nás minimálně tyto případy:

- Tam kde se vyučuje robotika a studentům připadá originální Lego software příliš primitivní nebo nepřehledný.
- Tam kde chtějí vyučující zajímavým způsobem oživit výuku.
- Jedná se navíc o zajímavou možnost, jak vyučovat Python anebo navázat na výuku Pythonu. Použití specifických knihoven, metod, objektů atd. studentům obvykle nedělá potíže.
- Zajímavá je rovněž možnost naučit tímto netradičním způsobem studenty práci s Linuxem. Studenti Linux obvykle vnímají velmi dobře a zde je možno jej zajímavě použít na neobvyklém hardware – Lego cihličce.

4.1 Pros and cons

Zkusme si nyní shrnout pro a proti použití těchto nástrojů:

Pro:

- Možnost naučit se (prohloubit si znalosti) Linux a Python v praxi – výsledky programování hned vidíme.
- Oproštění se od grafického programování – v určitém okamžiku již studenti chtějí zkusit něco jiného.
- Výuka robotiky – příprava na reálné programování robotických systémů (příkazový řádek, Python).
- Další využití robotických Lego Mindstorms stavebnic.
- Možnost vyzkoušet i systém Open Roberta.

Proti:

- Není to grafické programování – nutnost posunout se o stupínek výše.
- Nutnost naučit se něco nového, zejména pro učitele.
- Práce v příkazovém řádku může být pro studenty obtížná – nejsou na ni zvyklí.

4.2 Zkušenosti s použitím na VŠ

Rádi bychom ještě shrnuli naše zkušenosti s použitím tohoto software při výuce předmětu Robotika 1 na naší fakultě.

Studenti dostávají za úkol v rámci semestrálních projektů jednoduché roboty, přičemž si mohou zcela vybrat jaký hardware použijí. Jednou z možností je i Lego Mindstorms, kde máme k dispozici čtyři studentské a dvě učitelské stavebnice. Kromě Lego Mindstorms mohou například použít Raspberry Pi, Arduino a na konstrukce například Merkur.

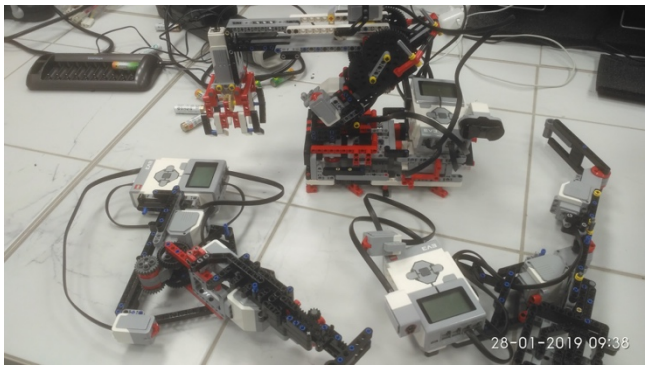
Studenti si rádi volí Lego Mindstorms, pozorujeme na nich že si rádi „hrají s Legem“. Studenti si mohou volit mezi originálním softwarem a popsanou kombinací Ev3dev Linux a EV3 Python. Studenti si bez výjimky zatím zvolili Ev3dev Linux. Přijde jim to jaksi „dospělejší“. Kromě toho si někdy stěžují, že originální grafický software je poměrně nepřehledný.

Co se týče volby EV3 Pythonu, tak není tak jednoznačná, zaznamenali jsme i pokusy použít Javu, ale obecně mají studenti Python rádi, oceňují na něm jednoduchost a přehlednost. Rovněž na studentech pozoruji, že si rádi odpočinou od striktně objektového programování Javy nebo C++. Na druhou stranu i v EV3 Pythonu se objekty používají, tam kde to vede k zjednodušení a zpřehlednění.

Výhodou nasazení na VŠ je to, že studenti v této fázi studia (obvykle letní semestr druhého ročníku) již obvykle mají slušnou znalost jak Linuxu, tak Pythonu. Pro ostatní připravujeme jedno cvičení na seznámení s Linuxem a jedno na seznámení s Pythonem.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

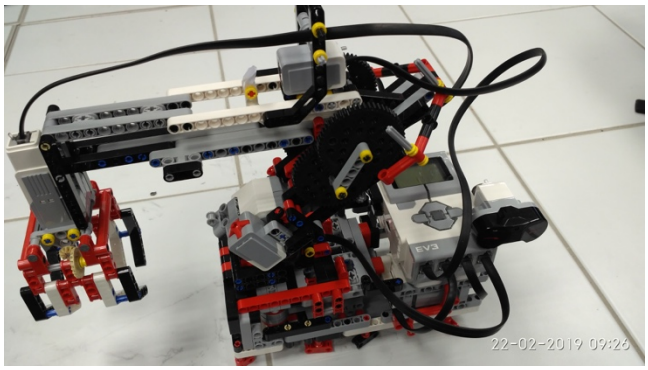
Obecně hodnotí studenti Lego Mindstorms s nainstalovaným Ev3dev Linuxem a EV3 Pythonem velice kladně. Což je i vidět na zajímavých výstupech z jejich prací.



Obrázek 2: Ukázky studentských prací – dvě 2D ruce (plottery), v pozadí 3D ruka

4.3 Příklad úlohy

Ze studentských úloh přikládáme příklad jednoho modelu, který považujeme za zdařilý a zajímavý. Jedná se o robotickou ruku pracující ve 3D prostoru. Viz následující obrázek.



Obrázek 3: 3D ruka

Tato ruka má použity dva velké motory pro pohyb vpravo a vlevo. Malý motor je použit pro otevírání a zavírání kleští na konci ruky.

Velmi vtipně jsou řešeny detekce koncových stavů. Jsou použity dva dotykové spínače. Jeden vidíme na dolním konci ramene, pro detekci maximálního pohybu směrem dolů. Druhý pro detekci maximálního vychýlení vpravo je zabudován uvnitř konstrukce základny. Pro detekci maximálního vychýlení směrem vlevo je použit detektor vzdálenosti, který je vidět zcela vpravo.

Ruku je možné používat v autonomním režimu, tedy předem jí naprogramovat na dané pohyby. Další možností je dávat jí pokyny pomocí čtečky barev, umístěné na horní straně nejdelšího ramene. Ta defaultně snímá bílou součástku a může tak reagovat na jakoukoliv jinou barvu.

Protože motorky měly problém s tím, že nedržely danou výšku anebo směr a například při zatížení sjelo rameno dolů nezávisle na požadované poloze, byla konstrukce vylepšena o díly tištěné na 3D tiskárně. Jedná se o ozubená kola různých velikostí.

5 ZÁVĚR

Použití Ev3dev Linuxu a EV3 Pythonu na Lego Mindstorms je zajímavou možností, jak doplnit použití této oblíbené výukové stavebnice v procesu vzdělávání.

Můžeme doufat, že náš článek zaujme případné čtenáře, kteří se rozhodnou Lego Mindstorms vyzkoušet a případně jím i obohatit výuku.

BIBLIOGRAFICKÉ ODKAZY

[1] Ev3dev Home. Ev3dev is your EV3 re-imagined [online]. [cit. 2019-01-25]. Dostupné z: https://www.ev3dev.org/

[2] EV3dev Python. EV3 Python v2 [online]. [cit. 2019-01-25]. Dostupné z: https://sites.google.com/site/ev3devpython/

[3] Open Roberta WIki [online]. [cit. 2019-01-26]. Dostupné z: https://jira.iais.fraunhofer.de/wiki/display/ORInfo

[4] Open Roberta Lab [online]. [cit. 2019-01-26]. Dostupné z: https://lab.open-roberta.org/

[5] BalenaEtcher [online]. [cit. 2019-01-26]. Dostupné z: https://www.balena.io/etcher/

[6] Robot Operating System [online]. [cit. 2019-01-26]. Dostupné z: http://www.ros.org/

[7] Explor3r Building Instructions [online]. [cit. 2019-01-26]. Dostupné z: http://robotsquare.com/2015/10/06/explor3r-building-instructions/

Problémy v přípravě učitelů informatiky: případová studie
Problems in Computer Science Teacher Preparation: A Case Study

Tomáš Průcha
Západočeská univerzita v Plzni
Univerzitní 8
30100 Plzeň
Česká republika
pruchat@kvd.zcu.cz

Zbyněk Filipi
Západočeská univerzita v Plzni
Univerzitní 8
30100 Plzeň
Česká republika
filipiz@kvd.zcu.cz

Lucie Rohlíková
Západočeská univerzita v Plzni
Univerzitní 8
30100 Plzeň
Česká republika
lrohlik@kvd.zcu.cz

ABSTRACT
In the Czech Republic there is a growing discussion about the preparation of CS (Computer Science) teachers, especially in connection with the integration of CS into the curriculum of primary and secondary schools. The aim of this paper is to describe problems in the preparation of CS teachers in the opinions of academic staff involved in the preparation of CS teachers, using a case study at the University of West Bohemia. The presented results are based on a qualitative research consisting of semi-structured interviews. The most common problems identified are mainly finances and insufficient numbers of students interested in the study of CS teaching, but also problems related to the creation of study programs, their composition and personnel provision of teaching.

Keywords
Computer science teacher preparation. Teacher training. Pre-service teacher. Case study

ABSTRAKT
V souvislosti s integrací informatiky do kurikula základních i středních škol v České republice sílí diskuze o přípravě učitelů informatiky. Cílem tohoto příspěvku je pomocí případové studie realizované na Západočeské univerzitě v Plzni popsat problémy v přípravě učitelů informatiky dle názorů akademických pracovníků, kteří se na její přípravě podílejí. Prezentované výsledky vychází z realizovaného kvalitativního výzkumu sestávajícího z polostrukturovaných rozhovorů. Nejběžnějšími nalezenými problémy jsou především nedostatek financí a nedostatečný počet zájemců o studium učitelství informatiky. Dále se jedná o problémy týkající se tvorby studijních programů, jejich skladby a personálního zabezpečení výuky.

Klíčové slova
Příprava učitelů informatiky. Vzdělávání učitelů. Pregraduální příprava učitelů. Případová studie

1 ÚVOD
Stále vzrůstá zájem o zlepšení kvality výuky informatiky na základních a středních školách a zvýšení počtu učitelů informatiky [1]. Inovace vysokoškolských studijních programů přípravy učitelů v oblasti informatiky je jedním z klíčových prvků k vyřešení prvního ze jmenovaných problémů. Velmi palčivý je ovšem i problém s nedostatečným počtem kvalifikovaných učitelů informatiky [1, 2]. Jakým způsobem by vlastně měla probíhat příprava budoucích učitelů informatiky a jaký vliv na to může mít fakt, že didaktika informatiky jako vědní obor není v České republice zatím jasně vymezena [1]?

Při přípravě budoucích učitelů informatiky je důležitou otázkou, jak studenty připravovat, aby byli schopni udržet krok s rychlým vývojem v oblasti informačních a komunikačních technologií po dokončení studia, který je v této disciplíně běžný [3].

Naléhavost tohoto tématu dokládá např. také aktuálně řešený projekt Podpora rozvíjení informatického myšlení (PRIM), do kterého jsou zapojeny všechny pedagogické fakulty v České republice. Projekt si mimo jiné klade za cíl tvorbu vzorových sylabů předmětů vysokoškolské přípravy učitelů informatiky a úpravu vzdělávání učitelů z praxe i výuky v pregraduální přípravě [4]. Aktuálnost dále dokládá také česko-slovenská konference DidInfo, kde příprava budoucích učitelů informatiky a jejich další vzdělávání je jedním z hlavních témat [5].

Všechny výše uvedené problémy a úsilí o jejich řešení mají globální povahu, což dokládají publikace z mnoha zemí, jako je např. Spojené království, Austrálie, Izrael a další [6, 7, 8].

2 METODOLOGIE
Zde popsaný výzkum byl realizován v rámci diplomové práce Srovnání vysokoškolského vzdělávání budoucích učitelů informatiky v ČR a USA [9]. Cílem práce bylo provést srovnávací studii zaměřenou na možnosti vysokoškolského vzdělávání budoucích učitelů informatiky v České republice a ve Spojených státech. Pro srovnávací studii jsme vycházeli z designu mnohonásobné případové studie tak, jak ji popisují Hendl [10] a Goodricková [11].

Základním cílem výzkumu bylo provést empirickou deskripci obou stanovených kontextů. V ní byla pozornost soustředěna na vlastnosti kontextů, které jsou vnímány jako klíčové v souvislosti s přípravou učitelů informatiky. Následně proběhl samotný výzkum zaměřený na přípravu učitelů informatiky s vlastním sběrem dat.

Srovnávanými případy pro nás byla příprava učitelů informatiky v ČR a USA. V souladu s tím, co uvádí Lor [12], jsme naše případy vybrali tak, aby zde daná problematika ještě nebyla hlouběji studována, a zároveň jsme si byli jistí, že pro realizaci výzkumu budeme mít vhodné podmínky a dobrý přístup.

Podle Hendla [10] mají být případové studie pružné, co se týče množství a typu dat. Ve zvolených případech jsme provedli analýzu všech souvisejících dokumentů, zejména kurikulárních a poté vlastní sběr dat. Pro vlastní sběr dat jsme využili hloubkových rozhovorů s tvůrci a garanty studijních programů pro učitele informatiky. Dle Hendla [10] získané výsledky zobecníme při popisu dnešních paradigmat v přípravě učitelů informatiky.

Na základě výše uvedených cílů byly stanoveny tyto obecné otázky výzkumu.

O_1: Jaká paradigmata dnes můžeme pozorovat v souvislosti s přípravou učitelů informatiky?

O_2: Jaké rozdíly lze pozorovat ve vysokoškolské přípravě učitelů informatiky mezi vybranými kontexty?

V tomto příspěvku se nesnažíme podat odpověď na obě výše uvedené otázky, ale zaměřujeme pozornost zejména na otázku O_1 a vyhodnocení dat získaných od respondentů ze Západočeské univerzity v Plzni.

2.1 Sběr dat

V rámci výzkumu byly na Západočeské univerzitě v Plzni realizovány polostrukturované rozhovory zaměřené na přípravu učitelů informatiky. Proběhla rovněž analýza studijních plánů programů přípravy učitelů informatiky.

Pro analýzu studijních programů jsme využili volně dostupné informace na stránkách univerzity, kde je k dispozici popis studijních programů i jednotlivých předmětů [13].

Respondenty rozhovorů byli akademičtí pracovníci univerzity, kteří se v rámci své výuky a výzkumu věnují přípravě budoucích učitelů informatiky. Jejich prvotní seznam byl sestaven výzkumným týmem vzhledem k dobré znalosti tohoto kontextu a jejich nábor probíhal pomocí osobního kontaktu, případně pomocí e-mailu. Nábor dalších respondentů probíhal metodou sněhové koule, kdy byl každý respondent po rozhovoru dotázán, zda může doporučit nějaké další respondenty vhodné do tohoto výzkumu. Výzkumník pak uvedené osoby na základě doporučení kontaktoval s žádostí o zapojení se do výzkumu.

Rozhovory byly realizovány osobně v soukromých prostorách (např. univerzitní kancelář či konferenční místnost). Délka rozhovoru zpravidla nepřesahovala jednu hodinu. Každý rozhovor byl zvukově zaznamenán na diktafon a později přepsán pro potřeby kvalitativní analýzy pomocí software ATLAS.ti.

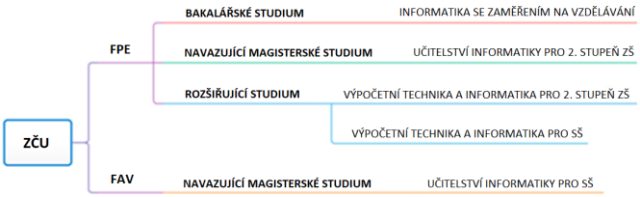
Pro tematické rozkrytí textu byly přepisy zakódovány pomocí procedury otevřeného kódování, kdy v první fázi kódování došlo k označení částí textu v návaznosti na obecná témata vycházejících z pokládaných otázek v rámci rozhovorů. Následně jsme se zaměřili na axiální kódování, kdy jsme hledali vztahy, příčiny a důsledky mezi vniklými kategoriemi kódů. Následně jsme pokračovali procedurou selektivního kódování, kdy jsme se zaměřili na vyhledání hlavních kategorií a témat, které budou zastřešovat již vzniklé kategorie a podkategorie [10].

2.2 Kontext: struktura studia učitelství informatiky zkoumaného případu

Na Západočeské univerzitě v Plzni (ZČU) lze studovat učitelství informatiky na Fakultě pedagogické (FPE). Na bakalářské úrovni je zde pro studenty možnost studia Informatiky se zaměřením na vzdělávání, které je dvouoborové (student si volí druhou aprobaci) a profilem absolventa nemusí nutně být učitelství. Navazující magisterské studia v oblasti informatiky nabízí tato univerzita dvě. Jeden pro druhý stupeň základních škol, vyučovaný na Fakultě pedagogické a druhý pro střední školy. Zmíněnou aprobaci pro střední školy zde vyučuje Fakulta aplikovaných věd (FAV) v rámci svého navazujícího magisterského programu Matematika. V tomto programu lze studovat obor učitelství matematiky, který je spojen s dalším aprobačním předmětem – učitelstvím informatiky. V tomto oboru se tedy jedná o přípravu učitelů informatiky pro střední školy.

Další možností jsou programy rozšiřujícího studia (jak pro ZŠ, tak SŠ), které jsou koncipovány pro již aktivní učitele, kteří si chtějí svojí aprobaci o informatiku rozšířit. Rozšiřující studium zajišťuje Fakulta pedagogická.

Celková struktura možností studia pro získání kvalifikace pro učitelství informatiky na ZČU je patrná z obrázku 1.



Obrázek 1: Vizualizace studijních programů na ZČU

3 VÝSLEDKY A DISKUZE

Do výsledků prezentovaných v tomto příspěvku byly zahrnuty výpovědi respondentů podílejících se na přípravě učitelů v programech Fakulty pedagogické i Fakulty aplikovaných věd Západočeské univerzity v Plzni a konkrétně se zaměřujeme na identifikované problémy v přípravě učitelů informatiky. V následujícím textu problémy diskutujeme a uvádíme autentická vyjádření respondentů.

Jedním z výrazných problémů, na které respondenti poukazovali je fakt, že střední školy nelze unifikovat a jejich velká diverzita, zejména u středních odborných škol klade i různé požadavky na přípravu učitelů.

> *Není tomu tak, že bych si představil, že odborné znalosti učitele na ZŠ a SŠ jsou ty samé, naopak, já si myslím, že musíme rozlišovat několik typů škol. A já ty SŠ nechci házet do jednoho pytle … na SŠ máme SŠ určitého typu, kde nejvýznamnější skupinou jsou asi gymnázia, kde je všeobecné vzdělaní, kde se ty učitele hodně blíží těm učitelům ZŠ v tom, že učí většinou i na víceletých gymnáziích v nižších stupních, ale zároveň potřebuji jít více do hloubky tak, aby dokázali předávat poznatky na těch výších stupních gymnázií. Pak tu máme SŠ odborné školy, které jsou velmi specifické a kde naopak mnohdy je lepším učitelem inženýr se znalostí didaktiky.*

Řešení problému přípravy učitelů informatiky pro střední školy by mohla poskytnout větší spolupráce Fakulty pedagogické s Fakultou aplikovaných věd, zejména pak v souvislosti s přípravou učitelů pro střední odborné školy (SOŠ), které se zpravidla profilují v jedné konkrétní oblasti, a zmíněna fakulta poskytuje dostatek odborníků, kteří tyto oblasti z hlediska výuky pokryjí.

> *Ten přínos, zejména při vzdělávaní pro střední odborné školy, tam by byl velký přínos … Výrazným kladem pro ty studenty by byla kvalifikovanost, nějakým způsobem, pokud by to učil prostě odborník na tu problematiku.*

Respondenti na druhou stranu vyjádřili obavu z možné spolupráce fakult při přípravě učitelů, zejména v souvislosti s přípravou učitelů pro 2. stupeň ZŠ. Argumenty byly zejména to, že silná odborná příprava v oblasti informatiky nemusí být u těchto učitelů klíčová.

> *U základních škol si to úplně nemyslím, a nemyslím si to z toho důvodu, že ta představa kvalitního učitele je zcela odlišná, na naší přírodovědné fakultě, Fakultě aplikovaných věd, od toho co my si myslíme, že by měl ten učitel umět, znát, jakým by měl byt člověkem. Takže tam to úplně zatím nevidím.*

Respondenti shodně označili jako jeden z problémů s nedostatkem učitelů, resp. učitelů informatiky, postavení učitelského povolání ve společnosti.

> *Ten hlavní problém je v přijímání učitelského povolání ve společnosti. Kdyby se zatraktivnilo učitelské povolání ve společnosti, tak by to vedlo k zvýšení zájmu, ať už v tom bakalářském, magisterském, nebo rozšiřujícím.*

Respondenti z Fakulty pedagogické upozorňují na problém s členěním učitelského studia na bakalářské a magisterské. Po absolvování bakalářského studia studenta nic nenutí pokračovat v magisterském studiu a často ti dobří studenti odcházejí na technicky zaměřené školy, nebo do firem.

> *Jsme si vědomi, že ty absolventi mají ty tři cesty.*

Dalším z problémů, který se promítá do přípravy učitelů informatiky, je finanční ohodnocení absolventů. Respondenti vyjádřili v tomto ohledu názor, že nízké finanční ohodnocení učitelů je u učitelů informatiky aktuálnější než u jiných oborů a je důvodem, proč velké množství potenciálních učitelů informatiky ze školství odchází. Možné řešení vidí respondenti v zatraktivnění studia. Zároveň ale upozorňují na to, že k řešení je více komplexních problémů.

> *Některé učitelské obory potřebují vetší podporu, stát by si měl říct, že učitelů informatiky je nedostatek, víme jak, proč to je a tak dále, tak prostě já nevím, budou nějakým způsobem stipendijně podpořeni studenti učitelství informatiky … To, že utečou do tý disciplíny samotný [informatiky] se může stát, protože je to začne hrozně bavit, ne ta kantořina. Anebo že je tam mnohem více peněz. To je, bych řekl, mnohem častější případ v našich končinách. Myslím, že ty peníze v tom IT jsou ten faktor, kterej bude jinej oproti, já nevím, historii třeba, tam asi nehrozí, že by odešli dělat historika, protože, ty peníze budou podobný … Často se stává, že u nás po tom bakalářském studiu skončí ti nejkvalitnější studenti, když se podívají na finanční ohodnocení učitele a podívají se do některé firmy, tak prostě k nám nejdou.*

Diferenciovat mezi přípravou učitelů pro 2. stupeň ZŠ a SŠ se ukázalo z pohledu respondentů jako klíčové. Zejména z pohledu odborných znalostí informatiky. Učitel 2. stupně ZŠ si může dovolit určitý nadhled a zjednodušení, při zachování celkového širokého rozhledu nad všemi oblasti informatiky.

> *Učitel pro ZŠ se na počítačové sítě dívá z určitého nadhledu a zjednodušení, kdežto učitel SŠ naopak potřebuje, pokud je na odborně zaměřené škole, jít více do hloubky, přesně specifikovat, tak aby studenti tomu porozuměli, ale i ten učitel na ZŠ, by měl mít některé základní znalosti z počítačových sítí a neměl by být takzvaně o kapitolu napřed … Ten přístup, kdy získají v těch bakalářských programech ten základ stejný a potom se to větví v těch magisterských programech, je správný. Je výrazný rozdíl, když ten váš absolvent učí, řeknu, nějaký šesťáky a je významný rozdíl, když pak učí nějaké čtvrťáky na obchodní akademii, vnímám v tom opravdu poměrně dost velký rozdíl.*

Učitelé pro 2. stupeň ZŠ se zaměřují spíše na práci s dětmi, na vytvoření si didaktického nadhledu a pracují zejména na tom, jak informace předávat.

> *Studenti často zvažovali, zda jejich odborné znalosti jsou vhodné úrovně pro střední školy a v tom navazujícím studiu rozlišovali, jestli se hlásí pro ZŠ, nebo pro SŠ a bylo to i ve vztahu k tomu, kde se oni vidí, jestli chtějí pracovat více s dětma, nebo chtějí více předávat informace. Dá se říct, že tam vidím jakoby rozdíl v prací na 2. stupni ZŠ, kde ti, kteří více dbali na ty odborné znalosti a jejich předávaní a na to, že se musí do hloubky problému, se více profilovali, více pro střední školy, kdežto ti, kteří hledali ten didaktický nadhled, to, jak to předávat, jak vůbec pracovat s dětma, jak je rozvíjet, se více seberealizovali v těch základních školách.*

Studium na Fakultě pedagogické je postavené tak, že student během bakalářského studia získá široký přehled v oblastech informatiky a pochopí její základní principy. Studium na bakalářské úrovni nerozlišuje mezi učiteli pro ZŠ a SŠ. Následně student naváže v rámci magisterského studia na tyto poznatky didaktikou v závislosti na zvoleném stupni školy.

> *U bakalářského studia nám šlo o to pochopit principy, né nutně jít do hloubky, pochopit principy … Pro ty střední školy, na tý magisterský úrovni, tam máme trošku jiné programovaní, jinak kreditově hodnocené a trošku jinak udělané … jsou to drobné rozdíly, významný rozdíl je, že vedeme jinak tu didaktiku.*

4 ZÁVĚR

Obě fakulty hovoří o možnosti dvou-fakultního studijního programu pro přípravu učitelů informatiky, který by mohl být odpovědí na některé problémy, zejména týkající se personálního, odborného a finančního zabezpečení.

V souvislosti s personálním zabezpečením se zde také velmi projevuje fakt, na který upozorňuje Vaníček [1], tedy v podstatě neexistence habilitačního pracoviště čistě v oblasti didaktiky informatiky a tedy problém se získáním dostatečné kvalifikace akademických pracovníků.

Jako o možném řešení odborného zabezpečení výuky a případně i finančním zabezpečení mluví respondenti o možnosti společných předmětů pro studenty učitelství informatiky a studentů odborných programů informatiky. Zejména pak v začátcích studia a pro získání odborného základu.

V názorech akademických pracovníků lze pozorovat také to, že jejich představa o přípravě učitelů informatiky může být zcela rozdílná v závislosti na tom, na které fakultě působí. Tento nesoulad může mít na finální podobu studijních programů a tedy samotnou přípravu výrazný vliv. Gal-Ezer a Habiballa v izraelském kontextu uvádí, že všichni učitelé informatiky musí mít nezbytné formální vzdělání v informatice, které odpovídá alespoň bakalářskému odbornému studiu informatiky [14]. Z názorů respondentů působících na Fakultě pedagogické lze ale usuzovat, že formální vzdělání v informatice nemusí být nutný předpoklad k tomu být dobrým učitelem informatiky, zejména na ZŠ. Stanovení potřebného odborného základu pro učitele informatiky tak může být klíčové a Fakulta pedagogická ZČU tak proto připravuje budoucí učitele ve vlastním bakalářském programu Informatika ve vzdělávání, který se ve srovnání s bakalářským odborným studiem informatiky na Fakultě aplikovaných věd podstatně liší.

Dále respondenti upozorňují na problém se SOŠ, kde je učitel zpravidla zaměřen na jednu danou oblast informatiky dle typu školy a měl by tak mít možnost volit si zaměření již během studia, čemuž by měly odpovídat i studijní programy.

Dalším z problémů, který respondenti v přípravě učitelů informatiky popisují, je v tom, že studium je strukturované, tedy dělené na bakalářské a magisterské a často se stává, že studenti po úspěšném dokončení bakalářského studia na Fakultě pedagogické, které poskytuje široké znalosti z oblasti informatiky, odchází jiným směrem, než je učitelství samotné. Často se to děje z důvodu lepšího finančního ohodnocení ve firemním sektoru. Tento problém respondenti popisují jako specifický právě pro oblast informatiky, jelikož ve firemním sektoru jsou enormně širší možnosti, jak se uplatnit, včetně platu.

Obecným problémem je pak ohodnocení učitelské profese, které neláká nové studenty oboru. Respondenti mluví o možnosti systematické podpory ze strany státu pro studenty vybraných učitelských aprobací, zároveň ale upozorňují, že se jedná o komplexnější problém, jehož řešení není snadné.

Tato případová studie byla pilotním vstupem do problematiky, její závěry tudíž nejsou zobecnitelné. Do budoucna je v plánu rozšíření o další kontexty a získání většího vzorku respondentů, což umožní širší možnosti zobecnění získaných výsledků.

PODĚKOVÁNÍ

Výzkum byl finančně podpořen z projektu SGS-2018-019 Studentské grantové soutěže Západočeské univerzity v Plzni.

BIBLIOGRAFICKÉ ODKAZY

[1] STUCHLÍKOVÁ, I., JANÍK, T., BENEŠ, Z., et al. *Oborové didaktiky: vývoj, stav, perspektivy*. Brno: Masarykova univerzita, 2015.

[2] FIDRMUC, J., "Jaké digitální dovednosti mají mít žáci na konci ZŠ?," *Digikoalice*, 2017.

[3] MENEKSE, M., "Computer science teacher professional development in the United States: a review of studies published between 2004 and 2014," *Computer Science Education*, vol. 25, no. 4, pp. 325-350. 2015. DOI: https://doi.org/10.1080/08993408.2015.1111645

[4] Projekt „Podpora rozvíjení informatického myšlení" [online]. [cit. 2019-01-28]. Dostupné z: https://imysleni.cz/o-projektu

[5] DIDINFO 2019 [online]. [cit. 2019-01-31]. Dostupné z: http://www.didinfo.net

[6] THE ROYAL SOCIETY, *Shut Down Or Restart? The Way Forward for Computing in UK Schools* [online]. London: The Royal Society, 2012. [cit. 2019-01-28]. Dostupné z https://royalsociety.org/~/media/education/computing-in-schools/2012-01-12-computing-in-schools.pdf

[7] FALKNER, K., VIVIAN, R., FALKNER, N., & WILLIAMS, S., "Reflecting on Three Offerings of a Community-Centric MOOC for K-6 Computer Science Teachers," *SIGCSE '17*, 2017. DOI: https://doi.org/10.1145/3017680.3017712

[8] GAL-EZER, J., STEPHENSON, C., "A Tale of Two Countries: Successes and Challenges in K-12 Computer Science Education in Israel and the United States," *ACM Trans. Comput. Educ.*, vol. 14, no. 2, 2014. DOI: http://dx.doi.org/10.1145/2602483

[9] PRŮCHA, T., *Srovnání vysokoškolského vzdělávání budoucích učitelů informatiky v ČR a USA*, Diplomová práce, Západočeská univerzita v Plzni, 2018.

[10] HENDL, J., *Kvalitativní výzkum: základní teorie, metody a aplikace*. Praha: Portál, 2016.

[11] GOODRICK, D., "Comparative Case Studies" [online]. *Methodological Briefs Impact Evaluation*, no. 9, 2014. [cit. 2019-01-28]. Dostupné z: https://www.unicef-irc.org/publications/pdf/brief_9_comparativecasestudies_eng.pdf

[12] LOR, P., *International and comparative librarianship*. Berlin: De Gruyter Saur, 2011.

[13] Západočeská univerzita v Plzni, Prohlížení IS/STAG [online]. [cit. 2019-01-28]. Dostupné z: https://portal.zcu.cz/portal/studium/prohlizeni.html

[14] GAL-EZER, J., HABIBALLA, H., "Unikátní učební osnovy pro SŠ v Izraeli," *Matematika-fyzika-informatika*, vol. 11, no. 2, pp. 104-113, 2006.

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Matematická zložka vo vyučovaní programovania
Mathematical component in teaching programming

Ľubomír Salanci
Fakulta matematiky, fyziky a informatiky
Univerzita Komenského
Mlynská dolina
842 48 Bratislava
Slovenská republika
salanci@fmph.uniba.sk

ABSTRACT
We have created a set of worksheets about programming in Python for students of high school. Worksheets are designed for beginners and they cover basic programming topics such us variables, subroutines, loops and branching. We were interested in cognitive density of our worksheets and they relationship to mathematics. We found out that students should solve nearly 250 tasks by themselves. The analysis pointed out that the mentioned topics, in a fact, consist of many smaller computer-science concepts, together 68. After all, students have to learn up to 30 syntactic rules. It implies the following densities: one concept per 3.5 tasks and one syntactic rule per 8 tasks. It is interesting, that 53 % of computer-science concepts and 80 % tasks have mathematical character, although it was not our objective.
.

Keywords
Teaching and learning. Programming. Mathematics. PRIM.

ABSTRAKT
Pre žiakov stredných škôl sme vytvorili sadu pracovných listov o programovaní v jazyku Python. Tie sú určené začiatočníkom a pokývajú základné témy z programovania, akými sú premenné, podprogramy, cykly a vetvenia. Zaujímala nás kognitívna hustota našich pracovných listov a ich vzťah k matematike. Zistili sme, že žiaci majú samostatne vyriešiť takmer 250 úloh. Analýza poukázala na to, že spomínané základné programátorské témy pozostávajú z množstva drobných informatických ideí či konceptov, dohromady ich je 68. Popritom sa žiaci stretnú s takmer 30 syntaktickými pravidlami. Z toho vyplývajú nasledujúce hustoty: na 1 koncept pripadá 3,5 úlohy a na 1 nové syntaktické pravidlo vyše 8 úloh. Je zaujímavé, že až 53 % informatických konceptov a až 80 % úloh má matematický charakter, hoci to vôbec nebolo našim cieľom.

Kľúčové slová
Vyučovanie a učenie sa. Programovanie. Matematika. PRIM.

1. ÚVOD

V rámci českého národného projektu PRIM sme sa podieľali na tvorbe vzdelávacích materiálov z programovania v jazyku Python [1]. Pre žiakov stredných škôl sme vytvorili pracovné listy s úlohami. Učitelia získali k jednotlivým pracovným listom aj metodické materiály. Našim zámerom bolo, aby žiaci sami riešili úlohy a popri tom sami spoznávali a objavovali programátorské koncepty. Učiteľ by mal vystupovať v roli pomocníka, radcu, na ktorého sa žiaci v prípade potreby obrátia.

Pri návrhu sme vychádzali zo situácie, ktorá je daná stavom vyučovania informatiky (a zvlášť programovania) v českých školách. Bolo prijaté rozhodnutie, že pracovné listy vytvoríme pre žiakov – začiatočníkov. Tí, ktorí programovať vedia alebo niektoré koncepty poznajú, môžu úlohy preskočiť, prípadne ich vyriešia rýchlejšie. Pôvodný zámer bol, aby jeden pracovný list zodpovedal približne jednej vyučovacej hodine.

Výsledkom je séria 19 pracovných listov. Tie prešli prvotným overovaním na troch stredných školách rôzneho zamerania. Už toto testovanie poukázalo na veľké rozdiely medzi žiakmi. Napríklad, úlohy z prvých pracovných listov dokázali niektorí žiaci vyriešiť za 35, iným trvalo riešenie 45 minút. Úlohy z nasledujúcich pracovných listov žiaci riešili 45 minút až 2 vyučovacie hodiny, individuálne aj viac. Podľa ohlasov od učiteľov, žiaci zvládnu 4 až 5 úloh za hodinu. Celkovo sme však z prvotného testovania získali pomerne pozitívne vyjadrenia.

Z testovania vyplynuli aj niektoré nečakané a vcelku zaujímavé zistenia. Napríklad, že žiaci majú problém s počítačovou súradnicovou sústavou nielen na začiatku, ale aj na niekoľkých nasledujúcich vyučovacích hodinách. Pritom profesionálni programátori považujú počítačovú súradnicovú sústavu za samozrejmú až banálnu záležitosť. Navyše aj v diskusii s nimi zisťujeme, že si akoby neuvedomujú, koľko veľa matematiky naše úlohy obsahujú. A hoci je táto matematika na úrovni základnej školy, žiaci ju teraz musia používať pre nich novým až nezvyčajným spôsobom.

Preto sme sa rozhodli vykonať kvalitatívnu a kvantitatívnu analýzu pracovných listov. Zamerali sme sa na programátorské informatické koncepty a ich vzťah k matematike.

2. OBSAHOVÁ ANALÝZA

Obsahovú analýzu pracovných listov sme rozčlenili do nasledujúcich častí:

- počet úloh = číslo, koľko úloh pracovný list obsahuje
- počet úloh s matematickým charakterom = číslo, koľko úloh vyžaduje prácu s matematickými objektmi alebo operáciami
- koncepty = zoznam nových programátroských konceptov, s ktorými sa žiak stretne (neznamená to, že ich musí zvládnuť naraz, ihneď a nazačiatku, ale zoznamuje sa s nimi postupne tak, ako rieši jednotlivé úlohy) v rámci pracovného listu

ISBN 978-80-557-1533-9, ISSN 2454-051X.

- syntaktické pravidlá = nové syntaktické pravidlá, zápisy alebo symboly, s ktorými sa žiak stretne
- zručnosti = nové zručnosti, ktoré sa vyskytnú pri riešení úloh – tie sa dajú rozdeliť na technické zručnosti (ako napríklad ovládanie počítača), alebo aj také, ktoré boli predtým považované za nový koncept, ale teraz sa trénujú na úrovni zručnosti (napríklad, kreslenie na papier za účelom rozmýšľania o riešení problému)

V niektorých pracovných listoch sa neobjavujú nové syntaktické pravidlá alebo zručnosti, takže tieto časti v daným pracovných listoch vo výsledkoch neuvádzame. Tiež neuvádzame triviálne zručnosti, ako je písanie na klávesnici alebo klikanie myšou.

Ak treba, v poznámke uvádzame dodatočné vysvetlenie.

Koncepty označené drážkou ■ majú matematický charakter. Podobne sme spočítali, koľko úloh má matematický charakter.

Slovné spojenie *„matematický charakter“* je nejasné. Na jednej strane môžeme o každom programe povedať, že má matematický charakter, lebo je to svojim spôsobom inak vyjadrená matematika, prípadne naopak, dá sa popísať aj matematickým aparátom.

Na druhej strane sa chceme vyhnúť tomu, aby bola každá úloha, v ktorej sa vyskytne číslo, automaticky označená za takú, ktorá má matematický charakter. Napríklad, pri cykloch sa uvádza počet opakovaní zápisom `range(5)`. To je pre riešenie niektorých úloh podstatné – vtedy úlohu započítame medzi úlohy s matematickým charakterom. Niektoré úlohy sú však také, že v nich treba vylepšiť predchádzajúce riešenie, ale žiak nemusí uvažovať ani meniť počet opakovaní. Musí skôr rozmýšľať o tom, čo program zopakuje, aké príkazy budú tvoriť telo cyklu.

Preto za úlohu, ktorá má matematický charakter, označujeme takú úlohu, pri riešení ktorej majú matematické objekty (čísla, súradnice, geometrické tvary, logické hodnoty, ...) podstatný význam (vykonávajú sa s nimi matematické operácie, zostavujú sa matematické výrazy, uvažuje sa o geometrických vlastnostiach, vzťahoch alebo logických hodnotách).

Podobne aj o programátorskom koncepte povieme, že má matematický charakter, ak sa popri informatických ideách vyžaduje alebo rozvíja i matematický pohľad na riešenie problémov.

2.1 Výrazy

Počet úloh: 16

Počet úloh s matematickým charakterom: 15

Koncepty:

- jednoduché čísla (nerozlišujeme celé ani desatinné)
- základné aritmetické operácie, zátvorky
- priorita operácií
- vyhodnocovanie výrazov
- počítač reaguje na nesprávne zápisy chybou

Syntaktické pravidlá:

- zápis čísel, operácií, používanie zátvoriek
- operátory `+ - * /`
- nepovinný operátor `**`

Zručnosti:

- čítanie a porozumenie chybovým hláseniam: `SyntaxError`, `ZeroDivisionError`

2.2 Premenné

Počet úloh: 17

Počet úloh s matematickým charakterom: 12

Koncepty:

- premenná – čo to je, ako funguje
- priradenie – ako sa správa
- použitie premennej vo výraze, vyhodnotenie výrazov s premennými
- zo slovných zadaní zostaviť matematický výraz

Syntaktické pravidlá:

- meno premennej
- priraďovací príkaz
- jednoduchý zápis desatinných čísel (napríklad `3.14`)
- nepovinné (len ak situácia nastane): nesprávne mená premenných (`for`, `def`, ...)

Zručnosti:

- kreslenie na papier pre znázornenie obsahu premenných

2.3 Program

Počet úloh: 13

Počet úloh s matematickým charakterom: 3

Koncepty:

- `print` - príkaz pre výpis textov a hodnôt
- program ako postupnosť príkazov – poradie je dôležité
- iný postup práce ako doposiaľ (líši sa od interaktívnej konzoly): zápis programu → vykonanie programu → výsledok
- texty sú iné ako čísla

Syntaktické pravidlá:

- príkaz `print` má parametre, ktoré sú uzavreté v zátvorkách a oddelené čiarkami
- texty sú uzavreté v apostrofoch
- možné chyby (len ak situácia nastane): ak v príkaze `print` zabudneme zátvorky (najmä ukončovaciu), čiarky, apostrof (pre ukončenie reťazca)

Zručnosti:

- vytvorenie nového programu (File ► NewFile)
- uloženie programu
- spustenie programu
- prepínanie sa medzi oknom editora a konzolou

2.4 Výpisy

Počet úloh: 11

Počet úloh s matematickým charakterom: 10

Koncepty:

ISBN 978-80-557-1533-9, ISSN 2454-051X.

- kombinácia výrazov s premennými a výpismi
- treba vymyslieť vzorce, ktorými sa formalizuje výpočet popísaný v prirodzenom jazyku

Poznámky:

- zatiaľ sa premenné používajú ako konštanty, ako vstupné hodnoty pre program, nechceme zavádzať `input()` a konverzie typov
- úlohy majú matematický charakter tak, aby svojou matematickou náročnosťou ostávali na úrovni základnej školy

2.5 Kreslenie

Počet úloh: 13

Počet úloh s matematickým charakterom: 11

Koncepty:

- grafická plocha
- príkaz na kreslenie obdĺžnika + význam parametrov
- počítačová súradnicová sústava (je iná, ako žiaci poznajú z matematiky)
- určovanie vzájomnej pozície a rozmerov obdĺžnikov:
  - stanovenie parametrov pre kreslenie (súradníc) podľa zadania – napríklad, aby sa geometrické tvary dotýkali, mali rovnaký stred
  - zistenie rozmerov obdĺžnika (šírka a výška) z parametrov príkazu a naopak, z pozície (ľavého horného rohu) a rozmerov treba dopočítať parametre pre príkaz na kreslenie obdĺžnika

Syntaktické pravidlá:

- príkaz `canvas.create_rectangle` a jeho parametre
- nepovinné: súradnice môžu byť aj výrazy s číslami

Zručnosti:

- skopírovať alebo opísať časť textu (programu), ktorá inicializujú grafickú plochu:

```
import tkinter
canvas = tkinter.Canvas()
canvas.pack()
```

- práca s 3 oknami – program (textový editor), grafický výstup, interaktívna konzola
- kreslenie na papier pre odvodenie súradníc

Poznámky:

- jednotlivé príkazy pre inicializáciu grafiky sa žiakom nevysvetľujú (ak si to sami nevyžiadajú)
- súradnice sa zatiaľ určujú iba pomocou jednoduchých výrazov s konštantami

2.6 Farby

Počet úloh: 10

Počet úloh s matematickým charakterom: 8

Koncepty:

- určovanie farby obdĺžnika
- treba správne zvoliť poradie kreslenia, ak sa majú obdĺžniky prekrývať

Syntaktické pravidlá:

- zápis parametera pre nastavenie farebnej výplne

Zručnosti:

- kreslenie si na papier pri uvažovaní o súradniciach
- zvolenie správneho názvu farby (text) z daného zoznamu farieb (pomôcka pre žiaka)

2.7 Kreslenie s premennými

Počet úloh: 11

Počet úloh s matematickým charakterom: 11

Koncepty:

- doposiaľ obsahovali parametre pre kreslenie obdĺžnikov iba výrazy s konštantami, teraz sa v nich budú vyskytovať aj premenné – program tak dokáže automaticky nakresliť správny obdĺžnik na rôzne miesta
- odvodenie súradníc vrcholov obdĺžnika, ak sú v premenných uložené súradnice jeho stredu a rozmerov
- vymýšľanie výrazov s premennými pre kreslenie rôznych obdĺžnikov tak, aby mali danú vzájomnú polohu (susedia, majú spoločný jeden vrchol, spoločný stred,…)

Zručnosti:

- kreslenie na papier pri uvažovaní a odvodzovaní výrazov

2.8 Podprogramy

Počet úloh: 10

Počet úloh s matematickým charakterom: 4

Koncepty:

- definícia podprogramu (ako hlavička s názvom a telom podprogramu s príkazmi)
- volanie podprogramu – až teraz sa vykonajú príkazy podprogramu
- dekompozícia s pomocou (je daný obrázok, jeho kreslenie treba rozdeliť na dané menšie časti, zadefinovať pre ne podprogramy a tie neskôr zavolať)
- odsadzovanie (doposiaľ nebolo potrebné) ako telo podprogramu

Syntaktické pravidlá:

- pre definíciu podprogramu + kam a ako zapisovať
- pre volanie podprogramu + kedy volať

Poznámky:

- podprogramy sú zatiaľ bez parametrov, bez premenných
- v úlohách s kreslením sa používa iba `canvas` ako globálna premenná (žiakom sa nevysvetľuje – zatiaľ nebolo treba)

2.9 Náhoda

Počet úloh: 18

ISBN 978-80-557-1533-9, ISSN 2454-051X.

Počet úloh s matematickým charakterom: 14

Koncepty:

- generátor náhodných čísel
- generovanie čísel z daného rozsahu, z danej množiny iba pomocou `randint(..., ...)`
- použitie premennej pre zapamätanie si vygenerovanej hodnoty:
  - pre jej neskoršie viacnásobné použitie (napríklad pre kreslenie štvorca náhodnej veľkosti)
  - zatiaľ fungovala premenná ako konštanta, teraz sa už nedá povedať, aká bude jej konkrétna hodnota
- použitie lokálnych premenná v podprograme (to zatiaľ nebolo potrebné)

Syntaktické pravidlá:

- zápis pre vygenerovanie náhodnej hodnoty

Zručnosti:

- napísať `import random`

Poznámky:

- žiaci používajú lokálne premenné podľa daných ukážok
- pojmy lokálna / globálna zatiaľ nevysvetľujeme – netreba to, keďže žiaci budú riešiť iba také úlohy, v ktorých by sa nemali vyskytnúť konflikty s premennými

2.10 Kreslenie textu

Počet úloh: 11

Počet úloh s matematickým charakterom: 10

Koncepty:

- doposiaľ sa texty vypisovali iba do konzoly pomocou príkazu `print` – do grafickej plochy sa dajú tiež vypisovať, ale iným spôsobom
- ako súvisia parametre príkazu `create_text` so súradnicami, kam sa text vypisuje

Syntaktické pravidlá:

- príkaz pre výpis textu a jeho parametre `create_text(x, y, text='...')`
- v poslednej úlohe je prezradený variant pre výpis číselnej hodnoty `create_text(x, y, text=číslo)`

2.11 Program s opakovaním

Počet úloh: 15

Počet úloh s matematickým charakterom: 10

Koncepty:

- opakovanie príkazu alebo skupiny príkazov pomocou cyklu + počet opakovaní
- rozpoznávanie častí, ktoré sa majú zopakovať od tých, ktoré sa opakovať nesmú
- zostavovanie vzorcov pre súradnice kreslenia veľkého počtu prvkov
- kombinovanie cyklu, premenných, náhodných čísel, grafických príkazov / príkazov `print`

Syntaktické pravidlá:

- zápis cyklus: `for i in range(n): …` *n* je počet opakovaní
- telo cyklu treba odsadiť
- nastavenie písma a jeho veľkosti

Zručnosti:

- odsadzovanie
- kreslenie si na papier konkrétnych tvarov s konkrétnymi číslami pri uvažovaní a odvodzovaní všeobecných vzorcov

Poznámky:

- iba úlohy s jednoduchým cyklom (nie je cyklus v cykle)
- stačí iba konštantný počet opakovaní
- stačí iba jeden cyklus
- zatiaľ sa nepoužíva riadiaca premenná cyklu
- v niektorých úlohách sa v tele cyklu môžu používať premenné (pre uloženie náhodného čísla), ale bez toho, aby sa využívali ich hodnoty z predchádzajúcej alebo v nasledujúcej iterácii (teda, nevyžaduje sa akumulácia, ani inkrementácia premenných)

2.12 Premenná cyklu

Počet úloh: 13

Počet úloh s matematickým charakterom: 13

Koncepty:

- automatická zmena hodnoty v premennej cyklu
- používanie premennej cyklu v príkazoch tela cyklu
- odvodenie závislosti nejakej inej hodnoty od premennej cyklu (t. j. zovšeobecňovanie)
- krokovanie / čítanie programu, v ktorom sa využíva premenná cyklu (ako sa menia hodnoty a výrazy)

Poznámky:

- stále iba jednoduchý cyklus (nie cyklus v cykle)
- stále iba konštantný počet opakovaní
- stále iba jeden cyklus

2.13 Výrazy v cykle

Počet úloh: 9

Počet úloh s matematickým charakterom: 9

Koncepty:

- vykonanie priradenia – vyhodnotenie výrazu v pravo, priradenie do premennej vľavo
- akumulácia výsledku + odvodenie závislosti

Zručnosti:

- kreslenie si na papier a znázorňovanie obsahu premenných
- vypĺňanie krokovacej tabuľky
- kreslenie si na papier pri uvažovaní a odvodzovaní vzorcov

ISBN 978-80-557-1533-9, ISSN 2454-051X.

2.14 Elipsy a kruhy

Počet úloh: 11

Počet úloh s matematickým charakterom: 8

Koncepty:

- kreslenie elíps a kruhov
- prvýkrát sa zavádza pojem parameter
- ako súvisí kreslenie elíps s kreslením obdĺžnikov (parametre v príkazoch `create_rectangle` a `create_oval`)
  - uvažovanie o niektorých jednoduchých vzájomných polohách kruhov, obdĺžnikov a textov + pre konkrétne rozmery treba určiť parametre (aj metódou pokus-omyl) tak, aby program približne nakreslil daný obrázok

Syntaktické pravidlá:

- príkaz `create_oval` a jeho parametre

Zručnosti:

- kreslenie si na papier pri uvažovaní a odvodzovaní vzorcov pre vzájomnú polohu

2.15 Kruhy a cykly

Počet úloh: 12

Počet úloh s matematickým charakterom: 8

Koncepty:

- odvodenie vzorcov pre niektoré jednoduché vzájomné polohy kruhov, obdĺžnikov a textov tak, aby mali kruhy neznámych rozmerov spoločné stredy, aby sa dva kruhy dotýkali na špeciálnom mieste (v horizontálnom alebo vertikálnom smere) a ďalšie
- náhodný výber prvkov z danej množiny (čísel, textov, znakov)

Syntaktické pravidlá:

- príkaz `random.choice` a jeho parametre (občas treba písať `[ ]`)

Zručnosti:

- kreslenie si na papier pri odvádzaní vzorcov so vzájomnou polohou kruhov

2.16 Vetvenie

Počet úloh: 13

Počet úloh s matematickým charakterom: 10

Koncepty:

- testy na porovnanie – iba jednoduché porovnania pomocou `<`, `>`
- pravdivostná hodnota
- vetvenie podľa podmienky + vykonanie/nevykonanie príkazov
- prepis situácie z prirodzeného jazyka do formálnej podoby a jazyka pre počítač

Syntaktické pravidlá:

- zápis príkazu `if` … `else` …
- testy na porovnanie `<`, `>`

Zručnosti:

- odsadzovanie

Poznámky:

- iba jednoduché testy (bez zložených podmienok), iba jeden príkaz v každej vetve
- iba úlohy, v ktorých stačí jeden `if` … `else` (zatiaľ nie bez vetvy `else`)

2.17 Vetvenie a konštrukcie

Počet úloh: 15

Počet úloh s matematickým charakterom: 13

Koncepty:

- výpočet a nastavenie premennej pomocou `if` … `else`
- kombinácia jedného cyklu s vnoreným príkazom vetvenia
- použite premennej cyklu pri teste a vetvení
- použitie vetvenia v podprograme
- objavovanie podmienok vetvenia a príkazov, ktoré sa vykonajú v jednotlivých vetách, podľa zadaného problému

Syntaktické pravidlá:

- viacnásobné odsadzovanie vnorenej konštrukcie
- dodržiavanie známych syntaktických pravidiel aj pri kombinovaní programových štruktúr

Poznámky:

- úlohy sú stále zostavené tak, aby sa dali riešiť bez vnorených cyklov alebo vetvení (kombinuje sa jeden cyklus s jedným vetvení)

2.18 Vnorené vetvenie

Počet úloh: 13

Počet úloh s matematickým charakterom: 13

Koncepty:

- použitie vetvenia v kombinácii s akumuláciou výsledku
- viacnásobné vetvenie podľa viacerých podmienok
- vnorené vetvenie (aj vo vetve `if`, aj vo vetve `else` je vnorené ďalšie vetvenie)
- rozpoznanie situácie, pre viacnásobné aj vnorené vetvenia

Syntaktické pravidlá:

- operácie `==`, `!=`

2.19 Podprogram s parametrom

Počet úloh: 18

Počet úloh s matematickým charakterom: 17

Koncepty:

- parameter podprogramu ako označenie neznámej hodnoty + všeobecnejšie zápisy riešenia
- volanie podprogramu s parametrom

ISBN 978-80-557-1533-9, ISSN 2454-051X.

Syntaktické pravidlá:

- definícia podprogramu s parametrom
- volanie podprogramu s parametrom

3. VÝSLEDKY A DISKUSIA

V tabuľke 1 uvádzame štatistické spracovanie obsahovej analýzy pracovných listov.

Tabuľka 1. Ú = počet úloh, z toho M = počet úloh s matematickým charakterom (M ≤ K); K = počet konceptov, z toho N = počet konceptov s matematickým charakterom (N ≤ K); S = počet syntaktických pravidiel; Z = počet zručností.

Názov pracovného listu	Ú	M	K	N	S	Z
1. Výrazy	16	15	5	4	3	1
2. Premenné	17	12	4	2	4	1
3. Program	13	3	4	0	4	1
4. Výpisy	11	10	2	2	0	0
5. Kreslenie	13	11	5	3	2	3
6. Farby	10	8	2	0	1	2
7. Kreslenie s premennými	11	11	3	3	0	1
8. Podprogramy	10	4	4	0	2	0
9. Náhoda	18	14	6	4	1	1
10. Kreslenie textu	11	10	2	1	2	0
11. Program s opakovaním	15	10	4	2	3	2
12. Premenná cyklu	13	13	4	4	0	0
13. Výrazy v cykle	9	9	2	2	0	3
14. Elipsy a kruhy	11	8	4	1	1	1
15. Kruhy a cykly	12	8	2	1	1	1
16. Vetvenie	13	10	4	2	2	1
17. Vetvenie a konštrukcie	15	13	5	3	2	0
18. Vnorené vetvenie	13	13	4	1	1	0
19. Podprogram s param...	18	17	2	1	2	0
Spolu	**249**	**199**	**68**	**36**	**31**	**18**

Pracovné listy obsahujú takmer 250 úloh, z nich má takmer 200 úloh matematický charakter, čo je približne 80% úloh.

Žiaci sa stretnú s 68 novými konceptami a 30 syntaktickými pravidlami. Ak tieto hodnoty porovnám s počtom úloh, vychádza nám hustota v pomere: na 1 nový koncept pripadá zhruba 3,5 úlohy a na 1 nové syntaktické pravidlo vyše 8 úloh. Veľmi zjednodušene sa dajú tieto čísla interpretovať tak, že jedno syntaktické pravidlo sa trénuje na 8 úlohách, prípadne, jeden koncept na 3 – 4 úlohách (teda, nie je to tak, že na jednej úlohe predvedieme 10 nových syntaktických pravidiel a ďalších 10 nových konceptov).

Vidíme aj, že matematický charakter má takmer 53% konceptov, s ktorými sa žiaci pri programovaní stretnú.

Je zrejmé, že jednotlivé koncepty a syntaktické pravidlá sa ťažko porovnávajú, keďže majú pre žiakov rôznu náročnosť. Napríklad, podprogramy s parametrom považujeme za výrazne náročnejší koncept ako výpis pozdravov alebo vizitiek pomocou príkazu `print`.

V našej analýze sme náročnosť nezohľadňovali, keďže ju zatiaľ nedokážeme exaktne odmerať.

Pri zostavovaní pracovných listov sme sa riadili našimi skúsenosťami. Preto sa napríklad téma cyklu objavuje prinajmenšom v štyroch pracovných listoch, a neustále trénuje.

Podobne je „rozriedená“ aj téma podprogramy. Dá vidieť, že

- koncept podprogramov otvárame v 8. pracovnom liste, podprogramy sa následne trénujú, a až 19. pracovnom liste sa prvýkrát definuje podprogram s parametrom
- nekoncentrujeme na jednom mieste veľa nových konceptov (v 8. liste sú štyri nové koncepty, ale žiaden z nich nie je matematický; v 19. liste iba dva, z ktorých je iba jeden matematický a súvisí s hodnotou parametra)

4. ZÁVER

Vidíme, že aj základné témy z programovania akými sú premenné, podprogramy, cykly a vetvenia nesú v sebe množstvo drobných konceptov, ideí. Tie musia žiaci postupne, krokov za krokom absorbovať a natrénovať, ak chceme, aby dokázali sami riešiť programátorské úlohy a problémy.

Našim zámerom nebolo vytvoriť pracovné listy, ktoré sú zamerané na matematiku. Snažili sme sa navrhnúť a zostaviť také úlohy, ktoré by boli pre žiakov zaujímavé. Matematický charakter však získali akosi automaticky.

Vysoký výskyt matematických úloh v našich pracovných listoch pripisujeme tomu, že matematika je neoddeliteľnou súčasťou programovania a od určitého okamihu sa jej nevieme vyhnúť.

Zdalo sa nám opodstatnené, aby sa žiaci postupne naučili používať im známy (= na úrovni základnej školy) matematický aparát aj v programovacom jazyku Python, aby sa naučili objavovať a popisovať matematické vzťahy, a aby si tento proces natrénovali. Aj preto, napríklad, žiaci spočiatku používajú Python ako múdru kalkulačku a trénujú si na ňom zápisy matematických výrazov alebo spoznávajú, ako počítač vykoná niektoré operácie. Tie nemusia byť rovnaké, ako žiaci poznajú z matematiky – niekedy býva Python „zlomyseľný“. Napríklad, ak sa žiak pomýli a namiesto `3.14 * r * r` napíše `3,14 * r * r` (správne má byť desatinná bodka, nie desatinná čiarka), zobrazí sa „divný“ výsledok.

Python pozná veľa operátorov (aj v závislosti na typoch), syntaktických pravidiel, funkcií a objektov. Pri vyučovaní začiatočníkov máme snahu minimalizovať ich počet. Preto sme sa obmedzili iba na operácie s číslami a pohľadu žiakov nerozlišujeme celé ani desatinné čísla. S textovými reťazcami pracujeme ako s konštantami. Je však možné, že sa žiaci stretnú s problémom: `10 * 10 ≠ "10" * 10`. V podobných situáciách by mal učiteľ vedieť správne zareagovať.

BIBLIOGRAFICKÉ ODKAZY

[1] Blaho, A., Salanci, Ľ., Šimandl, V. Základy programování v jazyce Python pro střední školy. https://imysleni.cz/ucebnice/zaklady-programovani-v-jazyce-python-pro-stredni-skoly

Informatika bez bariér
Accessibility of the Computer Science study

Mária Stankovičová
Centrum podpory študentov
so špecifickými potrebami
Univerzita Komenského
Šafárikovo nám. 6
814 99 Bratislava
Slovenská republika
Maria.Stankovicova@rec.uniba.sk

ABSTRACT
This paper deals with activities of the Support Centre for students with special needs at Comenius University and with possibilities of cooperation with Computer Science teachers at secondary schools. Students with visual impairment require special attention not only in technical-material support, but also in learning strategies. We want to share our experience gained from the qualitative research aimed at case study of the Computer Science students with visual impairment.

Keywords
Students with visual impairment. Assistive technology. Accessibility. Tactile graphics. Computer Science.

ABSTRAKT
V príspevku sa zameriavame na pôsobenie Centra podpory študentov so špecifickými potrebami na Univerzite Komenského a možnosti spolupráce s učiteľmi informatiky na stredných školách. Zrakové postihnutie študenta si vyžaduje osobitnú pozornosť nielen v technicko-materiálnom zabezpečení, ale aj v metodike a vzdelávacích postupoch. Chceme sa s vami podeliť o naše doterajšie skúsenosti, ktoré systematicky zbierame v rámci kvalitatívneho výskumu metódou prípadovej štúdie študentov informatiky so zrakovým postihnutím.

Kľúčové slová
Študenti so zrakovým postihnutím. Asistenčné technológie. Prístupnosť. Reliéfna grafika. Informatika.

1 ÚVOD
Centrum podpory študentov so špecifickými potrebami na Univerzite Komenského (CPŠ) poskytuje rôzne služby a poradenstvo pre študentov za účelom zlepšenia študijných podmienok a pre rozvoj ich študijných kompetencií. Zrakové postihnutie výrazne ovplyvňuje spôsob získavania informácií a vytváranie predstáv o okolitom svete. Technológie predstavujú nástroj, ktorý môže uľahčiť prístup k informáciám a príležitosť pre mladého človeka stať sa samostatným v štúdiu. Kľúčovými predmetmi počas štúdia informatiky sú matematika a programovanie, ktoré si vyžadujú určitú úroveň abstraktného myslenia. Veľkú časť informácií tvoria obrázky a grafické schémy, ktoré znázorňujú vzťahy medzi jednotlivými opisovanými prvkami a v niektorých prípadoch aj reálne priestorové súvislosti. Na mnohých kurzoch študenti používajú rôzny softvér, a teda otázka prístupnosti vybraného softvéru pre asistenčné technológie je pre nich aktuálna. Ak didaktická pomôcka sama predstavuje bariéru vo vzdelávaní, tak je naozaj nevyhnutné hľadať vyhovujúce alternatívne riešenia.

Hlbšiemu skúmaniu špecifických potrieb študentov informatiky so zrakovým postihnutím sa venujeme v rámci našej dizertačnej práce. Veľmi dôležitá je aj systematická príprava študenta počas strednej školy, preto sa o naše skúsenosti radi podelíme s učiteľmi, ktorí sa usilujú o zefektívnenie svojich vyučovacích metód v procese inkluzívneho vzdelávania. Zároveň radi privítame akékoľvek podnety zo strany učiteľov. Považujeme za dôležité poukázať na slabé a problémové miesta, no aj na úspešné riešenia.

2 CENTRUM PODPORY ŠTUDENTOV
CPŠ vzniklo pred 25 rokmi ako špecializované pracovisko so zameraním na podporu študentov so zrakovým postihnutím [1]. Vďaka tejto tradícii máme bohaté skúsenosti, ktoré môžeme sprostredkovať nielen súčasným študentom a uchádzačom o štúdium na vysokej škole, ale aj širokej odbornej aj laickej verejnosti, učiteľom vysokých, stredných a základných škôl, poradenským pracovníkom (výchovní poradcovia, pracovníci CPPPaP[1], CŠPP[2]) a študentom VŠ zaujímajúcich sa vo svojom odbore o problematiku zdravotného postihnutia (špeciálna pedagogika, sociálna práca, informatika, architektúra, a iné).

Našou snahou je odstraňovať informačné a fyzické bariéry, ktoré sa môžu vyskytnúť v akejkoľvek fáze vzdelávacieho procesu. Preto sa v našich aktivitách venujeme niekoľkým oblastiam:

- informačné, poradenské a konzultačné služby,
- akademická podpora počas štúdia,
- vzdelávacia a tréningová činnosť (práca s asistenčnými technológiami),
- technická podpora (spracovanie študijnej literatúry do prístupnej formy).

Náš záujem o spoluprácu so základnými a strednými školami vychádza z podnetov od žiakov s ťažkým zdravotným postihnutím (nielen) integrovaných do bežných škôl. Kontaktujú nás učitelia aj rodinní príslušníci. Dôvody sú rôzne. V prvom rade je to záujem, aby žiak bol v štúdiu úspešný a mohol pokračovať vo vzdelávaní na vysokej škole. Ďalšími dôvodmi sú vážne problémy, ktoré spôsobuje nedostatok alebo úplná absencia učebníc v prístupnej forme. Učitelia si uvedomujú nepostačujúce skúsenosti s uvedenou skupinou žiakov a slabú orientáciu v špeciálnych metodikách a technológiách. Obe strany – rodičia aj učitelia sa usilujú predovšetkým o kvalitnú prípravu žiaka, najmä ak ide o mladého človeka so študijnými ambíciami. Z našej strany

[1] CPPPaP – Centrum pedagogicko-psychologického poradenstva a prevencie

[2] CŠPP – Centrum špeciálno-pedagogického poradenstva

môžeme ponúknuť odborné konzultácie zamerané na konkrétny problém. Inú formu pomoci predstavujú jednorazové semináre, ktoré môžu byť tematicky širšie zamerané.

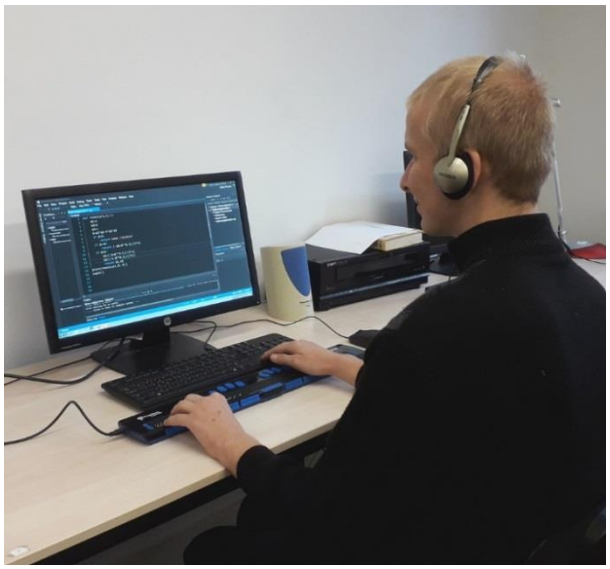
3 ASISTENČNÉ TECHNOLÓGIE PRE ŠTUDENTOV SO ZRAKOVÝM POSTIHNUTÍM

V širokej ponuke rôznych technologických zariadení majú významné postavenie tie, ktoré umožňujú kompenzovať dôsledky zrakového postihnutia. Súhrnne sa nazývajú asistenčné technológie (AT). [2]

3.1 AT pre nevidiacich študentov

Vo vzdelávacom procese je pre nevidiacich študentov vhodný počítač, v ktorom je nainštalovaný **čítač obrazovky**. K počítaču môže byť pripojený aj **braillovský riadok** (hmatový displej), ktorý zobrazuje text v Braillovom písme (Obrázok 1). Čítač obrazovky je program, ktorý cez hlasový výstup poskytuje študentovi textové informácie zobrazené na obrazovke počítača (názvy programov, text v dialógových oknách, text v dokumentoch). O obrázkoch vie poskytnúť informáciu iba z priradeného alternatívneho textu.

Všetky bežné úkony je možné na počítači vykonať pomocou klávesnice (spustenie aplikácie, posun po komponentoch dialógového okna, programovej ponuky). Čítač obrazovky dokáže simulovať niektoré udalosti myši na komponentoch, na ktoré je možné dostať sa pomocou klávesnice (napríklad tabulátorom). Ak aplikácia neumožňuje ovládanie pomocou klávesnice, tak je pre nevidiaceho študenta prakticky nepoužiteľná. Z metodického hľadiska je veľmi dôležité vyberať taký softvér, ktorý je pre čítač obrazovky prístupný, aby v ňom nevidiaci študent mohol pracovať.



Obrázok 1: Študent kontroluje správnu syntax na braillovskom riadku.

3.2 AT pre slabozrakých študentov

Pojem slabozrakosť vyjadruje široké spektrum zrakového postihnutia, takže skupina slabozrakých študentov je veľmi heterogénna. Vo vzdelávacom procese môže slabozraký študent používať počítač so **zväčšovačom obrazovky**. Je to program, ktorý umožňuje upraviť veľkosť písma, prispôsobiť farebnú schému a kontrast komplexne pre všetko, čo je zobrazené na monitore. Z metodického hľadiska je potrebné overiť u vybraného softvéru, ako sa zmení jeho prostredie a funkčnosť po spustení zväčšovača obrazovky. Problém v čitateľnosti môžu spôsobiť málo kontrastné farby, malé komponenty a pätkový font písma.

Uvedomujeme si, že výber univerzálneho softvéru nie je jednoduchá požiadavka a v praxi je učiteľ nútený hľadať rôzne alternatívy. Poznanie možností a obmedzení asistenčných technológií pomáha pedagógovi pri výbere vhodného edukačného softvéru. Tiež mu umožňuje odhadnúť adekvátne požiadavky na študenta, prípadne ho vhodne usmerniť. Študent na seba preberá zodpovednosť naučiť sa vybranú technológiu používať na takej úrovni, aby bola prínosom pre jeho štúdium.

4 PRÍSTUPNOSŤ ŠTUDIJNÝCH MATERIÁLOV

4.1 Texty

Asistenčné technológie sú spojené s používaním počítača. Preto je prirodzené, aby študent mal učebné texty k dispozícii v elektronickom formáte (napríklad txt, rtf, docx, html, pdf). Každý z uvedených formátov má v rôznych situáciách svoje výhody, napríklad možnosť editovať, vyhľadávať, orientovať sa v štruktúre dokumentu. Aby študent so zrakovým postihnutím mohol s pomocou asistenčných technológií efektívne pracovať so študijným textom, je pri tvorbe dokumentu potrebné dodržať niekoľko jednoduchých pravidiel [3]. Spomeňme aspoň základné:

- vytvoriť štruktúru dokumentu – pre jednotlivé časti použiť nadpisy s priradeným štýlom pre nadpis,
- pre text použiť bezpätkové písmo s jednoduchými nesplývajúcimi líniami,
- tabuľky navrhnúť tak, aby neobsahovali rôzne vnorené a zlúčené bunky,
- k obrázkom priradiť alternatívny text alebo opis, ak obrázok vyjadruje dôležitú informáciu, je potrebné zmieniť sa o tom aj v texte.

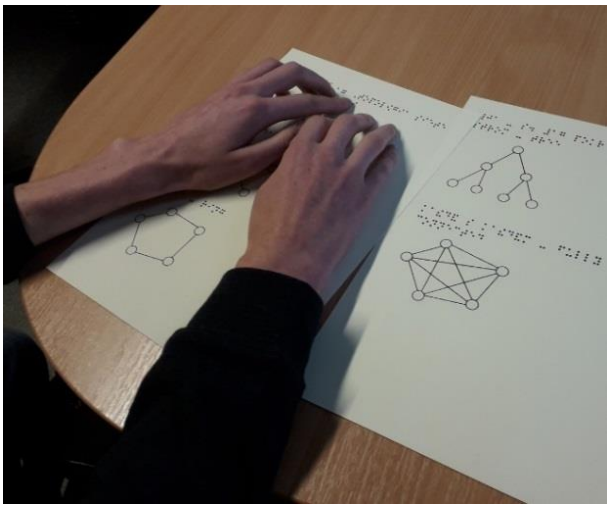
Jednou zo služieb CPŠ je spracovanie tlačenej literatúry do elektronickej formy. Konkrétne zásady a postupy uvádzame v príručke, ktorú zhotovili zamestnanci CPŠ. [4] Špecifickú pozornosť si vyžadujú učebné texty s matematickou symbolikou. Špecifickosť spočíva v tom, že častokrát dvojrozmerný zápis ako napríklad zlomky je potrebné linearizovať. [5] Alebo pre rôzne symboly ako napríklad znak odmocniny, je potrebné použiť alternatívny zápis. Do určitej miery je linearizácia matematického zápisu realizovaná v programovacích jazykoch. Komplexnejšie riešenie ponúka typografický systém TEX, no realizovať v ňom praktické výpočty je viac ako ťažkopádne. Preto sa zameriavame aj na oblasť sprístupňovania matematiky študentom so zrakovým postihnutím. [6]

4.2 Obrázky

Vo výchovno-vzdelávacom procese učiteľ potrebuje vziať do úvahy absenciu, neúplnosť, prípadne skreslenie zrakových vnemov u študenta so zrakovým postihnutím [7]. To znamená, že pri výklade nového učiva alebo preverovaní vedomostí by učiteľ nemal predpokladať vizuálnu skúsenosť porovnateľnú so zrakovým vnímaním u intaktných študentov. Otázka teda znie, ako treba upraviť obrázok, aby mal výpovednú hodnotu pre študenta so zrakovým postihnutím? Učiteľ sa potrebuje zamerať

na komplexné využitie zmyslov študenta, čiže hľadá spôsoby, ako je možné transformovať informáciu z obrázka do textu, zvuku alebo hmatateľnej formy – **reliéfnej grafiky**.

Obrázok upravený pre nevidiaceho študenta vo forme reliéfnej grafiky môže obsahovať body, línie aj plochy, no veľkosť a rozmiestnenie útvarov musia byť také, aby sa dali rozoznať hmatom. Na dobré rozlíšenie susedných plôch môžeme použiť rozdielne textúry. Pre prípravu reliéfnej grafiky môžeme použiť rôzne technológie. Medzi najrozšírenejšie patria **braillovské tlačiarne** na reliéfnu grafiku. Návrh obrázku môže učiteľ urobiť v ľubovoľnom grafickom editore. Najvýhodnejší je editor na tvorbu vektorových obrázkov, lebo tie môže učiteľ v prípade potreby neskôr upraviť. Ku každej braillovskej tlačiarni je k dispozícii obslužný softvér, v ktorom sa línie a plochy „vybodkujú“. To znamená, že po vytlačení sú tie miesta na papieri vyplnené vypuklými bodmi. Na niektorých typoch braillovských tlačiarní je možné kombinovať s bodmi aj atramentovú alebo laserovú tlač. Ďalšou technológiou na tvorbu reliéfnej tlače je zahriatie vzdúvacieho (bublinkového) papiera[3]. Predlohu tiež môžeme pripraviť v grafickom editore a vytlačiť na atramentovej alebo laserovej tlačiarni. Následne vložíme papier do ohrievača, kde sa všetky potlačené miesta vydujú (Obrázok 2).



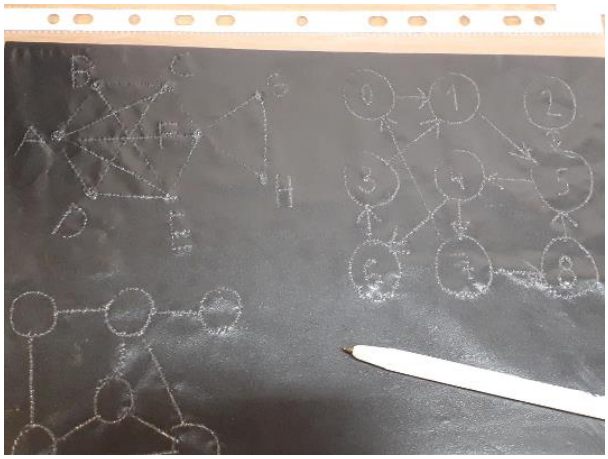
Obrázok 2: Študent číta grafy z obrázkov v reliéfnej forme.

V prípade, že učiteľ nemá možnosť používať braillovskú tlačiareň, tak najjednoduchší spôsob výroby reliéfnej grafiky je vyrývanie do fólie položenej na mäkkej podložke. Stačí, keď si zaobstará nepíšuce pero, polypropylénovú fóliu (napr. euroobaly) a vhodnú gumenú podložku (Obrázok 3). Výhodou je relatívne krátka doba prípravy a nízke náklady.

Pri úprave obrázku pre slabozrakého študenta môžeme použiť jednoduché línie a plochy odlíšené kontrastnými farbami. Veľkosť útvarov a textu prekonzultujeme so študentom a prispôsobíme podľa jeho individuálnych potrieb.

V každom prípade si treba uvedomiť, že študent so zrakovým postihnutím nevidí obrázok komplexne hneď na „prvý pohľad“. Učiteľ by mal so študentom obrázok prečítať a upozorniť ho na časti a súvislosti, ktoré sú dôležité a ktoré chce, aby študent pochopil. [8]

[3] Voľný preklad z anglického originálu „swell paper“.



Obrázok 3: Reliéfna grafika na euroobale.

5 Spolupráca so školami

Študenti so zrakovým postihnutím potrebujú byť dobre pripravení na štúdium na univerzite. Uvedomujeme si dôležitosť kvalitného vzdelávania na základných a stredných školách, a preto sa zúčastňujeme aj na projektoch zameraných na podporu výučby informatiky na týchto stupňoch vzdelávania.

5.1 Učebné materiály

V ostatných rokoch sme sa v spolupráci s KZVI[4] na FMFI UK[5] a Spojenou školou internátnou pre žiakov so zrakovým postihnutím podieľali na projekte KEGA č. 014UK-4/2016 *Rozvoj kompetencií nevidiacich žiakov ZŠ potrebných pre život v 21. storočí*. V rámci projektu sme sa venovali návrhu a tvorbe učebných materiálov v predmete informatika pre nevidiacich žiakov základných škôl [9]. V návrhoch a spracovaní učebných materiálov sme vychádzali z metodických usmernení pre vzdelávanie žiakov so zrakovým postihnutím. Z dlhodobého hľadiska si myslíme, že dobre položený základ v rozvoji informatických kompetencií môže neskôr mať výrazný vplyv na výber odboru vysokoškolského štúdia. Študijné materiály pokrývajú nasledovné tematické oblasti informatiky: Práca s textom, Práca so zvukom, Programovanie a riešenie problémov, Práca s matematickým textom, Práca s internetom. Súčasťou študijných materiálov sú aj reliéfne obrázky, ktoré nesú informáciu o priestorovom usporiadaní komponentov na obrazovke podľa danej témy. U nevidiacich žiakov je veľmi dôležité rozvíjať gramotnosť v čítaní reliéfnej grafiky, posilňuje to tvorbu mentálnych schém, pri ktorých sa spájajú teoretické vedomosti s hmatovými skúsenosťami. To predstavuje hodnotnú vedomostnú kompetenciu pre pochopenie znázornených priestorových aj abstraktných vzťahov.

5.2 Sústredenie pre stredoškolákov

Každoročne naše pracovisko organizuje *Sústredenie pre stredoškolákov s ťažkým zdravotným postihnutím*. Je určené pre študentov, ktorí majú seriózny záujem o štúdium na vysokej

[4] KZVI – Katedra základov vyučovania informatiky, od januára 2019 KDMFI – Katedra didaktiky matematiky, fyziky a informatiky

[5] FMFI UK – Fakulta matematiky, fyziky a informatiky Univerzita Komenského

škole. Pre záujemcov o štúdium informatiky sa snažíme sprostredkovať stretnutie s úspešnými absolventmi a tiež aj aktuálnymi študentmi. Za dôležitú považujeme aj komunikáciu s učiteľmi, ktorí študentov pripravujú počas strednej školy.

6 VÝSKUMNÝ PROJEKT

6.1 Motivácia a cieľ

Špecifické potreby študentov so zrakovým postihnutím, ktoré našou činnosťou napĺňame, sa viac či menej medzi jednotlivými študijnými odbormi nelíšia. No odbory technického zamerania ako informatika kladú väčšie nároky na hľadanie spôsobov sprístupnenia študijných textov a používaného softvéru. Zo strany študentov so zrakovým postihnutím je o štúdium informatiky záujem. Preto sme sa rozhodli v rámci nášho dizertačného výskumu venovať hlbšiu pozornosť špecifickým potrebám študentov informatiky so zrakovým postihnutím. Za cieľ si kladieme porozumenie jedinečného spôsobu štúdia.

6.2 Prehľad literatúry

Oblasti vzdelávania študentov so zrakovým postihnutím na úrovni vysokoškolského štúdia v odbore informatika[6] je v zahraničnej literatúre venovaná nemalá pozornosť. Neobchádzame domácu scénu, no zatiaľ sa nám nepodarilo nájsť štúdie relevantné vybranej oblasti. Budeme vďační za akékoľvek podnety.

Veľká pozornosť je venovaná prieskumu prístupnosti jednotlivých nástrojov v programovacích prostrediach pre asistenčné technológie [10, 11]. Nosnou myšlienkou je vízia, že zrakový deficit by nemal byť nevýhodou pri používaní akéhokoľvek vývojárskeho prostredia. Prieskum medzi nevidiacimi programátormi z praxe poukázal na zaujímavé spôsoby, ako si poradia s neprístupnosťou pomocných nástrojov. Napríklad počas ladenia programu môžu byť kontrolné výpisy vo vývojárskom prostredí neprístupné pre asistenčnú technológiu. Neprístupnosť konkrétnych pomocných nástrojov núti programátora so zrakovým postihnutím hľadať alternatívne riešenia, ktoré nemusia byť efektívne a ich realizácia môže zaberať výrazne dlhší čas (napríklad výpisy cez dialógové okná počas behu programu). Pre profesionála sa jeho nájdené alternatívy stávajú rutinou, no pre študenta, ktorý sa len učí programovať, nie je hľadanie alternatív jednoduché. Situáciu komplikuje aj vedomie, že jeho intaktní spolužiaci bezpečne používajú pomocné nástroje vývojárskeho prostredia a napredujú, zatiaľ čo on stagnuje, respektíve stráca čas riešením problémov, ktoré by nemusel mať.

Štúdie podobného charakteru, ako sme uviedli, sa stali inšpiráciou pre ďalšiu výskumnú oblasť, a to návrh a implementáciu prístupných pomocných nástrojov pre už existujúce vývojárske prostredia [12, 13]. V tomto kontexte uvedieme dva zaujímavé projekty. Prvým je vytvorenie a testovanie doplnkového modulu do vývojárskeho prostredia Eclipse s názvom StructJumper. [14] Myšlienka vytvoriť tento doplnok vychádza z úsilia kompenzovať nevidiacim programátorom deficit možnosti „prezrieť si“ naraz celý kód. V mnohých vývojárskych prostrediach je pre lepšiu orientáciu použité farebné odlíšenie syntaxe alebo častí kódu, no tieto odlíšenia čítač obrazovky nedokáže interpretovať. Vo výskume sa ďalej autori zaoberali vplyvu tohto doplnku na efektivitu vyhľadávania v kóde. Zúčastnení programátori so zrakovým postihnutím plnili úlohy s a bez použitia doplnku. Doplnok sa veľmi osvedčil pri vyhľadávaní v cudzom kóde. Študenti tvoria v tímoch a častokrát sa stane, že dostanú od spolužiaka kód, do ktorého majú niečo doplniť. Čím rýchlejšie sa dokážu v kóde zorientovať, tým je ich práca efektívnejšia.

Ďalším zaujímavým projektom je vytvorenie a testovanie doplnkového modulu do vývojárskeho prostredia Visual Studio s názvom CodeTalk. Vývojársky tím Microsoftu identifikoval problémy s prístupnosťou, navrhol riešenia, ktoré implementoval. Doplnok je súčasťou prostredia od verzie 2015/2017. [15]

Pre nás majú výsledky týchto štúdií dôležitý význam z hľadiska odstraňovania bariér. Niekto by možno mohol namietať, že toto všetko sú čisto technické záležitosti. No myslíme si, že nesprávne zvolené pracovné nástroje výrazne ovplyvnia aj študijné výsledky. Ak študent míňa čas a energiu na „zápas“ s neprimeranými podmienkami, uniká mu to podstatné – venovať sa štúdiu.

V neposlednom rade nás zaujíma aj oblasť didaktiky informatiky pre vysokoškolský stupeň vzdelávania. Z tohto pohľadu nás zaujali štúdie [16, 17] venované procesu poznávania v programovaní. Na základe teórie kognitívneho vývinu podľa J.Piageta autori opísali charakteristické znaky vývinových štádií v procese učenia sa programovať (senzomotorické, predoperačné, konkrétnych operácií, formálnych operácií). Podľa autorov je pre pedagóga veľmi dôležité vedieť rozpoznať, v ktorom štádiu sa svojím spôsobom myslenia študent nachádza.

6.3 Výskumná stratégia a metódy zberu dát

Rozhodli sme sa použiť stratégiu kvalitatívneho výskumu. Ako najvhodnejšiu výskumnú stratégiu považujeme prípadovú štúdiu. Pre zhromažďovanie relevantných údajov použijeme rozhovory a priame pozorovania účastníkov. [18]

Rozhovory chceme zamerať na prístupnosť učebných materiálov a softvéru, možnosti a limity používaných asistenčných technológií a vplyv zrakového postihnutia na vznik prípadných miskoncepcií pri osvojovaní učiva. Za dôležité považujeme aj rozhovory s učiteľmi, ktorí vedú povinné kurzy. Otázky zameriame na metodológiu, využívanie ponúkanej podpory zo strany CPŠ, nevyhnutné zmeny v metodike vyučovania, ktoré si vyžiadala prítomnosť študentov so zrakovým postihnutím.

Priame pozorovania budú prebiehať v čase cvičení alebo kurzov. Po pozorovaniach účastníci vyplnia štruktúrované dotazníky pre zvýšenie spoľahlivosti záznamov získaných z pozorovania.

6.4 Výskumná vzorka

Plánujeme požiadať súčasných študentov a ich učiteľov, ktorí vedú povinné kurzy o účasť na výskume. V tomto akademickom roku študujú odbor aplikovaná informatika dvaja slabozrakí študenti a jeden nevidiaci študent. Rôznorodosť ich zrakového postihnutia má odlišný vplyv na spôsob štúdia, požiadavky na spracovanie literatúry a používaný softvér. Okrem toho používajú rôzne asistenčné technológie.

Nevidiaci študent "A" vie čítať Braillovo písmo a tiež reliéfnu grafiku, používa počítač s čítačom obrazovky a braillovským riadkom. Študent nemá vizuálnu predstavivosť a domnievame sa, že to môže mať vplyv na pochopenie úloh s priestorovými vzťahmi.

Slabozrakosť u dvoch spomínaných študentov sa prejavuje odlišne. Prvý slabozraký študent "B" vie čítať Braillovo písmo a

[6] V zahraničnej literatúre sa v súvislosti so študijným odborom označenie „Informatika“ nevyskytuje, skôr sa uvádzajú termíny ako Computer Science, Software Developing, Programming.

používa počítač spolu so zväčšovacím softvérom, no má skúsenosti aj s čítačom obrazovky a braillovským riadkom. Študent vie narábať s vizuálnymi predstavami, ale nepoužíva vlastný rukopis.

Druhý zo slabozrakých študentov "C" nevie čítať Braillovo písmo a používa počítač len so zväčšovacím softvérom. Študent vie narábať s vizuálnymi predstavami a vie používať vlastný rukopis pri písaní a pri tvorbe náčrtov, ktoré nevyžadujú presnosť. Je zrejmé, že pre všetkých troch študentov je potrebná úprava úloh, ktoré vyžadujú presné kreslenie.

So študentmi sa pravidelne stretávame na konzultáciách. Počas týchto stretnutí sa rozprávame o priebehu vyučovania, požiadavkách na spracovanie študijných materiálov a používanom softvéri. Pravidelne sa stretávame aj s vyučujúcimi, s ktorými riešime technické zabezpečenie a tiež aj potrebné zmeny v metodike, ktoré si vyžaduje zrakové postihnutie študentov.

7 Záver

V našom príspevku sme priblížili činnosť Centra podpory študentov vo vzťahu k študentom so zrakovým postihnutím. Poskytli sme stručný prehľad špecifických potrieb, pričom sme sa zamerali na spôsob práce študentov s asistenčnými technológiami. Uviedli sme našu spoluprácu s učiteľmi informatiky a participáciu na projektoch zameraných na vyučovanie informatiky žiakov so zrakovým postihnutím. Predstavili sme zámer nášho výskumného projektu a načrtli najbližšie plány.

BIBLIOGRAFICKÉ ODKAZY

[1] MORAVČÍKOVÁ, Ľ. Support Centre for Visually impaired Students. In: Zagler W.L., Busby G., Wagner R.R. (eds) Computers for Handicapped Persons. ICCHP 1994. Lecture Notes in Computer Science, vol 860. Springer, Berlin, Heidelberg

[2] LECKÝ, P. Kompenzačné pomôcky a asistenčné technológie pre študentov so zdravotným postihnutím. Informačný materiál, UK Bratislava 2011, http://cezap.sk/publikacie/kompenzacne-pomocky-a-asistencne-technologie-pre-studentov-sozdravotnym-postihnutim/

[3] MAMOJKA, B., REGEC, V., TEPLICKÝ, P. (2006): Metodická príručka na vytváranie a používanie prístupných elektronických dokumentov, prístupná na internete, UNSS 2006, https://unss.sk/publikacie/metodicka_prirucka.rtf

[4] HANÁČIKOVÁ, L., LECKÝ, P. Úprava textov do prístupnej formy pre zrakovo postihnutých používateľov, CPŠ UK, 2012, https://www.minedu.sk/data/att/1909.pdf

[5] STANKOVIČOVÁ, M., PECNOVÁ, M. Praktické využitie matematického editora LAMBDA na vyučovaní matematiky pri integrácii nevidiacich žiakov na strednej škole; In: 48. konferencia slovenských matematikov 2016; EDIS, Žilina, s. 38. ISBN 978-80-554-1279-5.

[6] STANKOVIČOVÁ, M., MENDELOVÁ, E., Matematika bez bariér. In: Dva dni s didaktikou matematiky 2018, Zborník príspevkov 1. vyd. Bratislava : Fakulta matematiky, fyziky a informatiky UK, 2018 S. 93-98, ISBN 978–80–8147–087–5, https://www.ddm.fmph.uniba.sk/files/DvaDni/Zbornik2018.pdf

[7] Štátny pedagogický ústav Vzdelávací program pre deti a žiakov so zrakovým postihnutím 2009, http://www.statpedu.sk/files/sk/svp/statny-vzdelavaci-program/vp-deti-ziakov-so-zdravotnym-znevyhodnenim/vp-deti-ziakov-so-zrakovym-postihnutim/vp_zp-isced_1_2_3_5.pdf

[8] POLLY, K. EDMAN Tactile Graphics, American Foundation for the Blind, New York, 1992, ISBN 0-89128-194-0

[9] JAŠKOVÁ Ľ., STANKOVIČOVÁ M. Učebné materiály pre výučbu informatiky nevidiacich žiakov základnej školy, In: DidInfo and DidactIG 2017 Banská Bystrica: Univerzita Mateja Bela, 2017 S. 77-82

[10] ALBUSAYS, K., LUDI, S., HUENERFAUTH M. Interviews and Observation of Blind Software Developers at Work to Understand Code Navigation Challenges. In Proceedings of the 19th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '17). ACM, New York, NY, USA, 91-100. DOI: https://doi.org/10.1145/3132525.3132550

[11] MEALIN, S., MURPHY-HILL, E. An exploratory study of blind software developers. Visual Languages and Human-Centric Computing (VL/HCC), 2012 IEEE Symposium on. IEEE, (September 2012), 71-74.

[12] FRANCONI, J., M., SMITH, A., C., Computer science accessibility for students with visual disabilities. SIGCSE Bull. 34, 1 (February 2002), 91-95. DOI: https://doi.org/10.1145/563517.563372

[13] SIEGFRIED, R., M. Visual programming and the blind: the challenge and the opportunity, ACM Proceedings of the 37th SIGCSE technical symposium on Computer science education, Houston, Texas, USA, 2006, pp. 275-278.

[14] BAKER, C., M., MILNE, LADNER, R., E. StructJumper: A Tool to Help Blind Programmers Navigate and Understand the Structure of Code. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15). ACM, New York, NY, USA, 3043-3052. DOI: https://doi.org/10.1145/2702123.2702589

[15] POTLURI, V., VAITHILINGAM, P., IYENGAR, S., VIDYA, Y., SWAMINATHAN, M., SRINIVASA, G.,. CodeTalk: Improving Programming Environment Accessibility for Visually Impaired Developers. In Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18). ACM, New York, NY, USA, Paper 618, 11 pages. DOI: https://doi.org/10.1145/3173574.3174192

[16] LISTER, R. Toward a Developmental Epistemology of Computer Programming, WiPSCE '16 Proceedings of the 11th Workshop in Primary and Secondary Computing Education, ACM, Münster, Germany

[17] TEAGUE, D., LISTER, R. Blinded by their Plight: Tracing and the Preoperational Programmer, PPIG 2014 25th Workshop, Psychology of Programming Interest Group Annual Conference, Darwin College, Brighton, United Kingdom, pp. 53-64.

[18] YIN, R., K. Case Study Research and Aplications: Design and Methods, Los Angeles 2018: Sage Publications, Inc., 6th edition, ISBN 978-1-5063-3616-9

IKT vo vyučovaní na prvom stupni základnej školy
ICT in first degree of primary school

Veronika Stoffová
Katedra matematiky a informatiky
Pedagogická fakulta Trnavská Univerzita v Trnave
Priemyselná 4
91843 Trnava
Slovenská republika
nikastoffova@seznam.cz

Veronika Gabaľová
Katedra matematiky a informatiky
Pedagogická fakulta Trnavská Univerzita v Trnave
Priemyselná 4
91843 Trnava
Slovenská republika
veronika.gabalova@gmail.com

ABSTRACT
The article discusses the possibilities of using information and communication technologies at the first level of primary school. It briefly informs about the representation of ICT subjects in the Primary school teacher training program at the Faculty of Education of the Trnava University in Trnava. Based on the content of the subjects and the results of their teaching, it assess the readiness of future primary school teachers to use modern teaching and learning technologies in education. Particular attention is paid to the object of natural sciences. The possibilities of using ICT and didactic software are demonstrated on the subject of mathematics. Within the framework of teaching mathematics in the fourth grade at the a primary school in Trnava district, a pedagogical experiment was carried out focusing on the use of available didactic applications and tools for the acquisition of new knowledge: for its presentation, understanding, practice and fixation.

Keywords
ICT; First stage of primary school; ICT in primary education; Digital educational technologies; Educational software.

ABSTRAKT
Článok hovorí o možnostiach využitia informačno-komunikačných technológií na prvom stupni základnej školy. Stručne informuje o zastúpení predmetov IKT v študijnom programe pre učiteľov základnej školy na Pedagogickej fakulte Trnavskej univerzity v Trnave. Na základe obsahu predmetov a výsledkov ich vyučovania hodnotí pripravenosť budúcich učiteľov prvého stupňa ZŠ na využívanie moderných didaktických prostriedkov a vzdelávacích technológií vo vyučovaní. Zvláštna pozornosť je venovaná prírodovedným predmetom. Možnosti využívania IKT a didaktického softvéru sú demonštrované na predmete matematika. V rámci vyučovania matematiky vo štvrtom ročníku na základnej škole v okrese Trnava bol realizovaný pedagogický experiment orientovaný na využívanie dostupných didaktických aplikácií a dostupnej didaktickej techniky na osvojovanie nových poznatkov: na ich prezentovanie, pochopenie, precvičovanie a fixáciu.

Kľúčové slová
IKT; Prvý stupeň základnej školy; IKT v primárnom vzdelávaní; Digitálne vzdelávacie technológie; Edukačný softvér.

1 ÚVOD
Problematika využívania IKT je vo vyučovaní na prvom stupni základnej školy v súčasnosti mimoriadne aktuálna, nakoľko sa v dnešnej dobe s IKT ako súčasťou predškolskej prípravy stretávajú už deti v materskej škole, teda ešte predtým ako sú zaradené do povinného vzdelávacieho systému.
Deti narodené po roku 2010 sú niekedy označované ako deti digitálnej generácie. Tieto deti sú špecifickou skupinou s jedinečným postavením prvej generácie, ktorá sa narodila už do sveta, kde všade sú prítomné počítače s prístupom na Internet s neobmedzenými službami a s možnosťou využívania rozsiahlych informačných systémov [1, 2]. Deti sú prakticky od narodenia prostredníctvom rodičov a samotného prostredia priamo, prípadne sprostredkovane uvádzané do digitálneho sveta. Veková skupina súčasných študentov vysokých škôl sa narodila v druhej polovici 90. rokov minulého storočia, kedy sa na Slovensku začalo využívanie Internetu v školách. Do študijných programov študentov učiteľstva na všetkých fakultách sú zaradené predmety, ktoré budúcich učiteľov pripravujú na zvládnutie moderných vzdelávacích technológií, ukazujú im a usmerňujú ich ako využívať IKT v budúcej pedagogickej praxi.

2 ZASTÚPENIE PREDMETOV V PRÍPRAVE BUDÚCICH UČITEĽOV PREDŠKOLSKEJ A ELEMENTÁRNEJ PEDAGOGIKY
V študijnom programe na prípravu študentov v odbore predškolská a elementárna pedagogika na PdF TU v Trnave sa nachádza niekoľko predmetov, ktoré sú zamerané na získanie teoretických poznatkov a praktických zručností, potrebných na úspešnú integráciu IKT do vzdelávania. Ide konkrétne o predmety IKT vo vzdelávaní a informačné technológie pre učiteľskú profesiu. Tieto predmety pripravujú budúcich učiteľov aj na to, aby vedeli zhodnotiť použitie jednotlivých technologických nástrojov pre zvolený tematický celok so stanovenými vzdelávacími cieľmi, ktoré treba dosiahnuť. Dôležité je tiež, aby vedeli vybrať vhodné nástroje a pripraviť aj vlastné aplikácie s implementáciou IKT do vzdelávania na prvom stupni základnej školy. Zároveň sa v týchto predmetoch vytvára priestor, aby sa študenti dozvedeli o aktuálnych trendoch v oblasti vzdelávacích technológií a tak mohli doplniť a rozšíriť svoje poznatky v modernej pedagogike. Vyučovanie vyššie vymenovaných a podobných predmetov pre všetky učiteľské študijné programy na PdF TU v Trnave zabezpečujú učitelia informatiky z Katedry matematiky a informatiky. Je celý rad aj ďalších špeciálnych predmetov, ktoré sa ponúkajú študentom učiteľstva v kategórii B a C. Takýmto predmetom je napr. „ICT in language teaching“ pre študentov Katedry anglického jazyka a literatúry. Počas absolvovania uvedených predmetov budúci učitelia získajú nielen potrebné teoretické poznatky a praktické skúsenosti z využívania hardvérových a softvérových prostriedkov, ale získajú aj

dostatočné zručnosti z tvorby kompletných didaktických aplikácií. Zručne vytvárajú interaktívne prezentácie v prostredí PowerPoint (príp. Prezi), aplikácie na testovanie, nacvičovanie a fixovanie poznatkov v Hot Potatoes, Easy Test Maker, Moodle/Quiz a pod. Dokážu takto vytvorené moduly spojiť do jedného celku a zabezpečiť ich kompatibilitu. Odvážnejší z nich sa púšťajú aj do tvorby edukačných webových stránok. Podobne je to aj s vytváraním aplikácií pre interaktívnu tabuľu [3]. Na tvorbu aplikácií používajú štandardný, výrobcom dodávaný softvér, alebo voľne dostupnú verzie platforiem na vytváranie prezentácií s bohatou knižnicou interaktívnych prvkov. To, že budúci učitelia primárneho vzdelávania sú presvedčení o užitočnosti moderných vzdelávacích technológií, a že majú záujem o ich využívanie v budúcej pedagogickej praxi svedčí aj tá skutočnosť, že z roka na rok stále viac z nich má záujem o diplomovú či bakalársku prácu z danej oblasti. Praktická časť záverečných prác je orientovaná na tvorbu vlastných didaktických aplikácií vo vybratých predmetoch, ktoré sa vyučujú na prvom stupni základnej školy. Aj keď medzi predmetmi, na ktoré sa orientujú, dominuje matematika a exaktné vedy [4, 5, 6, 7], často nás študenti prekvapia interaktívnymi, obsahovo a formálne dobre štruktúrovanými audiovizuálnymi aplikáciami, ktoré prezentujú tematické celky z iných predmetov.

3 IKT VO VYUČOVANÍ NA PRVOM STUPNI ZÁKLADNEJ ŠKOLY

Výchovno-vzdelávací proces sa v súčasnosti nezaobíde bez využívania moderných informačno-komunikačných technológií ani na prvom stupni ZŠ. Nakoľko vývoj IKT neustále napreduje, objavujú sa nielen nové, na didaktické účely vhodné elektronické a digitálne zariadenia, ale sa vyvíjajú aj nové technológie ich uplatnenia. Cieľom nie je len optimalizovať vzdelávací proces, ale predovšetkým zvýšiť jeho efektivitu, účinnosť a atraktívnosť aj pre najnižšie vekové kategórie školákov. Aby deti do školy chodili radi, aby vyučovanie a učenie sa boli zábavné, aby pobyt škole bol príjemný, plný pozitívnych zážitkov. Preto v škole má byť samozrejmosťou správne využívanie moderných prostriedkov IKT, ktoré školu môžu premeniť na veľké ihrisko a učenie sa na príjemnú kolektívnu hru Tak možno zabezpečiť a spríjemniť prechod z bezstarostného detstva do obdobia povinnej školskej dochádzky. Preto je potrebné aby sa školy, ktoré chcú držať krok s dobou, snažili zaobstarať moderné IKT, aby oživili a spríjemnili samotný výchovno-vzdelávací proces, motivovali žiakov k učeniu, vzbudili ich záujem a zvedavosť a podporili ich kreativitu a aktivitu v získavaní nových poznatkov.

Prostredníctvom IKT u žiakov možno vybudovať pevné základy a rozvíjať mnohé schopnosti ako sú napr. vyjadrovanie, komunikácia, kreativita, práca v tíme, zvedavosť, bádanie, konštruovanie atď. Pritom možno použiť účinné a efektívne aktivizujúce metódy a prístupy aktívneho učenia ako napr. projektové a problémové vyučovanie, zvedavosťou riadené učenie sa, bádateľsky orientované učenie sa, zážitkové učenie sa a pod. s uplatnením princípov konštruktivizmu, objavovania, konektivizmu, kooperácie a kolaborácie a to všetko hravou a zábavnou formou.

Aká je situácia na školách, aká je „tvrdá realita“?

Na prvom stupni základnej školy sa využívanie IKT ešte stále nachádza v štádiu objavovania jeho veľkého didaktického potenciálu a možností. Spočiatku sa IKT využívali hlavne na hodinách informatiky, dnes sa IKT využívajú ako prostriedky k učeniu v rámci akéhokoľvek predmetu už aj na primárnom stupni. Od školského roka 2015/2016 vstúpil do platnosti inovovaný štátny vzdelávací program, ktorý reviduje štátny vzdelávací program (ŠVP) platný od školského roku 2008/2009. Tento reguluje aj hodinovú dotáciu vyučovania informatiky na ZŠ [8]. Veľkým prínosom ŠVP v oblasti informatiky je, že informatika sa ako povinný vyučovací predmet rozšírila už aj na 1. stupeň ZŠ.

Je známe, že oblasť informatiky zaznamenáva mimoriadny prudký rozvoj, preto v predmete informatika je potrebné dôkladnejšie sa zamerať na štúdium nadčasových základných poznatkov. Dostupné technológie majú poskytnúť vyučovaniu informatiky široký priestor na motiváciu a praktické získavanie aktuálnych informácií, na vybudovanie základnej informačnej gramotnosti a jej rozvoj, sústrediť sa aj na kritické myslenie, na informačnú bezpečnosť a na pestovanie návykov a potreby permanentného vzdelávania. Tak postupne už v útlom veku je potrebné začať budovať informatickú a informačnú kultúru, vychovávať a viesť žiakov k efektívnemu využívaniu prostriedkov informačnej civilizácie s rešpektovaním právnych a etických zásad používania informačných technológií a softvérových produktov. Tieto ušľachtilé dlhodobé ciele je potrebné dosiahnuť spoločným pôsobením predmetu informatika a aplikovaním informačných technológií vo vyučovaní iných predmetov. Systematické základné vzdelanie v oblasti informatiky a využívania jej nástrojov zabezpečí pre každého rovnaké príležitosti na uplatnenie sa a plnohodnotný život v informačnej a znalostnej spoločnosti, ktorú budujeme.

Profesor Ivan Kalaš, ktorý sa profesionálne venuje pedagogickému výskumu v oblasti teórie vyučovania informatiky a informatizácie poznávacieho procesu, v rozhovore pre časopis Quark fundovane hodnotil situáciu informatizácie vzdelávania na základnej škole. Naznačil, ako pomôcť učiteľom na prvom stupni ZŠ plniť svoju novú úlohu v tomto procese [9]. Dôležité je ale, aby učiteľ bol pokrokovým myšlienkam naklonený, aby to robil dobrovoľne a z presvedčenia a aby bol vnútorne dostatočne motivovaný. Dobrý učiteľ primárneho vzdelávania „*vie vytvárať situácie, v ktorých žiak pod jeho citlivou réžiou objavuje a poznáva. A vie, ako pomáhať svojim žiakom efektívne prepájať zážitky a malé objavy zo všetkých vyučovacích predmetov...*“ Vekovú kategóriu žiakov na prvom stupni základnej školy považuje profesor Kalaš za mimoriadne vhodnú na začatie výučby informatiky [9].

Moderné technológie, ktoré prenikajú do vzdelávania, tradičnú výučbu významne ovplyvňujú a od základov ju menia. Je už len na učiteľovi ako ich didaktický potenciál využije v prospech vyučovania na splnenie stanovených úloh a na dosahovanie vzdelávacích cieľov.

4 PEDAGOGICKÝ EXPERIMENT

V priebehu školského roka 2017/18 bol (študentkou našej fakulty Bc. Lenkou Vaškovou) pod vedením jednej z autoriek tohto článku zrealizovaný pedagogický experiment na Základnej škole s materskou škôlkou v Rakoviciach v Trnavskom okrese [10]. Školu navštevuje 161 žiakov. V každom ročníku je jedna trieda, a v triedach je 14 – 26 žiakov. Na škole pracuje 14 učiteľov, 1 asistent učiteľa a 2 vychovávateľky. Je to škola, ktorá tvorivosť a humánnosť stavia ako priority výchovy a vzdelávania, vychováva a vzdeláva žiakov, ktorí budú tvorivo a kriticky myslieť a riešiť problémy, vytvára podmienky na formovanie správnej hodnotovej orientácie detí, rešpektuje individualitu detí a

rozvíja národné a historické povedomie (čerpané z kréda školy zverejnenej na ich webovej stránke) [11]. Škola je (príp. bola) zapojená do 13 projektov, medzi ne patrí: Elektronizácia a revitalizácia školských knižníc, Počítače pre školy a Zvyšovanie kvality vzdelávania v základných a stredných školách s využitím elektronického testovania vedomostí.

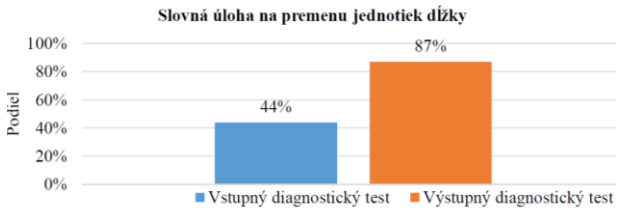
Škola je pomerne dobre vybavená didaktickou technikou, ktorá umožňuje uplatniť moderné vzdelávacie technológie vo vyučovaní. Medzi inými majú počítačovú učebňu vybavenú 21 počítačmi, jazykové laboratórium so 14 žiackymi pracovnými stanicami, počítačmi, interaktívnou tabuľou a novým nábytkom, moderne vybavené učebne s interaktívnymi tabuľami a projektormi.

Pedagogický experiment pozostával zo vstupného vedomostného testu, z 3-týždňovej starostlivo pripravenej a realizovanej výučby predmetu matematika vo štvrtom ročníku s použitím dostupných didaktických aplikácií pre počítač a interaktívnu tabuľu na prezentovanie, nacvičovanie a testovanie poznatkov. Triedu tvorilo 20 žiakov (7 dievčat a 13 chlapcov). Následne po uskutočnení experimentálnej výučby bol vykonaný dotazníkový prieskum spokojnosti medzi žiakmi a realizovaný výstupný vedomostný test. Výskumníčka prípravu na vyučovanie vypracovala v písomnej forme s uvedením didaktických aplikácií, ktoré chcela využiť a ktoré vyberala starostlivo a uvážlivo. Počas výučby sledovala reakcie a počínanie žiakov a svoje pozorovania a postrehy zaznamenala po každej vyučovacej jednotke. Každá vyučovacia hodina bola podrobne analyzovaná a hodnotená.

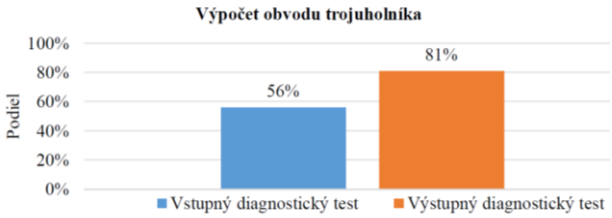
Didaktické aplikácie, ktoré sa využili vo vyučovaní, boli vyhotovené väčšinou v prostrediach PowerPoint a Hot Potatoes a čerpané z voľne dostupných webových stránok [12, 13, 14].

Vzhľadom na to, že výskumná vzorka bola malá, pedagogický experiment budeme hodnotiť len globálne a z dotazníka pre žiakov explicitne vyhodnotíme len odpovede na niektoré vybraté otázky, ktoré sa nám zdajú byť zaujímavé, a získané výsledky poučné. Rovnako uvedieme z vedomostného testu len úspešnosť riešení úloh z niektorých tematických celkov a porovnáme vedomostnú úroveň žiakov pred a po využívaní vybratých didaktických aplikácií na pochopenie, precvičovanie a upevňovanie vedomostí.
Vedomostný test obsahoval 10 úloh, väčšinou slovných. Niektoré úlohy mali aj podúlohy, kde bolo treba najprv priamo realizovať dané operácie (napr. sčítanie) s uvedenými číslami. Správne vyriešenie podúloh bolo predpokladom správneho riešenia celej úlohy. Napr. Zisti výsledok jednotlivých súčtov (ktoré sú usporiadané do dvoch stĺpcov) a v každom stĺpci zakrúžkuj najväčší výsledok.

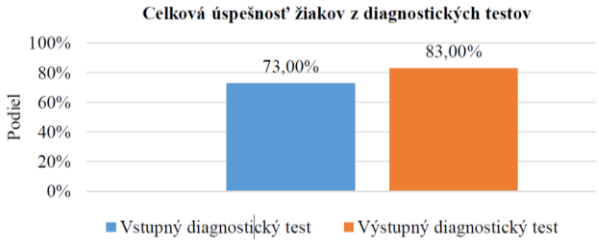
Typy úloh vo vstupnom aj výstupnom teste boli rovnaké, len formulácia úlohy a číselné hodnoty v nej sa zmenili. Riešenie úlohy bolo treba priamo zadať, alebo určiť správne riešenie výberom jednej z ponúknutých možností. V ďalšom uvedieme len dva príklady, keď sa žiaci výrazne zlepšili v správnych riešeniach.



Obrázok 1: Výsledky úlohy: Premena jednotiek dĺžky



Obrázok 2: Výsledky úlohy: Výpočet obvodu trojuholníka

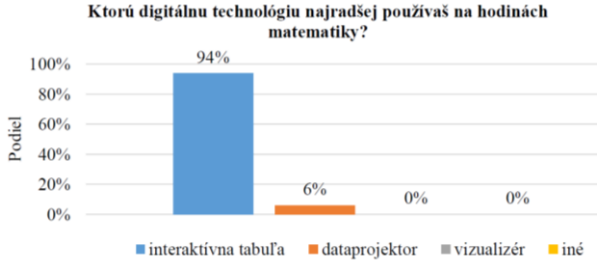


Obrázok 3: Percentuálne znázornenie úspešnosti žiakov z riešenia slovných úloh

Percentuálny rozdiel celkovej úspešnosti žiakov vo vstupnom a výstupnom texte je len 10 % v prospech výstupného testu. To ale naznačuje, že rozdiely v riešení niektorých úloh neboli také výrazné a dokonca niektoré úlohy boli horšie vyriešené vo výstupnom teste ako vo vstupnom. Tak tomu bolo napr. pri úlohe z tematických celkov Vyjadrenie popoludňajšieho času a Zlomky.

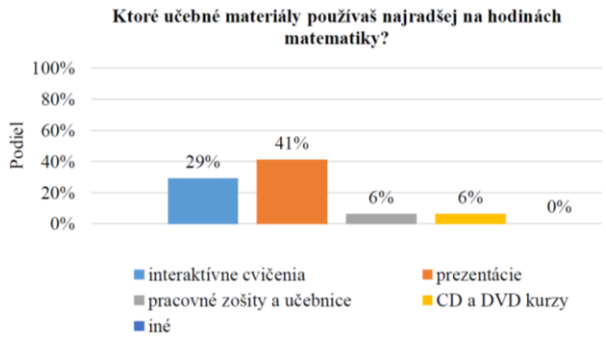
Dotazník, ktorý bol orientovaný na prieskum názorov žiakov na používanie vzdelávacích technológií vo vyučovaní, obsahoval 12 otázok. Na hodnotenie vyberáme z nich len dve.

Na otázku: *„Ktorú digitálnu technológiu najradšej používaš na hodinách matematiky?“* až 94 % respondentov zvolilo ako odpoveď interaktívnu tabuľu. Táto voľba nás veľmi neprekvapila, pretože tento prostriedok je na zvolenej škole bežne dostupný a pravdepodobne najviac používaný. Prostredníctvom interaktívnej tabule sa žiakom prezentuje učivo atraktívnejšou formou, z vyučovania majú príjemné zážitky, majú možnosť aktívne sa zapojiť, spolupracovať a pritom sa rozvíjajú aj ich zručnosti v oblasti používania počítačov a informačných technológií [10, 15, 16].



Obrázok 4: Hodnotenie obľúbenosti použitých digitálnych technológií

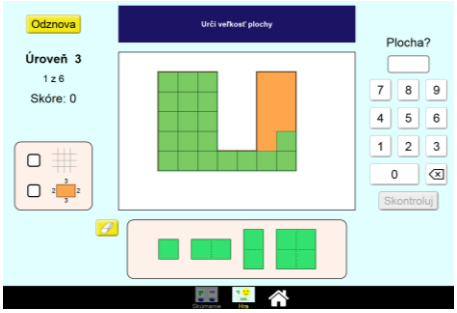
Druhá vybratá otázka znela: *Ktoré učebné materiály používaš najradšej na hodinách matematiky?* Ako odpoveď bolo ponúknutých 5 možností, z toho 41 % žiakov zvolilo elektronické prezentácie, 29 % interaktívne cvičenie, 6 % pracovné zošity a učebnice a 6 % CD a DVD nahrávky (6 % predstavuje 1 žiaka).



Obrázok 5: Najobľúbenejšie didaktické materiály a aktivity na hodinách matematiky

Z dotazníkového prieskumu vyplýva, že najčastejšie používané vzdelávacie technológie na základných školách sú viazané na interaktívnu tabuľu a počítač (príp. notebook) s pripojením na Internet.

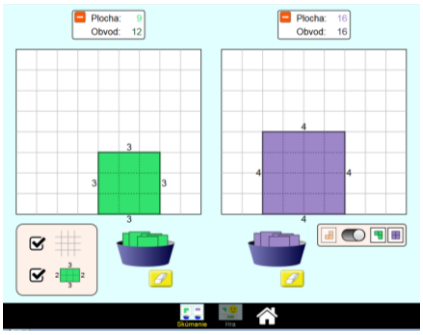
V prieskume sme ďalej pozorovaním zistili, že žiaci sú počas vyučovania matematiky s využitím IKT pozornejší, aktívnejší, nevyrušujú, lepšie si osvoja učivo a samotné vyučovanie prebieha zábavnou a hravou formou. Na základe vyplnených dotazníkov možno povedať, že žiaci štvrtého ročníka najradšej používajú prezentácie vytvorené v programe PowerPoint, pretože obsahujú rôzne zaujímavé animácie, obrázky a zvukové sekvencie, ktoré ich zaujmú a zabavia. Za druhú najčastejšie používanú učebnú pomôcku, ktorú si obľúbili, označili interaktívne cvičenie dostupné na Internete, príp. vytvorené pomocou softvéru Hot Potatoes. Využívali sa niektoré interaktívne cvičenia a prezentácie, ktoré nám odporúčala cvičná učiteľka (matematiky) vyučujúca v tejto triede. Niektoré cvičenia boli pripravené a doplnené vzhľadom na potrebu posilniť vyučovanie niektorých tematických celkov. Softvérové aplikácie, ktoré sa používali počas pedagogického experimentu, sa žiakom veľmi páčili, každý žiak s nimi chcel pracovať a vyskúšať si ich. Každodenné používanie IKT počas pedagogického experimentu bolo pre žiakov veľmi zaujímavé, pretože ich učiteľka matematiky nepoužíva na každej hodine interaktívnu tabuľu. Všetky edukačné činnosti, ktoré boli overované v pedagogickom experimente, boli primerané veku a v súlade s inovovaným štátnym vzdelávacím programom pre prvý stupeň ZŠ a so vzdelávacou oblasťou Matematika a práca s informáciami [8].



Obrázok 6: Riešenie úlohy: Urči veľkosť plochy

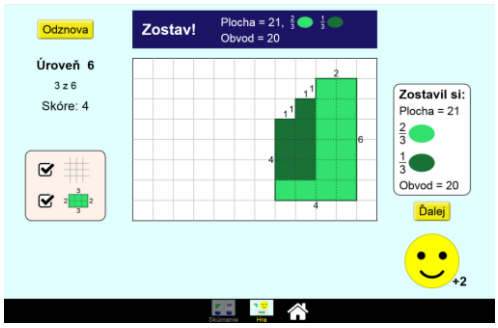
Pre učiteľov základných škôl možno odporúčať ako vhodné alternatívne interaktívne učebné pomôcky animačno-simulačné modely na stránke univerzity v Colorade.

https://phet.colorado.edu/sims/html/area-builder/latest/area-builder_sk.html



Obrázok 7: Porovnanie veľkosti plochy a obvodu štvorcov

Obrázky 6. – 7. demonštrujú možnosti spracovanej témy na výpočet obvodu a obsahu plošných útvarov vytvorených z jednotkového štvorca (s dĺžkou strany 1). Žiak môže pracovať v dvoch režimoch: v režime skúmania problematiky a v režime interaktívneho riešenia úloh hravou formou. Úlohy sú rôzneho stupňa náročnosti a precvičuje sa pritom nielen stanovenie a výpočet veľkosti plochy a obvodu útvaru, ale je do riešenia úloh zaintegrovaná aj práca so zlomkami – rozdelenie plochy útvaru v určitom pomere.



Obrázok 8: Rozdelenie plochy v určitom pomere

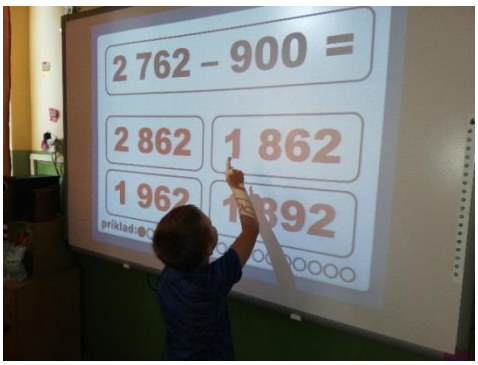
Obrázok 9: Matematická rozcvička na tému Obvod štvorca
(z vlastnej produkcie)

5 DISKUSIA

Informatika má dôležité postavenie vo vzdelávaní, pretože podobne ako matematika rozvíja myslenie žiakov, ich schopnosť analyzovať a syntetizovať, zovšeobecňovať, kriticky posudzovať, porovnávať, rozhodovať, hľadať vhodné stratégie riešenia problémov a overovať ich v praxi. Ak informatika ašpiruje na pozíciu ďalšej základnej vednej disciplíny popri

klasických vedných disciplínach, musí svoju pozíciu vybojovať a neustále ju upevňovať a udržiavať. Zavedením informatiky do vzdelávania ako povinného vyučovacieho predmetu už od prvého stupňa ZŠ sme boj ešte nevyhrali. Školy a náš školský systém na vyučovanie informatiky neboli pripravené. Chýbali kvalifikovaní učitelia, a to hlavne na prvom stupni základných škôl. Nadšencov, obetavých učiteľov, ktorí svoje povolanie učiteľa berú ako poslanie, je málo. I napriek tomu možno poslúžiť „dobrými príkladmi“, keď učiteľ prvého stupňa základnej školy „svojpomocne“ z vlastnej iniciatívy sa vypracoval na moderného učiteľa, ktorý zručne využíva vo svojej pedagogickej praxi nové vzdelávacie technológie na báze digitálnej techniky [15].

Vďaka mnohým iniciatívam, projektom, štátnej a sponzorskej podpore a pomoci boli školy vybavené základnými technickými prostriedkami a získali prístup na Internet. Vyškolil sa (príp. sa preškolil) veľký počet učiteľov, ktorí sa naučili nové technické zariadenie nielen obsluhovať a moderné technológie využívať, ale získali základné zručnosti aj z tvorby didaktických aplikácií [16, 17, 18, 19, 20, 21, 22]. Aj keď optimizmus je na mieste, nemožno nereagovať na alarmujúcu správu Štátnej školskej inšpekcie o stave vzdelávania na Slovensku za školský rok (2016/17). Zo správy vyplýva, že informatiku na každej druhej základnej škole vyučuje telocvikár, slovenčinár či učiteľ zemepisu. „Za výrazne negatívnu možno považovať skutočnosť, že až v 45 percentách kontrolovaných škôl bol tento predmet v plnom rozsahu vyučovaný neodborne a v ďalších viac ako desiatich percentách škôl vyučovanie informatiky zabezpečovali s nižšou ako 50-percentnou odbornosťou“ uvádza vo výročnej správe štátna inšpekcia [23]. Aj keď pedagogické fakulty pripravujúce učiteľov vo svojich nových študijných programoch adekvátne reagovali na potreby praxe a zaradili do učiteľských programov predmety s príslušným obsahom, nie každá škola môže zamestnať učiteľa „z novej generácie“, čo by mohlo situáciu čiastočne riešiť.



Obrázok 6: Interaktívna tabuľa na hodine matematiky

6 ZÁVER

Pre prosperitu krajiny je kľúčové rozširovanie digitálnej gramotnosti. Odborníci na informatiku a informačné technológie v súčasnosti patria medzi najhľadanejšie profesie a obsadzujú najlepšie platené pozície nielen na lokálnom, ale aj na medzinárodnom trhu práce. Základnú úroveň digitálnych zručností a informačnej gramotnosti bude podľa Európskej komisie v blízkej budúcnosti vyžadovať až 90 % pracovných pozícií. V roku 2020 bude podľa IT asociácie Slovenska v IT sektore chýbať až 20 tisíc zamestnancov [23]. Preto je spoločenskou objednávkou zabezpečiť informačnú a informatickú výchovu od raného detstva, keď sa vytvárajú a formulujú základné postoje, názory a vzťahy k spoločnosti, ľuďom, okoliu, vzdelávaniu, práci, povolaniu a pod. Začať s tým neskôr môže byť neskoro. Informatické myslenie je pre vzdelávanie, život a prácu v tomto storočí novou nevyhnutnosťou, novou špecifickou gramotnosťou.

POĎAKOVANIE

Článok bol publikovaný s podporou projektov KEGA 012TTU – 4/2018: Interaktívne animačno-simulačné modely vo vzdelávaní a KEGA 015TTU – 4/2018: Interaktivita v elektronických didaktických aplikáciách.

BIBLIOGRAFICKÉ ODKAZY

[1] KALAŠ, I. a kol. 2013. *Premeny v školskom digitálnom veku.* 1. vyd. Bratislava : SPN Mladé letá, 2013. 256 s. ISBN 978-80-0100-2409-4.

[2] PRIBILOVÁ, K. 2013. *Informačná gramotnosť a počítačová gramotnosť.* [online]. Trnava : Pedagogická fakulta Trnavskej univerzity v Trnave, 2013. [cit. 2018. 15.12.]. Dostupné na internete: http://pdf.truni.sk/e-ucebnice/informacna-gramotnost/. ISBN 978-80-8082-682-6.

[3] MARTINKOVÁ, A. Tvorba učebních pomůcek pomocí interaktivní tabule Smart Board. [online]. Ostrava : Ostravská univerzita v Ostrave, 2010, s. 10 – 12 [cit. 2018.24.07.]. Dostupné na internete: http://edu.vsb.cz/interaktivni_tabule/_kurz_interaktivni_tabule/05_dalsi_materialy/2010_01_31_OU_Ostrava_Distan%C4%8Dn%C3%AD_opora_Tvorba_uceb_pomucek_pomoci_itab.pdf

[4] HÍC, P., POKORNÝ, M. E-learning ako efektívny nástroj vo vyučovaní matematiky na PdF TU. In: *INFOTECH 2007*, Olomouc, PF UP. s. 239-242. ISBN 978-80-7220-301-7.

[5] VIDERMANOVÁ, K. Možnosti využitia programu Hot Potatoes pri tvorbe úloh z matematiky. In: *Aktivizujúce prvky vo výučbe matematiky.* Nitra : FPV UKF v Nitre, 2013. s. 388-397, ISBN 978-80-558-0471-2.

[6] ŽILKOVÁ, K. *Školská matematika v prostredí IKT.* Bratislava : Univerzita Komenského. 2009. 138 s. ISBN 978-80-223-2555-4.

[7] PARTOVÁ, E. *Vyučovanie matematiky pomocou moderných technológií.* Bratislava: UK, 2011. 94 s. ISBN 978-80-223-3144-9.

[8] MINISTERSTVO ŠKOLSTVA SR, *Štátny vzdelávací program pre primárne vzdelávanie – 1. stupeň základnej školy. 2015.* [online]. Bratislava : ŠPÚ, 2015. [cit. 2019-01-14]. Dostupné na internete: http://www.statpedu.sk/files/articles/dokumenty/inovovanystatny-vzdelavaci-program/svp_pv_2015.pdf

[9] http://www.quark.sk/vyucba-informatiky-na-1-stupni-zs/ [cit. 2019-01-14].

[10] VAŠKOVÁ, L. *IKT vo vyučovaní na prvom stupni základnej školy*. [Diplomová práca]. Trnavská univerzita v Trnave, Pedagogická fakulta, Katedra školskej pedagogiky. Trnava : TU PdF, 2018. 111 s.

[11] https://zssmsrakovice.edupage.org/[cit. 2018-01-14].

[12] http://www.andreaucitelka.wz.sk/jednotky_dlzky/jednotky_dlzky1.htm [cit. 2018-01-14].

[13] http://hotpot.wbl.sk/jednotky_dlzky_o_zvieratkach.htm [cit. 2018-12-14].

[14] http://megym.wbl.sk/uspstart.htm [cit. 2018-01-14].

[15] DOSTÁL, J. Reflections on the Use of Interactive Whiteboards in Instruction in International Context. In: *The New Educational Review*. 2011, Vol. 25, No. 3. p. 205 – 220. ISSN 1732-6729.

[16] CHRÁSKA, M. Informační technologie ve škole. In KROPÁČ, J. a kol. Didaktika technických předmětů. 1. vyd. Olomouc : Pdf UP, s. 154 – 157. ISBN 80-244-0848-1.

[17] NEMCOVÁ, J. *Edukačný potenciál interaktívnych technológií v primárnom vzdelávaní*. [Dizertačná práca], Univerzita Komenského v Bratislave, Pedagogická fakulta, Katedra predprimárnej a primárnej pedagogiky. 2016. 142 s.

[18] KALAŠ, I. Učebné aktivity žiakov pre 21. storočie. In: *DidInfo 2012* [CD]. Banská Bystrica : Univerzita Mateja Bela, Fakulta prírodných vied, 2012, s. 35-46. ISBN 978-80-557-0342-8.

[19] ČERNOCHOVÁ, M, KOMRSKA, T., NOVÁK, J. *Využití počítače při vyučování: náměty pro práci dětí s počítačem.* Praha : Portál, 1998. ISBN 80-717-8272-6.

[20] ŽILKOVÁ, K. Interaktívne prostredie v perspektívach vzdelávania. In: *Acta Facultatis Paedagogicae Universitatis Tyrnaviensis*. Trnava: TU, 2007, s. 14 - 19. ISBN 978-80-8082175-3.

[21] PŠENÁKOVÁ, I. Interactive applications in the work of teacher. In: *XXIXth DidMatTech 2016.* Budapest : Eötvös Loránd University in Budapest, Faculty of Informatics, 2016. ISBN 978-963-284-800-6. pp. 92 – 100. https://www.mii.lt/informatics_in_education/htm/infedu.2017.07.htm. WOS:000399818000007

[22] BASLER, J. Počítačové hry a způsob jejich využívání u žáků základních škol. In: *Trendy ve vzdělávání*, 9(1), 2016, s. 10-19.

[23] https://www.etrend.sk/ekonomika/vyse-polovica-skol-vyucuje-deti-informatiku-neodborne.html [cit. 2018-01-14].

Inovativní výuka chytrých měst
Innovative Teaching of Smart Cities

Miroslav Svítek
Fakulta prírodných vied
Univerzita Mateja Bela
Tajovského 40
97401 Banská Bystrica
Slovensko
Miroslav.Svitek@umb.sk

ABSTRACT
The goal of the paper is to describe practical experiences of teaching smart cities based on common synergies of individual specializations within a given territory. Since it is a systemic and broad issue, pedagogical activity needs to work with the appropriate structure of knowledge divided into individual modules. The common interconnection of modules asks for the innovative approaches.

Keywords
Smart cities, smart region, smart solution, smart technology.

ABSTRAKT
Cílem tohoto příspěvku je popsat praktické zkušenosti z výuky chytrých měst (smart cities), kde dochází k synergiím jednotlivých oblastí na daném území. Jelikož jde o systémové zachycení velmi široké problematiky, pro pedagogickou činnost je velmi důležitá struktura dílčích znalostí rozdělená do jednotlivých modulů a využití inovativních přístupů pro prezentaci jejich vzájemného provázání.

Kľúčové slová
Chytrá města, chytrý region, chytré řešení, chytrá technologie.

1. Úvod
Pokročilé informační systémy lze již dnes nalézt ve všech oblastech našeho hospodářství – ve výrobní sféře hovoříme o konceptu Idustry 4.0, v energetice o Smart gridech, v dopravě o Inteligentních dopravních systémech, ve státní správě o systému eGovernment, ve zdravotnictví o eHealth, atd.

Chytrá (smart) řešení se snaží o účelnou koordinaci všech těchto oblastí s ohledem na společné kritérium, kterým je udržitelný rozvoj společnosti a kvalita života občanů. Toho lze dosáhnout tím, že chytrá řešení budou usilovat v první řadě o minimalizaci použitých zdrojů (voda, plyn, elektrická energie, zábor země) a zároveň o maximální efektivitu stávajících infrastruktur (dopravní síť, energetická síť, radiová síť, atd.).

Dopad chytrých řešení se projeví v různých sférách života společnosti, protože umožňují přechod od globalizace k lokalizaci. Díky moderním technologiím (spojeným s 3D tiskem) se může výroba distribuovat do mnoha malých měst, čímž vzniknou lokální pracovní příležitosti, které udrží mladou generaci v těchto městech. Díky tomu se sníží požadavky na dopravu osob do velkých měst, ale na druhou stranu vzrostou nároky na informační propojení a datová centra.

Podobný efekt bude možno sledovat i na globální úrovni, kdy místo dopravy levného zboží z Číny (diskutovaná hedvábná stezka) budou důležitější informační koridory pro přenos know-how a jeho zabezpečení proti zneužití. Obdobně bude třeba řešit dopravu a logistiku na národní úrovni. Národní dálniční síť je navržena na požadavky z minulého století, které vycházejí z existence několika velkých továren v blízkosti velkých měst. Obdobná situace je v energetice i v dalších odvětvích.

Jedná se skutečně o (čtvrtou) revoluci [1] a ne pouze o evoluci, protože dopad propojených systémů Internetem věcí, služeb a lidí bude mít dopad na všechny procesy. Hovoří se proto o Společnosti 4.0 nebo též o Myšlení 4.0. Tyto společenské změny si vyžádají také inovativní přístupy ke vzdělávacímu procesu, o kterém se často hovoří jako o Vzdělání 4.0.

2. Chytrá města
Příkladem chytrých řešení je i oblast chytrých měst (smart cities), kde celosvětově žije a pracuje stále více a více obyvatel. Z ekonomického hlediska jsou samy města se svým produktivním prostředím produktem s nejvyšší přidanou hodnotou, což historicky prokázali Jan a Tomáš Baťa, kdy importovali celá funkcionalistická města do celého světa.

Současná doba si vyžaduje, aby v takto komplexních systémech, jako jsou města, byly co možná nejvíce využity nejmodernější technologie pro podporu jejich rozšiřování a managementu. Aplikace chytrých měst je možno rozdělit na cílové a druhové. Cílové aplikace se snaží optimalizovat energetickou spotřebu, zlepšit čistotu ovzduší, snížit hlukovou zátěž, regulovat dopravu, atd. Druhové aplikace naopak podporují identitu daného místa (urbanistické struktury), tj. jeho vlastní historickou, kulturní, ekologickou či estetickou podstatu. Získané poznatky lze rozšířit na další typy územních celků, jako jsou chytré vesnice (Smart Villages), nebo chytré regiony (Smart Regions).

3. Výuka chytrých měst v USA
V rámci vzájemné spolupráce mezi několika mezinárodními univerzitami [2] vznikl požadavek na vytvoření vysokoškolského předmětu v rozsahu 6+6 hodin týdně, který by uvedl základní teoretické poznatky a připravil studenty na plánování, projektování, implementaci a provoz chytrých měst (smart cities) tak, aby studenti mohli působit v inovativním výzkumu a pohybovat se v interdisciplinárním prostředí.

Nový předmět s názvem Smart Cities Fundamentals byl realizován v zimním semestru 2018 na University of Texas at El

Paso v USA. Obsah předmětu byl rozdělen pro lepší srozumitelnost do následujících sedmi modulů:

3.1 Základy

Studenti si osvojili základní terminologii a seznámili se s přehledem stávajícího vývoje v různých oblastech. Speciální důraz byl kladen na problematiku Industry 4.0, ze které koncept chytrých měst filozoficky vychází a také na inovace v oblasti urbanismu, kde jsou čím dál více používány metody umělé inteligence při návrhu sídelních struktur. Jako příklady byly představeny koncepce různých měst, výsledky různých projektů a také přehled organizací, které se danou problematikou zabývají, např. www. smartcities.ieee.org , www.smartcitiescouncil.com , www.eu-smartcities.eu .

3.2 Technologie

Studenti se seznámili se základními principy nových technologií, které se používají při návrhu a implementaci chytrých řešení. Jde zejména o oblast architektur informačních systémů (Cloud-Fog, BigData), informačních technologií (IoT, semantický web), bezpečnostních technologií (blockchain, identifikační systémy), komunikačních technologií (SigFox, LoRa, 5G), satelitních technologií (Galileo, GPS, Copernicus), energetických technologií (solární panely, baterie), technologie umělé inteligence (neuronové sítě, fuzzy systémy, multi-agentní systémy), technologie virtuální či rozšířené reality, pokročilé detektory (senzory, drony), atd. Znalost jednotlivých technologií se neomezovala pouze na technické parametry, ale zahrnovala též způsoby implementace, možné ekonomické modely či různá legislativní omezení.

3.3 Základní komponenty

Znalost jednotlivých technologií umožňuje popis dílčích komponent chytrého města dle jednotlivých sektorů. V rámci chytré dopravy byly prezentovány způsoby dopravního modelování, vytváření a verifikace různých scénářů dopravního řízení, sdílení dopravních prostředků či zvyšování atraktivity veřejné dopravy. V oblasti chytrých energetických sítí (smart grids) byly ukázány různé způsoby optimalizace energické energie pomocí alternativních zdrojů a baterií. Chytré osvětlení popisovalo osvětlování města dle aktuální potřeby a viditelnosti nízkoenergetickými svítidly. Chytrá logistika může využít sdílených vozidel případně dronů pro rychlou přepravu nákladů po městě. Chytré zdravotnictví bylo popsáno na základě informačního propojení mezi pacienty a lékařskými zařízeními, kde je možno on-line monitorovat stav pacientů a předcházet neočekávaným událostem. Chytrý bezpečnostní systém může monitorovat a předvídat nebezpečné situace a zajistit jejich včasné řešení. Chytré vodní hospodářství monitoruje nakládání s vodou a reaguje na poruchy rozvodné sítě či na přítomnost škodlivých látek. Chytré odpadové hospodářství provádí včasný svoz a recyklaci odpadů. Z výše uvedeného přehledu je zřejmé, že každý sektor využívá svá chytrá řešení pro management svých oblastí.

3.4 Integrace do vyšších celků

Základním znakem chytrého města (smart city) je schopnost vzájemné integrace dílčích komponent do vyšších celků (subsystémů). Tímto vyšším celkem může být chytrá ulice, chytré náměstí, chytrá čtvrť, chytrý univerzitní kampus, atd. Při integraci je třeba řešit vzájemná rozhraní pro výměnu a sdílení dat, která jsou standardizována (CEN, ISO, IEEE standardy). Pro porozumění technickým standardům rozhraní je třeba znát metodiku UML (Unified Modelling Languague), která zachycuje architekturu daného informačního systému a metodiku ASN.1 (Abstract Syntax Notation Number One), která definuje datové rozhraní mezi dvě subsystémy. Na konkrétních příkladech byly ukázány platné CEN a ISO standardy a jejich využití při integraci dílčích subsystémů chytrého města. Dalším oblastí integrace bylo vytvoření ontologického modelu, aby nedocházelo k neregularitě rozhraní díky sémanticky nekorektnímu popisu komponent chytrého města. Ontologický model ulice, parku, náměstí, atd. umožňuje holistický popis pomocí informací z mnoha sektorů, což je základním požadavkem pro vznik virtuálního obrazu daného území (twin city). Ve virtuálním prostoru lze realizovat celou řadu simulací napříč jednotlivých sektorů a pomocí nástrojů umělé inteligence modelovat synergie mezi jednotlivými sektory tak, aby docházelo k optimalizaci kriteriální funkce daného území. Ve většině případů je snaha o co nejvyšší kvalitu života, která se projevuje dobrým životním prostředím a minimalizací obecné energetické náročnosti, kam lze zahrnout i ztracený čas strávený například při cestování.

3.5 Lidský faktor

Chytrá řešení se neomezují pouze na technické systémy, ale musí být schopna dobře komunikovat s uživateli prostřednictvím HMI (Human Machine Interface). V případě chytrých měst je třeba řešit komunikaci s různými skupinami obyvatel, jako jsou děti, hendikepovaní spoluobčané, senioři, atd. Každý musí mít právo se rozhodnout, zdali chce či nechce moderní technologie používat. Ti, co nechtějí mít s moderními technologiemi nic společného, budou mít sice horší nabídku služeb, ale přesto musí být schopni ve městě standardně fungovat. Lidský faktor lze studovat i v komunikaci magistrátu města (governance) se svými občany. Nové technologie umožňují různé formy hlasování, projednávání, upozorňování na problémy (rozbitý chodník, atd.) nebo vytváření různých komunit, které mohou vzájemně lépe komunikovat a vyjadřovat potom svůj názor při komunikaci se státní správou nebo veřejnou samosprávou. Součástí této oblasti je i testování různých kritických scénářů typu black-out na umělém polygonu (virtuální prostředí) avšak s reálnými odpovědnými lidmi. Výhodou této strategie je, že lze zpětně analyzovat vzniklé problémy a při dalším opakování simulace odstranit slabá místa.

3.6 Hodnocení

Koncept chytrého města je způsob myšlení a cesta, kterou by si mělo každé město zvolit vzhledem ke svým problematickým oblastem i kulturní a historické tradici. Pokud má město zpracován svůj koncept, jak se chce vyvíjet, je třeba stanovit způsob vyhodnocování dílčích výsledků. Žádné město nemá ekonomické ani lidské zdroje na to, aby svůj chytrý koncept naplnilo ihned. Jde proto o postupné naplňování dílčích cílů, mapování dopadů a přijetí případných změn, pokud se v pilotních projektech ukáže, že řešení nepřináší kýžené výsledky. Technologie lze koupit, ale chytré řešení se musí léta budovat a tuto etapu nelze moc urychlit. V rámci komunity chytrých měst vznikla celá řada metodik, jak měřit chytrost města (smartness). Každá použitelná metodika se zaměřuje na oblast, pro kterou vznikla. Některé metodiky akcentuje kvalitu života (well-being), od které odvíjí měření, jak chytrá řešení přispívají k jejímu zlepšení. Dalším přístupy posuzují technickou vybavenost měst, další metodiky jsou více sektorově orientované, nejčastěji na posouzení kvality dopravní obslužnosti města. Řada měst si tvoří své indikátory sama dle svých problémů a sleduje jejich vývoj, což však omezuje vzájemné porovnání chytrosti měst mezi sebou. V rámci tohoto modulu jsou porovnány jednotlivé metodiky včetně způsobu užití a prezentace výsledků.

3.7 Příklady implementace

Studenti se seznámí s typickými řešeními chytrých měst z různých částí světa. Je třeba říci, že všechna města řeší shodné problémy, ale různými způsoby. Asijská města více využívají nejmodernější technologie. Vzorovou ukázkou je město Singapur, kde jsou nasazovány nejmodernější řešení a obyvatelé tento přístup akceptují. Na druhou stranu jsou města, která se snaží, aby technologie nebyly viditelné a aby občané ani nevěděli, že je město chytře řízené. Ukázkou je město Vídeň, která se zaměřuje na chytrou komunikaci s různými komunitami občanů a snaží se akcentovat lidský rozměr moderních technologií. Na druhé straně přelidněná indická města využívají moderní technologie, aby odvrátily ekologickou katastrofu, a řeší de facto krizový management těchto měst. Některá města jsou zaměřena pouze na určitý kritický prvek, jako například chytrá hranice (smart border) v El Paso, kde hranice je nejslabším a nejdiskutovanějším problémem obou měst Ciudad Juarez a El Paso. Jako příklad implementace chytrých řešení jsou prezentovány různé pilotní projekty vzniklé v rámci Smart Prague (www.smartprague.eu), kde jsou k dispozici výsledky těchto projektů v podmínkách města Prahy. Na dílčích projektech je možno popsat všechny poznatky získané ve všech předchozích modulech.

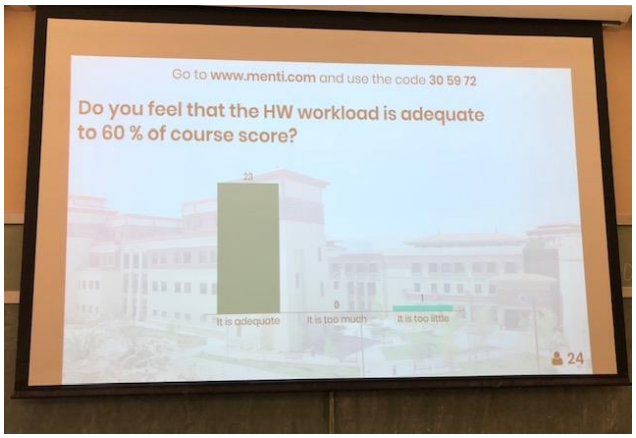
4. Inovativní pedagogické přístupy

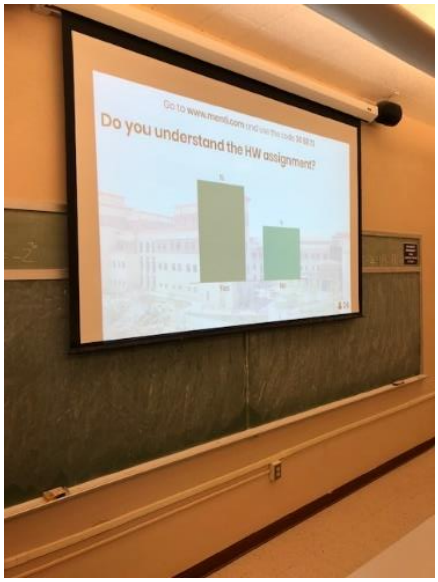
Vzhledem k rozsahu a inovativnosti oboru chytrých měst, bylo třeba studentům poskytovat novinky z tohoto oboru. Proto byla založena uzavřená skupina Smart Cities Fundamentals na Facebooku, jejíž členové byli jak studenti, tak i různí odborníci na oblast chytrých měst z různých zemí světa. Vzniklo tak informační prostředí pro jak pro sdílení novinek, tak i pro vzájemnou diskusi nad jednotlivými příspěvky (obr. 1).

Obr. 1 Speciální skupina na Facebooku věnovaná Smart Cities Fundamentals

Další inovací bylo využívání Mentimeteru (www.mentimeter.com), který umožňuje sbírání anonymních odpovědí na položené otázky, průzkumy přímo na přednáškách, vkládání různých termínů, atd. Tento software byl použit pro získání zpětné vazby od studentů přímo na přednáškách (obr. 2). Výhodou je, že lze položit konkrétní otázku, každý student si může otevřít webovou stránku (www.menti.com) na svém mobilním telefonu, vložit kód a může se začít vyjadřovat k danému problému.

Další zajímavostí bylo, že studentům byla nabídnuta účast na letní škole (summer school) v Praze, kde mohou řešit konkrétní praktické problémy pomocí znalostí, které získali v tomto předmětu. Letní škola je samozřejmě zpoplatněná, ale studenti kromě zkušeností získají i kredity pro své studium. Zároveň mohou výsledky práce na projektu prezentovat v rámci IEEE konference Smart City Symposium Prague 2019. Aby se studenti mohli zúčastnit letní školy, museli projít výběrovým řízením, protože o tuto aktivitu byl v USA velký zájem.





Obr. 2 Využití mentimeteru pro získávání on-line odpovědí od všech studentů přímo na přednáškách

5. Propojení pedagogické a výzkumné činnosti

V rámci společných výzkumných projektů, na kterých budou participovat i studenti bude např. řešen model hraniční oblasti El Paso (USA) / Ciudad Juárez (Mexiko) včetně identifikace silných procesů a řízení hraničních prvků tak, aby bylo co nejlépe splněno společné kritérium celého územního celku (bez ohledu na hranici). Je zřejmé, že toho nelze docílit zcela, protože hranice je velkou bariérou pro dané území, ale lze v rámci daných možností koordinovat prvky systému alespoň tak, aby sledovaly společné cíle.

Navzdory vytvořené umělé hranice má území společné přírodní vlastnosti jako jsou klimatické podmínky, zásoby energie atd. Na druhou stranu často existují velké rozdíly v politickém uspořádání, daňovém systému, ale i technologické vybavenosti rozdělených územních celků. Samostatným problémem je

překračování hranic, působení obyvatel v obou hraničních pásmech, rozvoj přeshraničního podnikání, řešení bezpečnosti, krizový management atd. Výsledky takto pojatého výzkumu budou určitě aplikovatelné i v dalších oblastech, protože tento stav může nastat pro různě rozdělené územní celky město-region, region-region, stát-stát a může vykazovat různě závažné problémy na svém rozhraní.

Český institut informatiky, robotiky a kybernetiky ČVUT v Praze (www.ciirc.cvut.cz) nabídne studentům a výzkumným pracovníkům pro řešení projektů své zkušenosti při budování Centra měst budoucnosti CCF (Center of the City of the Future), kde je zkoumáno město jako komplexní systém s jeho individuálními subsystémy, ale také jako část větší geografické a administrativní jednotky. CCF bude sloužit studentům jako výuková laboratoř, tak i jako testovací základna nových technologií pro transformaci měst na chytrá města (smart cities).

Na rozdíl od El Pasa je Praha historické, turistické město, jejíž virtuální model a model vybraného území čtvrti Praha 6 bude důležitou součástí společných projektů. Výsledné simulace budou prezentovány nad ortomapou celého města a jeho okolí a nad fyzickým modelem čtvrti Praha 6 včetně areálu ČVUT. Oba modely budou sloužit k vizualizaci procesů odehrávajících se v dané oblasti a umožní detailně porozumět jevům, které se odehrávají uvnitř městských struktur a rozpoznat jejich příčiny.

6. Závěr

Praha a El Paso jsou dvě města s různým klimatem, historií, kulturou, životním stylem, ekonomickým rozvojem, životním standardem atd. Čelí rovněž svým vlastním územním a politickým výzvám i omezením ze strany přírodních zdrojů. Studiem dvou různých měst získají studenti mezinárodní zkušenost, která jim umožní inovativně řešit problematiku chytrých měst (smart cities).

Celý výukový proces oblasti chytrých měst má naplánováno celou řadu etap, kde zkušenosti z vytvoření a realizace prvního samostatného předmětu Smart Cities Fundamentals byly shrnuty v tomto článku. Závěrem lze říci, že celá oblast chytrých řešení je pro studenty aktuální a je oprávněný předpoklad, že dokáže zlepšit atraktivity technického vzdělávání. Jako součást pedagogického procesu je vhodné maximálně využívat inovativních přístupů a nových technologií a jít tak příkladem s jejich praktickým nasazením.

Publikace

[1] Lom M., Přibyl O., Svítek M.: Industry 4.0 as a Part of Smart Cities, IEEE Smart Cities Symposium Prague 2016.

[2] Cheu R. L., Horák T., Ferregut C., Svítek M., Postránecký M.: International Dual Master Degrees Program in Smart Cities, IEEE Smart City Symposium Prague 2018.

[3] Svítek M., Postránecký M. a kolektiv.: Města budoucnosti, NADATUR, 2018, 392 stran, ISBN 978-80-7270-058-5.

[4] Postranecky M., Svitek M., Carrillo E. Z.: SynopCity Virtual HUB – A testbed for Smart Cities, IEEE Intelligent Transportation Systems Magazin, Vol. 10, Issue 2, 2018, pp. 50-57.

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Informatika v Slovenskej terminologickej databáze
Informatics in the Slovak terminology database

Martin Šechný
Prešovská univerzita v Prešove, Fakulta humanitných a prírodných vied
17. novembra 1
080 01 Prešov
Slovensko
martin.sechny@shenk.sk

ABSTRACT
Terms are basic elements of a learning content examined by didactics. The didactics of informatics should therefore determine what terminology (set of terms) in the informatics should be used. We need a reference register of terms, definitions and translations into other languages for that – the Slovak terminology database. A good definition is concise, unambiguous, contains one general term and a specific characteristic to it. With an increasing number of terms, the definitions may be finer, more accurate. The terminology is alive, evolving. Here are some terms from the Slovak terminology database, from the IT (Informatics) category, which I edit and fill up. Some terminology records are approved and published, some are pending approval. The terminology database can become a language source for a machine language translation. The terminology database will benefit a wide range of public administration institutions in Slovakia and the EU, companies, the public.

Keywords
Informatics. Terminology. Terminological database. Term. Definition. Wiki.

ABSTRAKT
Pojmy sú základné prvky obsahu vzdelávania, skúmaného didaktikou. Didaktika informatiky by preto mala určiť, aká terminológia (množina odborných pojmov) v informatike sa má používať. K tomu potrebujeme referenčný register termínov, s definíciami a prekladmi do iných jazykov – Slovenskú terminologickú databázu. Dobrá definícia je stručná, jednoznačná, obsahuje jeden všeobecnejší termín a k nemu špecifickú charakteristiku. So zväčšujúcim sa počtom termínov môžu byť definície jemnejšie, presnejšie. Terminológia je živá, vyvíja sa. Uvádzam niekoľko termínov zo Slovenskej terminologickej databázy, z kategórie IT (Informatika), ktorú editujem a dopĺňam. Niektoré terminologické záznamy sú schválené a zverejnené, niektoré ešte čakajú na schválenie. Terminologická databáza sa môže stať jazykovým zdrojom pre strojový jazykový preklad. Z terminologickej databázy bude mať úžitok široký okruh inštitúcií verejnej správy v SR a EÚ, firmy, verejnosť.

Kľúčové slová
Informatika. Terminológia. Terminologická databáza. Termín. Definícia. Wiki.

1 ÚVOD
Podľa Slovníka cudzích slov **didaktika** je časť pedagogiky, zaoberajúca sa teóriou vyučovania. [1] Didaktika informatiky je teda teóriou vyučovania informatiky. Význam slov (pojmov) v texte je závislý od kontextu. Čo je informatika? V kontexte tohto úvodu **informatika** je študijný predmet alebo študijný odbor. Pojmy sú základné prvky obsahu vzdelávania, skúmaného didaktikou. [2] V súlade s didaktickými zásadami pojmy by mali byť jednoduché, jednoznačné, zrozumiteľné. Obsah vzdelávania prepchatý novými pojmami a ich definíciami je nudný, ťažký, nezáživný. Dajú sa nové pojmy a ich definície vynechať?

Didaktická zásada aktivity, uplatnená najmä v aktivizačných metódach pri napĺňaní špecifických vzdelávacích cieľov, presúva prípadnú formuláciu definície pojmu z učiteľa na žiaka/študenta, ktorú učiteľ iba koriguje. Obsah vzdelávania vo forme učebného textu a ďalších materiálov môže mať redukovanú sadu (nových) pojmov na nevyhnutné minimum, ak žiak/študent aktívne konštruuje svoje poznanie (metódou konštrukcionizmu).

Didaktika informatiky by preto mala určiť, aká **terminológia** (množina odborných pojmov) v informatike sa má používať. Z celej množiny pojmov sa vyberie podmnožina základných pojmov, formálne vymenovaných v štátnom vzdelávacom programe, alebo rozšírená podmnožina základných pojmov, formálne vymenovaných v školskom vzdelávacom programe. Tento článok sa ale nezaoberá didaktickým spracovaním pojmov.

Didaktika informatiky pre svoj budúci rozvoj potrebuje referenčný register termínov (odborných pojmov), s definíciami a prekladmi do iných jazykov. Autori učebných textov, metodík, ďalších materiálov a učitelia by mali dôsledne dodržiavať takto nastavenú terminológiu, pretože škola vychováva novú generáciu. Informatika a informačná technológia sa vyvíjajú rýchlo. Bolo by veľmi zlé, keby sa terminológia pre didaktiku informatiky líšila od terminológie pre prax, pre odbornú verejnosť. Referenčný register termínov musí byť preto verejný a dostupný. Je užitočné do procesu tvorby terminológie zapojiť učiteľov a didaktikov, lebo môžu reagovať na jazyk mladej generácie.

2 SLOVENSKÁ TERMINOLOGICKÁ DATABÁZA
Vo viacerých učebniciach a učebných textoch (z informatiky), skriptách, knihách nájdeme krátke zoznamy použitých termínov a k nim definície. Slovníky alebo encyklopédie preberajú definície a preklady z rôznych starších zdrojov, aj zastarané a nesprávne. Takto tvorená a udržiavaná terminológia má nedostatočnú kvalitu.

Môj prvý príspevok k terminológii v informatike bol krátky *IT slovník anglicko-slovenský* pre SŠ zverejnený na webe[1]. Odborná aj laická verejnosť ho používa, napríklad aj pre preklady medzi angličtinou, slovenčinou, češtinou a ruštinou[2]. Moja práca na terminológii v informatike pokračovala externou spoluprácou s Jazykovedným ústavom Ľ. Štúra Slovenskej akadémie vied (JÚĽŠ SAV)[3] a pokračuje naďalej.

1 http://www.shenk.sk/skola/informatika/it-slovnik.pdf

2 Компьютерные термины на словацком языке, http://slovenskyjazyk.info/komputernie-terminy/

3 http://www.juls.savba.sk

ISBN 978-80-557-1533-9, ISSN 2454-051X.

JÚĽŠ SAV prevádzkuje Terminologický portál a Slovenskú terminologickú databázu (obrázok 1). Cieľom tohto projektu je zabezpečiť, aby predovšetkým odborná komunikácia prebiehala jasne a bez nedorozumení. Termíny v databáze sú zatriedené do niekoľkých kategórií pre ľahšie vyhľadávanie.

Obrázok 1:
https://terminologickyportal.sk/wiki/Hlavná_stránka

Terminologický záznam v terminologickej databáze obsahuje ku každému termínu niekoľko povinných a niekoľko nepovinných položiek. Najdôležitejšie položky sú: kategória, definícia, zdroj definície, cudzojazyčný ekvivalent (preklad), oblasť. To umožňuje pracovať s terminologickou databázou aj strojovo.

Oblasť použitia termínu je určená kľúčovým slovom slovníka *EuroVoc (European Vocabulary thesaurus)*[4], ktorý slúži ako referenčný kategorizačný prekladový slovník legislatívy EÚ. Väčším projektom je IATE (*InterActive Terminology for Europe*)[5], terminologická databáza legislatívy, s prekladom pre inštitúcie EÚ a verejnosť. Projekt ELRC (*European Language Resource Coordination*)[6] zastrešuje službu strojového jazykového prekladu pre inštitúcie verejnej správy EÚ. Všetky podobné projekty, či už pre inštitúcie verejnej správy alebo pre firmy a verejnosť (napr. Wikislovník[7], Google prekladač[8]), potrebujú kvalitné lingvistické a terminologické zdroje (gramatiky, slovníky, texty). Slovenská terminologická databáza sa môže stať takým zdrojom. Didaktici, autori učebných textov a učitelia sú dôležitou cieľovou skupinou používateľov i tvorcov terminológie. JÚĽŠ SAV garantuje dostatočnú kvalitu a stabilitu zostavenej terminologickej databázy.

Terminologický portál poskytuje bohatý zoznam slovenských a zahraničných prekladových a výkladových slovníkov[9], aj teoretickej literatúry. Napriek tomu, že slovenská terminológia pre informatiku (alebo výpočtovú techniku, kybernetiku) má dostatočne dlhú históriu, nemáme kvalitné prekladové a výkladové slovníky z tejto oblasti. Obsahujú neaktuálne, nepresné, nesprávne definície. Typický príklad zlej definície:

používateľ – kto niečo používa [3]

Dobrá definícia je stručná, jednoznačná, obsahuje jeden všeobecnejší termín a k nemu špecifickú charakteristiku, ktorá zúži význam všeobecnejšieho termínu. Dobrá definícia neobsahuje slovný základ obsiahnutý v termíne. Avšak niektoré definície nie sú dostatočne presné, ak majú byť stručné. Potom definícia rozšírená o definičný kontext môže obsahovať aj slovný základ obsiahnutý v termíne, ak sa týka len kontextu.

Ak sa nepodarí nájsť dostatočne dobrú definíciu, môže sa existujúca slovenská definícia upraviť alebo cudzojazyčná definícia preložiť a pred zdroj definície sa pripojí slovo *podľa*.

Encyclopaedia Beliana[10] je kvalitnou všeobecnou slovenskou encyklopédiou, ale obsahuje málo termínov z informatiky, tie sú prevzaté z iných slovníkov. Mnohých prekvapí, že Wikipédia je považovaná za akceptovateľný zdroj definície. Wikipédia nie je anarchia. Wikipédia má komunitu editorov, redaktorov, správcov, ktorí dbajú na dodržiavanie zásad[11], najmä citovanie zdrojov a použitie verejnej licencie. Wikipédia a Slovenská terminologická databáza bežia na rovnakom softvéri (redakčnom systéme *wiki*[12]), ktorý umožňuje prepojenia rôznych zdrojov. Registrovaní používatelia Slovenskej terminologickej databázy môžu vytvárať a editovať terminologické záznamy a verejnosť môže čítať zverejnené terminologické záznamy. Obsah je zverejnený s licenciou GNU FDL[13], ak nie je uvedené inak. Najlepšie zverejnenie zdroja je na webe, strojovo spracovateľné.

Presnosť definície termínu závisí aj od veľkosti terminológie (počtu termínov v databáze). Ak máme málo termínov, tak sú definície hrubé. So zväčšujúcim sa počtom termínov môžu byť definície jemnejšie, presnejšie. Terminológia je živá, vyvíja sa.

3 TERMÍNY

V nasledujúcich kapitolách uvádzam niekoľko termínov zo Slovenskej terminologickej databázy, z kategórie IT[14] (Informatika), ktorú editujem a dopĺňam. Niektoré terminologické záznamy sú schválené a zverejnené, niektoré ešte čakajú na schválenie kolektívom JÚĽŠ SAV. Následne budú zapracované pripomienky recenzentov a verejnosti. Formát terminologického záznamu pre účel tohto článku je zjednodušený:

slovenský termín (*český ekvivalent; anglický ekvivalent*) – definícia; zdroj; doplňujúce informácie

Najprv dôležité termíny pre informatiku a didaktiku informatiky. Potom problematické, nesprávne a zastarané termíny, termíny nové z pohľadu didaktiky. Na záver skratky a jednotky.

3.1 Dôležité termíny

informatika (*informatika*; *informatics*) – veda o získavaní, ukladaní a automatizovanom (počítačovom) spracovaní digitálnych dát; podľa: [3]; synonymum: **počítačová veda** (*počítačová věda*; *computer science*)

informatizácia (*informatizace*; *informatization, informatisation*) – zavádzanie alebo zavedenie počítačovej techniky a automatizovaného spracovania dát do praxe [4]

informatik (*informatik*; *informatician, informaticist*) – odborník v automatizovanom spracovaní digitálnych dát a v práci s výpočtovou technikou; podľa: [3]; synonymum: **itečkár**;

4 https://publications.europa.eu/sk/web/eu-vocabularies

5 https://iate.europa.eu, http://cdt.europa.eu/sk/iate-sk

6 http://www.lr-coordination.eu

7 https://sk.wiktionary.org/wiki/Wikislovník:Hlavná_stránka

8 https://translate.google.sk

9 https://terminologickyportal.sk/slovniky.html, https://terminologickyportal.sk/odkazy.html

10 https://sk.wikipedia.org/wiki/Encyclopaedia_Beliana, http://www.encyclopaedia.sk/index.php?option=com_content&view=article&id=8:projekt-encyclopaedia-beliana&catid=6:bel-matej&Itemid=9

11 https://sk.wikipedia.org/wiki/Wikipédia:Zásady

12 https://sk.wikipedia.org/wiki/Wiki

13 https://www.gnu.org/licenses/fdl.html

14 https://terminologickyportal.sk/wiki/Kategória:It

ISBN 978-80-557-1533-9, ISSN 2454-051X.

poznámka: V angličtine sa používa radšej konkrétna špecializácia, napríklad: IT technik (*IT technician*), programátor (*programmer*), softvérový inžinier (*software engineer*, *software developer*), počítačový inžinier (*computer engineer*), administrátor (*administrator*, *admin*).

informačná technológia (*informační technologie*; *information technology*) – priemyselný odbor zaoberajúci sa výrobou počítačov a automatizovaným (počítačovým) spracovaním dát; podľa: cs.wikipedia.org; synonymum: **IT** (*IT*; *IT*)

informačná a komunikačná technológia (*informační a komunikační technologie*; *information and communications technology*) – priemyselný odbor zaoberajúci sa výrobou počítačov a prenosových zariadení, automatizovaným (počítačovým) spracovaním dát a prenosom dát; podľa: sk.wikipedia.org; synonymum: **IKT** (*ICT*; *ICT*); poznámka: IT (informačná technológia) a IKT (informačná a komunikačná technológia) možno používať ako synonymá. Termín IT je viac používaný priemyslom a termín IKT je viac používaný vo verejných politikách, v legislatíve a v školstve. Termín IKT zdôrazňuje integráciu počítačovej techniky a telekomunikácií.

digitálna gramotnosť (*digitální gramotnost*; *digital literacy*) – schopnosť efektívne a bezpečne používať výpočtovú techniku v bežnom živote, zahŕňa základné znalosti a zručnosti z informatiky; podľa: cs.wikipedia.org; synonymum: **počítačová gramotnosť** (*počítačová gramotnost*; *computer literacy*); poznámka: Presnejší obsah počítačovej gramotnosti je vymenovaný napríklad v sylabe ECDL[15].

algoritmické myslenie (*algoritmické myšlení*; *computational thinking*) – schopnosť nájsť alebo vytvoriť vhodné postupy a metódy na automatizované riešenie danej úlohy, najmä pomocou počítača; podľa: slovnik-cizich-slov.abc.cz; synonymum: **výpočtové myslenie**, **informatické myslenie** (*výpočetní myšlení*; *computational thinking*); poznámka: Slovenská (a česká) terminológia v didaktike informatiky sa vyvíja mierne odlišne od anglickej terminológie. Cieľom algoritmického (výpočtového) myslenia (*computational thinking*[16]) je použitie počítača (*computing*) pri automatizovanom riešení úloh (*problem solving*). Informatické myslenie je zriedkavo používaným synonymom k algoritmickému mysleniu. Charakteristické črty algoritmického myslenia: dekompozícia, nájdenie vzoru alebo reprezentácie dát, zovšeobecnenie a abstrakcia, použitie algoritmov.

algoritmizácia (*algoritmizace*; *algorithmization*, *algorithmisation*) – zostavovanie presného a logicky jednoznačného predpisu s postupnosťou krokov na riešenie úlohy; podľa: sk.wikipedia.org

počítačové inžinierstvo (*počítačové inženýrství*; *computer engineering*) – technický odbor zaoberajúci sa návrhom a vývojom zariadení a metód pre automatizované spracovanie dát, kombinujúci informatiku a elektrotechniku; podľa: en.wikipedia.org

dátová veda (*datová věda*; *data science*) – interdisciplinárny odbor zaoberajúci sa získavaním znalostí z údajov, kombinujúci informatiku, analýzu, štatistiku, strojové učenie; podľa: en.wikipedia.org

dáta (*data*; *data*) – postupnosť znakov (symbolov) určená na spracovanie počítačom; podľa: en.wikipedia.org; synonymum: **údaje** (*údaje*; *data*); poznámka: Digitálne dáta tvoria postupnosť číslic. Ak sa použije číselný základ 2, tak ide o číslice 0 a 1.

informácia[(1)] (*informace*; *information*) – správa, údaj, obsah komunikácie; podľa: [5]

informácia[(2)] (*informace*; *information*) – číselné vyjadrenie množstva dát, udáva sa s jednotkou bit; podľa: [1]

teória informácie (*teorie informace*; *information theory*) – veda o meraní, kódovaní, prenose, ukladaní, spracovaní a využívaní digitálnych dát; zdroj: sk.wikipedia.org; synonymum: **teória informácií**, **informačná teória** (*teorie informací*, *informační teorie*; *information theory*)

metadáta (*metadata*; *metadata*) – štruktúrované údaje o primárnych údajoch; podľa: sk.wikipedia.org; synonymum: **metaúdaje** (*metaúdaje*; *metadata*)

znalosť (*znalost*; *knowledge*) – komplexná informácia, súhrn súvisiacich aplikovaných poznatkov; podľa: sk.wikipedia.org

ontológia (*ontologie*; *ontology*) – formálna reprezentácia jazyka, zostavená z pojmov, definícií a väzieb; podľa: cs.wikipedia.org; synonymum: **jazyková doména** (*jazyková domena*; *language domain*); poznámka: Ontológia sa používa ako dátový model na uloženie znalostí a zatriedenie do kategórií, konceptov.

informačný systém[(1)] (*informační systém*; *information system*) – komplexný spôsob správy údajov, ktorý sa skladá z výpočtovej techniky, procesov, organizačnej štruktúry a ľudí; podľa: sk.wikipedia.org

informačný systém[(2)] (*informační systém*; *information system*) – počítačový program na automatizovanú správu údajov podľa danej organizačnej štruktúry a daných procesov; podľa: sk.wikipedia.org

počítač (*počítač*; *computer*) – stroj na automatizované spracovanie dát; podľa: [3]; poznámka: Počítač môže byť digitálny aj analógový.

server[(1)] (*server*; *server*) – počítač, ktorý riadi počítačovú sieť a poskytuje služby súčasne viacerým používateľom; zdroj: [3], sk.wikipedia.org; synonymum: **hardvérový server** (*hardwarový server*; *hardware server*)

server[(2)] (*server*; *server*) – softvér, ktorý poskytuje službu klientom v počítačovej sieti; podľa: sk.wikipedia.org; synonymum: **softvérový server**, **serverový softvér** (*softwarový server*, *serverový software; software server*, *server software*)

výpočtová technika (*výpočetní technika*; *computers*, *computer technology*) – súhrn zariadení na automatizované spracovanie dát, zahŕňa hardvér aj softvér; podľa: sk.wikipedia.org; synonymum: **počítačová technika** (*počítačová technika*; *computers*, *computer technology*); poznámka: **Digitálna technika** je zriedkavo používaným synonymom k výpočtovej technike.

3.2 Problematické termíny

K niektorým bežne používaným termínom sa dajú nájsť rôzne definície i rôzne preklady, a preto sú problematické z pohľadu didaktiky informatiky.

pamäť (*paměť*; *memory*) – elektronické zariadenie na uchovávanie dát v počítači; podľa: sk.wikipedia.org; poznámka: Pamäť počítača možno rozdeliť podľa funkcie na primárnu (internú, vnútornú) a sekundárnu (externú, vonkajšiu). Pamäť počítača možno rozdeliť podľa použitia na ROM a RWM. RWM možno rozdeliť podľa prístupu na RAM, SAM, CAM.

15 ECDL, https://www.ecdl.sk

16 https://en.wikipedia.org/wiki/Computational_thinking

ISBN 978-80-557-1533-9, ISSN 2454-051X.

súbor (*soubor*; *file*) – pomenované zoskupenie dát, uložené na disku počítača; podľa: sk.wikipedia.org

priečinok (*složka*; *folder, directory*) – špeciálny typ súboru, ktorý slúži ako pomenovaný zoznam iných súborov; podľa: sk.wikipedia.org; synonymum: **adresár**[(1)] (*adresář*; *directory*); poznámka: **Zložka** je synonymum, v tomto význame nesprávne.

adresár[(2)] (*adresář*; *address book, contacts, telephone directory, yellow pages*) – abecedne usporiadaný zoznam adries [6]

cesta (*cesta*; *path*) – znakový reťazec obsahujúci postupnosť priečinkov v strome súborového systému, oddelených špeciálnym znakom; podľa: en.wikipedia.org; poznámka: Cesta udáva umiestnenie súboru v súborovom systéme. Absolútna cesta začína v koreni súborového systému, relatívna cesta začína v inom priečinku. Cesta sa môže končiť súborom.

algoritmus (*algoritmus*; *algorithm*) – presný a logicky jednoznačný predpis s postupnosťou krokov na riešenie úloh rovnakého typu; podľa: [7]; poznámka: Niektoré definície obsahujú obmedzenie na konečnosť algoritmu (konečnosť postupnosti krokov), čo spôsobuje nekonzistentnosť definícií príbuzných termínov (napr. program). Preto táto definícia pripúšťa aj nekonečnú rekurzívne spočítateľnú postupnosť krokov.

program[(1)] (*program*; *program*) – úplná postupnosť inštrukcií alebo príkazov v určitom jazyku na riešenie danej výpočtovej úlohy; podľa: [7]; kontext: Program je algoritmus napísaný v programovacom jazyku.

program[(2)] (*program*; *program*) – súbor alebo balík súvisiacich súborov, určený na vykonávanie danej systémovej alebo aplikačnej úlohy v počítači; podľa: sk.wikipedia.org; synonymum: **spustiteľný program**, **vykonateľný program**, **spustiteľný súbor**, **vykonateľný súbor** (*spustitelný program, vykonatelný program, spustitelný soubor, vykonatelny soubor*; *executable program, executable file, executable, binary*)

proces[(1)] (*proces*; *process*) – bežiaca inštancia programu; zdroj: sk.wikipedia.org; poznámka: Proces je uložený v hlavnej pamäti počítača a vykonávaný procesorom. Proces môže byť rozdelený na vlákna.

proces[(2)] (*proces*; *process*) – plynulý priebeh deja [3]

strojový kód (*strojový kód*; *machine code*) – postupnosť inštrukcií vykonateľných procesorom; podľa: sk.wikipedia.org; synonymum: **vykonateľný kód**, **vykonateľný program** (*vykonatelný kód, vykonatelný program*; *executable code*)

inštrukcia (*instrukce*; *instruction*) – elementárny príkaz priamo vykonateľný procesorom; podľa: cs.wikipedia.org

zdrojový kód (*zdrojový kód*; *source code*) – program napísaný v programovacom jazyku; podľa: sk.wikipedia.org

programovací jazyk (*programovací jazyk*; *programming language*) – sústava symbolov a pravidiel na vyjadrenie jednotlivých operácií a ich postupnosti pri spracúvaní dát [7]

programovanie (*programování*; *programming*) – proces vytvárania softvéru, zahŕňajúci analýzu výpočtovej úlohy, návrh algoritmu, zápis v programovacom jazyku, testovanie, údržbu a tvorbu dokumentácie; podľa: sk.wikipedia.org

programátor (*programátor*; *programmer*) – špecialista, ktorý vytvára softvér; podľa: [7]; synonymum: **softvérový vývojár**, **softvérový inžinier** (*softwarový vývojář, softwarový inženýr*; *software developer, software engineer*)

kodér (*kodér*; *coder*) – špecialista, ktorý vytvára program v programovacom jazyku; podľa: [1]; synonymum: **kódovač** (*kodér*; *coder*); poznámka: V súčasnom slovenskom jazyku sa vyskytuje aj synonymum v tvare kóder, ktorý má bližšie k anglickému originálu. Variant kodér má bližšie k českému prekladu.

prenosová rýchlosť (*přenosová rychlost*; *information bit rate, data bit rate, bit rate, bitrate*) – množstvo informácie alebo dát, ktoré sa prenesie cez komunikačný kanál za jednotku času; podľa: sk.wikipedia.org; poznámka: Jednotka prenosovej rýchlosti je b/s, s násobkami podľa SI (k = 1000). Prenosová rýchlosť (*information bit rate, data bit rate, bit rate, bitrate*) môže byť: maximálna teoretická prenosová rýchlosť (šírka prenosového pásma, prenosová kapacita, *bandwidth*), maximálna skutočná prenosová rýchlosť (*peak bitrate, peak rate*), priemerná skutočná prenosová rýchlosť (priepustnosť prenosového kanála, *throughput*), efektívna prenosová rýchlosť (užitočná prenosová rýchlosť, prenosový výkon, *data transfer rate, goodput*).

elektronická pošta (*elektronická pošta*; *electronic mail*) – služba počítačovej siete zabezpečujúca posielanie predovšetkým textových správ; podľa: sk.wikipedia.org; synonymum: **e-mail** (*e-mail*; *e-mail*); poznámka: Nesprávne používané synonymum v tomto význame: **email** – povrchová náterová hmota.

nevyžiadaná pošta (*nevyžádaná pošta*; *unsolicited mail*) – každá pošta, ktorá nie je očakávaná príjemcom; užší význam má termín **spam** (*spam*; *spam, e-mail spam, junk e-mail*) – hromadne rozosielaná nevyžiadaná elektronická správa, obyčajne so zámerom šíriť reklamu alebo škodlivý softvér; podľa: [1]

tabuľkový editor (*tabulkový editor*; *table editor*) – počítačový program na štruktúrovanie textových alebo číselných dát do riadkov a stĺpcov; podľa: cs.wikipedia.org

tabuľkový procesor (*tabulkový procesor*; *spreadsheet*) – počítačový program na spracovanie dát, štruktúrovaných do riadkov, stĺpcov a hárkov; podľa: cs.wikipedia.org; synonymum: **tabuľkový kalkulátor** (*tabulkový kalkulátor*; *spreadsheet*); poznámka: Tabuľkový editor iba upravuje dáta vo forme textovej alebo číselnej tabuľky. Tabuľkový procesor spracúva dáta v tabuľke, napríklad vypočíta a zobrazí hodnotu vzorca.

textový editor (*textový editor*; *text editor*) – počítačový program na vytváranie a úpravu znakových súborov; podľa: sk.wikipedia.org; poznámka: Textový editor upravuje čistý text, bez štýlu, ako postupnosť znakov.

textový procesor (*textový procesor*; *word processor*) – počítačový program na vytváranie a úpravu znakových dokumentov so štýlom, alebo na generovanie znakových dokumentov zo zdrojového kódu v nejakom značkovacom jazyku; podľa: sk.wikipedia.org; poznámka: Textový procesor upravuje textový dokument so štýlom (formátovaný text).

3.3 Nesprávne a zastarané termíny

Excel alebo **excel** ako tabuľka – nesprávny termín; správny je: **tabuľka**, **tabuľkový dokument**; poznámka: Ak ide o aplikačný softvér a nie je to konkrétne MS Excel, má sa používať všeobecný termín: **tabuľkový editor**, **tabuľkový procesor**, **tabuľkový kalkulátor**. Podobne aj: **text**, **textový dokument**, **textový editor**, **textový procesor**.

webstránka – nesprávny termín; správny je: **webová stránka** (*webová stránka*; *web page, webpage*) – hypertextový dokument služby WWW uložený na webovom serveri; podľa:

ISBN 978-80-557-1533-9, ISSN 2454-051X.

sk.wikipedia.org; príbuzný termín: **webové sídlo** (*webové sídlo*; *web site, website*) – skupina súvisiacich hypertextových dokumentov, zvyčajne v rámci jednej internetovej domény; podľa: sk.wikipedia.org; poznámka: Webové sídlo obyčajne pozostáva z viacerých webových stránok.

užívateľ – nesprávny termín v informatike; správny je: **používateľ** (*uživatel*; *user*) – človek alebo agent (stroj), ktorý pracuje s počítačom, programom, službou; podľa: sk.wikipedia.org; poznámka: Užívateľ spotrebúva alebo opotrebúva niečo.

rozhranie – nesprávny termín v informatike; rozhranie alebo hranica delí niečo na časti; v informatike potrebujeme vyjadriť odlišný až opačný význam: **prostredie** (*prostředí, rozhraní*; *environment, interface*) – súhrn podmienok, v ktorých niečo existuje, v ktorých sa niečo koná [8]; **spojenie** (*spojení*; *connection*) – styk, kontakt [3]; synonymum: prepojenie, pripojenie

používateľské rozhranie – nesprávny termín; správny je: **používateľské prostredie** (*uživatelské prostředí, uživatelské rozhraní*; *user interface*); synonymum: UI

aplikačné programovacie rozhranie – nesprávny termín; správny je: **aplikačná programovacia knižnica** (*aplikační programovací rozhraní, rozhraní pro programovaní aplikací*; *application programming interface*) – knižnica funkcií, protokolov a nástrojov na programovanie aplikácií; podľa: sk.wikipedia.org; synonymum: API

zdieľať (*sdílet*; *share*) – zastaraný a nejednoznačný termín, treba ho nahradiť vhodným synonymom podľa kontextu; prvý význam: spoločne niečo mať, mať účasť, súhlasiť, pozorovať, kontrolovať, podieľať sa, využívať, spolupracovať, využiť, používať, použiť; podľa: [9]; druhý význam: zdeliť sa, rozdeliť sa, podeliť sa, deliť sa, zdôverovať sa, poskytnúť, sprístupniť [9]

zdieľaný disk – zastaraný termín; správny je: **spoločný disk** (*sdílený disk, sdílený prostředek*; *share, shared disk, shared resource*) – úložisko dát sprístupnené iným používateľom alebo iným počítačom v počítačovej sieti; podľa: en.wikipedia.org; synonymum: **spoločný priečinok** (*sdílená složka, sdílený adresář*; *shared folder*), **sieťový disk** (*síťový disk*; *network disk, network share*)

USB kľúč (*USB klíč*; *USB key*) – nesprávny termín, kľúčom sa zamyká/odomyká, šifruje/dešifruje; správny termín: **USB disk** (*USB flash disk*; *USB flash disk, USB flash drive*) – prenosné prepisovateľné pamäťové médium s kapacitou v GiB, ukončené USB konektorom; podľa: sk.wikipedia.org

operačná pamäť (*operační paměť*; *main memory*) – nesprávny termín; správny je: **hlavná pamäť** (*hlavní paměť*; *main memory*) – elektronické zariadenie na krátkodobé uchovávanie dát v počítači, priamo adresovateľné programom, obvykle pripojené k procesoru zbernicou; podľa: sk.wikipedia.org; poznámka: Pamäť počítača, ktorá je nazývaná hlavná podľa jej funkcie, je zároveň nazývaná RAM podľa použitia a podľa technológie.

technické vybavenie počítača, **technické prostriedky počítača** – zastaraný termín; máme ustálený termín: **hardvér** (*hardware*; *hardware*) – konštrukčné mechanické a elektronické riešenie počítača a jeho častí; podľa: [4]

programové vybavenie počítača, **programové prostriedky počítača** – zastaraný termín; máme ustálený termín: **softvér** (*software*; *software*) – súhrn všetkých programov v počítači; podľa: sk.wikipedia.org

východzí – nesprávny termín; správny je: **východiskový** (*výchozí*; *default*) – počiatočný, základný, úvodný, prednastavený, predvolený [8]

3.4 Nové termíny vo vzdelávacom obsahu

Nasledujúce termíny z informatiky nie sú nové celkom, ale môžu byť nové v školskom vzdelávacom obsahu. Súvisia s novými trendmi, novou technológiou, novými predpismi.

dataset (*dataset*; *dataset, data set*) – kolekcia štruktúrovaných strojovo spracovateľných údajov, obyčajne v štandardizovanom formáte; podľa: cs.wikipedia.org

otvorené dáta (*otevřená data*; *open data*) – štruktúrované strojovo spracovateľné údaje, zverejnené s verejnou licenciou, obyčajne v štandardizovanom formáte; podľa: cs.wikipedia.org; synonymum: **otvorené údaje**

prepojené dáta (*propojená data*; *linked data*) – distribuované štruktúrované strojovo spracovateľné údaje, obyčajne v štandardizovanom formáte, zviazané vhodnou metódou alebo aplikačným komunikačným protokolom; podľa: en.wikipedia.org

veľké dáta (*velká data*; *big data*) – kolekcia údajov, ktorá je príliš rozsiahla alebo komplexná pre bežné spracovanie; podľa: cs.wikipedia.org

virtualizácia (*virtualizace*; *virtualisation, virtualization*) – technológia na vytvorenie softvérového počítača a prevádzkovanie počítačovej služby na softvérovom počítači, nie na hardvéri; podľa: sk.wikipedia.org; poznámka: Z pohľadu technológie virtualizácie rozlišujeme simuláciu, emuláciu, preklad inštrukcií a hardvérovú virtualizáciu. Hardvérovou virtualizáciou možno vyrobiť virtuálny stroj alebo kontajner. Z pohľadu používateľa rozlišujeme virtualizáciu hardvéru, virtualizáciu pracovnej plochy, virtualizáciu aplikácie. Do virtualizácie hardvéru možno zaradiť aj virtualizovanú počítačovú sieť.

rozšírená realita (*rozšířená realita*; *augmented reality*) – priamy alebo nepriamy pohľad na fyzicky skutočné prostredie, ktorého objekty sú zmyslovými podnetmi obohatené o dodatočné relevantné informácie; podľa: sk.wikipedia.org; synonymum: **obohatená realita** (*obohacená realita*; *augmented reality*)

otvorený softvér (*otevřený software*; *open software, open source software, open-source software, opensource software*) – počítačový program ako autorské dielo s verejnou licenciou podľa OSI; podľa: en.wikipedia.org; synonymum: OSS; poznámka: Použijú sa verejné licencie vyhovujúce kritériám OSI (*Open Source Initiative*)[17].

slobodný softvér (*svobodný software*; *free software, free and open source software, free/libre and open source software*) – počítačový program ako autorské dielo s verejnou licenciou poskytujúcou 4 slobody podľa FSF; podľa: sk.wikipedia.org; synonymum: FOSS, FLOSS; kontext: Princípy slobody podľa FSF[18] umožňujú: spustiť program na akýkoľvek účel, študovať a upravovať program, kopírovať program, vylepšiť program a zverejniť ho, aby bol na úžitok celej spoločnosti.

príspevkový softvér (*shareware*; *shareware*) – autorské dielo programátora, ktoré možno najprv vyskúšať a po zaplatení používať; podľa: sk.wikipedia.org

17 OSI (*Open Source Initiative*), https://opensource.org/licenses

18 FSF (*Free Software Foundation*), https://www.fsf.org

ISBN 978-80-557-1533-9, ISSN 2454-051X.

zverejniť (*zverejnit*; *publish*) – sprístupniť všetkým; podľa: [3]; poznámka: Právny základ zverejnenia je definovaný v §6, § 75, § 76 zákona 185/2015 Z. z. (autorský zákon).

webinár (*webinář*; *webinar, web seminar*) – interaktívna komunikácia viacerých používateľov cez internet s multimediálnym a hypertextovým obsahom; podľa: cs.wikipedia.org

hashtag (*hashtag*; *hashtag*) – slovo alebo fráza začínajúca znakom #, ktoré sa používa ako hypertextový odkaz na slovo, frázu alebo nadpis v dokumente; podľa: sk.wikipedia.org; poznámka: Tento termín je neologizmus, nemáme slovenský preklad.

emotikon (*emotikon*; *emoticon*) – grafický symbol alebo obrázok vyjadrujúci pocit, náladu pisateľa používaný v písomnej elektronickej komunikácii [6]; synonymum: **smajlík** (*smajlík*; *smiley*)

elektronický podpis (*elektronický podpis*; *electronic signature, e-signature*) – informácia pripojená k elektronickému dokumentu, ktorá identifikuje podpisovateľa (fyzickú osobu), umožňuje overiť pravosť podpísaného dokumentu a platí, že iba podpisovateľ ho môže efektívne vyhotoviť; podľa: zákon č. 215/2002 Z. z. o elektronickom podpise; synonymum: **digitálny podpis** (*digitální podpis*; *digital signature*); poznámka: V presnejšom význame elektronický podpis je digitálny podpis vyhotovený podľa legislatívy. Digitálny podpis nie je viazaný legislatívou.

heker (*hacker*; *hacker*) – osoba, ktorá neoprávnene vniká do chránených počítačových systémov, programov, údajov; podľa: [4]; synonymum: **haker**, **hacker**

softvérové pirátstvo (*softwarové pirátství*; *software piracy*) – nelegálne používanie, editovanie, kopírovanie, rozširovanie počítačových programov; podľa: [3], sk.wikipedia.org

škodlivý softvér (*škodlivý software, malware*; *malware, malicious software*) – počítačový program, ktorý zneužíva dostupnosť služieb, alebo spôsobuje škodu na službách, dátach, súkromí, inom softvéri, alebo hardvéri; podľa: sk.wikipedia.org; synonymum: **malvér**; poznámka: Doslovný preklad je zákerný softvér, škodlivosť je však dôležitejšia vlastnosť ako pôvodný úmysel škodiť.

špionážny softvér (*špionážní software, špehovací software*; *spyware*) – škodlivý počítačový program, ktorý bez vedomia používateľa zbiera citlivé dáta z počítača a odosiela ich inej osobe; podľa: sk.wikipedia.org

3.5 Skratky

Namiesto niektorých termínov v plnom znení sa často používajú skratky. Keďže originálna terminológia v informatike je prevažne anglická, sú zaužívané anglické skratky, a to aj v slovenskej terminológii. Slovenský preklad skratiek môže spôsobiť zmätok, ak slovenská skratka je totožná s inou anglickou skratkou. V takom prípade uprednostňujem použitie anglickej skratky.

AI – umelá inteligencia (*umělá inteligence*; *artificial intelligence)* – schopnosť počítača alebo systému napodobňovať niektoré funkcie ľudského mozgu, rozumové schopnosti a správanie sa človeka; podľa: [1]; poznámka: Slovenská skratka UI je totožná s nasledujúcou, preto ju neodporúčam používať.

UI – používateľské prostredie (*uživatelské prostředí, uživatelské rozhraní*; *user interface*); príbuzný termín: **UX – používateľský zážitok** (*uživatelský zážitek*; *user experience*)

URL (*URL, internetová adresa*; *URL, uniform resource locator*) – univerzálny formát adresy internetového zdroja; zdroj: sk.wikipedia.org; príbuzné termíny: **URI**, **URN**

DB – databáza (*datbáze*; *database*) a **DBMS – databázový riadiaci systém** (*databázový řídící systém, systém řízení báze dat, SŘBD*; *database management system*) tvoria **DBS – databázový systém** (*databázový systém*; *database system*)

3.6 Jednotky

bit (*bit*; *bit, binary digit*) – základná jednotka informácie, ktorá má vždy jednu z dvoch možných logických hodnôt; zdroj: sk.wikipedia.org; synonymum: **b**, **dvojková číslica**; poznámka: Bit nie je v sústave jednotiek SI. Násobky pre číselný základ 2 sú definované v norme IEC 60027-2[19], napr. Ki = 1024.

bajt (*bajt*; *byte*) – jednotka informácie, pozostávajúca z 8 bitov; synonymum: **B**; zdroj: sk.wikipedia.sk; poznámka: Bajt nie je v sústave jednotiek SI. Násobky pre číselný základ 2 sú definované v norme IEC 60027-2, napr. Ki = 1024.

4 ZÁVER

V čase písania tohto článku bolo v Slovenskej terminologickej databáze, v kategórii IT (Informatika) 635 termínov. Terminológia je živá, vyvíja sa. Berme to ako výzvu na spoluprácu. Každý môže poslať návrhy na korekcie existujúcich termínov a návrhy na nové termíny. Návrhy budú spracované mnou a kolektívom JÚĽŠ SAV.

Didaktika informatiky pre svoj budúci rozvoj potrebuje referenčný register termínov. Terminologická databáza sa môže stať dobrým jazykovým zdrojom pre strojový jazykový preklad, pre štandardizáciu informatizácie verejnej správy, vzdelávanie, vedu a prax. Z terminologickej databázy bude mať úžitok široký okruh inštitúcií verejnej správy v SR a EÚ, firmy, verejnosť.

Ďalšia práca: didaktické spracovanie pojmov do konceptov.

BIBLIOGRAFICKÉ ODKAZY

[1] PETRÁČKOVÁ, V., KRAUS, J. a kol. *Slovník cudzích slov (akademický)*. Bratislava: SPN – Mladé letá, 2005. ISBN 80-10-00381-6. http://slovniky.korpus.sk

[2] PETLÁK, E. *Všeobecná didaktika*. Bratislava: IRIS, 1997. ISBN 80-88778-49-2.

[3] KAČALA, J., PISÁRČIKOVÁ, M., POVAŽAJ, M. (red.) *Krátky slovník slovenského jazyka.* Bratislava: Veda, 2003. ISBN 80-224-0750-X. http://slovniky.korpus.sk

[4] JAROŠOVÁ, A., BUZÁSSYOVÁ, K. (red.) *Slovník súčasného slovenského jazyka H – L.* Bratislava: Veda, 2001. ISBN 978-80-224-1172-1. http://slovniky.korpus.sk

[5] REIFOVÁ, I. *Slovník mediální komunikace.* Praha: Portál, 2004.

[6] BUZÁSSYOVÁ, K., JAROŠOVÁ, A. (red.) *Slovník súčasného slovenského jazyka A – G.* Bratislava: Veda, 2006. ISBN 978-80-224-0932-4. http://slovniky.korpus.sk

[7] ŠALING, S., IVANOVÁ-ŠALINGOVÁ, M., MANÍKOVÁ, Z. *Veľký slovník cudzích slov.* Veľký Šariš: SAMO – AAMM, 1997. ISBN 80-967524-0-5.

[8] PISÁRČIKOVÁ, M. (red.) *Synonymický slovník slovenčiny.* Bratislava: Veda, 2004. ISBN 80-224-0801-8. http://slovniky.korpus.sk

[9] PECIAR, Š. (red.) *Slovník slovenského jazyka.* Bratislava: Vydavateľstvo SAV, 1965. http://slovniky.korpus.sk

19 https://sk.wikipedia.org/wiki/IEC_60027-2

Ochrana osobních údajů podle GDPR v soutěži Bobřík informatiky

Personal data protection according to GDPR in the Bebras contest

Václav Šimandl
KIN PF JU
Jeronýmova 10
371 15 České Budějovice
Česká republika
simandl@pf.jcu.cz

ABSTRACT

The aim of this article is to describe the adoption of measures according to General Data Protection Regulation (GDPR) in the Bebras contest and its impact on this competition. Until now we as the organizers of the contest did not have the explicit permission of neither school coordinators of the contest nor contestants from the last years to process their personal data. Therefore, the measures had to be adopted to process these data according to GDPR. Furthermore, we had to solve the issue of contestants´ data in the current year of the contest. We were looking into a problem how to apply students who do not want to provide their personal data or they cannot do that without their parents´ permission due to their age. Apart from that, we had to decide which data we should require from the contestants who want to be stated in official lists of results. In this article we concentrate on concrete solutions which we have suggested, implemented and put into practice. Furthermore, we describe the complications and advantages of these solutions.

Keywords

GDPR. Bebras contest. Personal data.

ABSTRAKT

Článek se zabývá zaváděním opatření podle Obecného nařízení o ochraně osobních údajů (GDPR) v soutěži Bobřík informatiky a jejich dopady na tuto soutěž. Protože jsme jako organizátoři soutěže doposud neměli explicitní souhlas školních koordinátorů soutěže ani soutěžících v předchozích ročnících soutěže se zpracováním jejich osobních údajů, bylo nutno učinit opatření, aby zpracování údajů těchto osob nebylo v rozporu s GDPR. Následně bylo potřeba vyřešit otázku týkající se údajů soutěžících v aktuálním ročníku soutěže. Byl řešen problém, jakým způsobem by se měli přihlašovat do soutěže žáci, kteří nechtějí nebo vzhledem k věku nemohou bez souhlasu rodičů zadávat své osobní údaje. Kromě toho bylo potřeba určit, které údaje požadovat po soutěžících, kteří chtějí být uvedeni v oficiálních výsledkových listinách. V článku se zabýváme konkrétními řešeními, které jsme navrhli, implementovali a v praxi vyzkoušeli. Dále popisujeme komplikace i výhody, které daná řešení s sebou přinesla.

Klíčová slova

GDPR. Soutěž Bobřík informatiky. Osobní údaje.

1. ÚVOD

Od 25. května 2018 vstoupilo v platnost Obecné nařízení o ochraně osobních údajů (GDPR) [1]. Toto nařízení má, kromě jiného, nemalý dopad také na školy. Že je význam této normy pro školství nezanedbatelný, ukazuje mimo jiné vznik hned dvou materiálů, které byly v loňském roce vytvořeny Ministerstvem školství, mládeže a tělovýchovy České republiky a které obsahují doporučení pro školy, jak dostát požadavkům GDPR na ně kladeným [2, 3].

Aspektem velmi důležitým pro školství je věková hranice, od níž mohou o souhlasu se zpracováním svých osobních údajů rozhodovat jednotlivé osoby samostatně. Tento věk je v GDPR stanoven v rozmezí 13 až 16 let s tím, že jednotlivé státy si ji prostřednictvím adekvátního zákona mohou zpřesnit [4]. V praxi opravdu některé státy zvolili hranici 13 či 14 let, zatímco jiné 15 nebo 16 let [4]; přehled pro vybrané státy ukazuje Tabulka 1. Ačkoliv by v Česku dle návrhu příslušného zákona touto hranicí měl být věk 15 let [5], protože daný zákon doposud nebyl schválen, je touto hranicí prozatím věk 16 let. Tato situace přináší komplikace nejen jednotlivým školám, ale také pořadatelům školních soutěží.

Tabulka 1. Věková hranice, od níž může jedinec udělit souhlas se zpracováním svých osobních údajů, ve vybraných státech Evropské unie (podle [4])

Stát	Věková hranice
Itálie	14 let
Maďarsko	16 let
Německo	16 let
Polsko	16 let
Rakousko	14 let
Slovensko	16 let
Velká Británie	13 let

Ačkoliv je ze strany MŠMT znatelná snaha pomoci školám vyrovnat se s GDPR, podpora, která by byla směřována pořadatelům školních soutěží, je spíše slabá. Organizátorům těchto soutěží tak nezbývá, než se s GDPR vypořádat individuálně. Jelikož to může být poměrně nesnadný úkol, rozhodli jsme se na příkladu soutěže Bobřík informatiky ukázat realizaci konkrétních opatření vedoucí k souladu s GDPR. Cílem tohoto článku je ukázat, jak se s požadavky GDPR vypořádali organizátoři soutěže Bobřík informatiky a jaké dopady měla přijatá opatření na průběh soutěže a tedy i na samotné soutěžící.

2. SOUTĚŽ BOBŘÍK INFORMATIKY

Bobřík informatiky je online soutěž, která probíhá jako jednorázový test přímo ve školách [6]. Soutěž probíhá

v 5 věkových kategorií, které zahrnují žáky od 4. třídy základní školy až po maturitní ročníky [7]; bližší rozdělení je uvedeno v Tabulce 2. Ve většině kategorií soutěž probíhá jednokolově, v nejstarší kategorii Senior je pořádáno ještě ústřední kolo, do nějž postupuje přibližně 30 nejúspěšnějších soutěžících z každého kraje České republiky.

Tabulka 2. Věkové kategorie soutěže Bobřík informatiky (podle [7])

Typ školy	Ročník školy	Kategorie soutěže
Základní škola	4. ročník	Mini
	5. ročník	
	6. ročník	Benjamin
	7. ročník	
	8. ročník	Kadet
	9. ročník	
Střední škola	1. ročník	Junior
	2. ročník	
	3. ročník	Senior
	4. ročník	

Do soutěže se mohou registrovat pouze školy, nikoliv jednotliví soutěžící [8]. Školu v rámci soutěže zastupuje tzv. školní koordinátor soutěže (dále jen koordinátor), kterým je obvykle učitel působící na dané školu. Ten školu registruje, komunikuje s organizátory soutěže a zabezpečuje průběh soutěže na škole [8]. V jeho rámci spravuje seznamy soutěžících z dané školy. V těchto seznamech může provádět opravy údajů zadaných soutěžícími (např. opravit překlepy) a také vyřadit soutěžící, kteří nesoutěžili podle stanovených pravidel (např. podváděli, soutěžili opakovaně, soutěžili bez dozoru učitele, soutěžili v mladší věkové kategorii apod.).

Každý koordinátor při registraci školy musí zadat své jméno a příjmení, emailovou adresu a vyplňuje informace o škole, kterou zastupuje. Z pohledu GDPR je každý koordinátor jednoznačně identifikovatelný a jím zadávané informace je nutné považovat za osobní údaje. Důvodem, proč musí koordinátor zadávat tyto údaje, je zajištění řádného průběhu soutěže na jím spravované škole (zejména možnost být kontaktován organizátory soutěže a transparentnost vůči soutěžícím).

Samotný soutěžící se do soutěže nemusí dopředu registrovat, ale pouze se do soutěže přihlásí těsně před začátkem soutěžení na základě tzv. kódu školy, který je společný pro všechny soutěžící na škole a který mu sdělí koordinátor [9]. Až do loňského ročníku soutěže zadával soutěžící při přihlášení do soutěže své jméno a příjmení, třídu, pohlaví, rok narození a na základě kódu školy byl přiřazen ke škole, za níž soutěžil. Výše uvedené informace byly vyžadovány zejména z důvodu:

- zajištění průběhu soutěže podle pravidel
- tvorby celostátního pořadí soutěžících a výsledkových listin
- tvorby diplomu

Kromě toho byly uvedené informace využívány také k:

- tvorbě anonymních statistik a výzkumů
- případným interním účelům školy (konkrétní použití určoval koordinátor)

Specifickou oblastí bylo zpracování osobních údajů v ústředním kole soutěže v kategorii Senior, v němž byla od soutěžících vyžadována navíc emailová adresa. Uvedené informace byly kromě výše uvedených účelů použity také k přímému oslovení několika nejlepších soutěžících ohledně zaslání diplomů a drobných cen. Údaje cca 10 nejlepších soutěžících byly předávány Ministerstvu školství, mládeže a tělovýchovy v rámci programu Excelence.

Z pohledu GDPR bylo nutné informace zadávané soutěžícím v národním i ústředním kole soutěže považovat za osobní údaje. Jelikož bylo v podstatě nemožné zajistit souhlas zákonných zástupců se zpracováním osobních údajů soutěžících mladších 16 let, bylo nutno vzhledem ke GDPR přehodnotit dosavadní praxi nakládání s osobními údaji soutěžících.

3. ZPŮSOB IMPLEMENTACE OPATŘENÍ NA OCHRANU OSOBNÍCH ÚDAJŮ

Implementace opatření na ochranu osobních údajů v soutěži Bobřík informatiky byla rozdělena do několika fází podle skupin osob, jichž se týkala:

- Aktivní i neaktivní koordinátoři
- Soutěžící v předchozích ročnících soutěže
- Soutěžící v národním kole aktuálního ročníku soutěže
- Soutěžící v ústředním kole aktuálního ročníku soutěže

V každé fázi byly kromě konkrétních technických opatření aktualizován dokument transparentně informující o způsobu zpracování osobních údajů - Prohlášení o ochraně osobních údajů [10]. V následujícím textu popíšeme, jaká konkrétní opatření byla přijata.

3.1 Úpravy týkající se údajů koordinátorů

Základním nedostatkem původního přístupu k osobním údajům koordinátorů bylo nevyžadování explicitního souhlasu se zpracováním osobních údajů během registrace. Předpokládalo se, že koordinátor potvrzením registrace souhlasí se zpracováním svých osobních údajů. Jelikož je tento přístup podle GDPR nedostatečný, bylo do registračního formuláře přidáno zaškrtávací políčko *Souhlasím se zpracováním osobních údajů*, bez jehož zaškrtnutí není možno registraci dokončit. Tím byly požadavky GDPR týkajících se registrace nových koordinátorů naplněny.

Problém nastal v případě stávajících koordinátorů, kteří žádný explicitní souhlas nedali, a tak by bylo jakékoliv zpracování jejich osobních údajů ode dne platnosti GDPR nezákonné. Bylo proto rozhodnuto, že do formuláře sloužícího k editaci údajů koordinátora bude taktéž přidáno zaškrtávací políčko *Souhlasím se zpracováním osobních údajů* a informace o jeho zaškrtnutí se budou ukládat do databáze. Všem stávajícím koordinátorům byl ještě před vstupem GDPR v platnost rozeslán email s informací, že je potřeba na příslušné stránce udělit souhlas se zpracováním svých osobních údajů v rámci soutěže a že pokud nebude souhlas udělen, bude jejich účet smazán, čímž se jimi spravovaná škola nadále nebude moci soutěže účastnit. Účty koordinátorů, kteří do stanovené lhůty souhlas se zpracováním osobních údajů neudělili, byly skutečně odstraněny, čímž došlo k nevratnému vymazání jejich osobních údajů. Informace o škole, kterou koordinátor zastupoval, však vymazány nebyly, neboť jsou uloženy v jiné databázové tabulce. V případě, že se koordinátor znovu registruje (popř. se jako koordinátor registruje jiný učitel dané školy), je možno mu školu znovu přiřadit, aniž by bylo nutno informace o škole znovu vyplňovat.

Aby byl naplněn požadavek GDPR týkající se odvolání souhlasu se zpracováním osobních údajů a dalších práv týkajících se osobních údajů koordinátora, byl vytvořen nový formulář, který je koordinátorům přístupný po přihlášení. Jeho prostřednictvím může

koordinátor uplatnit výše uvedená práva. Zpracování formuláře z technického hlediska funguje tak, že po zvolení příslušného požadavku koordinátorem je na emailovou adresu organizátorů soutěže odeslán email s informací, že daný koordinátor žádá o uplatnění určitého práva. Daný požadavek je následně ručně vyřízen organizátory soutěže.

Jelikož koordinátory jsou obvykle učitelé informatiky, lze platformu soutěže Bobřík informatiky považovat za kanál, jímž lze tyto učitele velmi efektivně oslovit s novinkami z oblasti výuky informatiky (např. s informací o školeních k výuce informatiky nebo o nových učebnicích informatiky). Výše popsaný souhlas se zpracováním osobních údajů se však týká pouze takových akcí, které jsou nezbytné k zajištění průběhu samotné soutěže. Aby bylo možno koordinátorům zasílat informace týkající se výuky informatiky, je k tomu potřeba od nich získat speciální souhlas[1]. K tomuto účelu bylo do registračního formuláře a do formuláře sloužícího k editaci údajů koordinátora přidáno ještě druhé zaškrtávací políčko *Souhlasím se se zasíláním informací týkajících se výuky informatiky*, jehož zaškrtnutím koordinátor indikuje, že mu mohou být takovéto hromadné emaily zasílány. Pokud se koordinátor rozhodne souhlas se zasíláním těchto emailů odvolat, stačí zrušit zaškrtnutí daného políčka v příslušném formuláři.

Osobní údaje koordinátora mohou být ručně vymazány kdykoliv na jeho žádost nebo budou vymazány po třech letech od běhu posledního ročníku soutěže, jehož se jím spravovaná škola účastnila.

3.2 Úpravy týkající se údajů soutěžících v předchozích ročnících soutěže

Základním nedostatkem původního přístupu k osobním údajům soutěžích bylo, stejně jako v případě koordinátorů, nevyžadování explicitního souhlasu se zpracováním osobních údajů. Bylo předpokládáno, že vyplněním požadovaných údajů před přihlášením do soutěže soutěžící se zpracováním svých osobních údajů souhlasí a že se případně o zadání osobních údajů poradil se svými rodiči. Tento přístup je však podle GDPR nedostatečný a jakékoliv zpracování takových osobních údajů (včetně uložení v databázi) by bylo ode dne platnosti GDPR nezákonné. Proto bylo rozhodnuto o anonymizaci veškerých dat o soutěžících v databázi a odstranění listin vítězů z webu soutěže.

K anonymizaci dat v databázi, namísto celkového odstranění veškerých dat o soutěžících, bylo přistoupeno proto, aby bylo možno v budoucnu realizovat anonymní výzkumy nad daty vzniklými během uplynulých ročníků soutěže. Anonymizace dat v databázi proběhla prostým vymazáním jména, příjmení a roku narození soutěžících. Domníváme se, že ze zbylých dat, které soutěžící zadali (tj. pohlaví, škola a třída), nelze jednoznačně identifikovat konkrétní osobu, a tudíž se nejedná o osobní údaje.

3.3 Úpravy týkající se údajů soutěžících v národním kole aktuálního ročníku soutěže

Významným problémem v národním kole aktuálního ročníku (a všech dalších ročnících) soutěže se ukázala obtížnost získat souhlasy zákonných zástupců se zpracováním osobních údajů u soutěžících mladších 16 let. Protože se soutěžící předem do soutěže neregistruje (a tuto možnost jako organizátoři soutěže ani do budoucna nepreferujeme), není možné souhlas zákonných zástupců požadovat online cestou. Přenést odpovědnost za získání souhlasu zákonných zástupců na koordinátory je dle našeho názoru pro koordinátory zbytečně zatěžující a neefektivní. Kromě toho není zájmem organizátorů sbírat osobní údaje soutěžících, ale umožnit soutěžení co nejširšímu okruhu žáků, což se s vyžadováním zadání osobních údajů vylučuje. Z těchto příčin jsme se zabývali důvody, proč jsou osobní údaje soutěžících vlastně vyžadovány. Hlavním důvodem je tvorba celostátního pořadí soutěžících a s tím spojená potřeba zajistit férový průběh soutěže (např. možnost identifikovat a vyřadit soutěžící, kteří podváděli). Dalším důvodem je tvorba diplomu, který slouží jako ocenění za účast v soutěži a na němž je řada osobních údajů (jméno, příjmení, pohlaví, škola) uvedena.

Abychom umožnili soutěžení i těm soutěžícím, které nechtějí (resp. vzhledem k svému věku bez souhlasu zákonných zástupců nemohou) udělit souhlas se zpracováním osobních údajů v rámci soutěže, vytvořili jsme dva typy soutěžících – *oficiální soutěžící* a *anonymní soutěžící*.

Oficiální soutěžící je takový soutěžící, který je jednoznačně identifikovatelný na základě poskytnutých osobních údajů a dal souhlas se zpracováním svých osobních údajů. Těmito údaji jsou jméno, příjmení, pohlaví, ročník školy a škola, za kterou soutěží. Oproti tomu *anonymní soutěžící* je soutěžící, který není identifikovatelný na základě poskytnutých údajů. Tento soutěžící při přihlášení do soutěže na rozdíl od *oficiálních soutěžících* neuvádí své jméno a příjmení, ale přezdívku. Protože přezdívka může být libovolná a kombinace ostatních údajů neumožňuje identifikaci soutěžícího, nepovažujeme zadané údaje za osobní údaje a nepožadujeme od takového soutěžícího souhlas se zpracováním osobních údajů. Jako přezdívku může soutěžící zadat libovolný text, který není organizátory soutěže kontrolován. S ohledem na pravidla, která daná škola přijala v oblasti zpracování osobních údajů žáků, však může požadavky na přezdívku zpřesnit koordinátor, který může např. žákům doporučit (resp. nařídit), aby jako přezdívku zadali své školní uživatelské jméno, studijní číslo apod.[2] Škola pak může seznam svých soutěžících na vlastní zodpovědnost využít k dalším účelům, např. k ocenění žáků, kteří se stali úspěšnými řešiteli soutěže.

Vzhledem k věkové hranici, od níž může souhlas se zpracováním svých osobních údajů udělit samotný soutěžící, je možné jako *oficiální soutěžící* soutěžit pouze ve dvou nejstarších věkových kategoriích - Junior a Senior. V těchto kategoriích se soutěžící starší 16 let (popř. soutěžící mladší 16 let mající souhlas zákonného zástupce se zpracováním svých osobních údajů) může rozhodnout, zda chce být *oficiální soutěžící* nebo *anonymní soutěžící*. Soutěžící mladší 16 let, který nemá souhlas zákonného zástupce se zpracováním svých osobních údajů, by měl v těchto kategoriích soutěžit jako *anonymní soutěžící*. V mladších věkových kategoriích (Mini, Benjamin, Kadet) jsou všichni soutěžící automaticky považováni za *anonymní soutěžící*.

Jelikož *oficiální soutěžící* je jednoznačně identifikovatelný a koordinátor tak může potvrdit, že soutěžil podle pravidel, je tento soutěžící (na rozdíl od *anonymních soutěžících*) uveden

[1] Podle GDPR není možné tyto dva souhlasy spojit v jeden, neboť jedinec má mít právo se rozhodnout, k jakému zpracování svých osobních údajů souhlas udělí a k jakému nikoliv

[2] Takovýto údaj z pohledu soutěže nepovažujeme za osobní údaj, neboť není zřejmé, co znamená a neidentifikuje tak daného soutěžícího. Koordinátor však díky němu může žáka rozpoznat

v celostátním pořadí soutěžících a ve výsledkových listinách[3]. Pokud soutěží v kategorii Senior, má šanci účastnit se ústředního kola soutěže. Lze tak říci, že *oficiální soutěžící* soupeří s ostatními soutěžícími o co nejlepší umístění v soutěži, zatímco *anonymní soutěžící* se snaží překonat jen sám sebe. Další rozdíl mezi typy soutěžících je v obdržení elektronického potvrzení o účasti v soutěži. Zatímco *oficiální soutěžící* získává diplom, kde je uvedeno jeho jméno a příjmení, *anonymní soutěžící* získává pamětní list, na němž je jeho přezdívka. Místo této přezdívky však může být zobrazen libovolný text, který soutěžící zadá před vytvořením pamětního listu[4].

Soutěžící v kategoriích Junior a Senior musí rozhodnutí, zda chce být *anonymním* či *oficiálním soutěžícím*, učinit při přihlášení do soutěže, tj. těsně před započetím soutěžního testu. Protože se některý soutěžící může chtít stát po skončení soutěže *anonymním soutěžícím* namísto *oficiálního* (nebo naopak), je tato změna možná prostřednictvím koordinátora. Ten může v rámci oprav údajů soutěžících ze své školy mimo jiné změnit typ konkrétního soutěžícího z *oficiálního* na *anonymního* (nebo naopak) a zadat následně jeho přezdívku namísto jména a příjmení (resp. jeho jméno a příjmení namísto přezdívky). Předpokládáme, že změny z *oficiálního soutěžícího* na *anonymního* využije soutěžící, který nadále nechce poskytovat své osobní údaje v rámci soutěže. Naopak změny z *anonymního soutěžícího* na *oficiálního* může využít soutěžící, který byl v soutěži úspěšný, avšak vzhledem k tomu, že je veden jako *anonymní soutěžící*, by nebyl uveden ve výsledkových listinách a nemohl by se účastnit ústředního kola soutěže. Tyto změny však koordinátor může provést pouze během omezené doby po skončení soutěže, později je mohou provést pouze organizátoři.

Anonymizování údajů zadaných soutěžícími je plánováno vždy ke konci školního roku, během něhož soutěž proběhla, a je realizováno prostým vymazáním jména a příjmení (resp. přezdívky) z databáze. Pokud by soutěžící chtěl uplatnit své právo odvolat souhlas se zpracováním osobních údajů popř. jiné právo týkající se jeho osobních údajů, může tak učinit sám emailem nebo prostřednictvím koordinátora školy, za níž soutěžil.

3.4 Úpravy týkající se údajů soutěžících v ústředním kole aktuálního ročníku soutěže

V ústředním kole soutěže je problematika zpracování osobních údajů oproti národnímu kolu jednodušší, neboť se soutěží pouze v kategorii Senior. Tato kategorie je určena žákům 3. a 4. ročníků středních škol a lze předpokládat, že všichni soutěžící jsou starší 16 let. Z tohoto důvodu není potřeba vyžadovat souhlas zákonných zástupců se zpracováním osobních údajů soutěžících, ale soutěžící mohou souhlas udělit sami.

V ústředním kole soutěže je dovoleno soutěžit pouze *oficiálním soutěžícím*, neboť je nezbytně nutné zajistit průběh soutěže podle pravidel. Soutěžící zadávají stejné údaje jako v národním kole (tj. jméno, příjmení, pohlaví, ročník školy a školu, za kterou soutěží) a navíc uvádějí rok narození a emailovou adresu. Ačkoliv soutěžící své osobní údaje vyplňují v elektronické podobě v soutěžní aplikaci, souhlas se zpracováním osobních údajů poskytují (na rozdíl od národního kola soutěže) v písemné podobě.

Osobní údaje soutěžících jsou použity k podobným účelům jako u *oficiálních soutěžících* v národním kole soutěže, navíc ale slouží k přímému oslovení několika nejlepších soutěžících ohledně zaslání diplomů a drobných cen. Vybrané osobní údaje (jméno a příjmení, škola, rok narození) přibližně 10 nejlepších soutěžících jsou kromě toho předávány Ministerstvu školství, mládeže a tělovýchovy v rámci programu Excelence.

Pokud by soutěžící chtěl uplatnit své právo odvolat souhlas se zpracováním osobních údajů popř. jiné právo týkající se jeho osobních údajů, může tak učinit sám emailem nebo prostřednictvím koordinátora školy, za níž soutěžil.

4. PŘIJETÍ OPATŘENÍ KOORDINÁTORY A SOUTĚŽÍCÍMI

Výše uvedená opatření na ochranu osobních údajů byla vyzkoušena během roku 2018. Zatímco opatření týkající se koordinátorů a soutěžících v předchozích ročnících soutěže byla použita během května 2018, opatření týkající se soutěžících v národním kole aktuálního ročníku soutěže byla použita v listopadu 2018. V následujícím textu popíšeme zjištění vzešlá z tohoto praktického nasazení. Protože ústřední kolo soutěže doposud neproběhlo, zjištění z tohoto kola v tomto článku neprezentujeme.

4.1 Přijetí opatření koordinátory

Během května 2018 byly koordinátorům rozeslány dva emaily s informací, že se soutěže budou moci účastnit jen ty školy, jejichž koordinátoři nám udělí souhlas se zpracováním svých osobních údajů. Tyto emaily byly zaslány přibližně 1200 koordinátorům, kteří spravovali nějakou školu (přesný počet nelze určit, neboť jsme příslušná data kvůli GDPR vymazali). Na základě emailu potvrdilo souhlas se zpracováním osobních údajů 626 koordinátorů. Ačkoliv se tento počet může zdát poměrně nízký, lze předpokládat, že řada koordinátorů, kteří souhlas neudělili, byla neaktivních (např. na dané škole již nepracují, popř. se soutěže účastnit nechtějí) – v ročníku 2017 se totiž soutěže zúčastnilo 655 škol. Z výše uvedených 626 koordinátorů, kteří udělili souhlas se zpracováním osobních údajů, souhlasilo se zasíláním emailů týkajících se výuky informatiky 520 osob (tedy 83 %).

Jelikož školy, které nespravuje žádný koordinátor, se nemohou soutěže účastnit, řada koordinátorů se registrovala po zrušení svého účtu znovu a následně zažádali organizátory o přiřazení své původní školy. Počet těchto koordinátorů odhadujeme na 105 (jde o koordinátory, kteří se registrovali po vstupu GDPR v platnost, avšak jimi spravované školy byly registrovány dříve, než GDPR vstoupilo v platnost).

K lednu 2019 stoupl počet koordinátorů, kteří spravovali nějakou školu, na 789. Z nich nám udělilo souhlas se zasíláním emailů týkajících se výuky informatiky 675 osob (tedy 86 %). Počet škol, které se zúčastnily posledního ročníku soutěže, byl 648. Tento počet je přibližně stejný jako v ročníku 2017, kdy se soutěže zúčastnilo 655 škol. Lze se tedy domnívat, že GDPR a námi přijatá opatření na ochranu osobních údajů neměla vliv na zájem škol účastnit se soutěže.

Svůj souhlas se zpracováním osobních údajů odvolali doposud dva koordinátoři (z nichž jeden se registroval omylem dvakrát)

[3] V kategoriích Mini, Benjamin a Kadet nejsou celostátní pořadí soutěžících ani výsledkové listiny vytvářeny, neboť v nich nesoutěží žádní *oficiální soutěžící*

[4] Tento zadaný text není nikde uchováván a slouží pouze k tvorbě pamětního listu.

a o přístup ke svým osobním údajům požádalo 12 koordinátorů. Kromě výše zmíněných žádostí jsme nezaznamenali žádné podněty týkající se nespokojenosti se zpracováním osobních údajů koordinátorů.

4.2 Přijetí opatření soutěžícími v národním kole soutěže

V listopadu 2018 proběhlo národní kolo soutěže Bobřík informatiky, jehož se zúčastnilo celkem 76345 soutěžících, kteří nebyli vyřazeni koordinátory. V kategoriích Junior a Senior soutěžilo 15772 nevyřazených soutěžících. Z tohoto počtu bylo *oficiálními soutěžícími* 13665 osob (tj. 87 %) a *anonymními soutěžícími* 2107 osob (tj. 13 %). V této podkapitole (mimo Tabulky 3) pod *oficiální soutěžící* zahrnujeme soutěžící, kteří jsou od zahájení testu *oficiálními soutěžícími*, a soutěžící, kteří se sice před zahájením testu rozhodli být *anonymními soutěžícími*, ale koordinátor je později změnil na *oficiální*. Naopak pod *anonymní soutěžící* zahrnujeme soutěžící, kteří jsou od zahájení testu *anonymními soutěžícími*, a soutěžící, kteří se sice před zahájením testu rozhodli být *oficiálními soutěžícími*, ale koordinátor je později změnil na *anonymní*. Bližší údaje jsou uvedeny v Tabulce 3.

Tabulka 3. Počty nevyřazených soutěžících v kategoriích Junior a Senior podle typu soutěžícího

Typ soutěžícího	Junior	Senior	Celkem
Oficiální	8633	4972	13605
Původně anonymní, ale koordinátor změnil na oficiální	43	17	60
Anonymní	1565	402	1967
Původně oficiální, ale koordinátor změnil na anonymní	124	16	140
Celkem	10365	5407	15772

Kromě změn typu soutěžícího z *anonymního soutěžícího* na *oficiálního* (nebo naopak), které provedli koordinátoři, jsme nezaznamenali žádnou žádost o vymazání osobních údajů soutěžících či uplatnění jiných práv týkajících se osobních údajů soutěžících.

Abychom lépe porozuměli zastoupení *anonymních soutěžích* podle věku, rozhodli jsme se zjistit podíl *anonymních soutěžících* v kategoriích Junior a Senior podle ročníku školy, jenž navštěvují. Jelikož jsou kategorie pouze doporučené a žák může soutěžit ve starší věkové kategorii, jsou v Tabulce 4 zahrnuti i žáci základních škol, pro které tyto kategorie nejsou určené.

Tabulka 4. Počty nevyřazených soutěžících v kategoriích Junior a Senior podle typu soutěžícího a ročníku školy

	Oficiální soutěžící	Anonymní soutěžící	Celkem
Žáci ZŠ	137	136	273
1. ročník SŠ	4524	1008	5532
2. ročník SŠ	4315	622	4937
3. ročník SŠ	2732	167	2899
4. ročník SŠ	1957	174	2131

Jak vyplývá z Tabulky 4, mezi žáky základních škol, kteří soutěžili v kategorii Junior či Senior, bylo přesně 50 % *anonymních soutěžících*. Tento stav byl zřejmě způsoben tím, že žáci základních škol (ZŠ) jsou obvykle mladší 16 let a řada z nich neměla souhlas zákonných zástupců se zpracováním svých osobních údajů v rámci soutěže. Důležitější je však zastoupení *anonymních soutěžících* mezi žáky středních škol (SŠ), protože právě pro ně jsou kategorie Junior a Senior určeny.

Zatímco mezi soutěžícími navštěvujícími 1. ročník SŠ bylo 18 % *anonymních soutěžících*, mezi žáky navštěvujícími 2. ročník SŠ jich bylo 13 %. Tento stav může být způsoben stejně jako u žáků ZŠ tím, že někteří žáci 1. ročníku SŠ byli v době konání soutěže mladší 16 let a neměli souhlas zákonných zástupců se zpracováním svých osobních údajů v rámci soutěže. Mezi soutěžícími navštěvujícími 3. ročník SŠ bylo 6 % *anonymních soutěžících* a mezi soutěžícími navštěvujícími 4. ročník SŠ jich bylo 8 %. Protože tito žáci soutěžili výhradně v kategorii Senior, rozhodnutí být *oficiálními soutěžícími* mohlo být u řady z nich motivováno snahou postoupit do ústředního kola soutěže a případně vyhrát některou z cen.

5. ZÁVĚR

Realizovali jsme online soutěž Bobřík informatiky, v níž je potřeba zpracovávat osobní údaje školních koordinátorů soutěže i některých soutěžících. Protože doposud nebyl od koordinátorů vyžadován explicitní souhlas se zpracováním osobních údajů, bylo nutno jej získat dodatečně. V případě, že koordinátor včas souhlas neudělil, byl jeho účet odstraněn. Ačkoliv toto opatření přineslo řadu komplikací, došlo ke zpřehlednění seznamu koordinátorů, kdy byli vymazáni neaktivní koordinátoři. Současně se žádostí o uvedený souhlas se zpracováním osobních údajů v rámci soutěže byl koordinátor požádán i o souhlas se zasíláním informací ohledně výuky informatiky. Protože nám přibližně čtyři koordinátoři z pěti souhlas udělili, můžeme nyní většinu koordinátorů efektivně oslovovat i se záležitostmi mimo vlastní soutěž.

Soutěže Bobřík informatiky se každoročně účastní několik desítek tisíc žáků, z nichž většina je mladších 16 let. Po těchto soutěžících nelze požadovat zadání osobních údajů bez souhlasu zákonných zástupců a vzhledem k povaze soutěže není možné předpokládat, že by potřebný souhlas zákonní zástupci dopředu udělili. Abychom naplnili požadavky GDPR, vytvořili jsme dvě skupiny soutěžících. První z nich tvoří soutěžící starší 16 let, kteří zadávají své osobní údaje, souhlasí s jejich zpracováním a kteří tak mohou být uvedeni v oficiálních výsledkových listinách. Druhou skupinu tvoří soutěžící, kteří souhlas se zpracováním osobních údajů udělit nechtějí nebo nemohou a kteří tak soutěží bez možnosti porovnat se s ostatními.

Jak bylo zjištěno v uplynulém ročníku soutěže, naprostá většina soutěžících souhlasí se zpracováním svých osobních údajů organizátory soutěže, pokud to mohou učinit. Problém však nastává u soutěžících mladších 16 let, kteří nemohou sdělit své osobní údaje bez souhlasu zákonných zástupců, a proto musí soutěžit pod přezdívkou. Pokud je daná přezdívka libovolná, situace je komplikovaná i pro koordinátory, neboť nemohou své žáky identifikovat a např. je ocenit za to, že byli úspěšnými řešiteli soutěže. Na druhou stranu jako organizátoři nemůžeme zabránit tomu, aby žáci jako přezdívku uvedli např. své celé jméno, školní uživatelské jméno nebo studijní číslo popř. nemůžeme zabránit koordinátorům v tom, aby jim zadání těchto údajů nařídili.

Nabízí se tak otázka, zda by bylo možno změnit zpracování osobních údajů v rámci soutěže tak, aby správci osobních údajů u žáků mladších 16 let byly samotné školy a organizátoři soutěže se stali zpracovateli údajů. Pro školu by totiž bylo jednodušší nejen

získat souhlasy zákonných zástupců, ale také je archivovat. Je však otázkou, zda by taková administrativa nebyla zbytečně zatěžující a jak by soutěžili žáci, jejichž zákonní zástupci by potřebný souhlas neudělili.

BIBLIOGRAFICKÉ ODKAZY

[1] EVROPSKÁ UNIE. Nařízení Evropského parlamentu a Rady (EU) 2016/679 ze dne 27. dubna 2016 o ochraně fyzických osob v souvislosti se zpracováním osobních údajů a o volném pohybu těchto údajů a o zrušení směrnice 95/46/ES (obecné nařízení o ochraně osobních údajů). In *Úřední věstník Evropské unie* [online]. 2016, L 119/1, s. 1-88. ISSN 1977-0626. https://eur-lex.europa.eu/legal-content/CS/TXT/PDF/?uri=CELEX:32016R0679

[2] MŠMT. *Metodická pomůcka k aplikaci obecného nařízení o ochraně osobních údajů a zákona o zpracování osobních údajů v podmínkách školství* [online]. Praha: Ministerstvo školství, mládeže a tělovýchovy České republiky, 2017 [cit. 2019-01-10]. http://www.msmt.cz/file/44592_1_1/

[3] MŠMT. *Stručný návod na zabezpečení procesů souvisejících s GDPR ve školách (nástin pracovního postupu)* [online]. Praha: Ministerstvo školství, mládeže a tělovýchovy České republiky, 2018 [cit. 2019-01-10]. http://www.msmt.cz/file/46214_1_1/

[4] MILKAITE, I., LIEVENS, E. The changing patchwork of the child's age of consent for data processing across the EU (January 2019). In *Better Internet for Kids* [online]. 2019 [cit. 2019-01-25]. https://www.betterinternetforkids.eu/en_US/web/portal/practice/awareness/detail?articleId=3017751

[5] Vládní návrh zákona o zpracování osobních údajů. In *Sněmovní tisk 138/0, Poslanecká sněmovna, VIII. Volební období* [online]. 2018 [cit. 2019-10-01]. http://www.psp.cz/sqw/text/tiskt.sqw?O=8&CT=138&CT1=0

[6] VANÍČEK, J. Rozvoj informatických kompetencí žáků pomocí soutěže: kauza Bobřík informatiky [online]. In *Sborník konference Počítač ve škole*. Nové Město na Moravě: Gymnázium Vincence Makovského, 2009 [cit. 2019-01-12]. https://www.bebras.org/sites/default/files/documents/publications/Vanicek%20-2009.pdf

[7] Věkové kategorie. In *Bobřík informatiky* [online]. 2018 [cit. 2019-12-01]. https://www.ibobr.cz/o-soutezi/24-kategorie

[8] Jak se přihlásit do soutěže. In *Bobřík informatiky* [online]. 2018 [cit. 2019-12-01]. https://www.ibobr.cz/o-soutezi/27-jak-se-prihlasit-do-souteze

[9] Jak zorganizovat soutěž na škole. In *Bobřík informatiky* [online]. 2018 [cit. 2019-12-01]. https://www.ibobr.cz/o-soutezi/26-pokyny-pro-skolni-koordinatory

[10] Prohlášení o ochraně osobních údajů. In *Bobřík informatiky* [online]. 2018 [cit. 2019-13-01]. https://www.ibobr.cz/kontakty/soukromi/220-prohlaseni-o-ochrane-osobnich-udaju

Procesy operačného systému vo vyučovaní informatiky
Processes of operating system in the teaching of computer science

Jarmila Škrinárová
Univerzita Mateja Bela
Tajovského 40
974 01 Banská Bystrica
Slovenská republika
jarmila.skrinarova@umb.sk

Adam Dudáš
Žilinská univerzita v Žiline
Univerzitná 8215/1
010 26 Žilina
Slovenská republika
adam.dudas@fri.uniza.sk

ABSTRACT
Presented paper is motivated by survey, which was part of KEGA project called "Innovative methods in subject informatics in secondary education". The survey shows, and this is the goal of this paper, the need to develop methodologies for the teaching of processes in the operating system suitable for secondary education teachers. Designed methodology contains practical examples supplemented by the interpretation. This methodology can be used for teaching work with processes, process creation and modification of simple application with parameters. This article follows and further elaborates on educational materials of subject "Computer systems and networks", designed as a part of project IT akadémia.

Keywords
Process. Operating system. Secondary education.

ABSTRAKT
Prezentovaný článok je motivovaný prieskumom, ktorý sme robili v rámci KEGA projektu s názvom „Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní". Z prieskumu vyplýva, a to je aj cieľom tohto príspevku, potreba vypracovať metodiky vyučovania procesov v operačnom systéme, vhodné pre učiteľov informatiky sekundárneho vzdelávania. Navrhnutá metodika obsahuje praktické príklady, doplnené výkladom, ktoré na seba postupne nadväzujú. Pomocou metodiky je možné naučiť študentov pracovať s procesmi, vytvárať procesy a modifikovať jednoduché aplikácie zadávaním parametrov. Tento článok nadväzuje a podrobnejšie rozpracúva výučbové materiály predmetu „Počítačové systémy a siete", vytvorené v rámci projektu IT akadémia.

Kľúčové slová
Proces. Operačný systém. Sekundárne vzdelávanie.

1 ÚVOD
Tento príspevok je motivovaný prieskumom, ktorý sme robili v rámci projektu KEGA s názvom „Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní". Prieskumu sa zúčastnilo 47 učiteľov z rôznych typov škôl, z toho 45 učiteľov, ktorí spadajú do kategórie učiteľov sekundárneho vzdelávania. V prieskume sme učiteľom položili 82 otázok [1]. Otázky sme formulovali v súlade so Štátnym vzdelávacím programom, zameraným na predmet Informatika pre gymnáziá so štvorročným a päťročným vzdelávacím programom [2]. Z toho päť otázok sa týkalo témy „ Softvér a hardvér – práca v operačnom systéme".

Výkonový štandard témy uvádza [2]:

Žiak vie/dokáže:

- spravovať aplikácie (napr. inštalovať, odinštalovať softvér, doplnky, zisťovať parametre bežiacej aplikácie/procesov, zastavovať ich...),
- používať nástroje na prispôsobenie si (pracovného) prostredia v počítači a správanie sa počítača,
- skúmať nové možnosti operačného systému.

Obsahový štandard danej témy je špecifikovaný takto:

Pojmy: aplikácia

Vlastnosti a vzťahy: operačný systém ako softvér, operačný systém a správa prostriedkov (procesor, pamäť, ...) a poskytovanie služieb (pre aplikácie, ...), aplikácia ako softvér.

Vráťme sa však k prieskumu. Z odpovedí na otázky, týkajúce sa operačných systémov vyplýva, že štandard je všeobecný a abstraktný a učitelia by mali radi k dispozícii učebné materiály a metodiky s konkrétnymi príkladmi tak, aby mohli prakticky demonštrovať úlohy na bežiace aplikácie (procesy).

Aktuálne pracujeme v projekte IT akadémia. Projekt IT akadémia – vzdelávanie pre 21. storočie sa, okrem iných aktivít a predmetov, zameriava aj na predmet „Počítačové systémy a siete". Približne jedna tretina predmetu je venovaná „Operačným systémom". Stručne môžeme stanoviť špecifické ciele, ktoré má absolvent predmetu (za časť operačné systémy) dosiahnuť, takto: Porozumieť základným princípom operačných systémov. Poznať metódy riadenia procesov, pamätí a súborov v operačných systémoch a vedieť prezentovať. Nadobudnúť zručnosť pri práci s operačným systémom, vedieť vytvoriť jednoduché skripty a použiť ich na tvorbu a monitorovanie procesov. Poznať nástroje a vhodne ich použiť na zníženie doby prístupu k súborom. Poznať adresárové štruktúry a súborové systémy a vedieť s nimi pracovať. Vedieť pracovať s virtuálnymi počítačmi a navzájom ich prepojiť.

V rámci tohto projektu sme vytvorili vzdelávacie materiály, ktoré po pilotnom vzdelávaní budú zaradené do učebnice „Počítačové systémy a siete". Preto je cieľom tohto príspevku ukázať časť metodického materiálu, ktorý sa zameriava na objasnenie pojmu proces a najmä predstaviť praktický metodický postup, vrátane príkladov. Metodika a príklady, ktoré tu prezentujeme, čiastočne

prezentuje a čiastočne nadväzuje na materiály, ktoré sme vytvorili v projekte IT akadémia.

Prezentovaný článok je členený do dvoch nosných kapitol. V kapitole 2 je stručne predstavená úvodná teória k pojmu proces v operačnom systéme. V kapitole 3 uvádzame navrhnutý metodický postup, súčasťou ktorého je uvedených niekoľko príkazov a príkladov práce s procesmi v operačnom systéme.

2 PROCESY V OPERAČNOM SYSTÉME

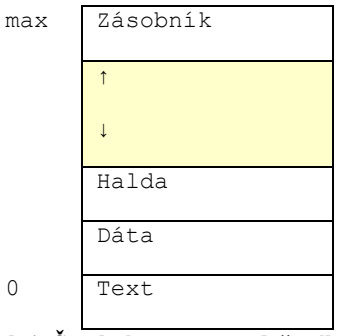
V tejto časti stručne definujeme pojmy operačný systém a proces. Operačný systém poskytuje prostredie, v ktorom sa vykonávajú programy (aplikácie). Operačný systém predstavuje súbor riadiacich programov (jadro operačného systému). Jadro operačného systému obsahuje softvérové komponenty, ktoré plnia základné funkcie systému. Medzi základné komponenty jadra patrí správa procesov, správa pamäte, súborových systémov a správa periférnych zariadení [3, 4]. Operačný systém musí komunikovať s používateľmi, aby mohol vykonávať ich požadované činnosti. Časť operačného systému, ktorá obsluhuje túto komunikáciu sa nazýva používateľské rozhranie.

Program je súbor uložený na disku, ktorý môžeme spustiť a keď ho spustíme vykonáva určitú činnosť. Proces je tzv. inštancia bežiaceho programu. Každý program, ktorý spustíme v systéme vytvorí jeden alebo viac procesov. Inak povedané proces je program v činnosti. Proces je definovaný kontextom, v ktorom sa vykonáva.

Proces je tvorený:

- kódom programu (tzv. text),
- aktuálnym stavom procesora a pamäte (počítadlo inštrukcií a obsah registrov), ktoré súvisia s procesom,
- zásobníkom, v ktorom sú uložené dočasné údaje (napríklad parametre funkcie, návratové adresy a lokálne premenné),
- dátovou časťou, ktorá obsahuje globálne premenné,
- haldou (nie vždy), ktorá je pamäťou a je dynamicky prideľovaná počas procesu spustenia.

Štruktúra procesu, ktorý je spustený a umiestnený v pamäti je zobrazená na obrázku 1.



Obrázok 1: Štruktúra procesu uloženého v pamäti

Samotný program nie je proces. Program je pasívny subjekt, napríklad súbor obsahujúci zoznam inštrukcií, uložených na disku (často nazývaný spustiteľný súbor). Na rozdiel od toho je proces aktívnou entitou, s počítadlom inštrukcií, ktoré ukazuje na ďalšiu inštrukciu na vykonanie a súbor pridružených zdrojov. Program sa stáva procesom spustiteľného súboru, keď je spustený a uložený do pamäte.

Z uvedeného vyplýva, že pojem proces je abstraktný pojem a je potrebné tvoriť príklady a metodické postupy tak, aby ich učitelia mohli použiť na hodinách informatiky.

3 METODICKÝ POSTUP A PRÍKLADY NA POROZUMENIE A TVORBU PROCESOV V OPERAČNOM SYSTÉME

V tejto kapitole uvedieme metodický postup, vrátene príkladov na objasnenie a tvorbu procesov v operačnom systéme Linux. Operačný systém Linux používame najmä preto, že je jednoducho použiteľný a ľahko čitateľný systém.

3.1 SKÔR AKO ZAČNEME

Skôr ako zaradíme túto tému do vyučovania je dôležité vedieť, aké má mať študent vedomosti a zručnosti. V prvom rade sú to základné pojmy, ako sú súbor, adresár, základné príkazy Linuxu (najmä `cd`, `pwd` a `ls`). Študent má disponovať základnou zručnosťou pri práci so systémom. Vie sa pohybovať po adresárovej (súborovej) štruktúre systému a vie zadávať základné príkazy do tzv. príkazového riadku. Ovládať základy tvorby skriptov nie je podmienkou. Základy skriptovania je, podľa nás, vhodné zaradiť práve do tejto témy. Treba si uvedomiť, že ak chceme pracovať s systéme, je potrebné ovládať základné programové konštrukcie ako sú sekvencie príkazov, vetvenia a cykly a mať základnú zručnosť v ich programovaní v ľubovoľnom programovacom jazyku, alebo prostredí. V tejto kapitole budeme používať príkazy uvedené v tabuľke 1.

Tabuľka 1: Prehľad príkazov používaných v kapitole 3

Príkaz	Význam
`ls /proc`	Výpis obsahu priečinka `proc`, ktorý obsahuje všetky spustené programy
`ps`	Výpis stavu procesov používateľa
`ps ax`	Výpis stavu všetkých procesov v systéme
`for`	Cyklus s pevným počtom opakovaní
`echo`	Výpis reťazca, ktorý za týmto príkazom nasleduje

3.2 PROCESY V PRÍKLADOCH

V tejto časti prezentujeme nielen príkazy a príklady, ale aj postup ako majú za sebou nasledovať. Okomentované príkazy a príklady tvoria výklad pre študenta a cieľom je, aby metodicky viedli učiteľa.

Všetky spustené programy (procesy) sa nachádzajú v operačnom systéme Linux v priečinku `/proc`. Na obrázku 2 vidíme príklad výpisu všetkých procesov, ktoré bežia v systéme počítača. Výpis urobíme pomocou príkazu `ls /proc`.

```
jskrinarova@labs:~$ ls /proc
1        10887  driver       kpageflags  softirqs
7227     95     execdomains  loadavg     stat
100      10889  1316         1350        14277     17
1318     1375   684          7826        acpi      fs        misc        sysrq-trig
cmdline  irq    mtrr         tty
1382     1668   522          693         91
crypto   keys   partitions   vmallocinfo
10884    31380  39           539         695       devices   key-users
```

Obrázok 2: Príklad výpisu všetkých procesov, ktoré bežia v systéme počítača (`/proc`)

Každý proces v systéme má svoje meno (číslo). Je to jedinečné číslo. To znamená, že žiadny iný proces v systéme nie je označený takým istým číslom. Pomocou príkazu `ps` (angl. process status) vypíšeme stav procesov, ktoré sú v systéme. Keďže sa nachádzame v určitom adresári, zistíme stav, najmä čísla procesov, ktoré spúšťame (pozri obrázok 3). Z výpisu sme zistili, že príkaz `ps` vytvoril proces `ps`, ktorý je v systéme označený číslom `3132`. Na výpise vidíme ešte proces `bash`, ktorý je označený číslom `32694`.

```
jskrinarova@labs:~$/priklady$ ps
  PID TTY          TIME CMD
 3132 pts/0    00:00:00 ps
32694 pts/0    00:00:00 bash
```

Obrázok 3: Spôsob výpisu vlastných procesov

Ak chceme získať zoznam všetkých procesov, ktoré sú v systéme napíšeme príkaz `ps ax` (pozri obrázok 4). Príkazy v systéme sú v skutočnosti vykonávateľné súbory. Po ich spustení rovnako aj vykonávaný príkaz v systéme vytvorí proces a vidíme ho vo výpise procesov. Preto príkaz `ps`, vytvorí proces `ps`.

```
jskrinarova@labs:~/skripty/priklady$ ps ax
PID TTY      STAT   TIME COMMAND
  1 ?        Ss   1730:45 /sbin/init
  2 ?        S      0:15 [kthreadd]
  3 ?        S    121:13 [ksoftirqd/0]
  .
  .
  .
31279 ?      S      0:00 [kworker/u8:0]
32358 pts/1  R+     0:00 ps ax
32395 ?      S      0:00 pickup -l -t unix -u-c
```

Obrázok 4: Spôsob zistenia všetkých procesov

Podobne vytvoríme proces aj spustením ľubovoľného programu. Program je vykonávateľný súbor, ktorého cieľom je po spustení riešiť určitú úlohu alebo vykonať daný výpočet [5]. Kód programu môže byť napísaný v rôznych programovacích jazykoch. V našom prípade, vytvoríme program vo forme skriptu. Skript tvorí množina za sebou idúcich príkazov operačného systému.. Na obrázku 5 je uvedený kód pre skript využívajúci príkazy operačného systému Linux v príkazovom interpreteri BASH. BASH (z angl. Bourne Again Shell) je program, ktorý umožní vykonávať používateľom zadané príkazy operačného systému Linux [6].

```
1    #!/bin/bash
2
3    for x in $(seq 1 10000)
4    do
5            echo $x >> VystupnySubor
6    done
```

Obrázok 5: Jednoduchý príklad skriptu využívajúceho cyklus s pevným počtom opakovaní

Na obrázku 5 je jednoduchý skript, v ktorom sme použili len jeden cyklus s pevným počtom (10000) opakovaní. Program v cykle zapisuje čísla od 1 do 10000 do riadku tak, že čísla ukladá do špecifikovaného súboru `VystupnySubor`.

Podstatnými časťami kódu na obrázku 5 sú príkazy v riadkoch 3 a 5. V riadku 3 je uvedená syntax príkazu cyklu FOR. Príkaz FOR slúži na vytvorenie cyklu s pevným počtom opakovaní. V našom prípade premenná `x` postupne (sekvenčne) nadobúda hodnoty `1` až `10000`, čo je definované vo výraze `$(seq 1 10000)`, pričom prvé číslo označuje počiatočnú hodnotu premennej x a druhé číslo konečnú hodnotu tejto premennej.

Pre každú hodnotu x sa vykonajú všetky príkazy uvedené medzi riadkami 4 a 6. V našom príklade sa na riadku 5 nachádza príkaz `echo $x >> VystupnySubor`, ktorý postupne vypisuje (príkaz echo) hodnoty premennej `x` (vyberáme ich pomocou `$x`). Pretože chceme, aby sa premenné zapisovali do súboru, použijeme znaky presmerovania a meno výstupného súboru (`>> VystupnySubor`).

Skript, obsahujúci kód z obrázku 5, sme uložili pod názvom `NazovSkriptu`. Skript (program) spustíme rovnako ako každý program, t.j. pomocou dvojice znakov `./` a jeho názvu (pozri

obrázok 6). Časť súboru `VystupnySubor` po spustení skriptu je uvedený na obrázku 7.

```
./NazovSkriptu
```

Obrázok 6: Spustenie skriptu z obrázku 5

```
9995
9996
9997
9998
9999
10000
```

Obrázok 7: Časť súboru VystupnySubor po spustení skriptu `NazovSkriptu`

Teraz chceme zobraziť naše procesy, ktoré bežia v systéme, preto použijeme príkaz `ps` počas behu nášho skriptu. Na obrázku 8 vidíme príkaz pre spustenie tohto skriptu na pozadí a zároveň spustenie príkazu `ps`.

```
./NazovSkriptu & ps
```

Obrázok 8: Spustenie skriptu `NazovSkriptu` na pozadí a vypísanie práve bežiacich procesov

Keďže beh nášho skriptu trvá len niekoľko sekúnd, pri vypísaní aktuálne bežiacich procesov môžeme vidieť aj novovzniknutý proces s názvom `NazovSkriptu` (pozri obrázok 9).

Po ukončení práce skriptu môžeme znova spustiť príkaz `ps`, pričom vidíme, že proces s názvom `NazovSkriptu` už viac nie je medzi aktívnymi procesmi (pozri obrázok 10). Po skončení aplikácie (programu) sa vytvorený proces ukončí a je vymazaný z `/proc` a tiež uvoľní pridelenú časť pamäte.

```
PID TTY          TIME CMD
 8601 pts/2    00:00:00 bash
20379 pts/2    00:00:00 NazovSkriptu
20380 pts/2    00:00:00 ps
```

Obrázok 9: Skript `NazovSkriptu` po spustení beží ako proces s PID 20379

```
PID TTY          TIME CMD
 8601 pts/2    00:00:00 bash
20857 pts/2    00:00:00 ps
```

Obrázok 10: Výpis bežiacich procesov po ukončení práce skriptu `NazovSkriptu`

Ďalším cieľom resp. úlohou učiteľa je, aby žiak vedel meniť parametre procesov (pozri výkonový štandard) [2]. Preto rozšírime náš skript (obrázok 5) o prácu s parametrami, ktoré môžu meniť beh skriptu. Parameter je hodnota, ktorú zadá používateľ pri spúšťaní skriptu a táto hodnota bude v skripte použitá tak, že sa priradí do premennej. Ak zadáme jeden parameter, jeho hodnota sa uloží do premennej s názvom `$1`. Ak zadáme za menom skriptu viac parametrov, budú sa postupne priraďovať do premenných `$1`, `$2`, `$3`, ...

Aby sme študentov naučili používať parametre skriptu, jednoducho upravíme skript z obrázku 5 tak, aby program nepočítal striktne do 10 000, ale po také číslo, aké mu používateľ zadá pri spúšťaní. Zmena v tele skriptu je uvedená na obrázku 11. Kde systém za premennú `$1` dosadí prvú zadanú hodnotu pri spúšťaní skriptu. Na obrázku 12 môžeme vidieť príklad spúšťania nášho skriptu s parametrom 20 000 t.j. aplikácia bude počítať po toto číslo.

```
1    #!/bin/bash
2
3    for x in $(seq 1 $1)
4    do
5            echo $x >> VystupnySubor
6    done
```

Obrázok 11: Úprava skriptu `NazovSkriptu`, ktorý nazveme `NazovUpravenehoSkriptu`

```
./NazovUpravenehoSkriptu 20000
```

Obrázok 12: Spustenie skriptu `NazovUpravenehoSkriptu` s parametrom 20000

Okrem premenných, ktoré zadal používateľ (`$1`, `$2`, `$3`, ...) sa v skriptoch bežne využívajú aj špeciálne premenné, ktoré nie je potrebné pri spúšťaní zadávať. Medzi najvyužívanejšie premenné tohto typu patrí najmä `$0` a `$#`. Premenná `$0` vždy obsahuje názov spúšťaného skriptu, v ktorom je použitá. Premenná, označená ako `$#` obsahuje počet parametrov zadaných používateľom pri spustení skriptu. Obe špeciálne premenné využívame v príklade na obrázku 13.

```
1      #!/bin/bash
2
3      echo Nazov tohto suboru je $0
4      echo Pocet parametrov je $#
5      echo Parameter 1 ma hodnotu $1
6      echo Parameter 2 ma hodnotu $2
```

Obrázok 13: Skript využívajúci štandardné aj špeciálne parametre

Na obrázku 14 vidíme príkaz na spustenie skriptu z obrázku 13 (`NazovSkriptuSParametrami`) s troma parametrami. Výstup z programu je zobrazený na obrázku 15.

```
./NazovSkriptuSParametrami jeden dva tri
```

Obrázok 14: Príkaz na spustenie skriptu `NazovSkriptuSParametrami`

```
Nazov tohto suboru je NazovSkriptuSParametrami

Pocet parametrov je 3

Parameter 1 ma hodnotu jeden

Parameter 2 ma hodnotu dva
```

Obrázok 15: Výstup z behu skriptu `NazovSkriptuSParametrami`

4 ZÁVER

V tomto článku sme objasnili pojem proces v operačnom systéme. Z teoretických východísk vyplýva, že ide o abstraktný pojem, ktorý je pre študentov ťažko porozumiteľný. Preto sme navrhli metodický postup, vrátane príkladov. Cieľom je, aby študent, ktorý postupne prechádza jednotlivé príklady (často príkazy operačného systému) nielen porozumel pojmu proces, ale vedel ho aj jednoducho sám vytvoriť.

POĎAKOVANIE

Príspevok vznikol v rámci riešenia projektu KEGA 009KU-4/2017 Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní.

Príspevok je čiastočne podporený projektom IT akadémia – vzdelávanie pre 21. storočie, ITMS 312011F057, ktorý je spolufinancovaný Európskym sociálnym fondom.

BIBLIOGRAFICKÉ ODKAZY

[1] Hedviga Petrušková, Janka Majherová, Jana Jacková. Rozvoj didaktických kompetencií učiteľov informatiky [online]. In DidInfo 2018. Liberec (Česko) : Technická univerzita v Liberci, 2018. - ISBN 978-80-7494-424-6, s. 211-216.

http://www.didinfo.net/images/DidInfo/files/Didinfo_2018.pdf

[2] Štátny pedagogický ústav. *Informatika – gymnázium so štvorročným a päťročným vzdelávacím programom* [online]. 2015.

http://www.statpedu.sk/sk/svp/inovovany-statny-vzdelavaci-program/inovovany-svp-gymnazia-so-stvorrocnym-patrocnym-vzdelavacim-programom/matematika-praca-informaciami/

[3] BROOKSHEAR, J. G.: *Computer Science: An Overview*. 11.vyd. New Jersey: Prentice Hall, 2012, 624 s. ISBN 0132569035.

[4] MARTINCOVÁ, P., GRONDŽÁK, K.: *Operačné systémy*. 1.vyd. Žilina: EDIS, 2004, 294 s. ISBN 80-8070-242-X.

[5] ROCHKIND, M.: *Advanced UNIX Programming.* 2.vyd. Addison Wesley Longman Publishing Co., Inc. Redwood City, 2004. ISBN 0131411543

[6] STONES, R., MATTHEW, N.: *Beginning Linux Programming.* 2.vyd. Wrox Pr Inc., 1999. ISBN 978-1861002976

Nový informatický predmet Programovanie mobilných zariadení: Koncepcia jeho výučby a ukážky etúd

New Informatics Subject Programming of Mobile Devices: Concept of its Teaching and Samples of Etudes

Ľubomír Šnajder
Prírodovedecká fakulta UPJŠ
Jesenná 5
041 54 Košice
Slovenská republika
lubomir.snajder@upjs.sk

Ján Guniš
Prírodovedecká fakulta UPJŠ
Jesenná 5
041 54 Košice
Slovenská republika
jan.gunis@upjs.sk

Gabriela Lovászová
Fakulta prírodných vied UKF
Trieda Andreja Hlinku 1
949 74 Nitra
Slovenská republika
glovaszova@ukf.sk

Viera Michaličková
Fakulta prírodných vied UKF
Trieda Andreja Hlinku 1
949 74 Nitra
Slovenská republika
vmichalickova@ukf.sk

ABSTRACT

In this paper, we deal with concept and implementation of the pilot teaching of subject Programming mobile devices at secondary schools. We develop this subject as one of the eight new informatics subjects within the frame of the National Project IT Academy – Education for 21st Century. The scope of the course is 33 lessons and it is oriented to event-driven programming of Android devices using embedded sensors to program useful STEAM applications in cloud block programming environment the MIT App Inventor 2.

The main line of teaching of this subject leads from programming first applications through programming simple etudes towards development of more demanding projects including a final team project (computer game, educational application, simulation or modeling application, health or sports supportive application). Teachers are provided with teaching materials, worksheets (without and with task solving), work files (source code of programs, multimedia files), reference materials. Pilot verification of the subject teaching is ongoing in the school year 2018/2019, while selection of programming etudes and projects is within the competence of teacher-verifiers.

Keywords

Programming. Mobile Devices. MIT App Inventor. Methodology of Teaching. STEAM. Constructionism.

ABSTRAKT

V príspevku sa zaoberáme koncepciou a realizáciou pilotnej výučby predmetu Programovanie mobilných zariadení na stredných školách. Tento predmet vyvíjame ako jeden z ôsmich nových informatických predmetov v rámci národného projektu IT Akadémia – vzdelávanie pre 21. storočie. Rozsah predmetu je 33 vyučovacích hodín a je zameraný na udalosťami riadené programovanie mobilných zariadení s OS Android s využitím vstavaných senzorov, výsledkom ktorého je naprogramovanie užitočných STEAM aplikácii v cloudovom blokovom programovacom prostredí MIT App Inventor 2.

Hlavná línia výučby predmetu vedie od naprogramovania prvej aplikácie cez programovanie jednoduchých programátorských etúd smerom k tvorbe náročnejších projektov až k vývoju tímového projektu (počítačovej hry, edukačnej aplikácie, simulačnej alebo modelovacej aplikácie, asistenčnej zdravotnej či športovej aplikácie). Učiteľom sme poskytli metodické materiály, pracovné listy (bez aj s riešeniami úloh), pracovné súbory (zdrojové kódy programov, multimediálne súbory), referenčné materiály. Pilotné overenie predmetu prebieha v školskom roku 2018/2019, pričom výber programátorských etúd aj projektov je v kompetencii učiteľov-overovateľov.

Kľúčové slová

Programovanie. Mobilné zariadenia. MIT App Inventor. Metodika výučby. STEAM. Konštrukcionizmus.

1. ÚVOD

Strategickým cieľom Národného projektu IT Akadémia – vzdelávanie pre 21. storočie je vytvorenie modelu vzdelávania a prípravy mladých ľudí pre aktuálne a perspektívne potreby vedomostnej spoločnosti a trhu práce so zameraním na informatiku a IKT. Cieľom aktivity 1.1b je vytvorenie modelu špeciálnej triedy so zameraním na informatiku hlavne na gymnáziách, ktorého

súčasťou je vytvorenie metodických a učebných materiálov ôsmich nových predmetov pre tieto triedy [1]:

- Riešenie problémov a programovanie,
- Programovanie mobilných zariadení,
- Počítačové systémy a siete,
- Informačná bezpečnosť,
- Databázy,
- Objektový prístup k riešeniu problémov,
- Tvorba a prezentácia dát,
- Programovanie webových stránok.

Predmet Programovanie mobilných zariadení vyvíjajú autori tohto článku na partnerských univerzitách – UPJŠ v Košiciach a UKF v Nitre v spolupráci s projektovými strednými školami a IT spoločnosťou T-Systems.

2. CHARAKTERISTIKA, CIELE A LÍNIA VÝUČBY PREDMETU

Predmet Programovanie mobilných zariadení nadväzuje na základný kurz stredoškolského programovania a je určený žiakom 2. alebo 3. ročníka strednej školy. Je navrhnutý ako 33-hodinový kurz udalosťami riadeného programovania mobilných zariadení v prostredí MIT App Inventor 2 (ďalej App Inventor) zameraný na tvorbu užitočných STEAM (skratka angl. Science, Technology, Engineering, Arts and Math) aplikácii využívajúcich vstavané senzory mobilných zariadení.

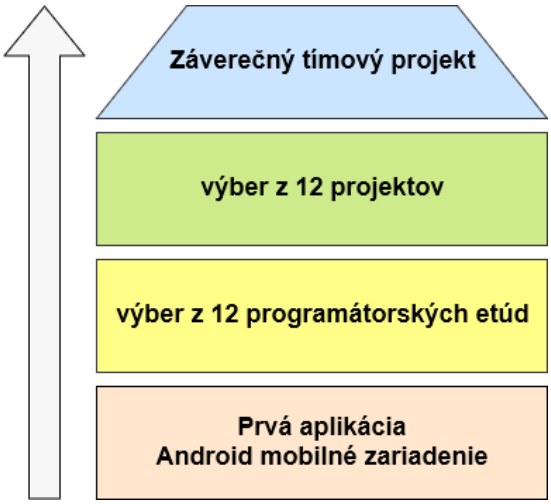
Hlavnými cieľmi predmetu sú: získať znalosti, zručnosti a skúsenosti pri programovaní udalosťami riadených praktických aplikácií, rozvíjať informatické myslenie, bádateľské spôsobilosti a tvorivosť žiakov.

Špecifickými cieľmi predmetu sú:

- byť zručný v práci s mobilným zariadením s OS Android: zisťovať parametre a meniť nastavenia mobilného zariadenia, spúšťať a ukončovať beh aplikácií, orientovať sa v štruktúre súborov v mobilnom zariadení a manipulovať so súbormi,
- porovnávať, kriticky hodnotiť funkcionality existujúcich aplikácií, tvorivo navrhnúť funkcionalitu vlastnej aplikácie,
- rozumieť princípom tvorby grafického používateľského rozhrania aplikácie a udalosťami riadeného programovania v prostredí App Inventor a vedieť ich aplikovať pri tvorbe jednoduchých aplikácií podľa zadania,
- samostatne navrhnúť a prakticky realizovať postup tvorby aplikácie v prostredí App Inventor a vytvárať aplikácie, v ktorých sa využívajú rôzne funkcionality mobilného zariadenia (multimédiá, senzory, komunikačné nástroje, aktivity externých aplikácií) a rôzne prvky programovacieho jazyka (algoritmické bloky a dátové štruktúry).

Výučbu tohto predmetu odporúčame realizovať podľa uvedenej línie (pozri obrázok 1):

- úvodné oboznámenie sa s Android mobilným zariadením (nastavenia, senzory, súborový systém) a tvorba prvej aplikácie v cloudovom vývojovom prostredí App Inventor,
- tvorba programátorských etúd (jednoduchých programov) zameraných na vybrané funkcionality vývojového prostredia App Inventor,
- tvorba programátorských projektov zameraných na vybrané problémy každodenného života využívajúce multimédiá, sieťové prepojenie a senzory mobilného zariadenia,
- vývoj a prezentácia záverečného tímového programátorského projektu.



Obrázok 1. Línia vyučovania predmetu Programovanie mobilných zariadení

Celkovú 33-hodinovú dotáciu predmetu odporúčame rozdeliť na jednotlivé časti línie vyučovania v pomere 2 : 6 : 20 : 5. Je však v kompetencii samotného učiteľa prispôsobiť si podľa svojich podmienok rozdelenie počtu hodín v jednotlivých častiach tohto predmetu a tiež vlastný výber programátorských etúd a projektov.

3. OBSAH PREDMETU

Obsah predmetu je rozdelený do 6 kapitol:

1. Úvod do práce s OS Android a tvorba prvej aplikácie v App Inventore,
2. Tvorba jednoduchých aplikácií – programátorských etúd,
3. Tvorba aplikácií s využitím multimédií,
4. Tvorba aplikácií s využitím sietí,
5. Tvorba aplikácií s využitím geolokácie.
6. Tvorba aplikácií s využitím senzorov a aktuátorov.

Metodický materiál obsahuje koncepciu výučby tohto predmetu zaradenú na začiatku materiálu a prílohy s referenčnými materiálmi a pracovnými súbormi zaradené na konci materiálu.

4. ŠTRUKTÚRA METODICKÉHO MATERIÁLU K PROGRAMÁTORSKÝM ETUDÁM

Programátorské etudy sú uvedené v kapitole 2, ktorá má oproti ostatným kapitolám mierne modifikovanú štruktúru:

- Kľúčové slová,
- Kognitívne ciele kapitoly (použité komponenty, udalosti, metódy a vlastnosti; použité programové konštrukcie a dátové štruktúry; použité algoritmy; prepojenie na iné predmety či oblasti spoločnosti),
- Príprava na výučbu,
- Odporúčaný priebeh výučby,
- Zoznam programátorských etúd s uvedeným počtom aplikácií a použitými komponentmi, udalosťami, metódami a vlastnosťami,
- Tabuľka s prehľadom komponentov, ktoré neboli pokryté programátorskými etudami,
- Tabuľka s pokrytím vybraných komponentov a jazykových konceptov App Inventora jednotlivými úlohami (pozri obrázok 2),
- Každá z 12 etúd obsahuje:
    - Prvky nového učiva (komponenty, udalosti s premennými, metódy, vlastnosti, jazykové koncepty),
    - Prehľad pracovných súborov (meno a použitie súboru),
    - Komentár k priebehu a výsledkom výučby,
    - Pracovný list so zadaním niekoľkých úloh,
    - Riešenie pracovného listu,
    - Sebahodnotiacu kartu,
    - Návrh tvorby aplikácie (návrh používateľského rozhrania, návrh správania),
- Bibliografia,
- Register pojmov.

Z metodického materiálu pre učiteľov sme vytvorili učebný materiál pre žiakov, ktorý neobsahuje metodické časti (Príprava na výučbu, Odporúčaný priebeh výučby), ani zdrojové kódy programov s riešením vybraných úloh, ale obsahuje pracovné listy, sebahodnotiace karty a pracovné súbory (zdrojové kódy programov a prípadne multimediálne súbory potrebné pre vyriešenie zadaných úloh).

5. ZOZNAM A UKÁŽKY VYBRANÝCH PROGRAMÁTORSKÝCH ETÚD

Aj keď hlavným cieľom programátorských etúd je, aby žiaci získali prvé skúsenosti s programovaním v App Inventore, každá z etúd je koncipovaná ako séria praktických úloh, rozpracovaná v rôznych úrovniach náročnosti, resp. s rôznym rozsahom rozšírenia.

Zoznam programátorských etúd:

2.1. Kresliaci editor,
2.2. Hra Postreh,
2.3. Hra Guľka,
2.4. Kalkulačka,
2.5. Zbierka vtipov,
2.6. Čítačka QR kód,
2.7. Asistent pri cvičení,
2.8. Generátor náhodných viet,
2.9. Zobrazovač aktuálnej polohy,
2.10. Asistent aktuálnej polohy,
2.11. Hlasovanie na internete,
2.12. Komunikačný asistent.

Zbierka 12 programátorských etúd obsahujúca 37 čiastkových úloh pokrýva 30 vybraných skupín komponentov a jazykových konceptov App Inventora (pozri obrázok 2 vyjadrujúci globálny pohľad na uvedené mapovanie):

	Používateľské rozhranie							Mmédiá			Senzory								Komunik.			Pamäť		Jazykové koncepty							
úloha	Screen, Canvas, Ball, Image	Multiple Screens	Button, TextBox, Label, CheckBox	H/V/T Arrangements	ListView, ListPicker	Slider, Spinner	Notifier	Sound	TextToSpeech	SpeechRecognizer	Clock	AccelerometerSensor	LocationSensor	Map	OrientationSensor	BarcodeScanner	ProximitySensor	Pedometer	ActivityStarter	Texting	PhoneCall	TinyDB	FireBase	Cyklus	Vetvenie, podmienky	Matematické operácie a funkcie	Globálne premenné (čísla, farby)	Zoznamy	Procedúry, funkcie	Lokálne premenné	
Kresliaci editor (1.1, 1.2)	■											■																			2
Kresliaci editor (1.3)	■											■																			2
Hra Postreh (2.1)	■							■			■															■					4
Hra Postreh (2.2)	■		■	■				■			■															■	■				7
Hra Postreh (2.3)	■		■	■				■			■															■	■				7
Hra Guľka (3.1)	■							■							■											■					4
Hra Guľka (3.2)	■														■											■					3
Stopky (3.3)			■								■															■	■				4
Počítadlo (3.4)	■		■																			■				■	■				5
Hra Guľka (3.5)	■		■								■				■							■			■	■	■		■		9
Kalkulačka BMI (4.1)			■	■																					■	■				■	5
Kalkulačka BMI (4.2)			■	■																					■	■			■	■	6
Kalkulačka BMI (4.3)			■	■		■																			■	■			■	■	7
Kalkulačka BMI (4.4)			■				■																		■	■	■		■		6
Zbierka vtipov (5.1)	■	■	■	■																					■						5
Zbierka vtipov (5.2)	■	■	■	■																					■						5
Zbierka vtipov (5.3)	■	■	■	■																					■						5
Čítačka QR kódu (6.1)			■						■							■															3
Čítačka QR kódu (6.2)			■			■			■	■						■															5
Čítačka QR kódu (6.3)			■	■		■			■	■						■															6
Asistent cvikov (7.1)	■							■				■					■														4
Asistent cvikov (7.2)	■		■	■				■				■					■								■	■	■				9
Asistent cvikov (7.3)			■	■				■										■								■					5
Asistent cvikov (7.4)	■		■	■														■							■	■			■		7
Generátor viet (8.1)			■						■																	■	■	■			5
Generátor viet (8.2)			■						■																	■	■	■		■	6
Spracovanie zoznamov (8.3)			■	■	■																			■	■	■	■	■	■	■	10
Spracovanie zoznamov (8.4)			■	■	■																			■	■	■	■	■	■	■	10
Zobrazovač GPS polohy (9.1)			■	■									■	■																	4
Zobrazovač GPS polohy (9.2)			■	■		■							■	■											■						6
Asistent GPS polohy (10.1)	■		■	■	■								■						■								■	■			8
Asistent GPS polohy (10.2)	■		■	■	■		■						■						■			■		■	■	■	■	■	■		14
Spúšťač externých apiek (10.3)			■	■															■								■	■			5
Webový kliker (11.1)	■		■																				■			■					4
Webové hlasovanie (11.2)	■		■	■																			■		■	■		■			7
Hlasná čítačka SMS (12.1)			■						■											■											3
Asistent SMS (12.2)	■		■	■					■	■		■					■			■	■				■						10
	20	3	31	21	4	4	2	7	7	3	5	5	4	2	3	3	3	2	3	2	1	3	2	3	16	22	14	8	8	6	

Obrázok 2. Mapa pokrytia vybraných komponentov a jazykových konceptov App Inventora jednotlivými úlohami

Jednotlivé problémy pokrývajú nasledovné komponenty a jazykové koncepty App Inventora (pozri tabuľka 1):

Tabuľka 1: Komponenty a jazykové koncepty rozvíjané v programátorských etudách

Používateľské rozhranie:	Jazykové koncepty:
• `Screen, Canvas, Ball, Image` • `Multiple Screens` • `Button, TextBox, Label, CheckBox` • `Horizontal/Vertical/Table Arrangements` • `ListView, ListPicker` • `Slider, Spinner` • `Notifier`	• Cyklus • Vetvenie, podmienky • Matematické operácie a funkcie • Globálne premenné (čísla, farby) • Zoznamy • Procedúry, funkcie • Lokálne premenné
Senzory:	**Komunikácia:**
• `Clock` • `AccelerometerSensor` • `LocationSensor` • `Map` • `OrientationSensor` • `BarcodeScanner` • `ProximitySensor` • `Pedometer`	• `ActivityStarter` • `Texting` • `PhoneCall`
Pamäť:	**Multimédiá:**
• `TinyDB` • `FireBase`	• `Sound` • `TextToSpeech` • `SpeechRecognizer`

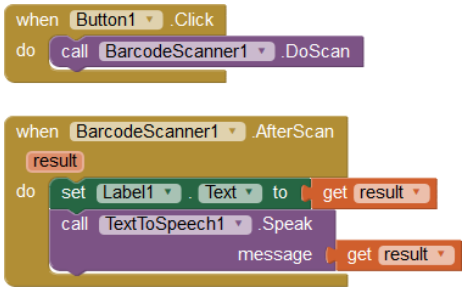
Pre lepšiu predstavu o metodike výučby programátorských etúd uvádzame ukážky troch vybraných etúd.

5.1 Čítačka QR kódu

Hlavným cieľom tejto etudy je vytvoriť aplikáciu na zosnímanie QR kódu, jeho zobrazenie a prečítanie syntetickým hlasom.

Pracovný list obsahuje tri úlohy. Prvé dve úlohy sú zamerané na analýzu hotového programového kódu, tretia na rozšírenie programového kódu.

V prvej úlohe majú žiaci analyzovať používateľské rozhranie a programový kód aplikácie, v ktorej sú použité nové komponenty `BarcodeScanner` a `TextToSpeech`.



Obrázok 3. Čítačka QR kódu – programový kód prvej úlohy

V pracovnom liste je k analyzovanému zdrojovému kódu (pozri obrázok 3) uvedená sada otázok, pomocou ktorej žiaci môžu zaznamenať, vysvetliť a prediskutovať svoje zistenia:

- Čo sa stane po spustení metódy `BarcodeScanner.DoScan`?
- Čo vyvolalo udalosť `BarcodeScanner.AfterScan` a kedy?
- Aký význam v udalosti `BarcodeScanner.AfterScan` má parameter result?
- Čo sa stane, ak v komponente `BarcodeScanner` zaškrtneme vlastnosť `UseExternalScanner`?
- Čo robí metóda `TextToSpeech.Speak`?

V druhej analytickej úlohe sú použité ďalšie nové komponenty `Spinner` a `SpeechRecognizer`. Odporúčame, aby pri týchto úlohách žiaci v dvojiciach skúmali hotové programové kódy, aby si ich vzájomne vysvetlili a prediskutovali. Rolu učiteľa vidíme ako konzultanta, ktorý dialógom usmerňuje a povzbuduje jednotlivé dvojice žiakov. Namiesto tradičného frontálneho výkladu všetkých možných nových funkcionalít a nepodstatných detailov, by mal učiteľ poskytnúť žiakom čas na ich samostatné objavovanie, ktoré by malo byť zavŕšené ich radosťou a sebavedomím, že prišli na nové veci sami, prípadne len s nevýznamnou pomocou učiteľa.

Tretia úloha je zameraná na rozšírenie druhej úlohy o ďalšie funkcionality – rozbaľovacie zoznamy s nastavením rýchlosti syntezovanej reči, výšky hlasu, krajiny a jazyka (pozri obrázok 4).

Obrázok 4. Čítačka QR kódu – snímka obrazovky s používateľským rozhraním výslednej aplikácie z tretej úlohy

Je vítané, ak žiaci doprogramujú aj svoje vlastné funkcionality, čo významne prispieva k slobodnej a kreatívnej atmosfére hodiny.

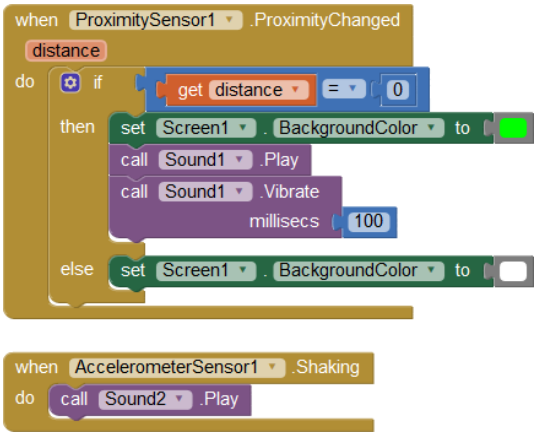
V závere hodiny môže učiteľ formou dialógu so žiakmi stručne zhrnúť čo nové sa naučili pri tejto etude, pochváliť žiakov za ich úsilie a za zaujímavé nápady v ich aplikáciách.

5.2 Asistent pri cvičení

Hlavným cieľom tejto etudy je vytvoriť aplikáciu pre počítanie fyzických cvikov a aplikáciu pre počítanie krokov s využitím senzora priblíženia a virtuálneho senzora `Pedometer`.

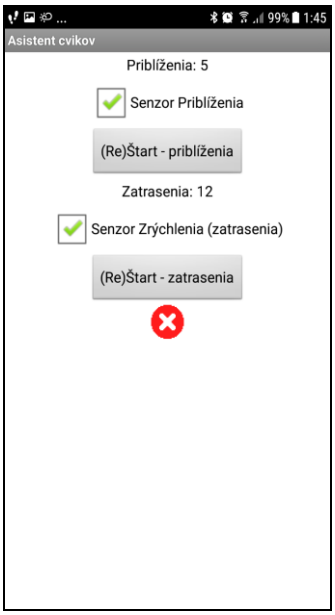
Pracovný list obsahuje štyri úlohy. Prvé dve sú zamerané na analyzovanie a doplnenie aplikácie pre počítanie fyzických cvikov využívajúcej senzor priblíženia. Ďalšie dve úlohy sú zamerané na analyzovanie a doplnenie aplikácie pre počítanie krokov využívajúcej virtuálny senzor `Pedometer`.

V prvej úlohe majú žiaci analyzovať používateľské rozhranie a programový kód aplikácie (pozri obrázok 5), v ktorej je použitý nový komponent `ProximitySensor`. Pomocou sady otázok majú žiaci zaznamenať, vysvetliť a prediskutovať svoje zistenia.



Obrázok 5. Asistent pri cvičení – programový kód prvej úlohy

Druhá úloha je zameraná na rozšírenie prvej úlohy o ďalšie funkcionality – resetovanie počtov priblížení a krokov, zapínanie a vypínanie senzorov pomocou zaškrtavacích políčok. Snímka obrazovky s používateľským rozhraním výslednej aplikácie je uvedená na obrázku 6.

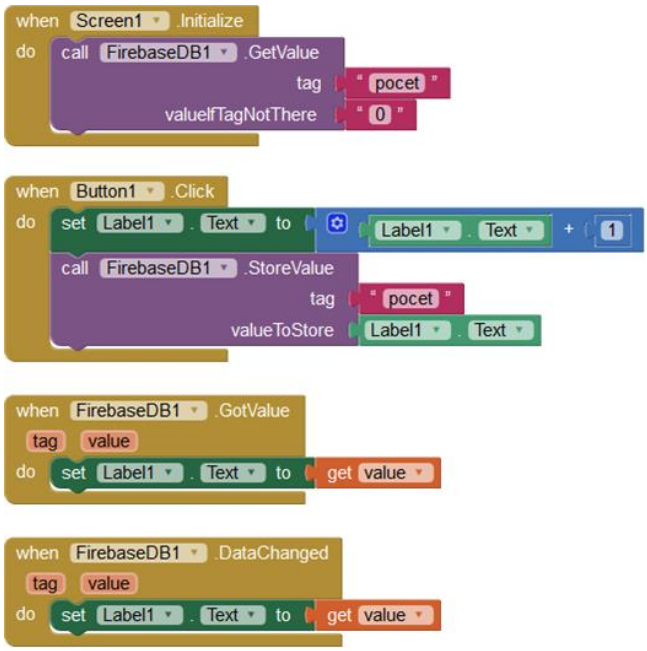


Obrázok 6. Asistent pri cvičení – snímka obrazovky s používateľským rozhraním výslednej aplikácie z druhej úlohy

5.3 Hlasovanie na internete

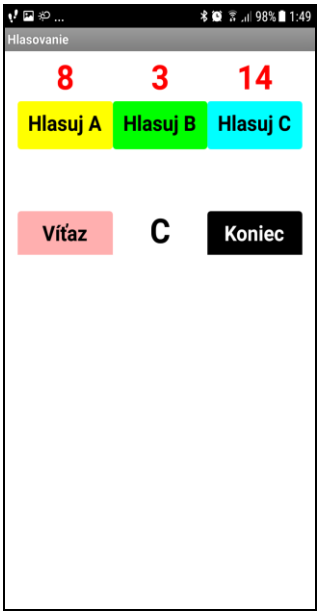
Hlavným cieľom tejto etudy je vytvoriť hlasovaciu aplikáciu využívajúcu webovú databázu (`Firebase`).

Pracovný list obsahuje dve úlohy. Vzhľadom na náročnosť problematiky webovej non-SQL databázy a viacero technických detailov je súčasťou zadania prvej úlohy tutoriál, ktorý dovedie žiakov k vytvoreniu vlastnej webovej databázy a naprogramovaniu jednoduchej hlasovacej aplikácie (s programovým kódom uvedeným na obrázku 7) obsahujúcej len tlačidlo a popisok zobrazujúci počet kliknutí.



Obrázok 7. Hlasovanie na internete – programový kód prvej úlohy

Druhá úloha je zameraná na rozšírenie prvej úlohy o ďalšie funkcionality – zaznamenie až troch počítadiel v databáze, doplnenie ďalších hlasovacích tlačidiel aj tlačidla s výpisom víťaznej voľby hlasovania (pozri obrázok 8).



Obrázok 8. Hlasovanie na internete – snímka obrazovky s používateľským rozhraním rozšírenej aplikácie z druhej úlohy

6. PRIEBEH PILOTNÉHO OVEROVANIA PREDMETU

Pilotné overenie predmetu prebieha v školskom roku 2018/2019. Pre jeho podporu sme vytvorili Moodle kurz [2], v rámci ktorého poskytujeme učiteľom metodické materiály, vyriešené pracovné listy, zdrojové kódy programov vyriešených úloh. Pre ich žiakov poskytujeme učebné texty, pracovné listy, zdrojové kódy programov k zadaniu úloh, multimediálne pracovné súbory, referenčné materiály. Pre komunikáciu s učiteľmi používame diskusné fórum. Pre získanie spätnej väzby po výučbe jednotlivých časti predmetu používame hodnotiaci dotazník vyplnený učiteľmi a zozbierané žiacke projekty.

7. ZÁVER

Predmet Programovanie mobilných zariadení umožňuje žiakom byť nielen používateľmi aplikácií na mobilných zariadeniach, ale stať sa aj tvorcami užitočných aplikácií, ktoré využívajú špecifické funkcionality mobilného zariadenia. Tento predmet vyvíjame tak, aby bol inovatívny nielen svojím zameraním a obsahom, ale tiež spôsobom výučby. Okrem programátorských zručností žiakov sa snažíme rozvíjať aj ich bádateľské spôsobilosti, kreativitu a schopnosti tímovej práce.

Pri vývoji metodických a učebných materiálov sme čerpali z vlastných pedagogických skúseností [3, 4, 5] aj z publikácii ďalších pedagógov a výskumníkov [6, 7, 8]. Veľmi podstatnou súčasťou vývoja materiálov pre tento predmet sú samotní učitelia a konzultanti z IT firiem, ktorí nám poskytli a poskytujú veľmi dôležitú spätnú väzbu k jednotlivým častiam predmetu po jeho preštudovaní a odučení v školskej praxi.

V príspevku sme sa zamerali hlavne na predstavenie prvej časti predmetu – programátorských etúd, ktoré tvoria východisko pre tvorbu náročnejších individuálnych či tímových projektov.

Pomocou ukážok troch programátorských etúd (Čítačky QR kódu, Asistenta pri cvičení, Hlasovania na internete) sme ukázali na využitie špecifických funkcionalít mobilného zariadenia (dotykovej obrazovky, kamery, senzorov zrýchlenia a priblíženia), internetového pripojenia, webovej databázy a aj na užitočnosť a využiteľnosť vytvorených aplikácii v praxi (vo vyučovaní, pri zábave aj pri fyzickom cvičení).

Veríme, že po finálnej aktualizácii metodických a učebných materiálov v júni 2020 na základe spätných väzieb od učiteľov a konzultantov z IT firiem, poskytneme učiteľom na všetkých našich školách materiály pre vyučovanie nového informatického predmetu, prípadne aj pre ich využitie vo všeobecnovzdelávacom predmete informatika, resp. na informatických krúžkoch.

POĎAKOVANIE

Tento článok bol vytvorený v rámci Národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje a v rámci projektu KEGA 029UKF-4/2018 Inovatívne metódy vo výučbe programovania v príprave učiteľov a IT odborníkov. Ďakujeme učiteľom-overovateľom a aj konzultantovi Ing. Martinovi Fedorovi zo spoločnosti T-systems za ich cenné pripomienky a komentáre k metodickým a učebným materiálom predmetu Programovanie mobilných zariadení.

BIBLIOGRAFICKÉ ODKAZY

[1] Národný projekt IT Akadémia – vzdelávanie pre 21. storočie. http://itakademia.sk/

[2] ŠNAJDER, Ľ., GUNIŠ, J., LOVÁSZOVÁ, G., MICHALIČKOVÁ, V.: Programovanie mobilných zariadení – Moodle kurz pre učiteľov – overovateľov a používateľov. https://vzdelavanie.itakademia.sk/course/view.php?id=16

[3] ŠNAJDER, Ľ. Metodika výučby programovania mobilných zariadení. In *DidInfo 2015* [CD]. Banská Bystrica : Univerzita Mateja Bela, Fakulta prírodných vied, 2015, s. 18-27. ISBN 9788055708522. Dostupné tiež na: http://www.didinfo.net/images/DidInfo/files/didinfo_2015.pdf

[4] ŠNAJDER, Ľ., GUNIŠ, J., ANTONI, Ľ.: Model of Teaching of Mobile Applications Programming in Non-Formal Education. In: *DidSci 2016 : proceedings of the 7th International Conference on Research in Didactics of the Sciences*.Cracow : Pedagogical University of Cracow, 2016. p. 151-154. ISBN 9788380840379.

[5] MICHALIČKOVÁ, V. *Programovanie mobilných aplikácií v prostredí MIT APP INVENTOR 2*. Nitra: Univerzita Konštantína Filozofa v Nitre, 2016, 93 s. ISBN 978-80-558-1101-7.

[6] TOMCSÁNYIOVÁ, M, TOMCSÁNYI, P. Vyučovanie programovania vývojom androidových aplikácií v App Inventore. In *DidInfo 2015* [CD]. Banská Bystrica : Univerzita Mateja Bela, Fakulta prírodných vied, 2015, s. 160-163. ISBN 9788055708522. Dostupné tiež na: http://www.didinfo.net/images/DidInfo/files/didinfo_2015.pdf

[7] TOMCSÁNYIOVÁ, M, Vytvárame mobilné aplikácie v prostredí App Inventor 2. http://edi.fmph.uniba.sk/~tomcsanyiova/SPIRIT/AI_kurz_pilot.pdf

[8] WOLBER, David, ABELSON, Hal, SPERTUS, Ellen, LOONEY, Liz. App Inventor 2: *Create your own Android Apps*. O'Reilly Media, 2014. ISBN 978-1491906842

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Umelá inteligencia ako súčasť kurikula informatiky na základných a stredných školách

Artificial intelligence as a part of Computer Science curriculum at primary and secondary schools

Zuzana Tkáčová

Prírodovedecká fakulta UPJŠ
Jesenná 5
040 01 Košice
Slovensko
zuzana.tkacova1@upjs.sk

Pedagogická fakulta UKF
Dražovská cesta 4
949 74 Nitra
Slovensko
zuzana.tkacova@ukf.sk

ABSTRACT

This paper presents some examples of approaches to teaching the topic of Artificial Intelligence at elementary and secondary schools, as even in the context of the Industrial Revolution 4.0, the issue of Artificial Intelligence is becoming important throughout society. Based on a questionnaire that we have worked out among secondary school students and computer science teachers, it shows that despite the interest of students and teachers, their awareness and readiness is not sufficient. Therefore, we present examples of three educational methodological materials that were prepared to teach Artificial Intelligence within the subject of computer science.

Keywords

Artificial intelligence. Curriculum. School informatics. Teaching methodology.

ABSTRAKT

V tomto článku predstavíme niektoré ukážky prístupov k výučbe témy umelá inteligencia na základných a stredných školách, nakoľko sa aj v kontexte priemyselnej revolúcie 4.0 stáva téma umelej inteligencie celospoločensky dôležitou. Na základe dotazníka, ktorý sme uskutočnili medzi žiakmi stredných škôl a učiteľmi informatiky sa ukazuje, že napriek záujmu zo strany žiakov aj učiteľov, ich informovanosť a pripravenosť nie je postačujúca. Preto ukážeme príklady troch edukačných metodických materiálov, ktoré boli pripravené pre výučbu umelej inteligencie v rámci kurikula predmetu informatika.

Kľúčové slová

Umelá inteligencia. Kurikulum. Školská informatika. Metodika výučby.

1 ÚVOD

Umelá inteligencia ako základ priemyselnej revolúcie 4.0 sa stala v posledných rokoch frekventovaným celospoločenským pojmom v médiách, ako aj v odborných kruhoch. Potreba pripraviť mladú generáciu na jej možné dopady v spoločnosti sa pomaly začína presúvať aj na oblasť vzdelávania. Zatiaľčo robotika si svoje miesto v školskej informatike aj vďaka popularite LEGO stavebníc a dobrej podpore robotických súťaží už našla, umelá inteligencia si zatiaľ ešte len hľadá svoj priestor, obsah a spôsob, ako sa dostať k žiakom na základných a stredných školách. Napriek tomu, že sa jedná o aktuálnu a atraktívnu tému, učitelia informatiky na ňu teoreticky, ani prakticky neboli dosiaľ pripravovaní. Preto sa postupne začínajú objavovať nové edukačné platformy, ktoré ponúkajú učiteľom vhodný didakticky spracovaný obsah a potrebné softvérové (ale aj hardvérové) nástroje na jeho sprístupňovanie, napr. MIT Cognimates [1] pre deti vo veku 7-10 rokov, Machine Learning for Kids [2] pre deti vo veku od 10 rokov, Microsoft AI School [3] alebo IBM TJBot [4] pre stredoškolákov. Ukazuje sa totiž, že tak ako sa považuje programovanie za súčasť digitálnej gramotnosti, nemenej dôležitý význam má aj umelá inteligencia (aj v prepojení na data science a strojové učenie) s ohľadom na jej silnejúci vplyv na spoločnosť.

Tomu však nezodpovedá v súčasnosti na Slovensku platný vzdelávací štandard pre predmet Informatika [5, 6], kde sa umelá inteligencia explicitne vôbec nevyskytuje - najbližšie k tejto téme nájdeme azda len tematickú oblasť Informačná spoločnosť – digitálne technológie v spoločnosti, avšak ani tu sa umelá inteligencia nedostala ani do obsahového štandardu. Kvôli úplnosti je možné poznamenať, že umelú inteligenciu nenájdeme ani v nových Cieľových požiadavkách na vedomosti a zručnosti maturantov z informatiky, ktoré vstupujú do platnosti v školskom roku 2018/2019. Preto je namieste otázka, ako vlastne budú naši žiaci pripravovaní na nástup umelej inteligencie, keď na to nemáme pripravených ani ich učiteľov, ani učebné materiály a nie je to špecifikované ani v platných vzdelávacích štandardoch.

Preto sme sa v rámci Národného projektu IT Akadémia – Vzdelávanie pre 21. storočie [7] rozhodli vytvoriť pilotnú sadu edukačných materiálov pre učiteľov, pomocou ktorých sa budú môcť otázkam umelej inteligencie na hodinách informatiky venovať. Nakoľko overovanie týchto materiálov vo výučbe je plánované na druhý polrok školského roka 2018/2019, pilotne sme ich otestovali počas našich workshopov pre žiakov stredných škôl a počas školení učiteľov informatiky, kde sme zároveň spracovali krátky prieskum postojov a poznatkov z oblasti umelej inteligencie u účastníkov spomínaných aktivít.

2 DOTAZNÍKOVÝ PRIESKUM

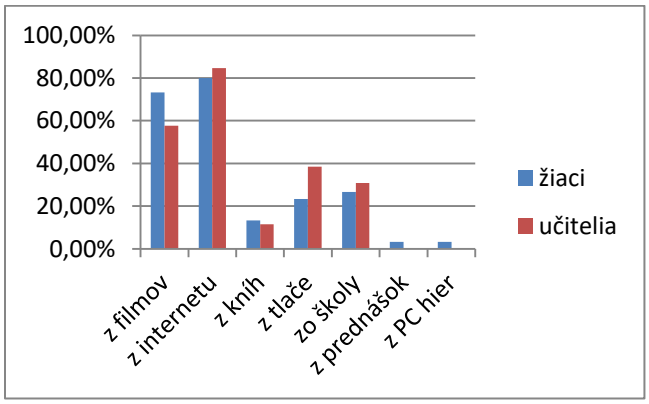
Prieskum bol realizovaný formou krátkeho online dotazníka na začiatku workshopu/školenia, v ktorom sme položili účastníkom (30 žiakov stredných škôl a 26 učiteľov informatiky) nasledovné otázky:

1. Odkiaľ ste dosiaľ čerpali informácie o umelej inteligencii?
2. Aký je Váš postoj k umelej inteligencii?

ISBN 978-80-557-1533-9, ISSN 2454-051X.

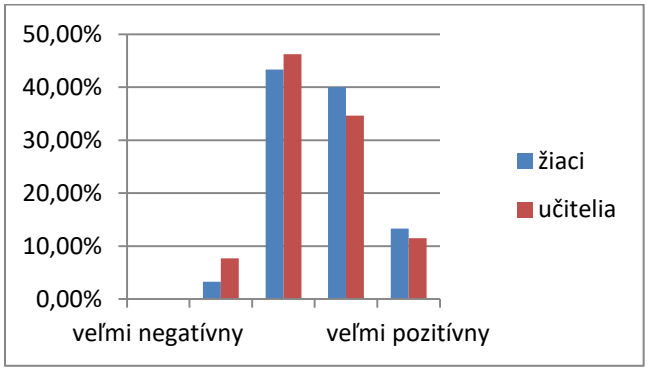
3. Aký je rozsah Vašich poznatkov o umelej inteligencii?
4. Uveďte niekoľko konkrétnych aplikácií, kde sa umelá inteligencia využíva už dnes.
5. Chceli by ste sa dozvedieť viac o umelej inteligencii?
6. Aké je podľa Vás dôležité učiť sa o umelej inteligencii?
7. V krátkosti vysvetlite, čo znamená "priemyselná revolúcia 4.0".

Na základe výsledkov nášho prieskumu vidieť, že dominantným zdrojom poznatkov pre učiteľov aj žiakov je internet a filmy (pozri Obrázok 1).



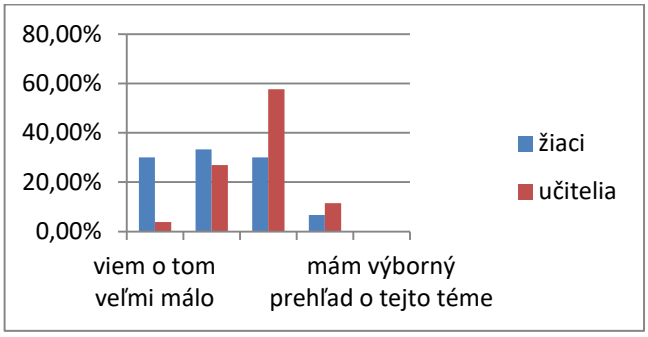
Obrázok 1: Odkiaľ ste dosiaľ čerpali informácie o umelej inteligencii?

Z pohľadu osobného postoja k umelej inteligencii, v oboch skupinách prevláda neutrálne alebo skôr pozitívne vnímanie tejto témy (pozri Obrázok 2).



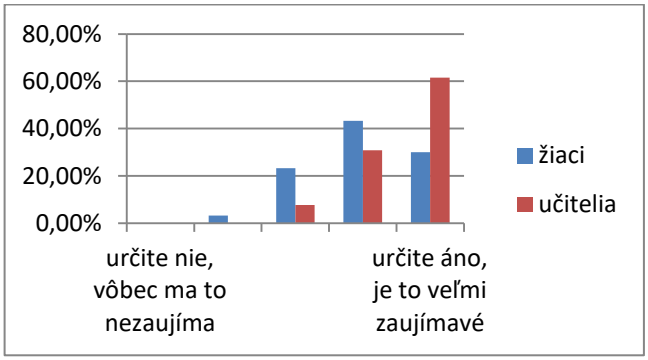
Obrázok 2: Aký je Váš postoj k umelej inteligencii?

Pri hodnotení svojich doterajších poznatkov o umelej inteligencii žiaci väčšinou priznali veľmi málo znalostí, zatiaľčo učitelia si verili trochu viac, aj keď väčšina z nich priznala skôr len priemerné vedomosti o tejto téme (pozri Obrázok 3) – napriek tomu bolo pomerne zarážajúce, že v otázke, kde mali uviesť nejaký konkrétny príklad na aplikáciu umelej inteligencie v súčasnosti až 15% učiteľov nedokázalo uviesť žiaden príklad. Žalostný bol výsledok v tejto otázke u žiakov, kde až 60% z respondentov nedokázalo uviesť žiadnu ukážkovú aplikáciu využitia umelej inteligencie.



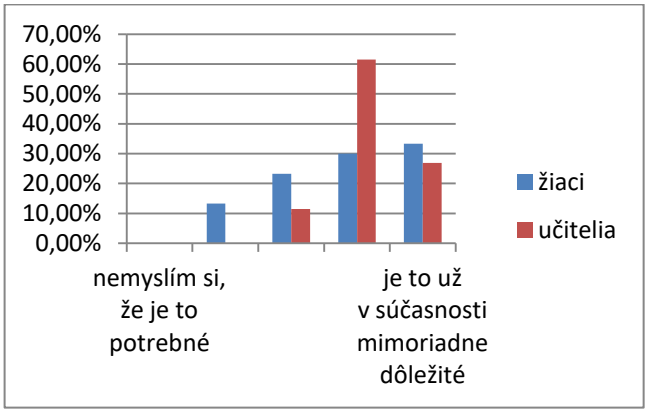
Obrázok 3: Aký je rozsah Vašich poznatkov o umelej inteligencii?

Pri otázke, či by sa chceli dozvedieť viac o umelej inteligencii, učitelia jednoznačne vyjadrili záujem o túto oblasť a pomerne veľký záujem je aj u žiakov (pozri Obrázok 4).



Obrázok 4: Chceli by ste sa dozvedieť viac o umelej inteligencii?

Pri otázke o učení sa o umelej inteligencii sa prakticky všetci respondenti zhodli na tom, že je to potrebné, resp. dôležité (pozri Obrázok 5), o čom svedčia aj výsledky poslednej, otvorenej otázky, v ktorej boli vyzvaní na krátke objasnenie termínu "priemyselná revolúcia 4.0" – u žiakov túto otázku zvládol len jediný žiak (3,3%), no aj v skupine učiteľov bola úspešnosť veľmi nízka – uspokojivé vysvetlenie poskytlo len 9 učiteľov z 26, t.j. len 34,6%.



Obrázok 5: Aké je podľa Vás dôležité učiť sa o umelej inteligencii?

Aj výsledky nášho malého prieskumu poukazujú na skutočnosť, že je potrebné začať riešiť túto situáciu.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

3 METODICKÉ MATERIÁLY

3.1 Základné informácie

Edukačné materiály k umelej inteligencii boli spracované pre základný stredoškolský kurz informatiky ako všeobecnovzdelávacieho predmetu na všetkých typoch škôl, avšak po drobných úpravách je možné niektoré jeho časti (predovšetkým prvú tému) odučiť aj v končiacich ročníkoch základnej školy (t.j. 8. alebo 9. ročník). Sú rozdelené tematicky do troch oblastí – objavujeme umelú inteligenciu, ako sa učí počítač a načo sú nám dáta – pričom každá z tém je rozpracovaná na jednu hodinu, príp. podľa uváženia učiteľa a časových podmienok školy aj na dve hodiny. Po metodickej stránke boli zvolené aktivizujúce a bádateľské prístupy, aby bolo žiakom umožnené počas hodiny aktívne skúmať a diskutovať o preberaných témach. Špecifické ciele, ktoré by mal žiak dosiahnuť po absolvovaní vyučovacích hodín s pripravenými materiálmi sú:

- uviesť niekoľko typických príkladov úloh (problémov) riešených pomocou umelej inteligencie,
- navrhnúť vhodné aplikačné využitie prvkov umelej inteligencie ako rozšírenie daného programu alebo aplikácie,
- vysvetliť vlastnými slovami základný princíp fungovania neurónovej siete,
- posúdiť možné dôsledky nevhodného výberu dát pre učenie neurónovej siete,
- vysvetliť vlastnými slovami základný princíp fungovania genetického algoritmu.

Pri napĺňaní uvedených cieľov sa nezameriavame na konkrétne softvérové produkty - tie slúžia len ako ukážkové nástroje na skúmanie. Taktiež neočakávame vlastnú programovú implementáciu neurónových sietí žiakmi – s ohľadom na plánovanú časovú dotáciu je hlavné ukázať na možnosti využitia prvkov umelej inteligencie pri rozšírení funkcionality iných programov (napr. hier, ovládacích programov pre Internet vecí, mobilných aplikácií a pod.). Nakoľko konkrétne softvérové nástroje sa postupne menia a vyvíjajú, zameriavame sa na generalizáciu konceptov a poznatkov, ktoré vzniknú pri samotnom žiackom skúmaní a experimentovaní.

3.2 Objavujeme umelú inteligenciu

Prvá téma ponúka úvod do problematiky, kde sa žiaci budú mať možnosť prakticky stretnúť s niektorými dostupnými aplikáciami umelej inteligencie.

Hodina sa začína krátkou brainwritingovou skupinovou aktivitou, v ktorej žiaci zapisujú na papiere čo najviac pojmov a príkladov, ktoré sa im spájajú s umelou inteligenciou. Pokračuje sa analýzou úvodného motivačného videa.

Dôležitou časťou hodiny je fáza skúmania rôznych webových nástrojov využívajúcich umelú inteligenciu:

1. Thing Translator [8] na rozpoznávanie objektov a preklad do iného jazyka,
2. IBM Speech-To-Text [9] na spracovanie a analýzu hovorenej reči,
3. IBM Watson Visual Recognition [10] na rozpoznávanie objektov na fotografiách,
4. IBM Natural Language Understanding [11] na analýzu textu.

Žiaci pracujú v skupinkách – každá skupina má zadaný konkrétny nástroj. Pri každom z týchto nástrojov majú žiaci krátke inštruktážne video a následne skúmajú daný nástroj, jeho vstupy, výstupy a navrhujú možnosti ďalšieho využitia.

Po krátkom predstavení výsledkov žiackeho skúmania sa pokračuje aktivitami pomocou nástrojov AutoDraw [12] a Quick, Draw! [13], kde vidia aj proces učenia sa umelej inteligencie na základe interakcie s užívateľom, čo bude východiskom pre ďalšiu hodinu (Ako sa učí počítač).

V závere hodiny si žiaci ešte rozšíria základný prehľad o rôzne možnosti praktických aplikácií umelej inteligencie, ktoré získajú štúdiom krátkych aktuálnych článkov z webu a spoločne vytvoria z prečítaných článkov prehľad pomocou interaktívnej nástenky, tzv. Padletu.

3.3 Ako sa učí počítač

Druhá hodina sa tematicky venuje predstaveniu fungovania dvoch dôležitých nástrojov umelej inteligencie – genetickým algoritmom a neurónovým sieťam. Cieľom nie je zvládnuť tieto postupy po programátorskej stránke, ale pochopiť ich princíp, výhody, ale aj úskalia, či obmedzenia. Preto sú aj aktivity zamerané na experimentovanie s parametrami už hotových programov a sledovanie ich vplyvu na správanie sa programov. Pri aktivitách sa bude používať aj online platforma Machine Learning for Kids [14] využívajúca Watson API.

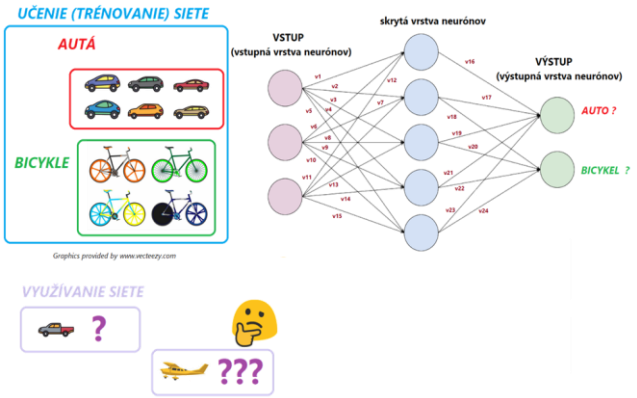
Úvodná bádateľská aktivita demonštruje spôsob rozpoznávania tvárí pri strojovom učení prostredníctvom extrakcie príznakov a následného prehľadávania databázy. Žiaci najprv analyzujú pripravené fotografie ľudí a vyplnia tabuľku so sledovanými príznakmi, na základe čoho potom identifikujú, kto je na novej fotografii.

Fáza skúmania pokračuje opäť prácou v skupinách – tentokrát žiaci musia pripraviť vhodné dáta na učenie neurónovej siete, natrénovať a otestovať neurónovú sieť a výsledok programátorsky zapracovať do jednoduchého programu v prostredí Scratch. Jednotlivé zadania programov sú z rôznych oblastí využitia umelej inteligencie:

1. analýza sentimentu (rozpoznáva text) – program mení výraz smajlíka na základe vety v dialógu s používateľom,
2. digitálny asistent (rozpoznáva text) – na základe dialógu s používateľom sa rozsvecujú/zhasínajú svetlá v triede, resp. sa spúšťa/vypína ventilátor,
3. biometrické rozpoznávanie tváre (rozpoznáva obrázky) – program slúži na odblokovanie/zablokovanie prístupu do virtuálneho mobilného telefónu,
4. OCR – rozpoznávanie písaného textu (rozpoznáva obrázky) – na určenie cieľa doručenia poštovej zásielky pomocou rukou písaného PSČ.

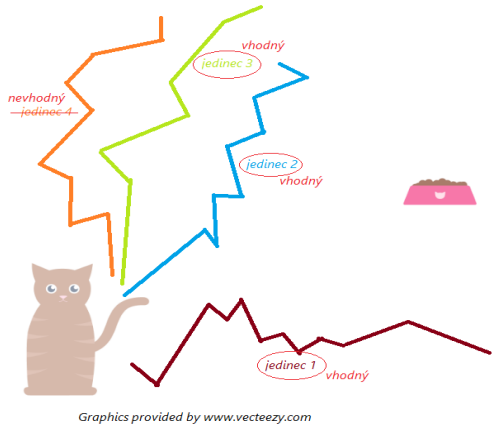
Po vytvorení jednotlivých programov je žiakom vysvetlená základná myšlienka neurónových sietí (pozri Obrázok 6). Ich vlastné bádanie by ich malo pri predošlých úlohách priviesť aj k potrebe začať sa intenzívne zamýšľať nad vhodnosťou dát, ktoré sa použijú pri učení neurónovej siete, ako aj k možným následkom, pri ich nevhodnej voľbe.

ISBN 978-80-557-1533-9, ISSN 2454-051X.



Obrázok 6: Objasnenie základnej myšlienky fungovania neurónovej siete

Pokračuje sa genetickými algoritmami, ktoré sú najprv prezentované v zjednodušenej podobe (pozri Obrázok 7) na príklade mačky hľadajúcej cestu k miske s potravou. Aby mačka cestu našla, najprv jej vytvoríme niekoľko náhodných ciest (budú predstavovať potenciálnych „jedincov" v novovzniknutej populácii), ktoré ju privedú do rôznej vzdialenosti od misky s potravou, na základe čoho vieme posúdiť o ich vhodnosti. Niekoľko najlepších ciest vyberieme do nasledujúceho kroku (v analógii s evolučným výberom tých najlepších jedincov na prežitie) a z nich môžeme vytvoriť ďalšie cesty k miske (t.j. ďalšiu generáciu jedincov) – opäť využijeme nástroje, ktoré žiaci už poznajú z hodín biológie, teda mutáciu (cestu v niektorých miestach náhodne pozmeníme) alebo kríženie (nová cesta vznikne spojením častí z rôznych ciest). Aby sme žiakom ukázali, že genetický algoritmus nemusí len hľadať cesty, využijeme nástroj na webovej stránke Grow Your Own Picture [15], kde žiaci formou krátkych experimentov skúmajú vplyv rôznych parametrov genetického algoritmu - pomocou rôznych geometrických tvarov je potrebné vyskladať portrét Mony Lisy. Cieľom tejto aktivity je okrem iného naučiť žiakov aj základom systematického experimentovania s pripraveným modelom.

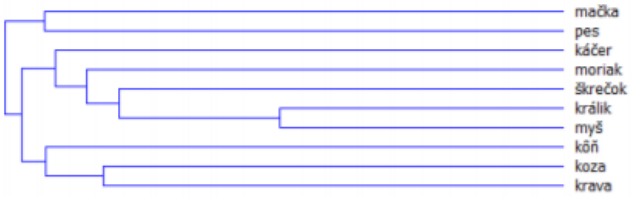


Obrázok 7: Objasnenie základnej myšlienky fungovania genetického algoritmu

3.4 Načo sú nám dáta

Posledná hodina je venovaná dátam a ich využitiu pri strojom učení a dolovaní v dátach. Na úvod hodiny diskutujú žiaci o krátkom motivačnom videu venovanom téme Big Data a následne skúmajú rôzne ukážky využitia dát na internete napr. pomocou sledovania dopravnej premávky v Google mapách, pomocou nástroja Google Trends [16] a SocialMention [17].

Nosnou časťou hodiny je bádateľská aktivita - práca v programe Orange [18]. Ide o praktickú ukážku strojového učenia – zhlukovanie obrázkov zvierat na základe ich podobnosti. Žiaci majú k dispozícii hotový tutoriál aj pracovný priečinok s dátami (obrázkami zvierat), pomocou ktorého vytvoria tzv. dendrogram (pozri Obrázok 8), v ktorom je však schovaná chyba, ktorá vznikla v dôsledku použitých vstupných dát. Úlohou žiakov je túto chybu odhaliť a identifikovať, prečo k nej došlo. Na samostatné skúmanie formou domáceho projektu môže učiteľ navrhnúť žiakom iné úlohy na zhlukovanie obrázkov podľa ich vlastných preferencií.



Obrázok 8 Ukážka výsledného dendrogramu

4 ZÁVER A DISKUSIA

Uvedené edukačné materiály sme začali testovali zatiaľ na žiakoch v rámci workshopov a na učiteľoch informatiky v rámci školení, kde sa nám potvrdilo, že pri spracovaní aktívnou a atraktívnou formou dokáže téma umelej inteligencie žiakov aj učiteľov zaujať a podporiť u nich záujem o diskusiu. Pilotné overovanie vo výučbe na stredných školách sa začne v druhom polroku školského roka 2018/2019.

Na rozšírenie aktivít z umelej inteligencie pre učiteľov aj žiakov plánujeme doplniť v budúcnosti aj ďalšie úlohy a nástroje, napr. z oblasti robotiky, generatívneho umenia, dátovej analytiky a strojového učenia.

POĎAKOVANIE

Tento článok bol vytvorený v rámci národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje.

BIBLIOGRAFICKÉ ODKAZY

[1] COGNIMATES. http://cognimates.me
[2] MACHINE LEARNING FOR KIDS. http://machinelearningforkids.co.uk
[3] AI SCHOOL. http://aischool.microsoft.com
[4] IBM TJBOT. http://ibmtjbot.github.io
[5] ŠTÁTNY VZDELÁVACÍ PROGRAM, *Informatika – nižšie stredné vzdelávanie.* [online]. Bratislava: Štátny pedagogický ústav, 2014 [cit. 2019-01-30]. 31 s. http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_nsv_2014.pdf
[6] ŠTÁTNY VZDELÁVACÍ PROGRAM, *Informatika – vyššie stredné vzdelávanie.* [online]. Bratislava: Štátny pedagogický ústav, 2014 [cit. 2019-01-30]. 18 s. http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_g_4_5_r.pdf

ISBN 978-80-557-1533-9, ISSN 2454-051X.

[7] IT AKADÉMIA – VZDELÁVANIE PRE 21. STOROČIE. http://itakademia.sk
[8] THING TRANSLATOR. https://thing-translator.appspot.com
[9] IBM SPEECH-TO-TEXT. https://www.ibm.com/watson/services/speech-to-text
[10] IBM WATSON VISUAL RECOGNITION. https://www.ibm.com/watson/services/visual-recognition
[11] IBM NATURAL LANGUAGE UNDERSTANDING. https://www.ibm.com/watson/services/natural-language-understanding
[12] AUTODRAW. https://www.autodraw.com
[13] QUICK, DRAW!. https://quickdraw.withgoogle.com
[14] MACHINE LEARNING FOR KIDS. https://machinelearningforkids.co.uk
[15] GROW YOUR OWN PICTURE. https://chriscummins.cc/s/genetics
[16] GOOGLE TRENDS. https://trends.google.com/trends
[17] SOCIALMENTION. http://www.socialmention.com
[18] ORANGE. https://orange.biolab.si

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Úspešnosť súťažiacich s rôznym počtom vynechaných úloh v súťaži iBobor

Achievements of Contestants Having Different Number of Skipped Tasks in the Bebras Challenge

Peter Tomcsányi
FMFI UK Bratislava
Mlynská Dolina
842 48 Bratislava
Slovensko
peter.tomcsanyi@fmph.uniba.sk

ABSTRACT

In this article, we analyze the results of the Bebras Challenge in Slovakia in order to better understand how contestants use the option of skipping a task, i.e. not solving it for not being penalized for an incorrect answer. We want to find out whether students are aware of this option, whether they use it in order to optimize their results and whether we can, by analyzing the contest results by discern the situations when the use of this option has helped the contestant and when it hasn't, and whether we can justify why it happened. The article gives some answers, suggest others as well as it provides new questions and suggestions for further exploration.

Keywords

Bebras Challenge. Evaluating Multiple-Choice Tasks. Skipping tasks in the Challenge.

ABSTRAKT

V tomto článku sa zaoberáme analýzou výsledkov súťaže Informatický Bobor na Slovensku za účelom lepšieho pochopenia ako súťažiaci využívajú možnosť vynechať úlohu, teda neodpovedať na ňu a nebyť prípadne penalizovaný za nesprávnu odpoveď. Chceme zistiť, či si žiaci uvedomujú túto možnosť, či ju používajú v snahe optimalizovať svoj výsledok a či vieme pomocou analýzy výsledkov rozlíšiť situácie kedy využitie tejto možnosti pomohlo súťažiacim, kedy nie, a či vieme zdôvodniť prečo sa tak stalo. Článok dáva niektoré odpovede, iné naznačuje, pričom dáva nové otázky a podnety na ďalšie skúmanie.

Kľúčové slová

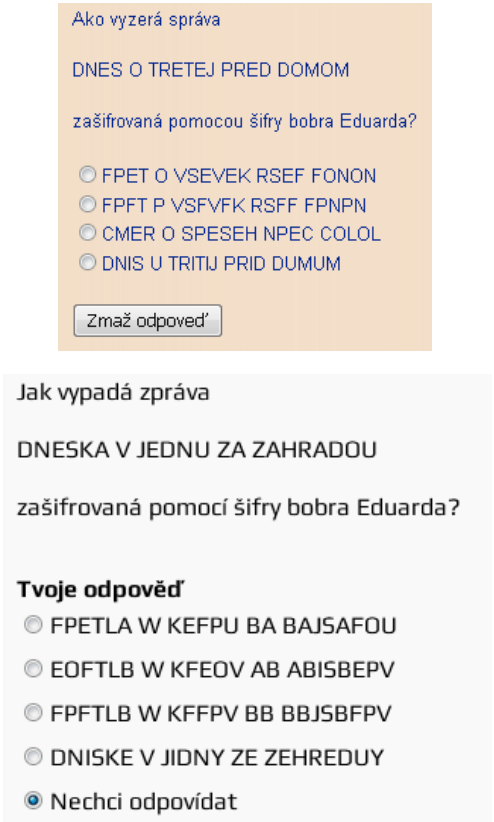
Súťaž iBobor. Hodnotenie úloh s výberom možností. Vynechanie úlohy v súťaži

1 ÚVOD

V súťaži Informatický Bobor (iBobor) [1] používame bodovanie, ktoré penalizuje nesprávne odpovede oproti vynechaniu odpovede. Používame všeobecne známy vzorec pre úlohy s výberom práve jednej z n odpovedí: ak správna odpoveď je za p bodov, tak nesprávna je penalizovaná p/(n-1) bodmi a za vynechanú odpoveď sa neudeľuje žiaden bod. Aby výsledok súťažiaceho nebol nikdy záporný, pripočíta sa k takto vypočítaným bodom úvodný bonus tak, aby minimálny možný počet bodov bol nula. Toto bodovanie sme prebrali z pôvodnej súťaže Bebras v Litve a pokiaľ vieme, aplikuje ho aj väčšina ostatných krajín zapojených do Informatického bobra (hnutia Bebras).

Už v prvých ročníkoch organizovania súťaže sme si kládli otázku, či si súťažiaci uvedomujú, že môžu vedome neodpovedať na úlohu keď majú vážne pochybnosti o správnosti svojej odpovede alebo o pochopení úlohy a ak aj áno, či naozaj túto možnosť vedome používajú (alebo vynechávajú úlohy keď nestíhajú), ale hlavne, či môžeme nejako kvantifikovať kedy im vynechanie odpovede skôr pomohlo alebo nie.

V štatistickom porovnaní výsledkov úloh použitých v národných súťažiach Informatického bobra na Slovensku a v Čechách v rokoch 2007 a 2008 [2] sme spolu s kolegom Jiřím Vaníčkom konštatovali, že súťažiaci v Česku vynechávajú odpovede častejšie než súťažiaci na Slovensku. Toto sme zdôvodnili tým, že organizátori v Česku kládli väčší dôraz na vysvetlenie možnosti neodpovedať.

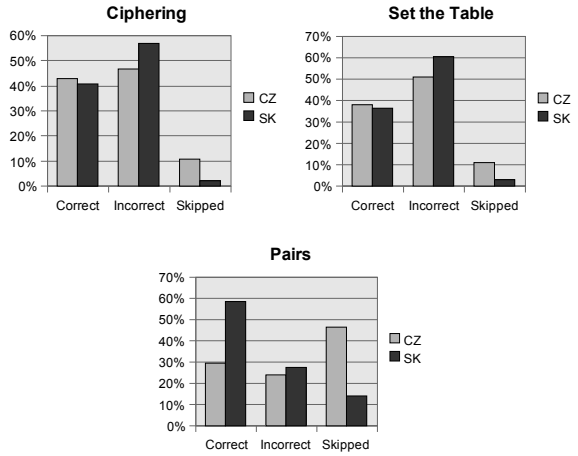


Obrázok 1: Vizualizácia možnosti neodpovedať v slovenskom (hore, tlačidlo Zmaž odpoveď) a českom (dole, voľba Nechci odpovídat) súťažnom systéme.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

Navyše rolu zrejme zohráva aj odlišná vizualizácia možnosti neodpovedať v českom a slovenskom súťažnom systéme (viď Obrázok 1).

Zároveň sme našli aj zaujímavé príklady úloh, kde stratégia radšej neodpovedať dopadla priaznivo pre českých súťažiacich (pri rovnakom podieli správnych odpovedí mali podiel nesprávnych odpovedí menší o to, o čo mali väčší podiel vynechaných odpovedí) aj príklady, kde boli práve o počet vynechaných odpovedí úspešnejší slovenskí súťažiaci (viď Obrázok 2).



Obrázok 2: Graf podielu správnych, nesprávnych a vynechaných odpovedí pre dve úlohy, kde bola úspešnejšia stratégia "keď nie som si istý tak neodpovedám" (hore) a jednu úlohu, kde nebola úspešnejšia (dole) [2]

V ďalších rokoch sme síce občas diskutovali o zmienených otázkach v kolektíve organizátorov súťaží, ale až v tomto článku sa budeme snažiť viac priblížiť k odpovediam analýzou údajov (zatiaľ len) zo slovenského iBobra.

2 PLÁN

Otázky sme si technicky upresnili, resp. sme sa zamerali užšie na jeden ich aspekt. Budeme skúmať, či sa v skupine súťažiacich, ktorí vynechali nejaký počet úloh, vyskytuje menší alebo väčší podiel úspešných riešiteľov, než v skupine súťažiacich, ktorí vyriešili všetky úlohy.

Úspešní riešitelia sú všetci súťažiaci, ktorí dosiahli aspoň polovicu "kladných bodov". V kategórii Drobec je to 45 bodov, v kategórii Bobrík je to 60 bodov, v kategórii Nevidiaci ZŠ a Nevidiaci SŠ je to 30 bodov a v ostatných kategóriách je to 50 bodov [3].

Úspešný riešiteľ je definovaný absolútnym počtom bodov a obtiažnosť úloh je v rôznych kategóriách a v rôznych ročníkoch súťaže rôzna. Preto sa líši aj podiel úspešných riešiteľov. Napríklad v kategórii Junior (1. a 2. ročník strednej školy, 15 - 17 roční súťažiaci) v roku 2017 bolo len 9,5% úspešných riešiteľov, zatiaľ čo v kategórii Kadet (8. a 9. ročník základnej školy, 13 - 15 roční súťažiaci) v roku 2018 to bolo až 50%.

Preto musíme posudzovať každú kategóriu a každý ročník súťaže zvlášť. Do výskumu sme zaradili všetky kategórie z posledných 4 ročníkov súťaže (2015 až 2018) okrem kategórií pre nevidiacich, ktorí majú príliš málo účastníkov pre účely štatistického skúmania.

3 VÝSKUM

Keď súťažiaci vynechá jednu úlohu, tak už zjavne nemôže dosiahnuť najvyšší počet bodov. Ale nemôže dosiahnuť ani najnižší počet bodov (lebo za tú jednu úlohu nebude penalizovaný). Ale keďže penalizácia za nesprávnu odpoveď je menšia než odmena za správnu odpoveď, intuícia nám hovorí, že sa vynechaním úlohy vzďaľujeme viac od hornej hranice než od dolnej. Teda medzi súťažiacimi, ktorí vynechajú úlohu by nemal byť väčší počet úspešných riešiteľov než medzi tými, ktorí zodpovedali na všetky úlohy. Výsledky niektorých kategórií ale ukazujú, že je to možné.

Napríklad v kategórii Senior (3. a vyššia trieda strednej školy, 17 - 19 roční súťažiaci) v roku 2018 súťažilo 4086 žiakov a z nich bolo 1057 úspešných riešiteľov, čo je 25,9%. Ak rozdelíme všetkých súťažiacich do 16 skupín podľa počtu nezodpovedaných otázok a pre každú spočítame počet úspešných riešiteľov, dostaneme na prvý pohľad prekvapivú tabuľku (viď Tabuľka 1). V nej vidíme, že len 24,7% súťažiacich, ktorí riešili všetky úlohy, boli úspešnými riešiteľmi, zatiaľ čo spomedzi tých, ktorí vynechali práve dve úlohy, bolo až 43,5% úspešných riešiteľov (toto *maximum* je zvýraznené červenou farbou). Stĺpec *% kumulatívne* ukazuje podiel úspešných riešiteľov medzi tými, ktorí vynechali jednu až n úloh. Vidíme, že *kumulatívne maximum* (zvýraznené zelenou farbou) je v riadku 3, teda že 37,7% tých, ktorí vynechali 1 až 3 úlohy bolo úspešných.

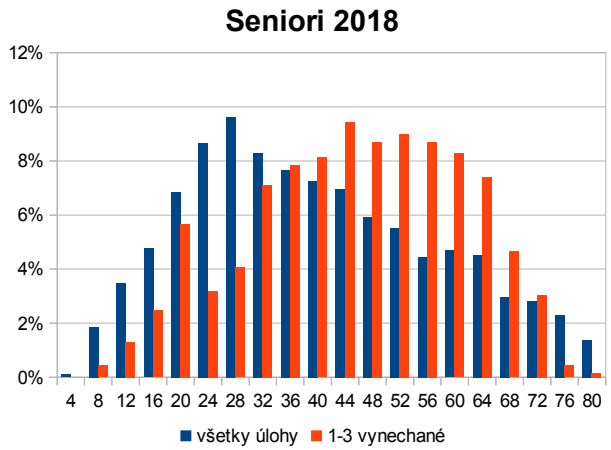
Tabuľka 1: Analýza počtu úspešných riešiteľov v kategórii Senior v roku 2018

Seniori 2018				
Vynechané	**Súťažiaci**	**Úspešní**	**% úspešných**	**% kumulatívne**
0	2922	721	24,7%	
1	297	97	32,7%	32,7%
2	207	90	**43,5%**	37,1%
3	185	73	39,5%	**37,7%**
4	156	46	29,5%	36,2%
5	105	21	20,0%	34,4%
6	87	9	10,3%	32,4%
7	47	0	0,0%	31,0%
8	23	0	0,0%	30,4%
9	22	0	0,0%	29,8%
10	11	0	0,0%	29,5%
11	3	0	0,0%	29,4%
12	7	0	0,0%	29,2%
13	1	0	0,0%	29,2%
14	2	0	0,0%	29,1%
15	11	0	0,0%	28,9%
spolu	4086	1057	25,9%	

Štatistický test nezávislosti pre kontingenčnú tabuľku [4] ukazuje, že rozdiely v podieloch úspešných riešiteľov, tak v prípade maxima ako aj v prípade kumulatívneho maxima, sú štatisticky významné na hladine významnosti 0,01.

Obrázok 3 porovnáva histogramy rozdelenia počtu bodov súťažiacich ktorí vyriešili všetky úlohy a súťažiacich, ktorí vynechali jednu až tri úlohy.

ISBN 978-80-557-1533-9, ISSN 2454-051X.



Obrázok 3: Rozdelenie počtu bodov súťažiacich, ktorí vyriešili všetky úlohy a tých, ktorí vynechali jednu až tri úlohy v kategórii Senior v roku 2018.

Takéto výsledky ale nie sú pravidlom. Napríklad v kategórii Benjamín v roku 2015 (vtedy to bola 5. - 7. trieda základnej školy, 10 - 13 roční súťažiaci) maximum podielu úspešných riešiteľov bol medzi súťažiacimi, ktorí nevynechali žiadnu úlohu (viď Tabuľka 2).

Tabuľka 2: Analýza počtu úspešných riešiteľov v kategórii Benjamín v roku 2015

Benjamíni 2015				
Vynechané	**Súťažiaci**	**Úspešní**	**% úspešných**	**% kumulatívne**
0	18008	3002	16,7%	
1	2363	388	16,4%	16,4%
2	908	129	14,2%	15,8%
3	581	79	13,6%	15,5%
4	415	35	8,4%	14,8%
5	217	17	7,8%	14,5%
6	133	1	0,8%	14,1%
7	83	0	0,0%	13,8%
8	41	0	0,0%	13,7%
9	22	0	0,0%	13,6%
10	17	0	0,0%	13,6%
11	12	0	0,0%	13,5%
12	11	0	0,0%	13,5%
13	9	0	0,0%	13,5%
14	21	0	0,0%	13,4%
15	85	0	0,0%	13,2%
spolu	22926	3651	15,9%	

V ďalšom sme výskyt *fenoménu* keď súťažiaci, ktorí nevynechali žiadnu úlohu, nemajú najväčší podiel úspešných riešiteľov, pričom rozdiel je štatisticky významný na hladine významnosti aspoň 0,05, sledovali vo všetkých kategóriách súťaže za posledné 4 roky (2015 - 2018). Výsledky zhŕňa Tabuľka 3.

V nej vidíme, že za jeden z faktorov môžeme uvažovať vek súťažiacich. V kategóriách Drobec a Bobrík nenastal v sledovanom období fenomén nikdy, ale v kategóriách Kadet, Junior a Senior nastal v troch zo štyroch sledovaných rokov.

V kategóriách Junior 2015 a Senior 2015 nie je pozorovaný rozdiel štatisticky významný.

Tabuľka 3: Zhrnutie analýzy všetkých kategórií za štyri roky

	feno-mén?	% úspešných					
		všetci	všetky úlohy	maximum		kum. max.	
				n	hodnota	n	hodnota
Drobec 2018	nie	13,0%					
Bobrík 2018	nie	36,2%					
Benjamín 2018	nie	44,3%					
Kadet 2018	nie	50,4%					
Junior 2018	áno	32,4%	32,2%	2	41,8%	2	41,3%
Senior 2018	áno	25,9%	24,7%	2	43,5%	3	37,7%
Drobec 2017	nie	28,1%					
Bobrík 2017	nie	35,0%					
Benjamín 2017	áno	27,2%	27,2%	1	31,3%	1	31,3%
Kadet 2017	áno	34,7%	34,2%	1	43,0%	1	43,0%
Junior 2017	áno	9,5%	9,3%	2	15,4%	2	12,3%
Senior 2017	áno	29,3%	29,3%	1	36,9%	1	36,9%
Bobrík 2016	nie	23,6%					
Benjamín 2016	áno	23,2%	22,9%	1	37,3%	1	37,3%
Kadet 2016	áno	36,9%	37,1%	2	46,5%	2	42,8%
Junior 2016	áno	15,1%	15,0%	1	20,1%	1	20,1%
Senior 2016	áno	26,7%	27,4%	2	35,3%	2	31,5%
Bobrík 2015	nie	16,4%					
Benjamín 2015	nie	15,9%					
Kadet 2015	áno	17,3%	16,8%	2	23,2%	2	21,9%
Junior 2015	nie	16,1%	16,6%	2	19,0%	3	18,1%
Senior 2015	nie	17,7%	18,5%	3	20,6%	3	19,3%

4 ZÁVER

Skúmali sme výsledky súťaže iBobor. Zistili sme, že v niektorých prípadoch bol najväčší pomer úspešných riešiteľov medzi tými súťažiacimi, ktorí nevynechali žiadnu úlohu, inokedy medzi súťažiacimi, ktorí niekoľko úloh (typicky jednu, dve alebo tri) vynechali.

Zdá sa, že viditeľný vplyv na tom, kedy nastane prvá a kedy druhá situácia, má vek súťažiacich. V dvoch najmladších kategóriách Drobec a Bobrík je vždy najviac úspešných riešiteľov medzi tými, ktorí sa pokúsili riešiť všetky úlohy. V najstarších kategóriách Kadet, Junior a Senior je to často naopak. Naskytá sa vysvetlenie, že mladšie deti nedokážu dostatočne vedome využívať možnosť preskočiť úlohu, kde hrozí strhnutie bodov za nesprávnu odpoveď, na optimalizovanie svojho výsledku.

Prečo bola v niektorých kategóriách v niektorých rokoch jedna situácia v iných druhá, sme nevysvetlili. Jedna z možností, ktorú sa oplatí preskúmať je počet veľmi ťažkých úloh v danom roku, prípadne výskyt úloh, kde viacerí súťažiaci nepochopili zadanie a preto sa im mohla zdať jednoduchšia než v skutočnosti bola.

BIBLIOGRAFICKÉ ODKAZY

[1] IBOBOR.SK. http://ibobor.sk

[2] TOMCSÁNYI, P., VANÍČEK, J. International Comparison of Problems from an Informatics Contest. In Mechlova, E. (ed.) *Proceedings of the Conference: Information and CommunicationTechnology in Education*, 2009, Ostrava University Editorial Centre, pp. 219 - 223.ISBN 978-80-7368-459-4

[3] IBOBOR.SK vysvetlivky k štatistickému vyhodnoteniu ročníka 2018. http://ibobor.sk/vyhodnotenie2018.php#vysv

[4] Chráska, M. *Metody pedagogického výzkumu.* Grada, Praha 2008, ISBN 978-80-247-1369-4

Programovanie v JavaScripte v príprave učiteľov informatiky

Programming in JavaScript in training of informatics teachers

Monika Tomcsányiová
Katedra didaktiky matematiky, fyziky a informatiky, FMFI Univerzity Komenského v Bratislave
Mlynská dolina
842 48 Bratislava
Slovensko
tomcsanyiova@fmph.uniba.sk

ABSTRACT

In the article, we describe the Programming in JavaScript in the bachelor study of future informatics teachers at the Faculty of Mathematics, Physics and Informatics of Comenius University in Bratislava. We mention the reasons why we chose JS programming language, what are the goals and content of this course. On specific assignments, we will show what topics we have been doing and how to solve some of the tasks we have prepared for the students. At the end of the article, we are focusing on the research that we performed during the course. We summarize our observations and suggest the changes we made in the course, respectively. which we will do in the school year 2018/19.

Keywords

JavaScript. Programming. Informatics Teacher.

ABSTRAKT

V článku popisujeme kurz Programovanie v JavaScripte v bakalárskom štúdiu budúcich učiteľov informatiky na Fakulte matematiky, fyziky a informatiky Univerzity Komenského v Bratislave. Uvádzame dôvody, prečo sme si vybrali práve programovací jazyk JS, aké sú ciele a obsah tohto kurzu. Na konkrétnych zadaniach ukážeme, akým témam sme sa postupne venovali a uvedieme riešenia niektorých úloh, ktoré sme pre študentov pripravili. V závere článku sa venujeme výskumu, ktorý sme zrealizovali počas dvoch behov kurzu. Zhrnieme naše pozorovania a navrhneme zmeny, ktoré sme v náplni kurzu urobili, resp. ktoré urobíme, v školskom roku 2018/19.

Kľúčové slová

JavaScript. Programovanie. Príprava učiteľov informatiky.

1. ÚVOD

Príprava budúcich učiteľov informatiky na FMFI UK v Bratislave má svoju už viac ako tridsaťročnú tradíciu. Jej nevyhnutnou súčasťou sú aj prednášky a cvičenia, ktoré učia študentov základy programovania. Tieto aktivity často nazývame aj ako „kurzy programovania“. V súčasnej akreditácii učiteľského štúdia, pozri [1], sa študenti vo svojom úvodnom trojsemestrálnom kurze najprv oboznamujú s jazykom Python. V kurze Algoritmy a údajové štruktúry, v štvrtom semestri, pokračujú v programovaní v Pythone, ale zameriavajú sa v ňom na pochopenie a programovanie rôznych algoritmov pre známe údajové štruktúry. Pri akreditovaní učiteľského bakalárskeho študijného programu v kombinácii s informatikou v roku 2014 sme sa pomerne dlho zamýšľali nad tým, aké ďalšie programovacie jazyky, okrem Pythonu, majú byť jeho súčasťou. Do piateho semestra sme zaradili kurz programovania v Imagine Logo. Tento programovací jazyk má na Slovensku svoju tradíciu predovšetkým pre žiakov na druhom stupni základných škôl. S ďalšími programovacími jazykmi ako Scratch, či Karel sa študenti oboznámia v rámci predmetu Digitálne technológie. Čo sa týka výberových predmetov, v predchádzajúcej akreditácii bol pre šiesty semester navrhnutý kurz Programovanie v Jave. V čase novej akreditácie však z výskumov pre jazyk Java, ako jazyk, ktorým sa dajú programovať aplikácie pre web, vyplynulo, že viaceré webové prehliadače prestávajú podporovať tzv. java-applety, na ktorých programovanie bol tento kurz orientovaný. Do popredia sa v tom čase dostával JavaScript (ďalej JS) ako jazyk, ktorým sa budú dať, vďaka nastupujúcej technológii HTML 5, programovať aj interaktívne aplikácie, v ktorých sa bude dať pomerne ľahko pracovať s obrázkami. Aj to bol jeden z dôvodov, prečo sme pre budúcich učiteľov informatiky navrhli, a v minulých dvoch rokoch aj implementovali, dvojhodinový kurz Programovanie v JavaScripte.

Pre tento kurz sme si stanovili pomerne špecifické ciele, ktoré nie sú úplne bežné pre iné kurzy programovania v JS, ktoré sa väčšinou sústreďujú na samotný jazyk a jeho využívanie pre úpravu HTML dokumentov, viac pozri časť 3.1. Na dosiahnutie týchto cieľov sme pre kurz navrhli sériu úloh, ktorá študentov cielene viedla k využívaniu grafickej plochy a prácu s obrázkami, resp. kreslenými útvarmi, a následne k spracovaniu udalostí myši, pozri časť 4 Ukážky zadaní. Pre účely výskumu sme študentov pri riešení nami navrhnutých úloh pozorovali, výsledky pozorovaní sme si zaznamenávali a následne sme vylepšovali a upravovali zadania úloh tak, aby vhodne a cielene viedli študentov k vytváraniu malých interaktívnych aplikácií s využitím grafiky.

2. PREČO JS

Pre programovanie grafických webových aplikácií sa programovací jazyk Python, s ktorým sa študenti stretávajú vo svojom základnom kurze programovania, vo svojej základnej inštalácii, celkom nehodí. S cieľom, ktorý sme mali, teda programovať interaktívne grafické webové aplikácie, sme mohli uvažovať o tom, že do vyučovania zaradíme niekoľko ďalších

modulov jazyka Python (napr. pygame) alebo aj smerom, že sa vrátime k Jave a jej knižniciam. Tomu sme sa však chceli pri príprave budúcich učiteľov, z ktorých nemajú byť profesionálni programátori, vyhnúť. Aj preto sme sa rozhodli pre programovací jazyk JS, v ktorom sa dajú programovať malé interaktívne grafické programy s podporou plátna canvas a technológie HTML 5.

3. O KURZE PROGRAMOVANIE V JS

Kurz je určený pre študentov učiteľského štúdia v kombinácii s informatikou, ktorí už majú skúsenosti s programovaním vo viacerých programovacích jazykoch, ktorí sa stretli aspoň so základmi práce s HTML kódom, ktorí majú hlbší záujem o programovanie a chceli by programovať aplikácie, ktoré fungujú vo webových prehliadačoch.

3.1 Ciele kurzu

Cieľom kurzu je navrhnúť a zrealizovať postupnosť tém a úloh, ktorá by viedla k tomu, aby si na jej konci študent dokázal pripraviť HTML stránku s javascriptovým programom, ktorý bude jednoduchou hrou (podobne ako to študenti robili v jazyku Python, Imagine Logo, či Scratch). Súčasťou týchto programov je vždy vykresľovanie niekoľkých obrázkov, ich animácia, klikanie a ťahanie obrázkov myšou. Preberané témy kurzu vedú študenta k tomu, aby bol na jeho konci schopný navrhnúť a naprogramovať svoj vlastný projekt – hru, v ktorej využije všetky získané vedomosti. Kurz je vnútorne rozdelený na tri časti približne v pomere 4:6:2. V prvých štyroch hodinách kurzu sa študenti oboznamujú s novým programovacím jazykom JS a jeho podobnosťou, resp. odlišnosťou, od jazyka Python, ktorý poznajú z predchádzajúcich semestrov svojho štúdia. Zároveň, ak sa doteraz nestretli s vytváraním webových stránok pomocou jazyka HTML, pri vytváraní aplikácií sa oboznamujú aj s jeho základnými príkazmi. Ďalších šesť hodín je venovaných spolupráci JS a canvas v HTML 5 a tiež využitiu polí pri návrhu a zobrazovaní rôznych malých grafických aplikácií, viac pozri v časti 4. Posledné dve hodiny sú venované vývoju hier a oboznamovaniu sa s knižnicou `sprites.js`, viac o knižnici pozri v článku [2].

3.2 Náplň kurzu

Prvé stretnutia kurzu sú zamerané na to, aby si študenti uvedomili aké sú rozdiely medzi programovacím jazykom Python, resp. Pascal, či Imagine Logo, a JS. V rámci riešení predložených úloh majú využiť rovnaké programátorské koncepty v JS s akými sa stretli v iných programovacích jazykoch. V čase priebehu kurzu majú študenti za sebou okrem základného kurzu v Pythone (Pascale) aj pomerne pokročilý kurz programovania v Imagine Logo. Pri riešení úloh kurzu študenti najprv používajú konzolový režim v prehliadači Chrome. Neskôr píšu riešenia úloh do webového online JS nástroja `JS.do`, pozri [3]. Následne od konca druhého stretnutia, resp. od tretieho stretnutia, využívajú vlastný HTML dokument, do ktorého vpisujú JS program. Tým získajú úplnú programátorskú kontrolu nad vlastnými dokumentami a programami vo webovom prehliadači. V tejto úvodnej časti kurzu študenti postupne prešli všetky základné programátorské konštrukcie ako postupnosť príkazov, cyklus a podmienky a dostali sa až k vytváraniu vlastných funkcií. Z údajových štruktúr pracovali s poľom, pri riešení úloh využili niektoré základné operácie na jednorozmerných číselných poliach. Od piateho stretnutia sa kurz venuje možnostiam komponentu canvas a spracovaniu udalostí myši a klávesnice v ňom. Úlohy sú zamerané na techniky spracovania kliknutia v JS a reakcií naň. Udalosť ťahania je náročnejšia a preto je študentom predkladaná formou šablóny. Neskôr začnú študenti využívať pripravenú knižnicu sprites.js, pozri [2], pomocou ktorej riešia zadania, v ktorých využívajú viaceré obrázky a zmenu obrázkov pre jeden objekt. Na dvoch posledných stretnutiach študenti samostatne programujú väčšie projekty, v ktorých využijú obrázky a objekty Sprite z knižnice sprites.js, pozri časti 4.4 a časť 5 o priebehu výskumu.

3.3 Obsah jednotlivých stretnutí

Obsah jednotlivých stretnutí sme navrhli pred začiatkom prvého behu kurzu a v rámci prvého behu sme ho mierne upravili. Po prvom behu sme urobili radikálnejšiu zmenu s ohľadom na náročnosť spracovania obrázkov v JS. V súčasnosti má kurz takýto obsah:

1 úvodné stretnutie, diskusia o dôvodoch, prečo sme prešli na JS | vytváranie premenných v JS | celočíselné premenné, reťazce | booleovské výrazy, operácie s nimi
2 JS a HTML tagy | cyklus for | generovanie náhodného čísla
3 polia, jednorozmerné a dvojrozmerné celočíselné polia | náhodné generovanie čísel do poľa | hra Obesenec, pozri 4.1
4 riešenie úloh, v ktorých sa využíva údajová štruktúra pole
5 vykreslenie poľa do canvas pomocou štvorčekov
6 myš a canvas
7 klikanie na konkrétne miesta v grafickej ploche | hra Vypni svetlá
8 knižnica sprites.js | ťahacie aktivity
9 knižnica sprites.js a tvorba interaktívnych programov
10 klikacie aktivity pomocou knižnice sprites.js
11, 12 programovanie náročnejších projektov

4. UKÁŽKY ZADANÍ

Pri učení sa programovať takmer v akomkoľvek programovacom jazyku, či prostredí, sú zrejme najdôležitejšie zadania úloh, ktoré študenti riešia počas behu kurzu. Závisí na námetoch úloh a na požiadavkách pri ich riešení, aké praktické skúsenosti študent z programovacieho jazyka a techník reálne nadobúda. Myslíme si, že pri učení sa programovať, sa využívajú dva hlavné prúdy: programovanie tzv. konzolových úloh alebo druhý prístup, v ktorom študenti riešia predovšetkým úlohy s grafikou a interakciou s používateľom ich programov napr. úlohy zamerané na zobrazovanie obrázkov a reagovanie na udalosti myši, pozri Obrázok 1.

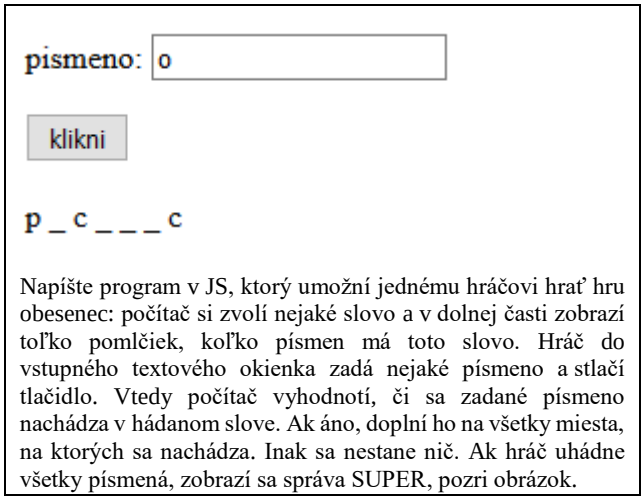
Obrázok 1: Počítač generuje odpovede „Áno.“, „Nie.“. Do pôvodnej úlohy zo súťaže iBobor s jedným konkrétnym riešením použitie JS vnieslo variabilitu, takže žiak musí pozorne čítať podmienky, overovať ich správnosť na obrázku a myšou kliknúť na obrázok, pre ktorý platia všetky tri výroky.

Námety konzolových úloh sa najčastejšie venujú matematickým výpočtom a transformácii vstupov na výstupy pomocou nejakého známeho alebo menej známeho algoritmu. Je nepochybné, že riešením takých úloh dokážu študenti zvládnuť základné programátorské konštrukcie takmer akéhokoľvek jazyka a rovnako aj prácu s údajovými štruktúrami. Pre úlohy, ktoré sa orientujú na grafiku môže byť situácia trochu odlišná. Pri ich riešení sa študenti musia oboznámiť aspoň so základnou grafickou knižnicou, ktorá im umožní zobrazovať obrázky, resp. aspoň kresliť základné geometrické útvary. Pri zostavovaní a riešení týchto úloh sa môže stať, že samotné riešenie nekladie dostatočný dôraz na precvičenie cyklov a podmienok, pretože sa zadania sa „ponáhľajú“ využívať tieto programátorské konštrukcie v grafike, ktorá môže zvádzať k rýchlejšiemu tempu a kvôli tomu menej hlbokému pochopeniu príslušného konceptu. Môže sa tiež stať, že niektorí študenti pri programovaní nedokážu dostatočne skĺbiť aj zápis cyklu a aj jeho pochopenie s tým, že výstupy cyklu vidia v grafickej ploche. Naopak, iní študenti môžu „zobrazovanie výsledkov cyklu“ v grafickej ploche vnímať ako výraznú pomoc k správnemu pochopeniu fungovania cyklu a výpočtov, ktoré sa v cykle vykonávajú.

Myslíme si preto, že námety zadaní úloh musí preto učiteľ veľmi starostlivo prispôsobovať cieľom kurzu. V našom prípade sme sa rozhodli, že urobíme akýsi kompromis. Na začiatku kurzu budeme viac využívať JS na úpravu dokumentu s využitím základných prvkov HTML jazyka ako sú textové okienko (input), štandardné tlačidlá (button) a následný výpis do HTML dokumentu, pozri 4.1. Potom sa budeme venovať grafickej ploche a základné konštrukcie jazyka JS budeme používať aj na vykresľovanie geometrických útvarov, pričom údajovú štruktúru pole budeme využívať v oboch prístupoch a následne aj v závere kurzu pri práci s knižnicou `sprites.js`.

4.1 Hra Obesenec – práca s poľom

Táto hra, pozri obrázok 2, je navrhnutá tak, aby študenti využili len základné elementy HTML jazyka (input, button) s ktorými sa stretli už v predchádzajúcich lekciách kurzu. V programe, ktorý majú vytvoriť využijú údajovú štruktúru pole a základné algoritmy s ním – vkladanie prvku do poľa, prechádzanie poľa a pomocou konštrukcie `if` zisťovanie toho, či sa nejaký prvok v poli nachádza.



Napíšte program v JS, ktorý umožní jednému hráčovi hrať hru obesenec: počítač si zvolí nejaké slovo a v dolnej časti zobrazí toľko pomlčiek, koľko písmen má toto slovo. Hráč do vstupného textového okienka zadá nejaké písmeno a stlačí tlačidlo. Vtedy počítač vyhodnotí, či sa zadané písmeno nachádza v hádanom slove. Ak áno, doplní ho na všetky miesta, na ktorých sa nachádza. Inak sa nestane nič. Ak hráč uhádne všetky písmená, zobrazí sa správa SUPER, pozri obrázok.

Obrázok 2: Zadanie, v ktorom využijeme polia a základné HTML elementy

Väčšina študentov túto hru poznala, resp. poznali jej zložitejší variant, v ktorej hráč vidí aj písmená, ktoré ešte nehádal. V zložitejšom variante hry by mal počítač k dispozícii slovník slov, z ktorých náhodne vyberá. Pre potreby kurzu sme však zadanie úlohy zjednodušili. Písmená, ktoré sme ešte nehádali sa nezobrazujú. Variant s viacerými slovami sme vôbec neuvažovali a slovo, ktoré má hráč uhádnuť je zvolené ako konštanta programu. K tejto hre sme v našom kurze pripravili pre študentov aj pomôcku v podobe slovného popisu riešenia. V ňom sme navrhovali, v akých údajových štruktúrach si majú uchovávať vypisované slovo a tiež postup (algoritmus), akým zistiť, či sa zvolené písmeno nachádza v slove.

4.2 Kreslenie útvarov do grafickej plochy

Zrejme najjednoduchší útvar, ktorý sa dá nakresliť do grafickej plochy je obdĺžnik, resp. štvorec. Podobne, ako iné programovacie jazyky, má aj JS príkaz, ktorým sa dá tento útvar nakresliť. V JS môžeme kresliť obdĺžnik pomocou funkcií `rect` alebo `fillRect`. Oba majú štyri parametre, tretí a štvrtý parameter však majú napr. iný význam ako v Pythone. V úlohách študenti vykresľovali štvorce do riadku, vytvárali z nich štvorčekovú sieť alebo ich kreslili do grafickej plochy pri stlačení ľavého tlačidla myši. Riešenie poslednej zo spomínaných úloh môže byť napr. také, ako na Obrázku 3. Samozrejme, ak chceme aby program fungoval musíme ho „obaliť“ HTML dokumentom, ktorý zabezpečí správne fungovanie javascriptového programu. Je však pomerne zrejmé, že práca s grafickou plochou v JS sa pomerne podobá práci s grafikou v Python. Aj pre ďalšie funkcie, ktoré umožňujú vykresľovanie útvarov do nej, študenti bez problémov chápali paralelu s inými programovacími jazykmi, predovšetkým Pythonom, pozri zadania úloh na prácu s grafikou v časti 5.1.

```
canvas = document.getElementById("canvas");
context = canvas.getContext("2d");

function mouseDown(event) {
    context.fillStyle = "#FF0000";
    context.fillRect(
      event.clientX, event.clientY, 10, 10);
}
    canvas.addEventListener
      ("mousedown", mouseDown);
```

Obrázok 3: Schéma na spracovanie kliknutia myšou v JS

4.3 Obrázky v JS

Jednou z kľúčových častí nášho kurzu Programovanie v JavasScripte je práca s obrázkami v komponente canvas. Pomerne dlho sme hľadali vhodnú cestu, ako načítať a zobraziť obrázok pomocou JS. Riešenia, ktoré sme našli na internete, často nefungovali alebo boli také zložité, že sme ich pre našu cieľovú skupinu, teda pre budúcich učiteľov informatiky, nechceli v kurze prezentovať. Nakoniec sme objavili jedno riešenie, ktoré sme sa rozhodli predložiť študentom ako šablónu, podľa ktorej budú vo svojich programoch načítavať a zobrazovať obrázky, pozri Obrázok 4. V predloženom riešení je použitá trieda `Image`. S triedami sa študenti stretli aj v iných programovacích jazykoch. Novinkou však pre nich bola funkcia, ktorá sa vykoná pri udalosti `onload`. Pochopenie celej funkčnosti tejto schémy nie je pre študentov úplne nevyhnutné. Navrhovali sme im, aby schému brali ako "technológiu", pomocou ktorej sa vykoná to, čo potrebujú, teda pomocou ktorej sa správne načíta a vykreslí obrázok do grafickej

plochy HTML dokumentu. Na overenie funkčnosti tejto schémy študenti riešili rôzne úlohy, ktoré sme pre nich pripravili.

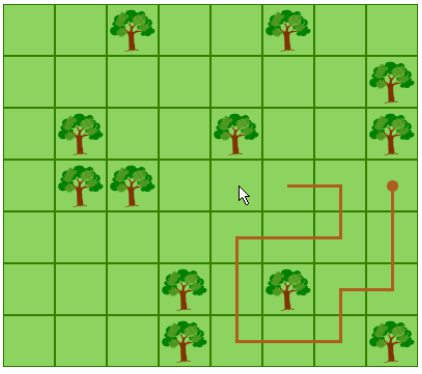
```
function loaded() {
    context.drawImage(image, 0, 0);
}
image = new Image();
image.onload = loaded;
image.src = "obrazok.png";
```

Obrázok 4: Časť riešenia, ktoré umožní načítať obrázok a zobraziť ho v grafickej ploche

Úlohy sme však vyberali veľmi starostlivo, pretože je zrejmé, že schéma je skôr určená na načítanie, zobrazenie a spracovanie kliknutia myšou pre jeden-dva obrázky. Ak by sme však chceli tento postup automatizovať pre viac obrázkov, potrebujeme ďalšiu šablónu, ktorá pomerne netriviálne vysvetľuje mechanizmus paralelného načítavania v jazyku JS pre pole obrázkov. V prvom behu kurzu sme schému spracovania viacerých obrázkov študentom poskytli, ale v druhom behu sme obsah kurzu upravili tak, že načítanie a vykresľovanie obrázkov v „holom" JS sme vynechali a sústredili sme sa na techniky práce s geometrickými útvarmi grafickej plochy a programovanie reakcií pri kliknutí na štvorce, ktoré boli umiestnené náhodne alebo vo vhodne zvolenej mriežke. Musíme priznať, že na jednej strane sme tým pomerne obmedzili škálu úloh, ktoré dokážu študenti v grafike riešiť, na druhej strane sme však vedeli o možnostiach, ktoré im na komfortnú prácu s obrázkami poskytne knižnica **`sprites.js`**, pozri časť 4.4, a aplikovaním tejto zmeny v obsahu kurzu nám zostalo viac času práve na riešenie úloh pomocou nej.

4.4 Knižnica sprites.js

Javascriptová knižnica s názvom **`sprites.js`** vznikla v rámci projektu KEGA. O jej vlastnostiach a práci s ňou sme písali v [4]. Knižnica sa ďalej vyvíja a v súčasnosti sa najviac používa pre programovanie úloh do súťaže iBobor, pozri [5]. Pre študentov v našom kurze sme využili dokument s návodom k práci s touto knižnicou, ktorý popisuje vytváranie malých interaktívnych aplikácií s využitím obrázkov. Výhodou knižnice z pohľadu kurzu pre učiteľských študentov je predovšetkým to, že knižnica vhodne zakrýva technické problémy s prácou s obrázkami v JS a umožňuje plne využívať nielen viacero objektov s obrázkom v jednej aplikácii, ale dokonca aj viacero obrázkov pre rovnaký objekt, ktoré sa „vymieňajú" pri kliknutí na objekt, pozri Obrázok 5.

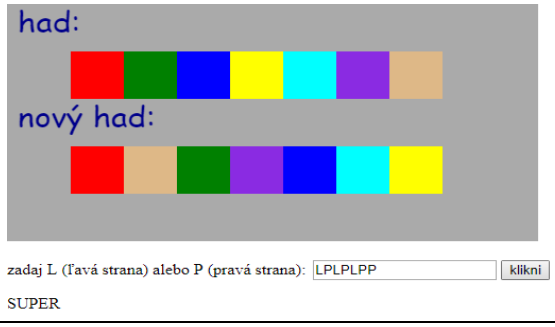


Obrázok 5: Klikaním na prázdne štvorčeky hráč vytvára spojitú cestu cez park. Kliknutie spracováva knižnica. Ostatné, pomerne netriviálne časti programu, však musí programátor zvládnuť v samotnom JS

4.5 Záverečné zadanie

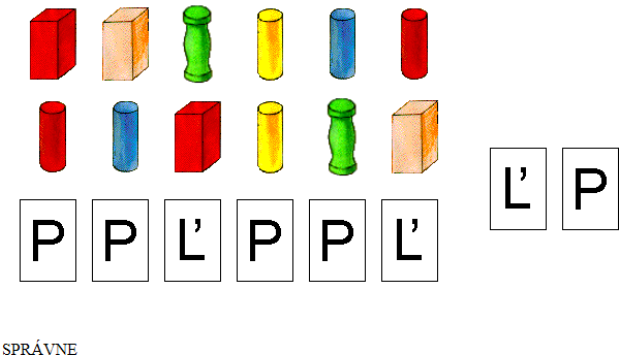
Zadanie záverečnej úlohy v prvom behu kurzu sme formulovali tak, aby ho študenti aj pre JS aj pre využitie knižnice **`sprites.js`** riešili pomocou načítavania a ťahania obrázkov. Z pozorovaní študentov pri riešení týchto úloh, a aj z analýzy ich riešení, sme však vyvodili, že predovšetkým spracovanie obrázkov v „holom" JS je technicky a aj časovo náročné.

Napíšte program, ktorý umožní žiakom riešiť takúto úlohu: V **hornej časti** grafickej plochy sa v jednom rade zobrazia farebné štvorce (každý inej farby), ktoré predstavujú hada. V **dolnej časti** je nový had, ktorý vznikne z horného hada tak, že niekedy zoberieme kocku z ľavého konca (Ľ) a niekedy z pravého konca (P) horného hada a vždy ju položíme na pravý koniec dolného hada. Úlohou hráča je určiť postupnosť písmen L a P podľa toho, z ktorej strany horného hada sme kocku zobrali.
Počítač najprv **vygeneruje** a **vykreslí** farebné kocky horného hada. Potom vygeneruje náhodnú postupnosť písmen L a P a podľa nej **vytvorí** a **vykreslí** dolného hada. Potom hráč vloží postupnosť písmen a stlačí "klikni". Program vypíše, či je hráčova postupnosť správna. Pozn. Uvedomte si, že hráčova postupnosť sa od vygenerovanej môže líšiť v poslednom písmene.



Obrázok 6: Záverečné zadanie, v ktorom sa využívajú elementy HTML a kreslenie útvarov do grafickej plochy

Do druhého behu sme obsah kurzu upravili a teda sme aj pre záverečné zadanie hľadali taký námet, ktorý sa bude dať riešiť pomocou HTML komponentov (textové okienko, tlačidlo) a vykresľovaním štvorcov do grafickej plochy, pozri Obrázok 6. Ale zároveň, aby malou obmenou tejto úlohy vznikla úloha, ktorá bude vhodná pre vytvorenie riešenia s ťahaním obrázkov a využitím knižnice **`sprites.js`**.



Obrázok 7: Záverečné zadanie, v ktorom sa využíva knižnica sprites.js a ťahanie kartičiek

Druhé zadanie, ktoré je súčasťou toho istého termínu skúšky, má rovnakú motiváciu, len program vykresľuje obrázky kociek a postupnosť L a P hráč vytvára interaktívne ich ťahaním do pripravených miest, pozri Obrázok 7.

5. VÝSKUM

V tejto časti nášho článku popíšeme a vyhodnotíme dva behy kurzu Programovanie v JavaScripte a uvedieme zmeny, ktoré plánujeme urobiť v školskom roku 2019/20.

5.1 Realizácia kurzu

Kurz Programovanie v JS prebehol dvakrát – v letnom semestri v školských rokoch 2016/17 a 2017/18. Žiaľ, kvôli malému počtu študentov v učiteľskom štúdiu v kombinácii s informatikou, kurz absolvovali iba štyria študenti. V prvom roku kurzu sme mali troch študentov, ktorí mali v prvých troch semestroch svoj základný kurz programovania v jazyku Pascal v prostredí Lazarus. V minulom školskom roku sme mali dve študentky, z ktorých kurz dokončila len jedna. Tie mali základný kurz programovania v jazyku Python.

Tabuľka 1: Šablóna – vloženie komponentu canvas do HTML dokumentu a zadania na programovanie v JS

```
<!DOCTYPE HTML>
<HTML>
<head> <meta charset="UTF-8"> </head>
<body style="margin: 0;
padding: 0; background-color:#000000">
  <canvas id="canvas" width=500 height=300>
  </canvas>
  <script>
    canvas = document.getElementById("canvas");
    context = canvas.getContext("2d");
    context.fillStyle = "#FF0000";
    context.fillRect(0, 0,
      canvas.width, canvas.height);
  </script>
</body>
</HTML>
```

Úloha *zmen_farbu*
Dokument si pozorne preštudujte a pochopte všetky použité konštrukcie. V prípade nejasností sa spýtajte na jednotlivé HTML elementy alebo časti JS kódu. Zmeňte časti dokumentu alebo časť JS kódu tak, aby bola grafická plocha vykreslená bielou farbou a aby mal dokument nejaké bledé pozadie.

Úloha *kreslime_utvary*
Nájdite na internete príkazy pre komponent canvas a zistite, aké útvary sa dajú do neho kresliť. Nakreslite aspoň 3 útvary, aspoň jeden z nich vyplňte nejakou farbou. *(nepovinne) gradientom

Úloha *piseme*
Zistite, aký príkaz JS dokáže vypisovať do grafickej plochy text. Napíšte niekde do grafickej plochy „JAVASCRIPT JE SUPER“.

Úloha *stvorce10*
Pripravte script, v ktorom bude môcť používateľ zadať do textového okienka veľkosť strany štvorca a po stlačení tlačidla program vedľa seba vykreslí v jednom rade 10 štvorcov s touto veľkosťou strany.

Na každé stretnutie sme pre študentov pripravovali pracovný list, ktorý obsahoval zadania úloh. V jeho úvode sme väčšinou uvádzali aj to, aké vedomosti sú potrebné k ich vyriešeniu. Niektoré vedomosti, ktoré boli v jednom pracovnom liste sa týkali len priamo JS, iné smerovali k tomu, aby sa študenti oboznámili v rámci riešenia úloh aj s vytváraním HTML dokumentov. Stretnutia kurzu prebiehali formou samostatnej práce študentov pri počítači tým, že riešili zadania. Deväťdesiat minút, ktoré trvalo jedno stretnutie, sa často delilo na štúdium vedomostí, ktoré boli potrebné k riešeniu úlohy a na samotné programovanie zadaní. V prípade štúdia vedomostí o HTML stránkach, ale aj o syntaxi základných riadiacich štruktúr jazyka JS, študenti väčšinou vyhľadávali informácie na webe v osvedčených kurzoch pre HTML a JS, najčastejšie na [6], resp. [7]. Ak sme chceli naštudovanie niektorej témy urýchliť, poskytli sme študentom v pracovnom liste návod alebo šablónu, ktorá ich viedla pomerne priamočiaro k tomu, aby boli schopní vyriešiť úlohy, ktoré v pracovnom liste nasledovali. Takto napr. študenti dostali šablónu na vloženie grafickej plochy do HTML dokumentu a sériu úloh, v ktorej sa canvas používa, pozri Tabuľku 1, v ktorej uvádzame šablónu a časť zadaní. Pre využitie šablóny im v rámci hodiny „stačilo“ skopírovať šablónu a vytvoriť z nej samostatný HTML dokument. Vzápätí tento dokument vyskúšať a podľa zadaní, uvedených v pracovnom liste, ho začať modifikovať. Myslíme si, že pochopenie a následná modifikácia hotových zadaní môže byť chápaná ako jeden zo základných postupov, ktorým sa študenti môžu učiť programovať, resp. využívať hotové (čiastočné) riešenia úloh.

V pracovnom liste bolo na každé stretnutie pripravených viac úloh ako sa stihlo počas deväťdesiatich minút prezenčného stretnutia. Študenti zadania dokončovali v rámci svojej domácej prípravy.

5.2 Vyhodnotenie priebehu kurzu

V nasledujúcej časti článku spomenieme spoločné črty oboch behov kurzu Programovanie v JavaScripte. Výskum sme robili formou priameho pozorovania študentov, pozri [8], pri programovaní nami navrhnutých zadaní, pozri časť 4 a skúmaním riešení, ktoré študenti posielali v rámci prezenčnej práce a domácej prípravy. Tento školský rok pripravujeme tretí beh kurzu a uvažujeme, aké zmeny je potrebné v kurze urobiť, aby bol pre študentov prístupnejší, ale aby sme aj napriek tomu zachovali jeho pôvodný cieľ, t. j. aby študenti po jeho skončení dokázali vytvárať grafické programy, resp. malé interaktívne hry, ktoré budú fungovať priamo vo webovom prehliadači. Jedným z možných riešení by bolo, že kurz sa bude viac venovať iba využívaniu knižnice **`sprites.js`** a pritom sa budú študenti učiť základy jazyka JS. Myslíme si však, že takáto radikálna zmena sa nedá urobiť, pretože je lepšie, aby sa v úvode študenti najprv, bez využívania grafiky a obrázkov naučili zvládať v JS základné programátorské koncepty a pomerne dobre zvládli prácu s poľom, pretože obe tieto kompetencie budú potrebovať aj pri práci s knižnicou **`sprites.js`**.

Pri pozorovaní priebehu oboch behov kurzov sme našli niekoľko problematických častí. V ďalšej časti ich popisujeme a uvádzame návrh zmien, ktoré plánujeme v ďalšom behu kurzu urobiť.

Z našich pozorovaní a vyhodnotení študentských riešení úloh vyplynulo:

- Študenti nemali skúsenosti s vytváraním webových stránok priamym písaním HTML príkazov. Keďže kurz, ktorý je na to zameraný nie je povinnou súčasťou ich štúdia, tento stav sme

očakávali. Aj preto sme do našich úloh vkladali odkazy na iný kurz, v ktorom sa študenti mali naučiť vytvárať vlastné webové stránky. Keďže tento kurz patrí k výberovým predmetom, predpokladali sme, že si ho budú vyberať študenti so záujmom o programovanie. Aj preto sme dúfali, že pre nich „doučenie sa“ HTML jazyka nebude problém. Žiaľ, kvôli malému počtu študentov si v súčasnosti vyberajú kurz takmer všetci študenti učiteľského štúdia a nielen tí, ktorí majú o programovanie zvýšený záujem. Z tejto skutočnosti vyplynulo, že študenti mali so samoštúdiom jazyka HTML problémy. *Návrh zmien*: Kurz upravíme tak, aby viac upozorňoval na dôležitosť HTML dokumentu a pridáme do neho úlohy, ktoré sa budú venovať tejto téme.

- Využívanie on-line prostredia `JS.do`, v ktorom sa výsledok objavil ihneď vo vedľajšom v okne, sme v ďalších stretnutiach kurzu vnímali ako pascu. Študenti mali neskôr, pri práci s HTML dokumentom, tendenciu písať iba riešenie úlohy v JS a nevenovať sa „obalu“, ktorý tvorili HTML tagy a komponenty. Dokonca v prípade, že v zadaní bolo potrebné pomocou JS modifikovať samotný HTML dokument, trvalo študentom oveľa dlhšie, kým pochopili, čo je ich úlohou. Pri riešení úloh zistili, že niektorým zápisom v HTML dokumente predsa len nerozumejú dostatočne, resp. niektoré úlohy, ktoré vyžadovali jeho modifikáciu, nedokážu vyriešiť. *Návrh zmien*: Je možné, že používanie on-line prostredia pre riešenie na prvých stretnutiach vynecháme a nahradíme ich tým, že riešenia budeme vytvárať od začiatku v HTML dokumente.
- Študentom robilo problémy pracovať s poľom. Toto bolo pre nás trochu prekvapujúce, keďže študenti absolvovali päť semestrov kurzov programovania a to dokonca v dvoch jazykoch – Pythone a Imagine Logo. Spracovanie a vykreslenie jednorozmerných polí robilo študentom menšie problémy, ale v prípade, ak mali prechádzať, resp. vykresliť dvojrozmerné pole, robilo im riešenie takýchto úloh pomerne veľké problémy. To sa nakoniec najviac prejavilo aj pri riešení záverečných zadaní. *Návrh zmien*: Do úvodnej časti kurzu pridáme ďalšie úlohy, ktoré sa budú cielene venovať práci s poľom.

6. ZÁVER

Nami navrhnutý kurz Programovanie v JavaScripte je koncipovaný tak, aby študenti dokázali programovať malé interaktívne grafické aplikácie. Naplnenie cieľov sme testovali po ukončení semestra tým, že študenti mali počas 120 minút naprogramovať dve záverečné zadania, ktoré mali rovnakú motiváciu, ale líšili sa spôsobom interakcie hráča pri práci s malou edukačnou aplikáciou, ktorú študenti naprogramovali a ktorá bola motivovaná úlohou súťaže iBobor, pozri Obrázky 4 a 5. Náš výskum, závery z neho a návrhy na zmeny nám pomôžu pri príprave budúcoročného behu kurzu Programovanie v JavaScripte pre učiteľské bakalárske štúdium v kombinácii s informatikou. Veríme, že absolvovanie tohto kurzu motivuje budúcich učiteľov informatiky navrhovať a programovať interaktívne zadania, napr. z úloh súťaže iBobor, pre svojich žiakov na základnej a strednej škole. Veríme, že tým budú žiakov nielen motivovať k riešeniu algoritmických informatických problémov, ale v budúcnosti aj k tomu, aby aj oni sami niečo programovali.

BIBLIOGRAFICKÉ ODKAZY

[1] Bakalársky študijný program učiteľstvo informatiky v kombinácii na FMFI UK v Bratislave. Dostupné na: https://fmph.uniba.sk/studium/programy/ucitelstvo-informatiky-v-kombinacii/

[2] Knižnica sprites.js. Dostupná na: http://salanci.sk/

[3] Online nástroj na spúšťanie javascriptových programov. Dostupné na: https://js.do/

[4] Tomcsányiová, M., Hrušecký, R.: Vývoj edukačných webových aplikácií pomocou knižnice SPRITES. DidInfo and DidactIG 2017 [elektronický zdroj]. - Banská Bystrica : Univerzita Mateja Bela, 2017. - ISBN 978-80-557-1216-1. - S. 64-67

[5] Hlavná stránka súťaže iBobor. Dostupné na: iBobor.sk

[6] Webové stránky o JavaScripte. Dostupné na: https://www.w3schools.com/js/

[7] Webové stránky o HTML. Dostupné na: https://www.w3schools.com/HTML/default.asp

[8] Gavora, Peter a kol. 2010. Elektronická učebnica pedagogického výskumu. [online]. Bratislava : Univerzita Komenského, 2010. Dostupné na: http://www.e-metodologia.fedu.uniba.sk/ ISBN 978–80–223–2951–4.

Budúci „Chief Cybercrime Officer“ ešte možno derie školské lavice

The future „Chief Cybercrime Officer“ may still sits at school

Ľudovít Trajteľ, Dana Horváthová, Miroslav Melicherčík, Alžbeta Michalíková, Vladimír Siládi, Jarmila Škrinárová, Michal Vagač, Patrik Voštinár
Katedra informatiky Fakulty prírodných vied Univerzity Mateja Bela
Tajovského 40
97401 Banská Bystrica
Slovensko
ludovit.trajtel@umb.sk, dana. horvathova@umb.sk, miroslav.melichercik@umb.sk, alzbeta.michalikova@umb.sk, vladimir.siladi@umb.sk, jarmila.skrinarova@umb.sk, michal.vagac@umb.sk, patrik.vostinar@umb.sk

ABSTRACT
The article presents BOV (Inquiry Based Learning) and its specific use in education of cyber security. In the next part article is devoted to the selected informatics subjects, which was addressed by the Department of Informatics of the Faculty of Natural Sciences of Matej Bel University in Banská Bystrica in the framework of the IT Academy – Education for the 21st Century.

Keywords
Inquiry Based Learning. Project IT Academy. Education for the 21st Century.

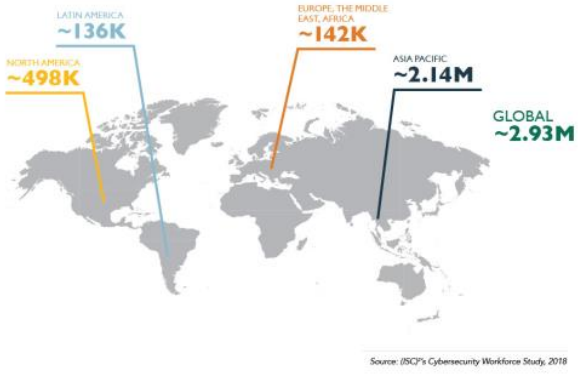
ABSTRAKT
Článok prezentuje bádateľsky orientovanú výučbu (BOV, Inquiry Based Learning) a jej konkrétne využitie pri edukácii kybernetickej bezpečnosti. V ďalšej časti sa venuje vybraným informatickým celkom, ktoré v rámci projektu IT Akadémia – vzdelávanie pre 21. storočie, riešila Katedra informatiky Fakulty prírodných vied Univerzity Mateja Bela v Banskej Bystrici.

Kľúčové slová
Bádateľsky orientovaná výučba. Projekt IT Akadémia. Vzdelávanie pre 21. storočie.

ÚVOD
Nedostatok špecialistov na počítačovú bezpečnosť je dnes jedným z hlavných dôvodov zraniteľnosti väčšiny organizácií, Obr.1.[1]

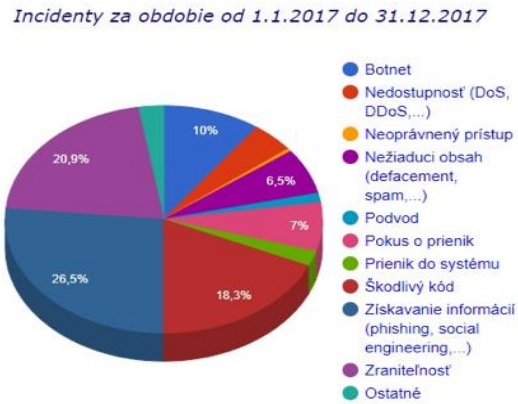
„Na Slovensku budú [...] v najbližších rokoch chýbať tisícky odborníkov na IT bezpečnosť a postupne začnú aj u nás vznikať roly riaditeľov pre kybernetickú kriminalitu (Chief Cybercrime Officer), ktorí budú zodpovedať za pripravenosť na kybernetické hrozby a aj za reakcie na útoky.“[2]

Množstvo, rovnako typ/rozmanitosť, kybernetických hrozieb veľmi rýchlo rastie. A hoci získať informácie o počte a type kybernetických útokov vykonaných na Slovensku je pomerne ťažké, môžeme sa, do istej miery, oprieť o údaje za rok 2017, ktoré formou grafu (o činnosti vládnej jednotky pre riešenie



Obrázok 1 Na celom svete dnes chýbajú 3 milióny špecialistov

počítačových incidentov CSIRT.sk) zverejnil Úrad podpredsedu vlády SR pre investície a informatizáciu, Obr.2.



Obrázok 2: Evidované počítačové incidenty na Slovensku v roku 2017[3]

[1] https://blog.isc2.org/isc2_blog/2018/10/cybersecurity-skills-shortage-soars-nearing-3-million.html
[2] https://blog.etrend.sk/roman-cupka/tieto-veci-nas-vo-svete-kybernetickej-bezpecnosti-neminu.html
[3] https://blog.etrend.sk/roman-cupka/akych-pocitacovych-utokov-sa-treba-naozaj-bat.html

Z grafu je jasné, že sa jednalo najmä o aktivity typu „phishing“ (doslova [rybo]lovenie citlivých údajov), sociálne inžinierstvo, zraniteľnosť, „defacement“ (forma webového vandalizmu), DDoS útok (zneprístupnenie cieľovej služby) a prienik do systémov, ..., tiež aktivity škodlivých kódov. Vidno, že približne polovicu incidentov tvorili pokusy o prieniky a získavanie informácií, útoky „botnetom“ (útočníkmi ovládaná sieť internetových robotov) a odstavovanie webových služieb. Naproti tomu aktivity škodlivých kódov mali na štruktúre bezpečnostných incidentov na Slovensku len menej ako pätinový podiel.

Od apríla 2018 platí na Slovensku Zákon o kybernetickej bezpečnosti, ktorý od verejných organizácii a firiem vyžaduje viaceré opatrenia.[4] Do dvoch rokov od začiatku účinnosti zákona musia mať dotknuté subjekty vytvorenú celú organizačnú štruktúru pre riadenie kybernetickej bezpečnosti (pozostávajúcu z viacerých rolí). Mali by už byť schopné bezpečne využívať adekvátne technológie (počítačové siete, počítačové systémy, robotiku, internet vecí, operačné systémy, pokročilé postupy programovania, ...).

Zoči-voči všetkým uvedeným faktom sa formuje rozhodnutie Katedry informatiky Fakulty prírodných vied Univerzity Mateja Bela v Banskej Bystrici (v texte ako KI FPV UMB) robiť výrazne viac v prospech gramotnosti v okruhoch kybernetickej bezpečnosti. A to nielen pri bezprostrednej príprave odborníkov pre segment aplikovanej informatiky, ale rovnako aj pri príprave učiteľov informatiky (aktívnych aj budúcich); ako v samotnom edukačnom procese, tak aj v rámci rôznych projektov.

1 K PROJEKTU IT-AKADÉMIA NA KI FPV UMB CEZ PROJEKT DVUI

Východiskom sú naše skúsenosti spred približne desiatich rokov, keď sa KI zapojila do projektu s názvom Ďalšie vzdelávanie učiteľov základných škôl a stredných škôl v predmete informatika (v texte ako DVUI). Spolu s projektovými partnermi sme vtedy vytvorili podmienky na to, aby v priebehu približne troch projektových rokov prešlo ročným školením celkom asi 1500 učiteľov (cca 1300 kvalifikovaných učiteľov informatiky a 200 „nekvalifikovaných“ učiteľov iných predmetov). Čo nás do istej miery prekvapilo, bolo to, že mnohí z vyškolených učiteľov po skončení kurzov DVUI zmenili prácu a zamierili do IT firiem. Bol to možno neželaný sprievodný jav samotného projektu DVUI (dôležitý však v dotyku s praxou), ale pre nás cenná informácia o tom, čo/koho prax potrebuje. Keď neskôr štát rozbehol projekt IT Akadémia – vzdelávanie pre 21. storočie (v texte ako IT-Akadémia), vedeli sme, aj vďaka DVUI, už oveľa viac aj o našich konkrétnych prioritách a edukačných cieľoch.

"Projekt [IT-Akadémia] sa bude venovať aj samotným učiteľom, pretože ak chceme mať na Slovensku dostatok dobre pripravených ľudí, ktorí budú schopní si nájsť dobre platené miesto na pracovnom trhu, bez [...] dobre pripravených učiteľov to nepôjde," povedal (v Bratislave, 13. novembra 2017) vtedajší podpredseda vlády SR pre investície a informatizáciu; podľa jeho ďalších slov musí Slovensko už dnes rozmýšľať, akým problémom bude čeliť za 15 rokov a na tieto trendy musí reagovať.[5] Od realizácie projektu sa očakáva ročný nárast na úrovni tisíc absolventov – špecialistov pre sektor IKT (informačných a komunikačných technológií). Projekt by mal byť vzorom spolupráce škôl a podnikov pôsobiacich v sektore IKT, keďže každá dotknutá aktivita projektu je podporovaná aj špecialistom z vybranej IT firmy. Celkový rozpočet národného projektu je 21 046 596,77 €, pričom nenávratný finančný príspevok (v texte ako NFP) zo štrukturálnych fondov Európskej únie je vo výške 20 672 504,80 €. Prijímateľom NFP je Centrum vedecko-technických informácií SR, ktoré projekt realizuje spoločne s ďalšími projektovými partnermi – Univerzitou Pavla Jozefa Šafárika v Košiciach, Technickou univerzitou v Košiciach, Univerzitou Konštantína Filozofa v Nitre, Univerzitou Mateja Bela v Banskej Bystrici a Žilinskou univerzitou v Žiline. Ďalšími partnermi projektu sú IT Asociácia Slovenska, Slovenská informatická spoločnosť, Košice IT Valley, Štátny inštitút odborného vzdelávania, Štátny pedagogický ústav a Metodicko-pedagogické centrum.[6] Prednaznačuje to úzku spoluprácu medzi školami, slovenskými univerzitami a prestížnymi IT spoločnosťami, prináša to najmä pilotné využívanie a testovanie IT ScienceLab pre základné školy a tvorbu edukačných materiálov pre budúce kurikulum informatiky a robotiky pre primárne a nižšie sekundárne vzdelávanie na Slovensku, tiež možné zaradenie školy do siete vybraných škôl Slovenska označených ako IT Akadémia partner. Na aktivitách národného projektu IT-Akadémia sa dnes už podieľa do 300 základných a 200 stredných škôl a päť univerzít, predpokladá sa zapojenie až celkom 36.000 žiakov a študentov a približne 2100 učiteľov.

1.1 Bádateľstvo ako základ projektu IT-Akadémia

Projekt IT-Akadémia stavia na bádateľsky orientovanej výučbe [1] pripravuje žiakov/študentov/respondentov pre aktuálne i perspektívne potreby vedomostnej spoločnosti, tiež na to, ako úspešne pokračovať v štúdiu alebo uplatniť sa na trhu práce, prináša inováciu vzdelávania na základných a stredných školách so zameraním na informatiku a IKT a má za cieľ posilniť záujem mladých ľudí o ďalšie štúdium v danom odbore. Venuje sa rozvoju a inovácii obsahu predmetov ako informatika, sieťové technológie, ... internet vecí. Neoddeliteľnou súčasťou aktivít projektu je formálne a neformálne vzdelávanie učiteľov, organizovanie krúžkov, IT táborov, seminárov, súťaží a ďalšie mimoškolské činnosti. Umožňuje tiež žiakom aj učiteľom získať certifikát ECDL. Generálny tajomník služobného úradu ministerstva školstva, vedy, výskumu a športu oceňuje prínos projektu slovami: „Ministerstvo školstva mimoriadne teší, že sa v ére digitálnych zručností v rámci projektu IT-Akadémia investuje do ľudských zdrojov. Investuje sa do žiakov, učiteľov, do kvalitného vzdelávania ľudí, aby mohli bádať a prichádzať s inováciami.“[7]

Tomu odpovedá aj zostava metodík[8] z KI FPV UMB predkladaná ako pomôcka, ktorej hlavným cieľom je zefektívniť prácu učiteľov pri zapracúvaní a aplikácii tém gramotnosti v okruhoch

[4] https://touchit.sk/stovky-firiem-sa-musia-do-konca-mesiaca-zaevidovat-na-nbu/194421

[5] http://skolskyservis.teraz.sk/skolstvo/projekt-it-akademia-ma-zlepsit-vzd/38281-clanok.html?combinedGlobalTab_webmagazin=0

[6] http://vedanadosah.cvtisr.sk/spoznajte-vitazne-projekty-ceny-itapa-2018

[7] https://itas.sk/do-digitalnej-koalicie-vstupilo-15-novych-clenov-it-asociacia-slovenska-ziskala-prestizne-ocenenie-v-bruseli-za-aktivnu-podporu-cinnosti-koalicie/

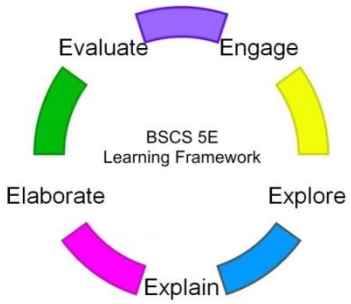
[8] Celkom šesť metodík smerujúcich do ISCED2A.

kybernetickej bezpečnosti do vyučovacieho procesu na druhom stupni ZŠ.

2 KI FPV UMB VYPRACOVALA ŠESŤ INOVATÍVNYCH METODÍK PRE KYBERNETICKÚ BEZPEČNOSŤ

Ponúkame (J. Jacková a Ľ. Trajteľ ako autori) inovatívne bádateľské metodiky postavené na využití moderných a bezpečných IKT. Východiskovým dokumentom je ISCED2. Je dôležité, aby si ho učiteľ prečítal a vytvoril si predstavu o tom, čo má žiak počas svojho vzdelávania na základnej škole dosiahnuť. Zdôrazňujeme skutočnosť, že aj gramotnosť v okruhoch kybernetickej bezpečnosti musí byť uceleným systémom, preto je nevyhnutné zaručiť nadväznosť medzi druhým a prvým stupňom základnej školy. Za veľmi vhodné považujeme uplatňovanie takého vyučovania, ktoré umožňuje predstaviť vzdelávací obsah v jeho komplexnosti, vedie k rozpoznaniu súvislostí, previazaniu poznatkov a umožňuje aplikáciu nadobudnutých zručností; aby žiaci bezprostredne videli/vnímali zmysel svojho vzdelávania. Len pripomíname, že takéto previazanie je možné realizovať ako na úrovni čiastkovej aktivity, jednoduchého projektu, tematického podcelku, tak aj na úrovni celého integrovaného učebného predmetu. Máme za to, že problematika výchovy a vzdelávania na podporu rozvoja gramotnosti v okruhoch kybernetickej bezpečnosti musí byť do ZŠ vzdelávania začlenená plošne. Znamená to, že každý účastník vzdelávania musí byť s ňou v určenom rozsahu oboznámený. Nepostačuje, ak je realizovaná len časťou triedy (alebo len krúžkami). V tomto zmysle treba gramotnosť v okruhoch kybernetickej bezpečnosti včleniť do výučby. Ak učiteľ pochopí, čo gramotnosťou v okruhoch kybernetickej bezpečnosti je (a čo nie je), lepšie vyberie spôsob výučby, tiež to, do ktorých predmetov/tém ju začlení a aké metódy vo výučbe bude uplatňovať.

V predkladaných metodikách zdôrazňujeme najmä zameranie sa na rozvíjanie bádateľských zručností žiakov a širokú podporu interaktívnej výučby metódou 5E.



Obrázok 3: Metóda 5E [2]

Zámerom predkladanej zostavy metodík však nie je direktívne nariadiť prístup a vynútiť si jeho uplatňovanie. Je na danej škole/na učiteľovi, pre ktorý prístup sa rozhodne, ako tvorivo si bude modifikovať učebný obsah (napr. koľko percent bude tvoriť tímová práca žiakov, koľko individuálna a pod.). Výučba však musí byť správna po odbornej stránke, je preto dôležité disponovať odbornými vedomosťami z predmetnej oblasti, schopnosťou adekvátne používať základné pojmy a vysvetľovať ich s prihliadnutím na jednotlivé vekové kategórie. Osvojenie pojmov však nemôže byť cieľom, ten spočíva v žiakom osvojených vedomostiach, rozvinutých zručnostiach a spôsobilostiach nevyhnutných pre aktívnu účasť na informatických aktivitách, pre využívanie produktov/výrobkov a služieb a zaručenie ich bezpečnosti.

Na vyššie spomenutých metodikách založené podporné kurzy sú určené pre tematický celok Komunikácia a spolupráca – práca s nástrojmi na komunikáciu a pre tematický celok Informačná spoločnosť – bezpečnosť a riziká[9]:

- 2stZŠ – Etika a bezpečnosť (3 podporné kurzy):
  - Etické aspekty IKT – Ako škodím ostatným?
  - Etické aspekty IKT – Ako niektorí škodia mne?
  - Autorské práva – Múdrejší/múdrejšia som silnejší/silnejšia.
- 2stZŠ – Riziká IKT (3 podporné kurzy):
  - Riziká IKT – Pre koho som zaujímavý/zaujímavá?
  - Riziká IKT – Čo potrebujem chrániť?
  - Riziká IKT – Čo nesmiem zanedbať?

2.1 Kurz „Etické aspekty IKT – Ako škodím ostatným?“

Pre ISCED 2, 7. – 8. ročník, 1. vyučovacia hodina, obsah:

- nekvalitne spracované webové stránky,
- spôsoby správania sa, ktoré môžu vyvolať u ostatných nepríjemné pocity,
- webová distribúcia násilníckeho/nelegálneho/nevhodného obsahu (reťazové maily/“hoax“),
- šikanovanie/urážanie/… ,
- šírenie, možno aj nevedomé, malware (nepoužívanie antivírusového SW).

Celkom 6 pripravených súčastí:

- Samotná metodika (titulný list, úvodný text k metodiky, podrobný opis priebehu vyučovacej hodiny, skúsenosti a zistenie z výučby, alternatívy výučby, prílohy).
- Pracovné listy (U, 1, 2, 3, 4).

2.2 Kurz „Etické aspekty IKT – Ako niektorí škodia mne?“

Pre ISCED 2, 7. – 8. ročník, 2. vyučovacia hodina, obsah:

- bezpečná práca s webovou stránkou a jej obsahom,
- bezpečné vyhľadávanie na webe,
- hrozby a riziká.

Celkom 2 pripravené súčasti:

- Samotná metodika (titulný list, úvodný text k metodiky, podrobný opis priebehu vyučovacej hodiny, skúsenosti a zistenie z výučby, alternatívy výučby, prílohy).
- Pracovný list 1.

[9]https://vzdelavanie.itakademia.sk/course/index.php?categoryid=33

2.3 Kurz „Autorské práva – Múdrejší/múdrejšia som silnejší/silnejšia“

Pre ISCED 2, 7. – 8. ročník, 3. vyučovacia hodina, obsah:

- citácia/citovanie,
- autorské práva; licencie, bezplatné a voľne šíriteľné programy, ...,
- softvérové pirátstvo (ale aj „crackovanie“, „cracker“, „hacker“, …).

Celkom 6 pripravených súčastí:

- Samotná metodika (titulný list, úvodný text k metodiky, podrobný opis priebehu vyučovacej hodiny, skúsenosti a zistenie z výučby, alternatívy výučby, prílohy).
- Pracovné listy (U, 1, 2, 3, 4).

2.4 Kurz „Riziká IKT – Pre koho som zaujímavý/zaujímavá?“

Pre ISCED 2, 7. – 8. ročník, 1. vyučovacia hodina, obsah:

- pravidlá a odporúčania pre bezpečné profily, pre príjem správ/dokumentov a pre ochranu súkromia; rešpektovanie súkromia ostatných,
- hrozby webových kontaktov s neznámymi ľuďmi, nekritické preberanie webového obsahu.

Celkom 5 pripravených súčastí:

- Samotná metodika (titulný list, úvodný text k metodiky, podrobný opis priebehu vyučovacej hodiny, skúsenosti a zistenie z výučby, alternatívy výučby, prílohy).
- Pracovné listy (1, 2, 3, 4).

2.5 Kurz „Riziká IKT – Čo potrebujem chrániť?“

Pre ISCED 2, 7. – 8. ročník, 2. vyučovacia hodina, obsah:

- práca s webovou stránkou,
- vyhľadávanie na webe,
- bezpečnosť a riziká.

Celkom 2 pripravené súčasti:

- Samotná metodika (titulný list, úvodný text k metodiky, podrobný opis priebehu vyučovacej hodiny, skúsenosti a zistenie z výučby, alternatívy výučby, prílohy).
- Pracovný list 1.

2.6 Kurz „Riziká IKT – Čo nesmiem zanedbať?“

Pre ISCED 2, 7. – 8. ročník, 3. vyučovacia hodina (pozn.: realizácia v špecifických prípadoch až po vyjadrení sa/súhlase rodičov), obsah:

- lepšie spoznanie elektronickej pošty, pochopenie jej mechanizmov,
- bezpečná elektronická komunikácia, riziká pri práci s elektronickou poštou/internetom,
- bezpečné správanie sa na internete.

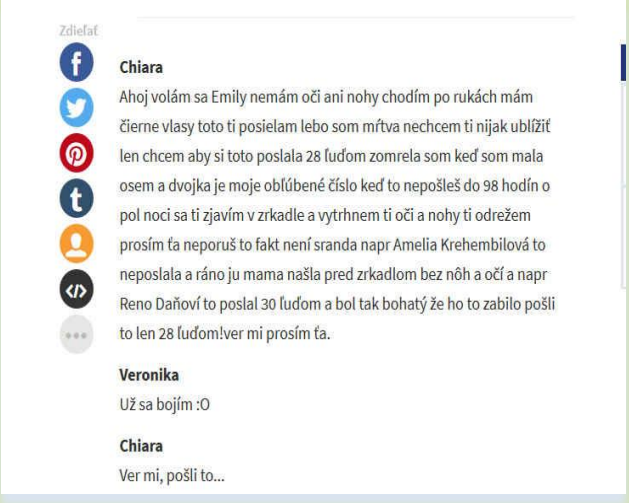
Celkom 5 pripravených súčastí:

- Samotná metodika (titulný list, úvodný text k metodiky, podrobný opis priebehu vyučovacej hodiny, skúsenosti a zistenie z výučby, alternatívy výučby, prílohy).
- Pracovné listy (1, 2).
- Odpoveďový hárok pre žiaka.
- (Odpoveďový hárok pre kontrolu správnosti odpovedí).

3 UKÁŽKA VYBRANÝCH ČASTÍ METODIKY „AKO ŠKODÍM OSTATNÝM?“

3.1 Ukážka textu k metodike „Ako škodím ostatným?“ – časť Skúmanie

Skúmanie (cca 10 minút): V ďalšej časti **žiakov zapojíme do** procesu skúmania/**bádania.** Môžeme to uviesť napríklad takýmto zaznamenaným necitlivým a násilníckym internetovým dialógom (https://www.wattpad.com/543895464-scary-messages-sk-8-chain-management resp. aj https://najmama.aktuality.sk/clanok/256340/kontrolujete-detom-co-a-s-kym-si-pisu-ja-som-dcere-v-mobile-nasla-hroznu-spravu/; vo vybranej časti je na obrázku uvedený v pôvodnom internetovom znení):



Evidentne **sa tu jedná o nevhodný/násilnícky obsah** a zrejme aj o porušovanie pravidiel na internete.

Poznámka

Pre učiteľa

- Je na nás, či v rámci výučby na tomto mieste ako úvodný príklad použijeme práve tento pomerne strašný/násilnícky dialóg/hoax (žiakov však určite upozorníme na to, že nie je určený pre citlivé povahy).
- Jeho použitie prípadne radšej vopred prekonzultujeme s rodičmi.

Žiakom zdôrazníme, že ak by sa tiež zapojili do tejto/takejto aktivity a obsah ďalej šírili, vedome by škodili ostatným. Lepšie je, keď v podobných prípadoch ostatných používateľov **upozornia na nelegálnosť/nevhodnosť** obsahu.

Väčšina najobľúbenejších stránok je v súčasnosti vybavená jednoduchými nástrojmi, ktoré používateľom umožňujú označiť nevhodný obsah, ktorý na stránku umiestnil niekto iný**. Je dôležité, aby žiaci vedeli, ako nelegálny/nevhodný obsah označiť**, pokiaľ by sa s ním náhodou stretli. Zo žiakov preto vytvoríme dvojice, ktoré následne **vyzveme k Aktivite 1.** Požiadame ich, aby **vyhľadali stránky ponúkajúce možnosť nahlasovať nelegálny/nevhodný webový obsah**.

Počas vyhľadávania a skúmania **je v aktivite dovolené**: diskutovať v rámci dvojice, spolupracovať v rámci dvojice a v dvojici spoločne zapisovať URL/linky vyhľadaných stránok a stručný komentár k ich ponuke; **nepovolené:** diskutovať medzi dvojicami, „odkukávať a/alebo odpisovať“.

Aktivita 1.

*Využite webový prehliadač a **vyhľadajte webové stránky, prostredníctvom ktorých možno nahlasovať nelegálny/nevhodný obsah**. Do **Pracovného listu 1** zapíšte URL/link takýchto stránok. **Ku každej z nájdených stránok pripíšte krátku charakteristiku** jej určenia.*

3.2 Ukážky textu k metodike „Ako škodím ostatným?“ – časť Rozpracovanie

Rozpracovanie (cca 10 minút): Žiakov vedieme spôsobom typickým pre reálne bádanie/reálny výskum, čím im (každému samostatne a/alebo v kooperácii so spolužiakmi) umožňujeme **aktívne získavať** primerané kompetencie/zručnosti/schopnosti.
Preto v tejto časti hodiny vytvárame podmienky na to, aby **žiak pracoval aj s ďalšími zdrojmi**, **tímovo komunikoval** (so spolužiakmi, tiež s učiteľom/rodičom/...) a **argumentoval**.
Možnosť **slobodne vyjadriť svoj názor** je pre žiaka kľúčová.

Formulovanie zmysluplných stanovísk a záverov od žiakov vyžaduje premýšľanie, porovnávanie/rozoberanie, interpretáciu získaných informácií a ich použitie v novom kontexte. Žiaci si majú **uvedomiť riziká a nástrahy** pri práci s internetom – virtuálnym svetom, mobilom, ... a následne majú **pochopiť aj sociálne, etické a právne aspekty online svetov**, v ktorých sú najmä deti/žiaci najviac ohrození.
Všetky **tieto činnosti pomáhajú žiakovi začleniť/doplniť nové poznatky do jeho vlastnej** (už existujúcej) **štruktúry poznatkov v okruhu „Ako škodím ostatným“**.

3.3 Ukážka textu k metodike „Ako škodím ostatným?“ – časť Hodnotenie/Sebahodnotenie

Hodnotenie (cca 5 minút).
V **Pracovnom liste 4** si žiaci vyplnia **Sebahodnotenie**, ktoré ponúkne pohľad na stav vedomostí a zručností, ktoré si mali na vyučovaní osvojiť.

Celý textový dokument metodiky „Ako škodím ostatným?“ a všetky textové dokumenty ostatných metodík nájde záujemca na Vzdelávacom portáli[10] projektu IT-Akadémia.

3.4 Ukážka pracovných listov k metodike „Ako škodím ostatným?“

Pracovné listy všetkých metodík nájde záujemca na Vzdelávacom portáli[11] projektu IT-Akadémia.

PRACOVNÝ LIST 1
„STRÁNKY PRE NAHLASOVANIE NEVHODNÉHO OBSAHU“

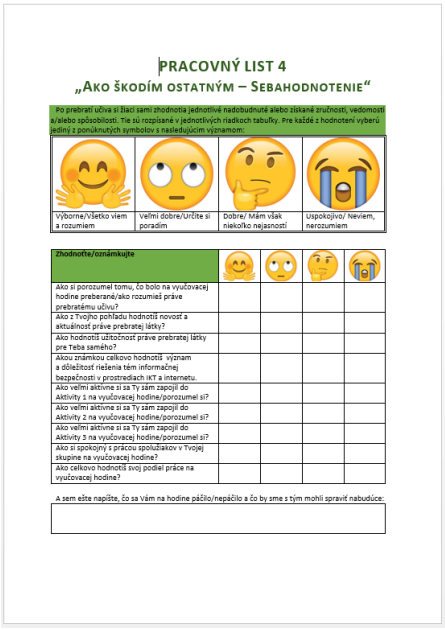
Cieľom aktivity je **pomôcť v aktivitách namierených proti nezákonnému webovému obsahu a takto prispieť k odhaľovaniu najmä prípadov zneužívania detí, prejavov rasizmu a xenofóbie na internete.** Žiaci, ktorí vedia, na koho (ktorú webovú stránku) sa v takých prípadoch môžu obrátiť (keďže používajú internet možno niekoľkonásobne viac ako niektorí dospelí), môžu nahlásením nevhodného obsahu prispieť k celkovej bezpečnosti internetu/sociálnych sietí. **Neškodia, naopak – pomáhajú ostatným.**

Do pracovného listu **zapíšte zistené URL/linky stránok pre nahlasovanie nevhodného obsahu** k stránke **pripíšte krátku charakteristiku.**

URL/link stránky pre nahlasovanie nevhodného/nezákonného/urážlivého/... obsahu	Stručná charakteristika stránky

Po vyplnení **spolu so žiakmi diskutujeme.**

Obrázok 4: Pracovný list 1



PRACOVNÝ LIST 4
„AKO ŠKODÍM OSTATNÝM – SEBAHODNOTENIE“

Po prebratí učiva si žiaci sami zhodnotia jednotlivé nadobudnuté alebo získané zručnosti, vedomosti a/alebo spôsobilosti. Tie sú rozpísané v jednotlivých riadkoch tabuľky. Pre každé z hodnotení vyberú jedinú z ponúknutých symbolov s nasledujúcim významom:

Výborne/Všetko viem a rozumiem	Veľmi dobre/Určite si poradím	Dobre/ Mám však niekoľko nejasností	Uspokojivo/ Neviem, nerozumiem

Zhodnoťte/označkujte				
Ako si porozumel tomu, čo bolo na vyučovacej hodine prebrané/ako rozumieš práve prebratému učivu?				
Ako z Tvojho pohľadu hodnotíš novosť a aktuálnosť práve prebratej látky?				
Ako hodnotíš užitočnosť práve prebratej látky pre Teba saméhо?				
Akou známkou celkovo hodnotíš význam a dôležitosť riešenia tém informačnej bezpečnosti v prostrediach IKT a internetu.				
Ako veľmi aktívne si sa Ty zám zapojil do Aktivity 1 na vyučovacej hodine/porozumel si?				
Ako veľmi aktívne si sa Ty zám zapojil do Aktivity 2 na vyučovacej hodine/porozumel si?				
Ako veľmi aktívne si sa Ty zám zapojil do Aktivity 3 na vyučovacej hodine/porozumel si?				
Ako si spokojný s prácou spolužiakov v Tvojej skupine na vyučovacej hodine?				
Ako celkovo hodnotíš svoj podiel práce na vyučovacej hodine?				

A sem ešte napíšte, čo sa Vám na hodine páčilo/nepáčilo a čo by sme s tým mohli spraviť nabudúce:

Obrázok 5: Pracovný list 4/Sebahodnotenie

[10]https://vzdelavanie.itakademia.sk/course/index.php?categoryid=33

[11]https://vzdelavanie.itakademia.sk/course/index.php?categoryid=33

4 ĎALŠIE AKTIVITY KI FPV UMB V PROJEKTE IT-AKADÉMIA

Okrem prípravy metodík vykonávajú členovia KI FPV UMB v projekte IT-Akadémia viaceré ďalšie významné činnosti v projektových aktivitách 1.1, 1.2, 1,3, 2.1 a 2.3.

4.1 Aktivita 1.1 – Tvorba predmetu Internet vecí

Katedra sa v rámci projektovej aktivity 1.1 zapojila (Ľ. Trajteľ ako spoluautor) do procesu tvorby predmetu Internet vecí. Okrem podporných materiálov je výstupom tejto aktivity zhodne nazvaná stredoškolská učebnica.

4.2 Aktivita 1.1 – Tvorba predmetu Počítačové systémy a siete

Do procesu tvorby predmetu Počítačové systémy a siete sa v aktivite 1.1 zapojili ďalší dvaja členovia KI (J. Škrinárová, V. Siládi ako spoluautori). V rámci tejto aktivity vzniká učebnica pre SŠ, tiež čiastkové materiály do vzdelávacieho obsahu pre podporu vzdelávania učiteľov zapojených do projektu IT-Akadémia. Úlohou autorov je nielen pripraviť SŠ učebnicu, ktorá obsahuje teoretický výklad, ale tiež navrhnúť a vytvoriť podklady pre viaceré vzdelávacie aktivity; na našej katedre konkrétne za časť venovanú operačným systémom: Práca v systéme a tvorba jednoduchých skriptov. Tvorba jednoduchých skriptov na identifikáciu, tvorbu, zrušenie a monitorovanie parametrov a vlastností procesov. Meranie doby prístupu k rôznym druhom pamäti pomocou primeraných nástrojov. Tvorba adresárových štruktúr, nastavovanie atribútov súborov a práca v prostredí so spoločným prístupom viacerých používateľov.

4.3 Aktivita 1.1 – Tvorba predmetu Tvorba a prezentácia dát

V rámci aktivity 1.1 sa KI podieľala (D. Horváthová ako spoluautorka) na spracovaní piatich kapitol (Multimédiá, Tvorba vektorovej grafiky a metagrafiky v editore Inkscape, Tvorba videa v editore VSDC, Multimédiá na webe a Programovanie multimédií v jazyku Python) SŠ učebnice Tvorba a prezentácia dát.

4.4 Aktivita 1.1 – Tvorba predmetu Programovanie webových stránok

Do procesu tvorby predmetu Programovanie webových stránok sa v aktivite 1.1 sa zapojili ďalší dvaja členovia KI (M. Melicherčík, P. Voštinár ako spoluautori). V rámci tejto aktivity vznikli/vznikajú čiastkové materiály do vzdelávacieho obsahu pre podporu vzdelávania učiteľov zapojených do projektu IT-Akadémia.

4.5 Aktivita 1.1 – Aktualizačné vzdelávanie ZŠ/SŠ

V tejto aktivite sa viacerí členovia KI (P. Voštinár, A. Michalíková, M. Vagač, D. Horváthová, Ľ. Trajteľ, V. Siládi, M. Melicherčík, J. Škrinárová) zapojili alebo v najbližšom čase zapoja do aktualizačného vzdelávania pre ZŠ/SŠ v celkom 9 predmetoch a inovatívneho vzdelávania v predmete Informatika v prírodných vedách.

Napríklad v predmetoch Databázy a Programovanie webových stránok učitelia získajú prehľad o aktuálnych postupoch a metódach používaných pri tvorbe dynamických webových stránok na strane klienta aj servera. Okrem tvorby dynamických webových stránok majú možnosť oboznámiť sa s princípmi práce s relačnými databázami a s prácou v jazyku SQL. V predmete Výučba pokročilých techník programovania sa zameriavame (P. Voštinár, M. Melicherčík ako lektori) na výučbu pokročilejších techník programovania a najnovších trendov v oblasti vývoja aplikácií. Cieľovou skupinou sú SŠ učitelia, pričom dôraz kladieme na praktickú využiteľnosť programátorských znalostí a zručností (najmä osvojenie si objektovo orientovaného programovania). Výučba prebieha za pomoci gradovaných a vizuálne orientovaných úloh v moderných programovacích jazykoch. Časť výučby je zameraná na programovanie pre mobilné platformy – platforma App Inventor. Prínosom výučby je možnosť oboznámiť sa s pokročilejšími technikami programovania, ktoré je následne možné využiť pri výučbe informatiky na stredných školách. Ďalej sme sa podieľali (D. Horváthová, M. Vagač) na lektorovaní aktualizačného vzdelávania pre ZŠ v predmete Programovanie robotických stavebníc v aktivite 1.2. Zatiaľ išlo o pilotné vzdelávanie prvej skupiny 15 učiteľov, ktorí sa zoznámili s programovaním robotov LEGO a prešli aj záverečnými skúškami.

V rámci prípravy aktualizačného vzdelávania boli na KI vyhotovené viaceré podporné materiály, ktoré majú učitelia k dispozícii. Prínosom takejto aktivity je zvýšenie informatickej odbornosti učiteľov informatiky a, samozrejme, inovácia predmetu informatika na stredných školách.

4.6 Aktivita 1.1 – Inovatívne vzdelávanie

Na tejto aktivite sa v predmete Informatika v prírodných vedách spolupodieľajú dvaja členovia KI (M. Melicherčík, A. Michalíková).

4.7 Aktivita 1.2 – Tvorba vzdelávacieho obsahu Programovanie mikroprocesorových systémov

Do procesu tvorby vzdelávacieho obsahu v aktivite 1,2 pre program kontinuálneho vzdelávania v predmete Programovanie mikroprocesorových systémov na SŠ sa na KI zapojil V. Siládi.

4.8 Aktivita 1.2 – Tvorba vzdelávacieho obsahu Programovanie robotických modelov

Do procesu tvorby vzdelávacieho obsahu v aktivite 1.2 pre program kontinuálneho vzdelávania v predmete Programovanie robotických modelov na ZŠ sa na KI zapojil M. Vagač.

4.9 Aktivita 1.2 – Tvorba vzdelávacieho obsahu Dáta a ich prezentácia

Do procesu tvorby vzdelávacieho obsahu v aktivite 1.2 pre program kontinuálneho vzdelávania v predmete Dáta a ich prezentácia sa na KI zapojil M. Melicherčík.

4.10 Aktivita 1.3 – Tábor, krúžky so zameraním na IT

Počas letných prázdnin prebiehal na FPV UMB letný IT tábor, zameraný na prírodné vedy, matematiku a informatiku, do ktorého sa prihlásilo v priebehu dvoch rokov približne 80 žiakov zo základných škôl Banskej Bystrice a blízkeho okolia. V rámci programu sa deti venovali programovaniu LEGO robotov

a vytváraniu vlastnej hry pod vedením členov KI (D. Horváthová, M. Vagač ako spoluorganizátori).

Obrázok 6: Žiaci – účastníci IT tábor pre budovou FPV UMB

Aktivitami Informatický krúžok UMB1 a UMB2 naša katedra (P. Voštinár ako autor myšlienky a organizátor) oslovila v školskom roku 2018/2019 žiakov 3. až 9. ročníka ZŠ. Na informatickom krúžku žiaci programujú roboty (v detských programovacích jazykoch), edukačné dosky a vytvárajú mobilné aplikácie v platforme App Inventor. Okrem nadobudnutia vedomostí z programovania nadobúdajú aj ďalšie zručnosti ako konštruovanie, testovanie robotov a učia sa pracovať v tíme. Katedra tak motivuje žiakov k budúcemu štúdiu informatiky.

4.11 Aktivita 2.1 – Inovácia predmetu pre informatikov

Traja členovia KI (M. Melicherčík, V. Siládi, M. Vagač,) sa zapojili do aktivít smerujúcich k inovácii predmetov v príprave VŠ študentov. V rámci aktivity 2.1 na KI vykonávame inovačný proces pre predmety: Gridové a cloudové počítanie, Business Intelligence, Programovanie v Jave.

4.12 Aktivita 2.3 – Odborná spolupráca s IT firmami

V tejto aktivite sa realizujeme prostredníctvom jedného člena KI (V. Siládi).

NAMIESTO ZÁVERU – POVEDALI O AJ NÁS PO KONFERENCII K PROJEKTU IT-AKADÉMIA

V závere roka (29. 11. až 1. 12. 2018) na konferencii v rámci projektu IT-Akadémia v Tatranských Matliaroch sme za účasti overovateľov z radov učiteľov ZŠ získali spätnú väzbu na viaceré z aktivít projektu IT-Akadémia.[12]

Obrázok 7: Účastníci konferencie, Tatranské Matliare 2018

- „Ďakujem za možnosť bližšie sa zoznámiť s projektom IT Akadémie - o tom, čo projekt ponúka som mala úplne inú predstavu vytvorenú na základe niektorých referencií a tiež vlastných skúseností s inými podobnými iniciatívami z minulosti. Vďaka tomu, že ma presvedčili sme ako škola využili možnosť vstúpiť do projektu ako partnerská škola a následne som sa zúčastnila Konferencie národného projektu pre učiteľov ZŠ. Bola som nesmierne očarená týmto podujatím: atmosféra konferencie, obsah vzdelávacích aktivít, cenné diskusie s kolegami i plenárne prednášky uvoľnili obrovské množstvo energie, ktorú som si odnášala domov [...].“
- „Po hodine učenej podľa metodiky: „ hodina sa nám páčila pretože sme sa neučili“ – oni sa tak zaujali metodikou, že brali to ako hru a nie ako učenie... Pani učiteľka Vy nás učíte myslieť...nie vzorce... . Vyjadrenie učiteľov: Sme všetci nabudení...stretnutie - konferencia nám dala novú pozitívnu energiu, motiváciu pre bádateľsky orientované učenie... Metodiky učia deti spolupracovať, majú prepojenie na praktický život... Žiaci majú radosť z objavovania...učíme deti klásť otázky [...]. Ak učíme 20 rokov rutinne učiteľ musí chcieť si pozrieť tú metodiku... Bola som prekvapená z toho ako aj nie najšikovnejší žiaci podľa metodik pekne pracovali [...].“
- „Pre pedagógov našej školy[13] je zaradenie našej školy do siete 60-tich vybraných škôl Slovenska označených ako "IT Akadémia partner" novou výzvou na skvalitnenie vzdelávacieho procesu, získavanie nových skúseností, ich šírenie a diseminiáciu až na úroveň zmien národného kurikula. Pre našich terajších a budúcich žiakov je to ale predovšetkým výnimočná príležitosť vzdelávať sa moderne a s novým obsahom v nadštandardných podmienkach na pomery ostatných slovenských škôl.“

POĎAKOVANIE

Tento článok bol vytvorený v rámci národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka

[12] https://de-de.facebook.com/pg/itakademia.sk/posts/

[13] (http://www.zshu.sk/index.php/projekty/it-akademia/item/673-sme-medzi-60-tkou-slovenskych-it-akademii)

podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje

BIBLIOGRAFICKÉ ODKAZY

[1] *What is Inquiry Based Learning (EBL)?* Centre for Excellence in Enquiry-Based Learning. University of Manchester. Retrieved October 2012. [online] [cit. 2019-01-28] Dostupné z: http://www.ceebl.manchester.ac.uk/ebl/

[2] *5E.* [online] [cit. 2019-01-28] Dostupné z: https://sites.google.com/site/jsaenzportfolio/_/rsrc/1472860875831/learning-cycle-siop-model/5E%20Learning%20Cycle%20pic.jpg?height=178&width=200

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Tvorba učebnice programování a vzdělávací cíle
Creation of programming textbook and educational goals

Jiří Vaníček
Jihočeská univerzita
Jeronýmova 10
37115 České Budějovice
Česko
vanicek@pf.jcu.cz

Ingrid Nagyová
Slovenská technická univerzita
Ilkovičova 2
842 16 Bratislava
Slovensko
ingrid.nagyova@stuba.sk

Monika Tomcsányiová
Univerzita Komenského
Mlynská dolina F1
842 48 Bratislava
Slovensko
tomcsanyiova@fmph.uniba.sk

ABSTRACT
In this paper, we describe creation of educational goals of created textbooks. We consider historical view of the role of programming education goals and relationship educational goals to obligatory state curricular documents. We discuss two authors' approaches to goals creation and show results of comparison of required and declared goals in an example of the textbook of lower secondary programming in Scratch. We add some experience about fulfilling the goals obtained in piloting the textbook.

Keywords
Programming. Lower secondary. Textbook. Scratch. Educational goals.

ABSTRAKT
V tomto článku popisujeme tvorbu vzdělávacích cílů u vytvářených učebnic. Uvažujeme historický pohled na roli cílů výuky programování a vztah vzdělávacích cílů k závazným státním kurikulárním dokumentům. Uvádíme dva přístupy autorů k tvorbě cílů a na příkladu učebnice programování pro 2. stupeň ZŠ ve Scratch ukazujeme výsledky komparace cílů požadovaných a deklarovaných, doplněné prvotními zkušenostmi z naplňování cílů při pilotním ověřování učebnice.

Klíčová slova
Programování. 2. stupeň ZŠ. Učebnice. Scratch. Vzdělávací cíle.

1 ÚVOD
Potřeba inovace kurikula informatiky na 2. stupni ZŠ v ČR a start projektu PRIM – Podpora rozvíjení informatického myšlení [1] svedla autory tohoto článku do týmu vytvářejícího učebnici programování ve Scratch. o vytvářené učebnici a jejích teoretických základech bylo referováno na minulé konferenci Didinfo [2].

Tvorba učebnice skýtá řadu aspektů, z nich se chceme věnovat tvorbě vzdělávacích cílů a jejich implementaci ve vytvářeném kurikulu.

2 VZDĚLÁVACÍ CÍLE
Na cíle výuky programování a jejich tvorbu se můžeme podívat ze dvou základních pohledů, které se u nás, co se týče dominance, v průběhu historie střídaly:

- dovednost programovat jako cíl,
- dovednost programovat jako prostředek.

Prvním pohledem je znalost programování u učícího se jako finální stav, kterého chceme dosáhnout. Tento cíl je prioritní u profesionálního vzdělávání, u vzdělávání programátorů. v minulosti bylo programování vnímáno pouze jako dovednost pro úzkou skupinu IT specialistů, proto tento pohled dominoval i ve vzdělávacích materiálech.

Druhý pohled vnímá programování jako součást všeobecného vzdělání [3], jako prostředek k rozvoji jedince. Programování rozvíjí mentální schopnosti, které zahrnujeme pod termín informatické myšlení, které bývá označováno za novou gramotnost pro 21. století [4]. Tento pohled se aktuálně promítá jako dominantní do českého i slovenského národního kurikula základního vzdělávání.

Český návrh revize Rámcového vzdělávacího programu (dále RVP) ve vzdělávací oblasti informatika a ICT [5] v části Algoritmizace a programování určuje očekávané výstupy takto:

Žák

1. po přečtení jednotlivých kroků algoritmu nebo programu vysvětlí celý postup; určí problém, který je daným algoritmem řešen;
2. rozdělí problém na jednotlivě řešitelné části a navrhne a popíše kroky k jejich řešení;
3. upraví daný algoritmus pro jiné problémy, ověří správnost postupu navrženého i někým jiným, najde a opraví v něm případnou chybu;

ISBN 978-80-557-1533-9, ISSN 2454-051X.

4. navrhne různé algoritmy pro řešení problému; vybere z více možností vhodný algoritmus pro řešený problém a svůj výběr zdůvodní;
5. v blokově orientovaném programovacím jazyce sestaví přehledný program pro vyřešení zadaného problému; program otestuje a opraví v něm případné běhové a logické chyby;
6. používá opakování, větvení programu, proměnné, podprogramy s parametry; používá události k paralelnímu spouštění podprogramů.

Slovenský Inovovaný státní vzdělávací program (dále iŠVP) určuje cíle oblasti Algoritmické řešení problémů následovně [6].

Žáci

- uvažují o algoritmech, hledají a nachází algoritmická řešení problémů, vytváří návody, programy podle daných pravidel,
- logicky uvažují, argumentují, hodnotí, činí odůvodnitelná rozhodnutí.

Dále iŠVP určuje výkonový standard v oblasti Algoritmické řešení problémů pro analýzu problémů, pro jazyk řešení, pro využití posloupnosti příkazů a cyklů a rovněž pro interpretaci zápisů řešení a hledání a opravu chyb. iŠVP neurčuje přímo programovací jazyk, pomocí něhož mají žáci určený standard naplnit. Je na rozhodnutí školy, resp. učitele, který jazyk pro zápis řešení použije. Některé základní školy na Slovensku pokračují v tradici a nadále využívají Imagine Logo. Jiné přešli nebo aktuálně přechází na Scratch.

3 PŘÍSTUPY K TVORBĚ UČEBNICE z POHLEDU CÍLŮ

Při tvorbě učebnice mohou její autoři, stejně tak jako např. učitel připravující své kurikulum, které bude realizovat se svými žáky, volit mezi dvěma přístupy.

- *Přístup naplňovací* znamená vzít napsané (předepsané) cíle či očekávané výstupy, případně strukturovat tematickou oblast, ke které se očekávané výstupy vztahují, a hledat k nim vhodné úlohy, které k nim budou směřovat, naučí je (takový přístup lze nalézt např. u učebnic [7] nebo [8]).
- *Přístup intuitivní* vychází často ze zkušeností s předchozí výukou, z precedentu nebo z primární orientace na zvládnutí nějakého prostředí či pomůcky, na motivaci a efekt. Na tomto základě si autor vědomě či podvědomě stanoví dovednosti a koncepty, které chce žáka naučit. Tedy nejprve sestaví kompaktní kurikulum a až poté se zabývá souladem jeho kurikula s předepsanými výstupy (takový přístup lze nalézt např. u učebnic [9] nebo [10]).

Oba přístupy mají své přednosti a nevýhody. Naplňovací přístup zaručí, že pokryje požadované výstupy. Může mít ale potíže s integritou obsahu, když roztříštěnost témat neposkytuje linku, která by procházela celou učebnicí a která by umožnila postupné naplnění metacílů, které leží nad příslušnou vzdělávací oblastí a řadíme je k životním dovednostem [11] nebo je zahrnujeme do klíčových kompetencí. Intuitivní přístup vytvoří ucelený materiál, který se ovšem může odchylovat od požadovaných vzdělávacích cílů a doplnění nebo dodatečná úprava obsahu k jejich naplnění může znamenat v některých případech zásadní problém, v každém případě je to časově náročnější.

Náš autorský tým šel cestou intuitivní: nejprve sestavil kurikulum na základě předem stanovených konceptů, které s programováním souvisí [2], a teprve ve druhém sledu se zabýval, zda a nakolik vytvořená učebnice naplňuje předepsané výstupy. To poskytovalo i možnost podívat se na tyto předepsané výstupy z pohledu, zda jsou adekvátní a zda jsou v rámci výuky programování pro daný věk žáka nenásilně realizovatelné.

4 KOMPARACE POŽADOVANÝCH A DEKLAROVANÝCH VZDĚLÁVACÍCH CÍLŮ

Cíle konkrétních úloh, jejich posloupnost a provázanost v rámci jednotlivých kapitol jsme analyzovali ze dvou pohledů – z pohledu složek informatického myšlení a z pohledu nově připravovaného RVP.

Existuje řada definic informatického myšlení. Posouzení komplexnosti cílů naší učebnice programování vycházelo z definice informatického myšlení jako myšlenkového procesu zaměřeného na řešení problémů, který rozvíjí abstrakci, dekompozici, algoritmický přístup, evaluaci a generalizaci [12]. Dagienė a kol. [13] specifikuje způsoby, jak jednotlivé složky informatického myšlení ve vzdělávacím procesu identifikovat.

Etapu komparace předepsaných a deklarovaných vzdělávacích cílů jsme realizovali od dílčího k celkovému, začali jsme u jednotlivých aktivit a úloh. U každé jsme určili hlavní vzdělávací cíle z pohledu programování, tyto cíle jsme v rámci každé kapitoly sdružovali a nejčastěji se objevující cíle jsme deklarovali jako cíle dané kapitoly. Tyto jakési mikrocíle jsme pak konfrontovali jednak s dvoudimenzionální taxonomií cílů, vztahující se k informatickému myšlení [12], jednak s předepsanými výstupy revidovaných RVP [14]. Tam, kde jsme následně zjistili, že naše kurikulum cíle nepokrývá dostatečně nebo se zde objevuje diskrepance, jsme obsah kapitoly upravovali.

Důvodem pro úpravu nebo i vyřazení úlohy se stávaly nejasné nebo nesouladné cíle (např. kdy cíl byl velmi lehký, ale k jeho dosažení pomocí dané aktivity žák potřeboval jiné pokročilé znalosti, které se učily až později) nebo kombinace náročných cílů s jejich slabou souvislostí s programováním (např. kdy žáky spíše trápila geometrie nebo výpočty).

Při rozhodováních o úpravách obsahu učebnice jsme také využívali průběžně probíhající ověřování na školách. Sledovali jsme, zda vytvořené aktivity jsou schopny autory deklarované nebo předepsané cíle naplnit, případně zda jsou učitelé schopni takovým způsobem výuku vést.

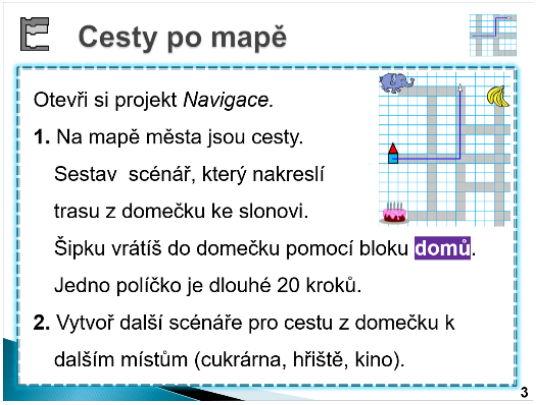
5 ILUSTRACE

Zde chceme výše uvedené teoretické koncepty ilustrovat na konkrétních úlohách a aktivitách ve třech vybraných didaktických prostředích, které používáme v počátečních kapitolách pro začátečníky. Pro ilustraci uvádíme několik konkrétních úloh, nad kterými jsme se pozastavili.

Příklad 1:

Při prvním ověřování žákovských listů jsme zjistili, že žáci mají problém s relativním otáčením (vlevo, vpravo). Rozhodli jsme se podpořit žákovo porozumění v této oblasti, pro které se ukázala vhodnou třetí kapitola zaměřena na práci s vlastními bloky. Vznikl tak námět na nové prostředí Cesty na mapě – viz obrázek 1.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

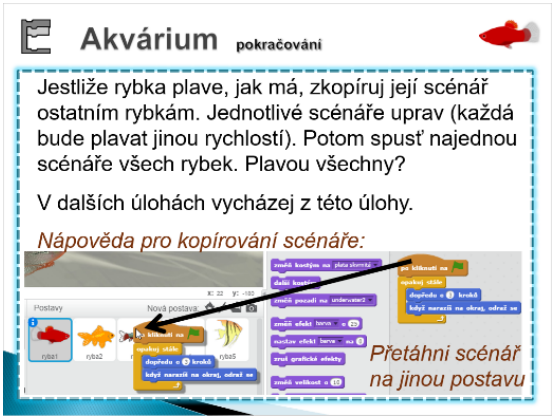


Obrázek 2: Prostředí Cesty na mapě

Kromě uvedených cílů se žáci v úlohách, které řeší v prostředí Cesty na mapě, obeznamují s datovou strukturou graf. Vrcholy grafu jsou zvířátka, hrany jsou cesty mezi nimi. Úloha, aniž by dlouze vysvětlovala teorii grafů, umožňuje žákům vyhledat cestu k jednotlivým vrcholům. Práce v prostředí je může dokonce motivovat k tomu, aby uvažovali, která z nalezených cest je nejkratší.

Příklad 2:

V projektu Akvárium si žáci hlouběji osvojí práci s více postavami. Po vytvoření scénáře pro jednu rybku zkopírují tento scénář ostatním rybám. Tím si uvědomí, že část scénáře, která definuje plavání, je pro všechny ryby stejná (viz obrázek 2). Některé vlastnosti tohoto scénáře (např. rychlost plavání) však mohou být pro jednotlivé rybky různé. v dalších částech této úlohy žáci definují jednotlivým rybkám vlastní scénář, v němž připraví různé chování pro konkrétní rybku – některá ryba se při kliknutí otočí, jiná se zmenší, další změní barvu svého kostýmu apod.



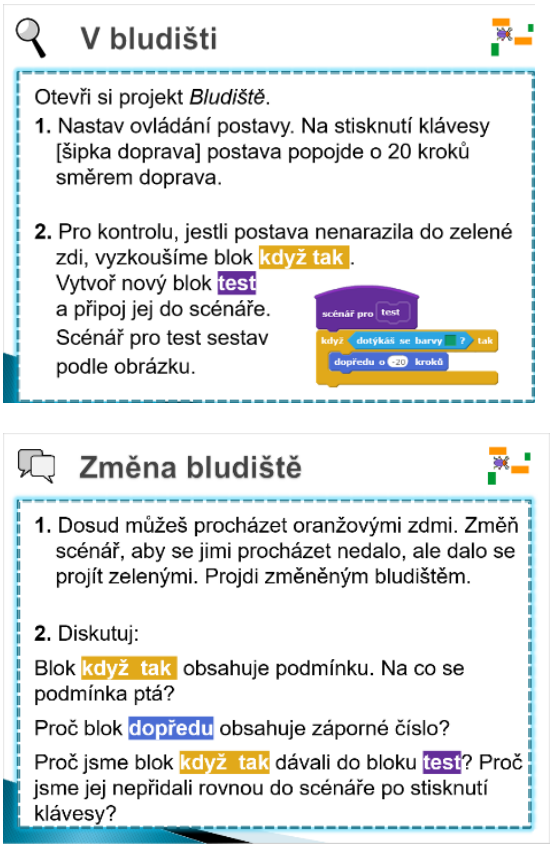
Obrázek 3: Prostředí Akvárium

Řešením takovýchto úloh žáci poznávají, jak rozdělit problém na části, učí se analyzovat, které scénáře mají postavy stejné a kde jsou rozdíly a následně odhalují, jak se to projevuje v chování jednotlivých postav. Přetahováním scénářů z jedné postavy na druhou si žáci z práce v prostředí Scratch osvojují techniku kopírování hotového scénáře. Tento přístup využijí v budoucnu při řešení dalších, často mnohem náročnějších úloh (např. [15]).

Z pohledu informatických principů tímto způsobem žáci zažívají propedeutiku objektově orientovaného programování – vyhledání stejných vlastností objektů a příprava metod, které zabezpečí nastavení a chování pro jednotlivé objekty.

První úlohou kapitoly 7 Rozhodování v její první verzi byla úloha Přistání na Marsu (viz [10]). Z diskuzí autorů k této kapitole vyplynulo, že by bylo vhodné vložit na začátek kapitoly jednodušší úlohu, která by se cíleně soustředila jenom na to, aby si žáci ujasnili, kdy a jak se postava rozhoduje. Vznikla úloha v bludišti (viz obrázek 3), v níž se žáci setkají s jednou postavou, která má připravený scénář test.

Příklad 3:



Obrázek 1: Prostředí Bludiště

Věříme, že prozkoumáním tohoto krátkého hotového scénáře pomocí zadaných úloh žáci lépe pochopí, k čemu podmíněný příkaz slouží. Poté, co si scénář vyzkouší, úloha pokračuje tím, že žáci scénář upraví. Návrh hodiny pokračuje diskuzí žáků o scénáři, čímž získají možnost lépe filosofii podmínek uchopit.

Z hlediska informatiky je podmíněný příkaz jeden ze základních programátorských konceptů, který umožňuje řídit běh programu. v prostředí Scratch se nejčastěji využívá tak, že žáci programují zadání, v nichž je důležité, zda se postava dotýká nějaké barvy nebo jiné postavy.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

6 VÝSLEDKY ANALÝZY VZDĚLÁVACÍCH CÍLŮ

Tabulka 1: Vzdělávací cíle jednotlivých kapitol učebnice

Kapitola		Informatické myšlení	RVP
1	**Scénář**	Algoritmizace: - spouštění programů - seřazení příkazů ve správném pořadí	(5) v blokově orientovaném programovacím jazyce sestaví přehledný program pro vyřešení zadaného problému (3) najde a opraví v programu případnou chybu
2	**Opakování**	Algoritmizace: - vytváření algoritmů a programů Generalizace - identifikace vzorů na základě podobnosti	(5) program otestuje a opraví v něm případné běhové a logické chyby (6) používá opakování (1) po přečtení jednotlivých kroků algoritmu nebo programu vysvětlí celý postup (1) určí problém, který je daným algoritmem řešen
3	**Procedury**	Dekompozice: - rozklad problému na podproblémy - přemýšlení o problému s ohledem na jeho součásti Algoritmizace: - vytváření algoritmů a programů	(6) používá podprogramy (2) rozdělí problém na jednotlivě řešitelné části a navrhne a popíše kroky k jejich řešení (3) ověří správnost postupu navrženého i někým jiným (3) upraví daný algoritmus pro jiné problémy (5) sestaví přehledný program pro vyřešení zadaného problému (1) po přečtení jednotlivých kroků algoritmu nebo programu vysvětlí celý postup
4	**Opakování s podmínkou**	Algoritmizace - přemýšlení o postupu a pravidlech řešení problému	(3) ověří správnost postupu navrženého i někým jiným (6) používá opakování (3) najde a opraví v programu případnou chybu (4) navrhne různé algoritmy pro řešení problému (5) program otestuje a opraví v něm případné běhové a logické chyby
5	**Více postav, vstupy**	Algoritmizace	(4) vybere z více možností vhodný algoritmus pro řešený problém a svůj výběr zdůvodní (6) používá události k paralelnímu spouštění podprogramů (5) program otestuje a opraví v něm případné běhové a logické chyby
6	**Postavy se spolu domlouvají**	Algoritmizace	(6) používá události k paralelnímu spouštění podprogramů (1) po přečtení jednotlivých kroků algoritmu nebo programu vysvětlí celý postup
7	**Rozhodování**	Algoritmizace	(6) používá větvení programu (5) program otestuje a opraví v něm případné běhové a logické chyby (3) upraví daný algoritmus pro jiné problémy (3) ověří správnost postupu navrženého i někým jiným
8	**Souřadnice**		(3) upraví daný algoritmus pro jiné problémy (6) používá opakování
9	**Procedury s parametry**	Generalizace: - na základě vyřešené úlohy řeší celou třídu podobných úloh Dekompozice - přemýšlení o problému s ohledem na jeho součásti	(6) používá podprogramy s parametry (6) používá události k paralelnímu spouštění podprogramů
10	**Proměnné**	Dekompozice - rozklad problému, řešení jeho součástí a spojení do celku	(6) používá proměnné (4) navrhne různé algoritmy pro řešení problému (3) upraví daný algoritmus pro jiné problémy

ISBN 978-80-557-1533-9, ISSN 2454-051X.

Tabulka 1 shrnuje výsledky analýzy, kterou jsme provedli s ohledem na zaměření a cíle jednotlivých kapitol učebnice. Ve sloupci RVP jsou vypsány jednotlivé očekávané výstupy nově připravovaného RVP jak odpovídají jednotlivým kapitolám. Očekávané výstupy jsou číslovány podle seznamu uvedeného výše a již letmý pohled napovídá, že nejlépe jsou v učebnici pokryty výstupy 3, 5 a 6.

6.1 Diskuse výsledků analýzy

Výstup 1 (analýza a vysvětlení postupu řešení) se v učebnici vyskytuje zejména v úlohách určených pro diskuzi, kdy žáci čtou scénář s porozuměním (viz obrázek 4). Při čtení scénářů žáci musí nejenom pochopit postup řešení, ale musí si představit postavu na scéně, její kostýmy a změny, které scénář způsobí.

Obrázek 4: Ilustrace diskusní aktivity k tématu opakování s podmínkou ukončení. Kostýmy postavy jsou písmena seřazená podle abecedy, blok další kostým nastaví v pořadí další písmeno abecedy, po Z následuje A.

Výstup 4, zaměřený na různorodost postupů řešení, je v učebnici pokryt zejména při tvorbě scénářů s opakováním. Žáci jsou nabádáni k tomu, aby zkoušeli a hledali možnosti a rozdíly ve využití bloků **opakuj**, **opakuj dokud nenastane** a **opakuj stále**.

Nejobtížnější se ukazuje pokrytí bodu 2 zaměřeného na dekompozici problému. Jedním z hlavních principů při tvorbě učebnice bylo vytvářet časově krátké úlohy zaměřené na jeden koncept, které bude schopen vést i méně zkušený učitel. Problémy prezentované v těchto úlohách však lze jen obtížně dekomponovat na menší problémy, tak aby výsledkem bylo řešení původního problému. Již jednoduchý problém kreslení domečku ze čtverce a trojúhelníku se ukazuje jako velice obtížný, a to jak pro samotného žáka, tak pro organizaci výuky ve třídě. Žáci jsou pouze v několika úlohách poslední kapitoly vedeni k tomu, jak si rozdělit problém na dílčí úlohy, které již umí vyřešit. Příkladem může být naprogramování hry simulující autodráhu, v níž žák postupně definuje pohyb auta po závodním okruhu, stanovuje a programuje pravidla pro start a cíl, činnosti pro postavu komentátora závodu atd.

Analýza cílů s pomocí výstupů z ověřování na školách výrazně přispěje k hledání cest, jak do učebnice doplnit např. úlohy na rozklad problému a stanovení postupu řešení. Podobně uvažujeme o doplnění dalších diskuzních úloh na čtení scénáře s porozuměním především u náročnějších konceptů. Další skupinou úloh, které je třeba posílit, jsou úlohy umožňující různorodost řešení, a to s vědomím, že tyto úlohy v sobě skrývají nebezpečí pro učitele – začátečníka, který musí být schopen v žákovských řešeních rozeznat a uznat za plnohodnotná i jiná než vzorová autorská řešení.

V tabulce 1 ve sloupci Informatické myšlení jsme se pokusili začlenit cíle kapitol učebnice podle jednotlivých složek informatického myšlení. Z tabulky je zřejmé, že učebnice velice dobře pokrývá oblast algoritmizace a algoritmického přístupu. v učebnici lze vysledovat také prvky generalizace (zobecnění) a dekompozice problémů – viz popis výše. Jako obtížné se jeví uplatnění prvků evaluace (návrh ověření správnosti sestaveného programu v různých situacích) a posouzení nejlepšího řešení. Snažíme se o to v několika málo diskuzních úlohách, například pro určení nejlepšího scénáře pro vykreslení domku.

6.2 Pilotní ověřování učebnice z pohledu naplnění cílů

Evaluace učebnice započala vloni na dvou základních školách v Českých Budějovicích a v Ostravě pod dohledem autorů. Po úpravách původního obsahu a zveřejnění betaverze [10] je letos (ve školním roce 2018/2019) učebnice evaluována postupně sedmi dalšími učiteli, kteří mají s autory kontakt pouze elektronicky (e-mail, skype, speciální software pro poskytování zpětné vazby, vyvinutý v rámci projektu PRIM partnery z Univerzity Palackého [16]). Učitelé pravidelně informují o průběhu výuky podle učebnice, hodnotí práci žáků – co se povedlo a s čím měli problémy.

Postřehy a zkušenosti učitelů cenným zdrojem podnětů pro metodickou příručku. Problémy, popisované učiteli, lze očekávat v dalších třídách, a tudíž je nutné na ně v metodice upozornit. Mezi řádky hodnocení lze často vyčíst způsob, jak učitelé s žáky pracují. Zde je cenné vidět, jak učitel sám musí mnohdy měnit zaběhlý styl a formy učení, jak je pro něho obtížné nechat žáky svobodně diskutovat, neopravovat chyby, ale vést žáky k pochopení chybného postupu a jeho nápravě. Žáci obvykle zvládají změnu stylu učení lépe, co potvrzuje vyjádření jedné paní učitelky o postupné proměně chování žáků v průběhu první kapitoly: „U vlaků se moc nediskutovalo, číslice a písmena už vyvolaly větší debaty, žáci chtěli diskutovat. Při čtení scénářů tvořících slova se už diskutovalo velmi živě, žáci si vzájemně vysvětlovali řešení.“

Žáci podle hodnocení učitelů pracují se zájmem, pomalu se učí spolupracovat ve dvojicích, diskutovat a argumentovat. Také se ukazuje, jak učitel sám proměňuje své poznání o této formě práce žáků. Jedna paní učitelka napsala, jak až v páté hodině pochopila, jak je důležité žáky ve dvojicích střídat, nechat spolupracovat lepší a slabší žáky, kteří se tak mohou vzájemně obohatit.

Z pohledu naplnění cílů ověřování jednoznačně ukazuje, že učebnice podporuje algoritmický přístup k řešení problémů. Ve zprávách od učitelů se objevují ukázky práce žáků, kteří do svých děl vkládají mnoho invence. Zajímavý je pohled učitelů na diskusní úlohy. Jedna učitelka popsala situaci ve třídě při kreslení kružnice: „Na 360° přišli (žáci) rychle, ale kruh byl zážitek - měli v pořádku úhel i opakování, ale zapomněli zmenšit krok dopředu a pavouček zmizel ze scény, takže jsme chvilku chodili po třídě a názorně jsme předváděli běhání pavoučka. Uvidíme příští hodinu, kdo si to zapamatoval.“. Ukazuje se, že diskuzní úlohy jsou časově velice náročné. Obvykle těmito aktivitami hodina končí, protože žáci jsou vzniklým problémem a argumentací plně zaujati.

Toto jsou pozitivní zprávy ze škol jak při práci s učebnicí, tak při samotném programování ve Scratchi. Tak, jak učitelé jen pomalu získávají důvěru v to, že vše bude v pro ně neznámém výukovém prostředí srozumitelné a bude fungovat, my autoři řešíme s ohledem na naplnění cílů učebnice doplnění vhodných úloh.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

Vyvstávají tyto otázky, týkající se otevřených úloh nebo úloh s více správnými řešeními:

- Jak učitel přijme jiné než „obvyklé“ (v učebnici deklarované) řešení? Vždyť toto řešení může být správné. Jak se postaví k řešení, kterému nerozumí? Jak přijme nejednoznačnou úlohu?

- Dokáže učitel vést žáky tak, aby každý vymyslel svoje řešení problému a toto řešení také implementoval? Lze to od učitele, který se v oblasti programování orientuje jenom zhruba, očekávat?

Doplnění vhodných úloh pro kompletní naplnění cílů učebnice předpokládá najít takové úlohy pro žáky, u kterých bude možné odpovědět na položené otázky „ano“. První otázka pokrývá celou oblast úloh pro hledání optimálního řešení bez ohledu na vnímání pojmu „optimální“. Druhá směřuje k uplatnění principu dekompozice, k rozkladu problému na podproblémy a k následnému spojování dílčích řešení v konečný výsledek.

7 ZÁVĚREM

V článku jsme prezentovali přístup k tvorbě učebnice programování, v němž se vzdělávací cíle, založené na konceptech vztahujících se k programování, dostávají do souladu s vyžadovanými vzdělávacími cíli až po vytvoření vzdělávacího obsahu.

Ukázalo se, že je to možný postup a že následné porovnání cílů umožní obohatit vzdělávací obsah přesto udržet jednotnou linii výuky. Jako velice důležité se jeví kvalitně provedené pilotní ověřování na školách, jestliže pokud možno co nejvíce simuluje očekávanou budoucí běžnou výuku.

PODĚKOVÁNÍ

Vývoj učebnice byl realizován z prostředků projektu CZ.02.3.68/0.0/0.0/16_036/0005322 PRIM – Podpora rozvíjení informatického myšlení.

BIBLIOGRAFICKÉ ODKAZY

[1] *Web projektu PRIM.* http://www.imysleni.cz. Jihočeská univerzita, 2017.

[2] VANÍČEK, J., NAGYOVÁ, I., TOMCSÁNYIOVÁ, M. Učebnice programování pro 2. stupeň ZŠ ve Scratchi, její teoretická východiska a charakter. In DRÁBKOVÁ J., BERKI, J. (eds*.) Sborník konference Didinfo 2018* [online]. Liberec, 2018, s. 25 - 32.

[3] GANDER, W. Informatics and general education. In Gülbahar, Y., Erinç, K. (Eds.) *Informatics in schools, teaching and learning perspectives*. Heidelberg: Springer LNCS: 1–7, 2014.

[4] WING, J. *Research notebook: Computational thinking: what and why?* Carnegie Mellon University, Pittsburgh: 2010. Dostupné na: https://www.cs.cmu.edu/~CompThink/resources/TheLinkWing.pdf

[5] *NÚV Návrh revizí rámcových vzdělávacích programů v oblasti informatiky a informačních a komunikačních technologií*. Praha: NÚV, 2018. Dostupné na: http://www.nuv.cz/file/3362/

[6] *Inovovaný štátny vzdělávací program - informatika.* Bratislava: Štátny pedagogický ústav, 2014. Dostupné na: http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_nsv_2014.pdf

[7] BERKI, J., DRÁBKOVÁ, J. *Základy informatiky pro základní školy.* Vzdělávací materiály, beta verze. Informatické myšlení: Web pro podporu školské informatiky, 2018. Dostupné na: http://imysleni.cz/ucebnice/zaklady-informatiky-pro-zakladni-skoly/

[8] FILIPI, Z., MAINZ, D., FADRHONC, J. *Práce s daty pro 5. až 7. ročník základní školy.* Vzdělávací materiály, beta verze. Informatické myšlení: Web pro podporu školské informatiky, 2018. Dostupné na: http://imysleni.cz/ucebnice/prace-s-daty-pro-5-az-7-tridu-zakladni-skoly/

[9] NOVÁK, M., PECH, J. *Arduino: Učebnice pro střední školy.* Vzdělávací materiály, beta verze. Informatické myšlení: Web pro podporu školské informatiky, 2018. Dostupné na: http://imysleni.cz/ucebnice/arduino-ucebnice-pro-stredni-skoly/

[10] VANÍČEK, J., NAGYOVÁ, I., TOMCSÁNYIOVÁ, M. *Programování ve Scratch pro 2. stupeň základní školy.* Vzdělávací materiály, beta verze. Informatické myšlení: Web pro podporu školské informatiky, 2018. Dostupné na: http://imysleni.cz/ucebnice/programovani-ve-scratchi-pro-2-stupen-zakladni-skoly/

[11] KOVALIK, S. *Integrovaná tematická výuka: Model*. Kroměříž: Spirála, 1995.

[12] SELBY, C., WOOLLARD, J. Computational Thinking: The Developing Definition. UK: University of Southampton, 2013. Dostupné na: http://eprints.soton.ac.uk/356481

[13] DAGIENE, V., SENTANCE, S., STUPURIENE, G. Developing a Two-Dimensional Categorization System for Educational Tasks in Informatics. In *INFORMATICA, 2017*, Vol. 28, No. 1, pp. 23-44. Dostupné na: https://www.mii.lt/Informatica/pdf/INFO1127.pdf

[14] *Informatika – rámec očekávaných výstupů*. Praha: Národní ústav pro vzdělávání, 2018. Dostupné na: http://www.nuv.cz/file/3361/

[15] ČERNOCHOVÁ, M., ŠTÍPEK, J., VAŇKOVÁ, P. Programování ve Scratch II (projekty pro 2. stupeň základní školy). Vzdělávací materiály, beta verze. Informatické myšlení: Web pro podporu školské informatiky, 2018. Dostupné na: http://imysleni.cz/ucebnice/programovani-ve-scratchi-ii-projekty-pro-2-stupen-zakladni-skoly/

[16] PRIM. *Evaluační systém pro zajištění zpětné vazby od učitelů z praxe.* http://prim.upol.cz/, 2018.

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Robotické modely s LEGO WeDo 2.0 v treťom a štvrtom ročníku

Robotic models with LEGO WeDo 2.0 in 3rd and 4th grade

Michaela Veselovská
KDMFI, Fakulta matematiky, fyziky a informatiky
Univerzita Komenského v Bratislave
Mlynská dolina
84248 Bratislava
Slovensko
veselovska@fmph.uniba.sk

Zuzana Kubincová
KDMFI, Fakulta matematiky, fyziky a informatiky
Univerzita Komenského v Bratislave
Mlynská dolina
84248 Bratislava
Slovensko
kubincova@fmph.uniba.sk

Karolína Mayerová
KDMFI, Fakulta matematiky, fyziky a informatiky
Univerzita Komenského v Bratislave
Mlynská dolina
84248 Bratislava
Slovensko
mayerova@fmph.uniba.sk

ABSTRACT

In this paper, we are dealing with the implementation of activities with the LEGO WeDo 2.0 robotic kit into primary education. In particular, we are reporting on an introductory activity for third and fourth grade pupils. On the basis of qualitative data analysis, we have discovered several differences in the creation of their first robotic models when comparing pupils in different grades. We have identified differences in terms of building, programming and presenting models. We plan to compare our findings from the introductory activity and also other activities with the findings of the Taiwanese researchers, with whom we collaborate in testing the activities with the robotic kit. We plan to compare the results of the analysis from a different cultural point of view and adapt the activities to both national curricula so that they can be part of the ordinary primary education.

Keywords

Educational robotics. Children. Programming education. LEGO WeDo 2.0. Comparative study.

ABSTRAKT

V príspevku sa zaoberáme implementáciou robotickej stavebnice LEGO WeDo 2.0 do primárneho vzdelávania. Podrobnejšie sa venujeme úvodnej aktivite vytvorenej pre žiakov tretieho a štvrtého ročníka. Na základe kvalitatívnej analýzy dát sme objavili niekoľko rozdielov pri tvorbe prvých vlastných robotických modelov medzi žiakmi jednotlivých ročníkov. Zistili sme odlišnosti z hľadiska stavby, programovania i prezentovania modelov. Naše zistenia z úvodnej, ale i ďalších aktivít plánujeme porovnávať so zisteniami taiwanských výskumníkov, s ktorými spolupracujeme na testovaní aktivít vytvorených s danou robotickou stavebnicou. Výsledky analýz plánujeme porovnať i z pohľadu odlišného kultúrneho zázemia a prispôsobiť ich obom národným kurikulám tak, aby mohli byť súčasťou bežného primárneho vzdelávania.

Kľúčové slová

Edukačná robotika. Deti. Výučba programovania. LEGO WeDo 2.0. Porovnávacia štúdia.

1. ÚVOD

Konštruktivizmus [1, 2] a konštrukcionizmus [3] ako dve z v súčasnosti asi najuznávanejších teórií učenia sa sú založené na myšlienkach, že učiaci sa si svoje poznanie aktívne konštruuje sám na základe jeho predchádzajúcich skúseností z reálneho sveta a jeho proces učenia sa je omnoho efektívnejší, keď pri ňom aj niečo aktívne tvorí. Nezáleží pritom na tom, či je výsledkom niečo jednoduché alebo zložité, najlepšie však je, ak ide o zmysluplný reálny produkt. Sociálny konštruktivizmus [4] ďalej rozširuje tieto teórie o myšlienku, že učenie sa nie je individuálny proces, ale ide o spoločenskú aktivitu, ktorá sa deje vždy v nejakom sociálnom kontexte.

Pre implementáciu konštruktivistických a konštrukcionistických metódy vo vyučovaní sa veľmi dobre hodia aktivity využívajúce robotické stavebnice [5]. Zvyčajne sú navrhnuté ako projektovo-orientované aktivity, v ktorých žiaci pracujú v menších tímoch, kde spoločne stavajú robotický model a spoločne programujú jeho správanie. Ide teda naozaj o sociálne učenie, pri ktorom žiaci prostredníctvom konštruovania hmatateľného objektu, ktorý je modelom objektu z reálneho sveta, kolaboratívne konštruujú svoje nové poznanie. Prostredie edukačnej robotiky je pre žiakov natoľko podnetné, že ich motivuje k nadobudnutiu akýchkoľvek zručností, ktoré potrebujú k dokončeniu svojho projektu [6]. Získavajú nielen hlbšie vedomosti o tom, ako fungujú technológie, ale zmysluplným a zábavným spôsobom sa učia spájať poznanie z rozličných disciplín a aplikovať vedomosti a zručnosti nadobudnuté v iných predmetoch. Navyše, sa pri kolaboratívnej práci u žiakov rozvíjajú aj interpersonálne kompetencie, zručnosti pre prácu v tíme, komunikačné zručnosti, kritické myslenie, kreativita a pod. Prednosťou edukačnej robotiky je, že môže byť využitá v rôznych predmetoch, nielen v predmetoch technického alebo prírodovedného zamerania, ale rovnako napr. v hudobnej alebo výtvarnej výchove, pri výučbe jazykov a inde [7, 8, 9].

Keďže edukačnú robotiku považujeme za veľmi perspektívnu oblasť vzdelávania žiakov už od prvého stupňa základnej školy, začali sme s jej implementáciou už pred niekoľkými rokmi. Vytvorili sme sériu aktivít s robotickou stavebnicou LEGO WeDo 1.0 spolu s metodickými materiálmi pre učiteľa pre prvý i druhý stupeň základnej školy a otestovali sme ich v podmienkach reálneho vyučovania [10, 11, 12, 13]. Výsledky boli veľmi

ISBN 978-80-557-1533-9, ISSN 2454-051X.

povzbudivé. Medzičasom vznikla nová verzia tejto robotickej stavebnice, s ktorou sa však vytvorené aktivity nedajú bez úprav použiť. Preto v súčasnosti vyvíjame novú sériu aktivít pre LEGO WeDo 2.0, zatiaľ pre prvý stupeň základnej školy.

V rámci spoločného projektu s taiwanskou National Tsing Hua University v Hsinchu plánujeme tieto aktivity overovať v oboch krajinách, porovnať výsledky z pohľadu rôzneho kultúrneho zázemia a prispôsobiť ich obom kurikulám tak, aby mohli byť súčasťou vyučovania. V pilotnej štúdii uskutočnenej v tomto projekte sme testovali úvodnú aktivitu s malou vzorkou detí bez predchádzajúcich skúseností s prácou s robotickými stavebnicami [14]. Na základe získaných výsledkov sme aktivitu upravili, aby bola bez problémov použiteľná v triede.

Na pilotnú štúdiu, ktorá bola realizovaná mimo vyučovania nadväzuje náš ďalší výskum, kde vyvíjame aktivity s robotickou stavebnicou LEGO WeDo 2.0 a testujeme ich už priamo na vyučovaní. Na tvorbu aktivít sme zvolili stratégiu výskumu vývojom. Aktuálne prebehlo testovanie upravenej úvodnej aktivity na slovenskej strane. V nasledujúcich týždňoch budeme realizovať vyučovanie so všetkými navrhnutými aktivitami ako na Slovensku tak i na Taiwane.

2. ÚVODNÁ AKTIVITA

Naším cieľom bolo realizovať aktivity primárne konštrukcionistickým spôsobom. Avšak na základe výsledkov pilotnej štúdie [14] sme sa rozhodli už do prvej aktivity zaradiť aj inštruktívne prvky výučby. Úvodná aktivita o pojme robot trvala jednu vyučovaciu hodinu (45 minút).

Cieľom tejto aktivity bolo v priebehu rozhovoru istým spôsobom usporiadať doterajšie neformálne poznatky žiakov o robotoch. Zistenia z podobných aktivít realizovaných s predchádzajúcou verziou zvolenej stavebnice - LEGO WeDo sme publikovali v jednom z našich predchádzajúcich článkov [12]. Na konci aktivity by mali žiaci vedieť, že hlavnou úlohou robota je pomáhať ľuďom. Roboti potrebujú pre svoje fungovanie zdroj energie, skladajú sa z rozličných častí a súčiastok, nemôžu sami myslieť a môžu robiť iba to, čo im naprogramuje človek. Ďalším cieľom tejto aktivity bolo oboznámiť žiakov s prácou so stavebnicou Lego WeDo 2.0. Žiaci mali skúmať ikony príkazov na ovládanie motora, pomocou ktorých mohli nastavovať napr. smer, rýchlosť a čas otáčania motora. Cieľom bolo aj spoločným konštruovaním a programovaním modelu prispieť k rozvoju nielen jemnej motoriky, ale i tímovej spolupráce a komunikačných zručností.

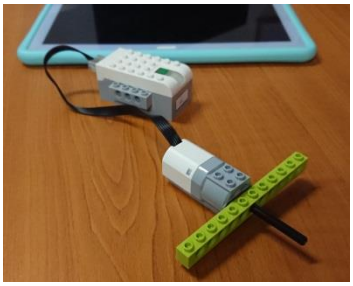
Úvodnú aktivitu sme rozdelili do štyroch častí. V prvej časti si učiteľka sadla so žiakmi do kruhu na koberec a spravila s nimi rozhovor za účelom zistenia ich predchádzajúcich skúsenosti a predstáv ohľadom robotov, viď. Obr. 1 vľavo.

Obrázok 1. Rozhovor o pojme robot (vľavo) a ukážka tvorby prvého programu učiteľkou (vpravo)

Učiteľka položila žiakom nasledujúce otázky:

1. Poznáte slovo robot? Poznáte nejakých robotov?
2. Stretli ste sa už s robotom? S akým? Kde?
3. Z čoho sa roboti skladajú? Čo musia mať, aby sa mohli hýbať?
4. Na čo slúžia? Aké je ich hlavné poslanie?
5. Môžu myslieť?
6. Môžu byť dobrí alebo zlí?

Ďalej nasledovala druhá časť, ktorá obsahovala inštruktívne prvky výučby. Učiteľka žiakom otvorila jednu škatuľu obsahujúcu súčiastky robotickej stavebnice a ukázala im niektoré elektronické a elektro-mechanické súčiastky ako hub a motor. Potom žiakom ukázala jeden zo spôsobov pripojenia motora na hub. Taktiež žiakom predviedla pripojenie dvoch LEGO súčiastok na motor, viď. Obr. 2.



Obrázok 2. Ukážka súčiastok pripojených k motoru

Učiteľka žiakom ukázala, kde v tablete nájdu aplikáciu na ovládanie danej stavebnice a ako ju spustia. Následne žiakom demonštrovala spárovanie stavebnice s uvedenou aplikáciou pomocou Bluetooth, a tiež vytvorenie a spustenie prvého programu na ovládanie robotického modelu, viď. Obr. 1 vpravo a Obr. 3.

Obrázok 3. Ukážkový program tvorený učiteľkou

Následne v tretej časti učiteľka žiakov požiadala, aby sa rozdelili do dvojíc a postavili svoj vlastný model robota, ktorý na konci vyučovacej hodiny predstavia spolužiakom. V úvode tejto časti učiteľka každej dvojici rozdala stavebnicu a tablet a každej dvojici vysvetlila spárovanie stavebnice s aplikáciou v tablete pomocou Bluetooth.

Vo štvrtej časti aktivity si žiaci aj s učiteľkou sadli v triede do kruhu na koberec a prezentovali, akých robotov postavili.

Výsledky analýzy dát z tretej a štvrtej časti aktivity popisujeme v nasledujúcej kapitole.

3. METODIKA VÝSKUMU

V tomto príspevku vychádzame z výsledkov testovania úvodnej aktivity. Jeho cieľom je popísať rozdiely pri tvorbe prvých vlastných robotických modelov u žiakov tretieho a štvrtého ročníka, ktorí nemali predchádzajúce skúsenosti s programovaním a s konštruovaním robotických modelov.

Vo výskume sme si položili nasledujúcu výskumnú otázku:

Aké sú rozdiely u žiakov tretieho a štvrtého ročníka pri tvorbe svojich prvých robotických modelov konštruovaných podľa vlastnej fantázie pomocou stavebnice LEGO WeDo 2.0?

Na zber a analýzu dát sme použili rozličné kvalitatívne metódy [15], medzi ktoré patrí zúčastnené neštruktúrované pozorovanie a

ISBN 978-80-557-1533-9, ISSN 2454-051X.

metóda terénnych zápiskov, pološtruktúrovaný rozhovor, vytváranie a analýza videozáznamov postupu práce každej dvojice žiakov i celej triedy a vytváranie fotografií postupu práce každej dvojice žiakov.

Testovanie aktivít sme začali realizovať v rámci bežnej výučby povinného predmetu informatika na súkromnej základnej škole v Bratislave s jednou triedou tretieho ročníka a jednou triedou štvrtého ročníka (išlo o žiakov vo veku osem až desať rokov). V triede sa okrem žiakov a ich učiteľky nachádzali tri výskumníčky, ktoré zbierali dáta. Tretiakov učila jedna z výskumníčok, pričom tretiacka učiteľka dohliadala na priebeh výučby a komunikovala so žiakmi. Štvrtákov učila ich bežná učiteľka informatiky a tri výskumníčky zbierali dáta. Učiteľka a výskumníčka učili podľa pokynov v metodických materiáloch, ktoré sme učiteľke poskytli v podobe webovej stránky. Túto webovú stránku vytvárala jedna z výskumníčok, ktorá viedla aktivitu u tretiakov. V treťom ročníku sa nachádzalo šesť žiakov (jedno dievča a päť chlapcov) a vo štvrtom ročníku bolo sedem žiakov (jedno dievča a šesť chlapcov). Žiaci v priebehu aktivity pracovali v dvojiciach, avšak vo štvrtom ročníku jedno dievča pracovalo samé. Toto dievča už malo predchádzajúce skúsenosti s prácou s danou stavebnicou z predošlej základnej školy, preto sa jej práci so stavebnicou nebudeme ďalej venovať. Výsledky analýzy dát zozbieraných počas tejto aktivity popisujeme v nasledujúcej kapitole.

4. POROVNANIE TVORBY PRVÉHO ROBOTICKÉHO MODELU

Na základe analýzy dát sme tvorbu prvého vlastného robotického modelu konštruovaného podľa fantázie žiakov rozdelili na tri časti: stavba, programovanie a prezentovanie robotického modelu. V nasledujúcich častiach popisujeme a porovnávame jednotlivé činnosti dvojíc tretiakov a štvrtákov.

4.1 Stavba podľa vlastnej fantázie

Prvá myšlienka, ktorá všetkým dvojiciam žiakov napadla na stavbu svojho prvého robotického modelu bolo auto. V treťom ročníku dve dvojice z troch postavili určitý typ auta (auto s vrtuľou a auto s vrtákom - vŕtač), viď. Obr. 4 vľavo v strede a vľavo dolu. Vo štvrtom ročníku jedna dvojica postavila model horského auta s laserom, viď. Obr. 4 vpravo v strede.

4.1.1 Stavanie a rozoberanie modelov

Všetky dvojice tretiakov a jedna dvojica štvrtákov sa snažili pripevniť na model kolesá tak, aby sa model hýbal. Avšak ani jednej dvojici sa do konca vyučovacej hodiny nepodarilo pripevniť kolesá na motor tak, aby sa auto hýbalo.

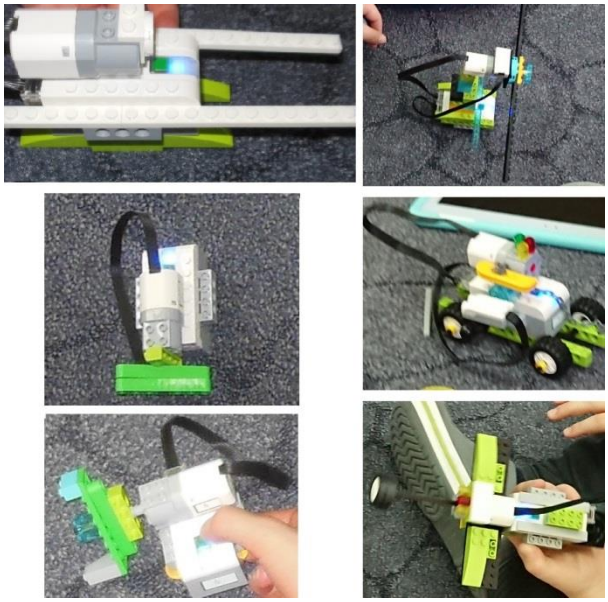
Väčšina dvojíc tretiakov úplne rozobrala a prestavala svoje modely trikrát. Finálne modely postavili do 11-tich minút. Pričom pri tvorbe tretieho modelu väčšina tretiakov postupovala nasledovne:

- najskôr konštruovali model (konštruovanie finálneho modelu im trvalo maximálne tri minúty),
- potom testovali správanie modelu pomocou vytvoreného programu (testovanie prebehlo do piatich minút),
- na záver ešte dostavovali konštrukciu modelu (dostavovanie trvalo maximálne tri minúty).

Väčšina dvojíc štvrtákov rozobrala v priebehu stavania svoje modely jedenkrát, pričom tieto modely nerozoberali úplne celé (okrem dvojice, ktorá konštruovala auto). Nechali si postavenú nejakú časť modelu (napr. vrtuľu). Potom do 5-tich minút (pred začatím prezentovania) postavili finálny model. Dvojica, ktorá stavala mixér, svoj model nerozoberala a postavila ho v priebehu 20-tich minút.

4.1.2 Zložitosť konštrukcie modelu

Konštrukcie robotických modelov štvrtákov boli o niečo zložitejšie ako konštrukcie modelov tretiakov, viď. Obr. 4 Táto zložitosť však môže byť zapríčinená hlavne tým, že štvrtáci sa stavbe jedného modelu venovali dlhší čas ako tretiaci.



Obrázok 4. Robotické modely tretiakov (vľavo) a štvrtákov (vpravo)

4.1.3 Spolupráca pri stavbe

Pri stavbe modelov u tretiakov (okrem zmiešanej dvojice - chlapec a dievča) buď staval hlavne jeden z dvojice, alebo stavali zo začiatku obidvaja, pričom neskôr už len jeden člen dvojice. U zmiešanej dvojice to vyzeralo tak, že obidvaja sa snažili niečo postaviť. Avšak nepracovali na spoločnom modeli. Keď dievča niečo postavilo, tak to chlapec rozobral.

Pri stavbe modelov u štvrtákov v jednej dvojici stavali obidvaja členovia a v dvoch dvojiciach staval prevažne jeden člen.

Žiaci oboch ročníkov prevažne nemali nacvičenú spoluprácu v dvojiciach. Model poväčšine staval jeden dominantnejší člen dvojice.

4.2 Programovanie

Všetky dvojice štvrtákov sa sústredili prevažne na stavbu modelov, ich programovaniu sa venovali len zlomok času v porovnaní so stavbou modelu. U štvrtákov neprebiehalo jednoznačne oddelené testovanie správania modelu od konštruovania stavby modelu ako u tretiakov.

Všetky dvojice tretiakov skúmali viaceré ikony príkazov a spájali ich do programu (išlo hlavne o ikony príkazov na ovládanie motora a ikonu príkazu na zmenu farby svetla na hube), viď. Obr.

ISBN 978-80-557-1533-9, ISSN 2454-051X.

5. Dve dvojice tretiakov vytvorili programy s viacerými ikonami príkazov, pričom využívali prevažne ikony príkazov na nastavenie smeru otáčania motora a rýchlosti otáčania motora, viď. Obr. 5.

Obrázok 5. Skúmanie viacerých ikon príkazov v programe a finálne programy (tretiaci)

Jedna dvojica tretiakov program vytvoriť nestihla. Bola to zmiešaná dvojica a chlapec v priebehu celej aktivity nechcel s dievčaťom spolupracovať.

Väčšina štvrtákov pri tvorbe svojich programov využila jednu ikonu príkazu, pričom skúmaniu ďalších ikon príkazov nevenovali pozornosť. Len u jednej skupiny sme spozorovali testovanie programu na ovládanie modelu, pričom táto skupina postavila model auta a zisťovala či sa auto hýbe (nehýbalo sa). V priebehu aktivity sme pozorovali, že tretiaci dlhšie skúmali ikony príkazov v programe, viac testovali správanie svojho modelu ako štvrtáci.

Väčšina dvojíc štvrtákov skúšala pridávať do svojich programov viaceré ikony príkazov až tesne pred prezentovaním alebo v priebehu prezentovania svojich modelov, viď. Obr. 6. Všetky dvojice štvrtákov pri prezentovaní mali programy obsahujúce dve ikony príkazov - otáčanie motora vpravo alebo vľavo a nastavenie rýchlosti otáčania motora. Tieto programy si vytvorili tesne pred prezentovaním.

Obrázok 6. Úprava programu počas prezentovania inej dvojice (štvrtáci)

4.3 Prezentovanie vytvoreného modelu

V priebehu prezentovania modelov všetky dvojice žiakov najskôr rozprávali o tom, čo plánovali postaviť, ale nepodarilo sa im to zrealizovať. Žiaci rozprávali napr. o tom, že plánovali: *postaviť auto či vrtuľník (obrázok vrtuľníka bol na krabici od stavebnice), pripevniť k autu kolesá, ale potom pripevnili lyže, zabezpečiť, aby sa kolesá auta točili, ale točí sa mi len vrtuľa a pod.*

Dve dvojice štvrtákov popisovali konštrukciu svojich modelov tak, že niektorým LEGO kockám pripísali vymyslené vlastnosti ako napr. na mixéri sa nachádzali senzory hustoty, aby mixér vedel ako silno mixovať a na horskom aute sa nachádzal laser.

Ani jedna dvojica tretiakov či štvrtákov nepopisovala svoj program na ovládanie modelu. Pri prezentovaní program bez komentára spustili. Z toho vyplýva, že nevieme jednoznačne posúdiť, do akej miery žiaci vytvoreným programom rozumeli a či len náhodne nespojili niekoľko ikon príkazov do programu.

5. DISKUSIA

Výsledky analýzy dát odhalili odlišnosti v práci tretiakov a štvrtákov. Tieto odlišnosti však mohli vzniknúť i rozdielnym prístupom učiteľov (výskumníčka a „bežná“ učiteľka informatiky). V priebehu analýzy dát sme sa pokúsili o porovnanie robotických modelov skonštruovaných pomocou stavebnice LEGO WeDo 1.0 a LEGO WeDo 2.0 u žiakov rovnakej vekovej kategórie. Avšak v týchto stavebniciach sa nachádza niekoľko odlišných LEGO kociek, a taktiež je odlišný aj hub. Novšia a staršia verzia LEGO WeDo stavebnice sa odlišuje aj tým, že staršia verzia musí byť pripojená k počítaču pomocou USB kábla, aby sme mohli model naprogramovať. Zatiaľ, čo novšia verzia už obsahuje bluetooth a model môžeme programovať s použitím tabletu. Pri porovnávaní programov vytvorených v týchto dvoch verziách stavebnice sme zistili, že keď žiaci vytvárali program na počítači, skúmali viacero ikon príkazov a niektorí využili napr. aj ikonu cyklu. Avšak pri tvorbe programu na tablete žiaci vytvárali kratšie a jednoduchšie programy zložené prevažne z ikon príkazov na ovládanie motora.

6. ZÁVER

V príspevku sa venujeme úvodnej aktivite na prácu s robotickou stavebnicou LEGO WeDo 2.0, ktorá je navrhnutá pre žiakov tretieho a štvrtého ročníka základnej školy. Rozoberáme prácu žiakov pri tvorbe ich prvého vlastného robotického modelu postaveného podľa ich fantázie, pričom sa zameriavame na hlavné rozdiely práce medzi jednotlivými ročníkmi. Na základe kvalitatívnej analýzy dát sme zistili niekoľko odlišností z hľadiska stavby, programovania i prezentovania robotického modelu.

Tretiaci postavili finálny model na tretí pokus, pričom svoje predchádzajúce modely kompletne rozobrali. Takmer všetky dvojice štvrtákov postavili finálny model na druhý pokus, avšak model nerozoberali úplne celý.

Tretiaci skúmali viaceré ikony príkazov, zostavovali programy s viacerými ikonami príkazov a pri tvorbe modelu testovali jeho správanie pri ovládaní vytvoreným programom. Štvrtáci sa viac sústredili na stavbu modelu a ikony príkazov veľmi neskúmali. Programy s viacerými ikonami príkazov (viac ako dve) tvorili až v priebehu prezentovania.

Pri prezentovaní štvrtáci pripisovali dizajnovým LEGO kockám vymyslené funkcie (senzory hustoty a laser). Tretiaci svojím

ISBN 978-80-557-1533-9, ISSN 2454-051X.

modelom síce pripisovali vymyslené správanie, ale nepripisovali vymyslené funkcie dizajnovým LEGO kockám.

Zistenia analýzy dát nám pomohli v úprave úvodnej aktivity, do ktorej sme sa rozhodli zaradiť stavbu robotického modelu podľa návodu a jeho programovanie podľa vlastnej fantázie. Taktiež prispeli k úprave obsahu a poradia ďalších vyvíjaných aktivít so zvolenou stavebnicou.

Výsledky analýzy dát z úvodnej, ale i ďalších aktivít ďalej plánujeme porovnávať s dátami zozbieranými v spolupráci s taiwanskými výskumníkmi. Pričom ich plánujeme porovnať i z pohľadu rôzneho kultúrneho zázemia a prispôsobiť ich obom národným kurikulám tak, aby mohli byť súčasťou bežného primárneho vzdelávania.

POĎAKOVANIE

Tento výskum bol podporený z projektu VEGA 1/0797/18 ako aj z bilaterálneho APVV projektu SK-TW-2017-0006

BIBLIOGRAFICKÉ ODKAZY

[1] Piaget, J. The principles of genetic epistemology. New York: Basic Books, 1972.

[2] VON GLASERSFELD, Ernst. An Introduction to Constructivism. In: WATZLAWICK, Paul. The Invented reality: how do we know what we believe we know? : contributions to constructivism. New York: Norton, 1984, pp. 17-40.

[3] Papert, S. The Children's Machine. New York: Basic Books, (1993).

[4] Palincsar, A.S. Social constructivist perspectives on teaching and learning. Annual Review of Psychology, 49, pp. 345–375, (1998).

[5] Alimisis, D., Moro, M., Arlegui, J., Pina, A., Frangou, S. and Papanikolaou, K., 2007, August. Robotics & constructivism in education: The TERECoP project. In EuroLogo (Vol. 40, pp. 19-24).

[6] Eguchi, A., 2014, July. Robotics as a learning tool for educational transformation. In Proceeding of 4th International Workshop Teaching Robotics, Teaching with Robotics & 5th International Conference Robotics in Education Padova (Italy)

[7] Mubin, O., Stevens, C.J., Shahid, S., Al Mahmud, A. and Dong, J.J., 2013. A review of the applicability of robots in education. Journal of Technology in Education and Learning, 1(209-0015), p.13.

[8] J.H. Han, D.H. Kim, and J.W. Kim, Physical learning activities with a teaching assistant robot in elementary school music class, Journal of Convergence Information Technology, 5(5), 2009, 1406–1410.

[9] C.W. Chang, J.H. Lee, P.Y. Chao, C.Y. Wang, et al., Exploring the possibility of using humanoid robots as instructional tools for teaching a second language in primary school, Educational Technology and Society, 13(2), 2010, 13–24.

[10] Veselovská, M., Mayerová, K.: Assessing robotics learning at lower secondary school. Information and Communication Technology in Education 2015, 240–247. Pedagogical Faculty, University of Ostrava, Ostrava (2015).

[11] Mayerové, K., Veselovská. M.: How to teach with LEGO WeDo at primary school. In: Robotics in Education, pp. 55–62. Springer International Publishing (2017).

[12] Mayerová, K., Veselovská, M.: How We Did introductory lessons about robot. In: Teaching Robotics, Teaching with Robotics, pp. 127–134. Padova (2014).

[13] Veselovská, M., Mayerová, K.: Pilot study: educational robotics at lower secondary school. In: Constructionism and Creativity Conference. Vienna (2014).

[14] Veselovská, M., Mayerová, K., Kubincová, Z., Chiu, F.Y. 2018. A pilot study: Comparing the work of children with LEGO WeDo 2.0 in Slovakia and Taiwan, In International Conference of Education, Research and Innovation (ICERI), (pp. 6030-6037), IATED.

[15] Creswell, J. W. Educational research: Planning, conducting, and evaluating quantitative. Upper Saddle River, New Jersey (2002).

Vývoj vysokoškolského kurzu edukačnej robotiky
Development of University Course about Educational Robotics

Karolína Mayerová
KDMFI FMFI UK
Mlynská dolina 1
84248 Bratislava
SR
mayerova@fmph.uniba.sk

Michaela Veselovská
KDMFI FMFI UK
Mlynská dolina 1
84248 Bratislava
SR
veselovska@fmph.uniba.sk

ABSTRACT
In the article, we describe the research that we have begun to implement in the past school year (2017/18) in which we develop content of a high-school course on educational robotics for future teachers. Based on the results of last year's research, we reduced number of used robotics kits to only one. Together with students we decided to use Ozobot Evo in the course of the semester. We have used project based learning method to help students develop the skills they need for a successful life in the 21st century. Students in our course created activities for pupils workshop in leisure time center, where were these activities tested during the semester. So students received immediate feedback, for example: if their time estimation is realistic, if demands are adequate and if content is attractive for pupils. We utilized qualitative data collection and processing for the nature and needs of our research.

Keywords
Robotics Course. After School Activities. Ozobot Evo. Educational Robotics. Activities.

ABSTRAKT
V článku opisujeme výskum, ktorý sme začali realizovať už minulý školský rok (2017/18), v ktorom vyvíjame obsah vysokoškolského kurzu o edukačnej robotike pre budúcich učiteľov. Na základe výsledkov výskumu z minulého roku sme použili vo výučbe jeden druh robota, a to Ozobot Evo. Využili sme projektové vyučovanie, vďaka ktorému si študenti rozvíjali viaceré zručnosti potrebné pre úspešný život v 21. storočí. Študenti na kurze vytvárali aktivity pre žiakov voľnočasového krúžku, ktoré boli počas semestra priamo testované. Teda dostávali okamžitú spätnú väzbu, či je napr. ich časový odhad reálny, náročnosť primeraná a obsah pútavý. Pre charakter a potreby výskumu sme využívali kvalitatívne zbieranie a spracovanie dát.

Kľúčové slová
Vysokoškolský kurz robotiky. Voľnočasový krúžok. Ozobot Evo. Edukačná robotika. Aktivity.

1 ÚVOD
Už niekoľko rokov (od roku 2013) vyučujeme na Univerzite Komenského predmet, na ktorom vzdelávame budúcich učiteľov v oblasti edukačnej robotiky. Ide o predmet Robotické stavebnice vo vzdelávaní (ďalej len RSVV), ktorý sa vyučuje v letnom aj v zimnom semestri (RSVV1 a RSVV2) a má za cieľ študentom sprostredkovať skúsenosť s edukačnými robotickými stavebnicami, pričom nadväzujeme na niekoľko ročnú prácu kolegýň Pekárovej a Kabátovej [1, 2]. Výučba je zameraná hlavne na praktické poznatky, ktoré študenti nemôžu získať z prednášok. Na tomto predmete majú možnosť sa stretnúť priamo s výhodami a nevýhodami robotických stavebníc, zažiť priamy kontakt s reálnymi problémami, ktoré technika prináša v kontexte vyučovania, či zlepšiť si tímovú spoluprácu a organizáciu svojich činností. V rámci nášho výskumu máme za cieľ zlepšovať návrh jeho obsahu a formy. Pred dvoma rokmi t.j. v zimnom semestri akademického roku (2017/2018) sme aplikovali do výučby prvky autentického učenia [3]. V tom istom roku, sme začali spoluprácu so študentkou (ďalej len Janka[1]), ktorá vedie robotický krúžok v centre voľného času. Janka sa podujala na tomto krúžku overovať aktivity, ktoré študenti vytvárali v rámci RSVV2. V tejto spolupráci sme pokračovali aj tento školský rok (2018/19). Na základe výsledkov z nášho minuloročného výskumu [4], sme upravili niektoré pravidlá a štruktúru predmetu RSSV2. Výsledky zmien uvádzame v nasledujúcich kapitolách.

2 KURZ RSVV2 A APLIKOVANÉ ZMENY
Predmet RSSV2 je pokračovaním predmetu RSVV1, ktorý patrí medzi povinne voliteľné predmety v študijnom pláne odboru učiteľstvo akademických predmetov v kombinácií s informatikou. Predmet RSVV1 študenti absolvujú v bakalárskom štúdiu. To znamená, že jeden semester už pracovali s robotickou stavebnicou a sú oboznámení so základnými charakteristikami edukačnej robotiky. RSVV2 je voliteľným predmetom v zimnom semestri magisterského štúdia pre študentov z odboru učiteľstvo akademických predmetov v kombinácií s predmetom informatika. Viac informácií o tomto predmete je popísaných v predošlom článku [4]. Štruktúru a prevedenie obsahu každý rok upravujeme podľa potrieb vyplývajúcich z počtu a charakteristiky študentov.

2.1 Projektové vyučovanie
Predminulý rok sme zaradili do výučby prvky autentického učenia sa. Tento rok sme sa rozhodli pre zaradenie projektového vyučovania, ktoré tiež pokrýva a rozvíja ciele kurzu. Dôvody, prečo sme si ho zvolili, spolu s jeho charakteristikami uvádzame v tejto kapitole. Projektové vyučovanie umožňuje prehlbovať a rozširovať kvalitu učenia sa a vyučovania [5,6]. Kladie dôraz na praktické skúsenosti, čo je pre náš predmet priam nevyhnutné. Študenti dokážu lepšie pochopiť praktický význam učiva. Ďalšou charakteristikou je, že projekty nebývajú krátkodobé, ale trvajú týždeň až mesiace - v našom prípade celý semester. Výhodou projektového vyučovania je aj skutočnosť, že projekt je a má byť sám o sebe pre študentov motivačný. Ide o princíp vzdelávania,

[1] Z dôvodu ochrany osobných údajov článku uvádzame zmenené meno.

kedy sa má obsah zabaliť do krásneho a zaujímavého obalu, najčastejšie do hry. Tento princíp presadzoval aj J. A. Komenský [7]. U študentov takýto typ práce s učivom rozvíja iniciatívu, samostatnosť, tvorivosť, kritické myslenie, schopnosť spolupráce, riešenia problémov a zručnosti vyhľadávať informácie.

2.2 Ozobot Evo

Minulý rok študenti na kurze pracovali každý týždeň s inou robotickou stavebnicou. Skúmali stavebnice ako napr. BeeBot, LEGO WeDo, LEGO WeDo 2.0 a LEGO Mindstorms NXT, ktoré sme mali na fakulte. Ďalej pracovali aj so stavebnicami, ktoré študentka Janka nosila na predmet RSVV2: AlbiRobot, CodeaPilar, Ozobot Evo a Sphero. Priebeh a výsledky práce študentov s uvedenými stavebnicami sme analyzovali a výsledky analýzy ukázali, že by bolo efektívnejšie pracovať s menším počtom stavebníc. Preto sme sa rozhodli, že tento rok budeme pracovať len s jedným, alebo maximálne s dvoma druhmi robotov. Tento semester sme im dali na výber z robotov Ozobot Evo, LEGO WeDo 2.0 a Sphero mini. V úvode semesta mali možnosť preskúmať ich potenciál a pohodlnosť práce s nimi. Vybrali si robota Ozobot Evo (Obrázok1), ktorý ponúkal viacero druhov aktivít a pomerne funkčné prostredie na programovanie.

Obrázok 1. Ozobot Evo[2] v dvoch farebných prevedeniach

2.3 Priebeh predmetu

Semester trval 13 týždňov, z čoho výučba predmetu prebehla v 11tich. Štruktúra nebola tento semester presne naplánovaná tak ako minulý rok. Vzhľadom na charakter projektového vyučovania študenti dostali v úvode semestra zadanie, ktoré mali na konci semestra odovzdať, pričom museli splniť určité požiadavky. Počas semestra sme sa stretávali pravidelne každý týždeň na 90 minút. Hodnotenie získali za aktívnu účasť na hodinách a odovzdaní nimi vytvorených aktivít. Výučby sa zúčastňovala aj študentka Janka.

Zadanie, ktoré študenti dostali znelo: *Na konci semestra odovzdajte overenú sériu na seba nadväzujúcich aktivít pre minimálne 4 vyučovacie hodiny. Overenia budú prebiehať na voľnočasovom krúžku študentky Janky, ideálne každý týždeň. Máte možnosť pracovať vo dvojiciach alebo všetci spolu. Na výber máte stavebnice LEGO WeDo 2.0, Ozobot Evo alebo SpheroMini. Rozdeľte si úlohy a vyberte si stavebnicu*.

3 METODOLÓGIA

Cieľom nášho výskumu bolo sledovať priebeh práce študentov a ich reakcie vzhľadom na vytvorené podmienky, ktoré poskytoval náš predmet. Chceli sme zistiť, aký bude priebeh rozdeľovania úloh, spolupráce, vývoj obsahu a formy vytvorených aktivít. Ďalej sme skúmali, aké dôsledky prinesie zavedenie len jedného druhu stavebnice či robota do výučby nášho predmetu a ako budú reagovať študenti na projektové vyučovanie. Rozhodli sme sa preto pre kvalitatívny výskum [8], ktorého nástroje nám umožňujú sledovať nami stanovené ciele. Zber dát sa preto zameriaval na archiváciu a analýzu kvalitatívnych dát [9] a to konkrétne: vytvorené materiály s aktivitami od študentov, spätnú väzbu z overovacích hodín od študentky Janky, záznamy a fotografie z vyučovacích hodín RSVV2 a nahrávku z diskusie študentov na poslednej hodine RSVV2.

Položili sme si dve výskumné otázky:

1. Čo prináša zavedenie projektového vyučovania do predmetu RSVV2?
2. Čím je charakteristická práca s robotom Ozobot Evo?

Výskumu na predmete RSVV2 sa zúčastnili štyria študenti (z toho jeden muž a tri ženy), pričom všetci okrem jednej z nich boli študentmi odboru učiteľstva akademických predmetov. Súčasťou výskumu bola i študentka Janka, ktorá overovala študentmi tvorené materiály na dvoch krúžkoch v centre voľného času. Vyučujúca predmetu zároveň plnila funkciu pozorovateľa a výskumníka. Na prvom krúžku bolo deväť detí, pričom tri z druhého ročníka, tri boli zo štvrtého ročníka, jedno z piateho ročníka, jedno zo šiesteho ročníka a jedno z ôsmeho ročníka. Išlo o žiakov, z ktorých sa niektorí zúčastnili výskumu už predchádzajúci rok, a teda už mali predchádzajúce skúsenosti s prácou s robotickými stavebnicami. V druhej skupine bolo celkovo menej detí a takmer všetky chodili na krúžok prvý krát. Išlo o šesť žiakov z nasledovných ročníkov: dvaja z tretieho ročníka, dvaja zo štvrtého ročníka, jeden zo šiesteho ročníka a jeden z ôsmeho ročníka.

4 Aktivity navrhnuté študentmi

Okrem samotného procesu, na ktorý sme sa počas semestra zameriavali, tak študenti samozrejme vytvorili na konci semestra aj aktivity pre tri 90-minútové krúžku robotiky s robotom Ozobot Evo. V tejto kapitole predstavujeme len stručný popis, ktorý pomôže čitateľovi vytvoriť si predstavu o povahe úloh. Keďže samotný výsledok nebol cieľom, neuvádzame aktivity v plnom rozsahu.

Študenti si zdieľali aktivity pomocou cloud úložiska Google Drive. Pôvodne vytvorili aktivity na štyri krúžky, ale po otestovaní aktivít v praxi a nepresný odhad, prvé dve hodiny spojili do jednej.

1. V prvej aktivite je séria úloh zameraná na oboznámenie sa s Ozobotom. V úlohách žiaci využívajú pohyb robota po čiarach na rôznych, vopred pripravených mapách. Ďalej dopĺňanie vytlačených farebných kódov, či ich dokresľovanie. Na záver hodiny sú zaradené úlohy zamerané na ovládanie Ozobota cez tablet.
2. V tejto druhej aktivite sú úlohy už s využitím ikonického programovacieho prostredia pre Ozobot - Ozoblockly. Žiaci začínajú s nahraním jednoduchého programu, oboznámením sa s príkazmi (Obrázok 2) a postupom nahrania vlastného programu. Ďalej úlohy pozvoľna gradujú, pričom sa využíva iba základná množina príkazov- t.j. s úrovňou 2.
3. Keďže ide o aktivity pre krúžok, na ktorom sú aj starší žiaci, do tejto série úloh zaradili študenti aj ťažšie úlohy, s názvom 8-uholník, bludisko, valčík a farebné bludisko. V úlohách sa využíva aj cyklus a podmienka.

[2] Zdroj: ozobot.com

Obrázok 2. Žiak pri práci s robotom Ozobot Evo

5 VÝSLEDKY

Výsledky, ktoré sme získali na základe analýzy dát, sme rozdelili do troch oblastí, ktoré opisujeme v nasledujúcich podkapitolách.

5.1 Všeobecné zistenia z pohľadu výskumníka

Z dlhodobého skúmania sme zistili, že študenti po absolvovaní našich dvoch kurzoch RSVV1 a RSVV2 väčšinou nechcú učiť v bežnom vyučovaní edukačnú robotiku pre jej časovú, technickú a finančnú náročnosť.

Na začiatku semestra si študenti ťažko organizovali svoju prácu a spoluprácu. Vykazovali nízku efektívnosť a trvalo im dosť dlho, kým začali aktívne samoštúdium. V priebehu prvých troch vyučovacích hodín ani jeden študent iniciatívne neprebral úlohu koordinátora. A to aj napriek tomu, že boli k tomu implicitne vyzývaní. Zmenu sme identifikovali na konci semestra, kedy na poslednej hodine, kde ešte mali pracovať na záverečnej úlohe, po oznámení zadania vedeli takmer okamžite začať pracovať na zadaní.

Minulý rok študenti potrebovali implicitne naviesť k tomu, aby do svojich materiálov začali zaraďovať odporúčania a riešenia pre učiteľa, či odhadovaný čas aktivít a pomôcky. Tento rok k tomu dospeli študenti sami, bez pomoci vyučujúceho.

Práca s jednou robotickou stavebnicou či robotom, resp. s jedným druhom, sa nám ukazuje ako vhodná. Študenti sa tak môžu dobre oboznámiť s robotickou stavebnicou / robotom a vedia vymyslieť lepšie aktivity, ošetriť nevhodné správanie a napísať do materiálov vhodné rady. Vzhľadom na rozsah semestra a dotáciu 90 min týždenne, nevidíme priestor pre hlbšie porozumenie viacerým stavebniciam. Študenti síce prejavili v závere semestra záujem pracovať s viacerými stavebnicami, pre obmedzený čas to však nepokladáme za vhodné.

Ukázalo sa, že náročnosť výučby edukačnej robotiky pre vyučujúceho v kontexte projektovej výučby je naozaj vysoká. Ide hlavne o technickú stránku. Treba zaobstarať funkčných robotov, nabité tablety a batérie, nástroje na tvorbu didaktických pomôcok (spárované grafické tablety, počítače, tlačiareň, rôzne druhy papierov, nožníc, fixiek, ...).

5.2 Závery študentov na konci semestra

Študenti sami vnímali ako veľmi náročné správne odhadnúť dĺžku trvania aktivity. Bolo pre nich ťažké vytvoriť aktivity s triviálnou náročnosťou, ktoré by sledovali zmysluplné vzdelávacie ciele a neslúžili len na zábavu. Na základe spätnej väzby od Janky si začali uvedomovať potrebu diferenciácie vzhľadom na vek a pohlavie žiakov. Postupom času začali vnímať limity robotov, resp. ich technické možnosti.

5.3 Práca s robotom Ozobot Evo

5.3.1 Výhody:

Ozobot Evo sa ukázal ako pomerne presný robot v porovnaní so stavebnicami, s ktorými sme doposiaľ pracovali (pozri kap. 2.2.). Ide o jednoduchého robota bez potreby konštruovania modelu, pričom ponúka široké možnosti programovania a to pomocou dvoch rôznych prístupov:

- postupnosti farebných kódov
- pomocou programovacieho prostredia Ozoblokly na tablete. Toto prostredie ponúka rôzne úrovne programovania s narastajúcim počtom príkazov. Prostredie je vizuálne podobné Scratchu - viacerí žiaci sa v ňom pomerne rýchlo zorientovali.

Robot je svojou náročnosťou vhodný už pre druhákov na ZŠ, ale rovnako vie pokryť niektoré časti SŠ učiva uvádzanom v ŠVP, samozrejme v primeranom rozsahu a forme. Podporuje široké spektrum aktivít. Existuje preň veľa rôznych zadaní [10, 11, 12, 13], ktoré podporujú rôzne zručnosti u žiakov, rovnako ako kreativitu u učiteľov [14]. Vo väčšine prípadov ide o blogy a práce učiteľov z praxe. Existuje veľa materiálov aj pre staršiu verziu Ozobot Bit [15, 16].

5.3.2 Nevýhody:

V druhej polovici semestra nás prekvapila slabá výdrž batérií. Pri vytváraní programu cez tablet a jeho následnom posielaní do robota, batéria vydrží necelých 60 minút. Jeden z dôvodov môže byť, že nefungovalo načítanie programu cez blikanie[3] a bolo treba zdĺhavé posielanie kódu cez aplikáciu. Vtedy však treba prepínať medzi prehliadačom a aplikáciou v tablete, pričom Ozobot musí byť stále zapnutý, aby ostal pripojený. Ak sa vypne, treba ho znovu spárovať a celý proces začne odznovu. To má za následok, že nemôžeme pracovať napríklad s dvoma skupinami/triedami za sebou. Pri práci s aplikáciou, svetlami či zvukom vytvárame dlhý kód a je pre žiakov potrebné častejšie si ho nahrať do robota a odskúšať, čo tiež znižuje výdrž už aj tak slabej batérie. Pri využití farebných kódov, vie Ozobot zostať dlhšie nabitý.

Zistili sme, že je veľmi prínosné mať všetky Ozoboty označené (napr. nálepkou s číslom), pretože inak žiaci dlho zisťujú, kto ovláda tabletom ktorého Ozobota.

Za nedostatky softvéru pokladáme aj občasné neočakávané správanie Ozobota. Niekedy sa nám nepodarilo pripojiť ho k tabletu, inokedy nefungovalo posielanie kódu z tabletu do Ozobota (vypísalo na krátko chybovú hlášku, ale nevedeli sme ju vyriešiť - niekedy sa táto chyba objavila pri slabšej batérii, inokedy zatiaľ z neznámych príčin). Občas sa nám stalo, že sa Ozobot pohyboval úplne náhodne a nie vždy sme ho vedeli "prinútiť", aby vykonal príkazy v programe. Deti mali hlboký zážitok, keď občas čudne skákal, alebo urobil zvláštny pohyb, ktorý sa nenachádzal v programe.

Je možné, že niektoré nedostatky sú odstránené v staršej generácií Ozobot Bit, avšak to nevieme overiť, lebo ju nemáme.

6 ZÁVER A DISKUSIA

Na záver by sme chceli skonštatovať, že nás najviac v celom výskume prekvapili technické problémy s Ozobotom. Napriek

[3] Jeden zo spôsobom nahrávania programu do Ozobota.

nesmiernym výhodám ako je jednoduchosť, využitie pre ZŠ aj SŠ a ďalšie, ktoré opisujeme v článku, sme museli pomerne často hľadať alternatívne riešenia pre chyby v softvéri, či chybné fungovanie funkcií ako spárovanie, načítanie programu, či rýchle vybíjanie batérií.

Na našu výskumnú otázku: *Čo prináša zavedenie projektového vyučovania do predmetu RSVV2*? by sme v zhrnutí zareagovali tak, že vidíme viaceré pozitíva, rovnako, ako pri použití autentického učenia. U niektorých študentov sme zaznamenali rozvoj schopnosti koordinovať členov skupiny. Rovnako sa zlepšili pri identifikovaní a prerozdelení čiastkových úloh. Ako najslabšie miesto projektového vyučovania na RSVV2 považujeme prácu s časom, ktoré má svoj pôvod v nedostatočnom uvedomení si zodpovednosti za dokončenie práce v časovom limite, ktorý je vzdialený niekoľko týždňov. Takmer úplne absentovala potreba odhadnúť a rozvrhnúť si prácu do časového harmonogramu.

Pri druhej výskumnej otázke: *Čím je charakteristická práca s robotom Ozobot Evo?* sme svoje zistenia spísali do kapitoly *5.3 Práca s Ozobot Evo*, kde opisujeme jeho výhody a nevýhody.

Zmenu - zaviesť jeden druh stavebnice či robota do kurzu namiesto viacerých druhov, pokladáme aj napriek nesúhlasu študentov za pozitívnu. Vzhľadom na rozsah a časovú dotáciu kurzu nie je priestor zaoberať sa do hĺbky viacerými stavebnicami. Radšej tento priestor využijeme na komplexnejšie oboznámenie študentov s problémami a výhodami edukačnej robotiky, ako na povrchné oboznamovanie sa s viacerými technickými riešeniami robotických stavebníc, ktoré sa aj tak v priebehu rokov, ba možno mesiacov zmenia.

POĎAKOVANIE

Tento výskum bol podporovaný zo Slovenského národného projektu VEGA č. 1/0797/18 rovnako ako z bilaterálneho projektu APVV č. SK-TW-2017-0006.

BIBLIOGRAFICKÉ ODKAZY

[1] KÁBÁTOVÁ, M., PEKÁROVÁ, J.: Learning how to teach robotics. In: *Constructionism 2010 conference*. Paris, 2010. ISBN 978-80-89186-66-2.

[2] KABÁTOVÁ, M. *Konštrukcionistický prístup vo vyučovaní robotiky v príprave budúcich učiteľov* [dizertačná práca]. Bratislava: Fakulta matematiky, fyziky a informatiky, Univerzita Komenského v Bratislave, 2010

[3] PASCH, M. et al. *Od vzdělávacího programu k vyučovací hodině: jak pracovat s kurikulem*. 2. Vyd. Praha: Portál, 2005. ISBN 80-7367-054-2.

[4] CSICSOLOVÁ, I., MAYEROVÁ, K., VESELOVSKÁ, M. *Ako univerzitní študenti pripravovali obsah robotického krúžku pre žiakov základnej školy.* DRÁBKOVÁ Jindra a Jan BERKI, eds. Sborník konference Didinfo 2018 [online]. Liberec, 2018. ISBN: 978-80-7494-424-6, ISSN: 2454-051X. Dostupné z: didinfo.net/images/DidInfo/files/didinfo_2018.pdf

[5] TUREK, I. *Didaktika*, Iura Edition spol sro, Bratislava, 2010. ISBN 978-80-8078-322-8.

[6] TKÁČOVÁ, Z. Projektové vyučování v informatice. *MATEMATIKA–FYZIKA–INFORMATIKA*, 2018, vol. 27, no.3, pp.223-230.

[7] ZORMANOVÁ, L. *Obecná didaktika: pro studium a praxi*. Grada Publishing, as. 2014

[8] CRESWELL, J. W. *Educational research: Planning, conducting and evaluating quantitative and qualitative research*. New Jersey: Pearson, 2002. ISBN: 978-0131367395.

[9] ŠVAŘÍČEK, R. et al. *Kvalitativní výzkum v pedagogických vědách*. Praha : Portál, 2007. ISBN 978-80-7367-313-0.

[10] CODE4KIDS. [online]. [vid 7.3.2019]. Dostupné na: https://www.code4kids.me/2017/12/24/step-2-ozobot-evo-manual-control/

[11] HOUR OF CODE. [online]. [vid 7.3.2019]. Dostupné na: https://storage.googleapis.com/ozobot-lesson-library/evo-the-troll/evo-the-troll.pdf

[12] Queensland Govermnet. [online]. [vid 7.3.2019]. Dostupné na: http://plconnect.slq.qld.gov.au/__data/assets/pdf_file/0003/383700/Ozobot-Evo-Activity-1.0.pdf

[13] OZOBOT Aktivity. [online]. [vid 7.3.2019]. Dostupné na: https://portal.ozobot.com/lessons/type/activity

[14] OZOBOT Blog. [online]. [vid 7.3.2019]. Dostupné na: https://blog.ozobot.com/2017/11/30/hour-code-inspiration-15-teachers-using-ozobot-enrich-young-lives/

[15] Ozobot Bit Aktivity. [online]. [vid 7.3.2019]. http://ozobot.sandofky.cz/

[16] Indiana K12 Maker Space. [online]. [vid 7.3.2019]. http://indianamakerspace.weebly.com/ozobots.html

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9 ISSN 2454-051X

Programovanie mobilných aplikácii – natívny, webový alebo hybridný vývoj?

Programming mobile applications - native, web or hybrid way?

Patrik Voštinár
Katedra informatiky FPV UMB
Tajovského 40
974 01 Banská Bystrica
Slovensko
patrik.vostinar@umb.sk

Martin Kováčik
Katedra informatiky FPV UMB
Tajovského 40
974 01 Banská Bystrica
Slovensko
martin.kovacik@studenti.umb.sk

ABSTRACT

We live in times where mobile applications became an indispensable part of our everyday life. There are many tools, technologies and ways how to create applications for mobile devices. The aim of this contribution is to show possibilities of programming mobile applications native, web and hybrid way. The contribution is focused on developing mobile application about the recipes. Application *What to Cook?* was created for comparing three times - native way with Android Studio, web way with HTML5 and hybrid way with tool PhoneGap. Each of these types of programming mobile applications has their advantages and disadvantages. The paper describes specific differences between these forms of development. It focuses on differences in programming user interface, information sharing, application architecture, and distribution of created apps. Our created app *What to Cook?* provides information about recipes, allows user registration, rating, sharing, adding and searching for recipes. It is therefore intended for users of all ages who do not know cooking advice words.

Keywords

Programming. JavaScript. Firebase. Bootstrap. Android. PhoneGap.

ABSTRAKT

Žijeme v dobe, kde sa mobilné aplikácie stali neoddeliteľnou súčasťou každodenného života. Existuje veľa nástrojov, technológií a spôsobov, ako vytvárať aplikácie pre mobilné zariadenia. Cieľom tohto príspevku je ukázať možnosti programovania mobilných aplikácií natívnym, webovým a hybridným spôsobom. Príspevok sa zameriava na vývoj mobilnej aplikácie o receptoch. Aplikácia *What to Cook?* bola vytvorená na porovnanie troch možností programovania - natívneho spôsobu s Android Studio, webovými technológiami s HTML5 a hybridným spôsobom s nástrojom PhoneGap. Každý z týchto typov programovania mobilných aplikácií má svoje výhody a nevýhody. Článok opisuje špecifické rozdiely medzi týmito formami vývoja. Zameriava sa na rozdiely v programovaní používateľského rozhrania, zdieľanie informácií, architektúru aplikácií a distribúciu vytvorených aplikácií. Naša vytvorená aplikácia *What to Cook?* poskytuje informácie o receptoch, umožňuje registráciu používateľov, hodnotenie, zdieľanie, pridávanie a vyhľadávanie receptov. Je preto určený pre používateľov všetkých vekových kategórií, ktorí nepoznajú poradenstvo varenia.

Kľúčové slová

Programovanie. JavaScript. Firebase. Bootstrap. Android. PhoneGap.

1 ÚVOD

Už takmer 40 rokov uplynulo od integrácie počítačov do vzdelávania. Za tento čas sa integrácia informačných a komunikačných technológie (IKT) postupne vyvíjala od použitia jednoduchých vzdelávacích programov na hodine, až po systémy na riadenie výučby [1]. V súčasnosti moderné vyučovacie metódy zahŕňajú e-learning, blended learning a mobile learning. Do popredia sa čoraz viac dostáva tzv. mobile learning, ktorý by sme mohli charakterizovať ako vzdelávanie s pomocou mobilných zariadení [2]. Technológie a trendy umožňujú používateľom nosiť so sebou a kedykoľvek využívať čoraz väčšie množstvo služieb a informácií, obsiahnutých hlavne v mobilných aplikáciách. Dnešné deti majú k hrám a aplikáciám na smartfónoch a iných mobilných zariadeniach veľmi blízko [3]. Pomocou týchto zariadení je teda možné ľahšie rozvíjať výpočtové myslenie [4].

Mobilné zariadenia sa môžu používať aj na ovládanie robotov. Problematika edukačnej robotiky v poslednom období veľmi rezonuje vo vzdelávaní v mnohých krajinách na všetkých stupňoch škôl. Pomocou nej sa lepšie darí motivovať žiakov a študentov k učeniu sa prírodovedne aj technicky zameraných predmetov a posilňovať tak zručnosti potrebné na riešenie problémov [5].

V príspevku sa zameriame na rôzne možnosti vývoja mobilných aplikácií, opíšeme výhody a nevýhody troch rôznych spôsobov vývoja mobilných aplikácií. V kapitole 2 si rozoberieme akými spôsobmi a nástrojmi je možné vyvíjať mobilné aplikácie. Kapitola 3 sa zameria na skúsenosti s vytvorením mobilnej aplikácie *What to Cook?* pomocou natívneho, hybridného a webového spôsobu vývoja mobilných aplikácií.

2 MOŽNOSTI VÝVOJA APLIKÁCIÍ

Pri rozhodovaní, na aký operačný systém budeme zameriavať vývoj mobilnej aplikácie, je dôležité vedieť, akých používateľov chceme zaujať. Na trhu s inteligentnými zariadeniami je dnes široký výber mobilných zariadení s rôznymi operačnými systémami. Medzi najpoužívanejšie operačné systémy patrí Android a iOS. Operačný systém Android je produktom firmy Google. Operačný systém iOS je vyvíjaný spoločnosťou Apple a určený je výlučne pre jej produkty. O zvyšok trhu sa delia ďalšie platformy ako Firefox OS, Windows mobile alebo Series 40, čo je mobilný operačný systém pre telefóny Nokia.

ISBN 978-80-557-1533-9 ISSN 2454-051X

Mobilné aplikácie môžeme vyvíjať viacerými spôsobmi. Konkrétne ide o natívny vývoj, ktorý sa špecializuje na jednu platformu. Webový vývoj umožňuje vývoj nezávisle na používanom zariadení alebo platforme. Existuje však aj hybridný spôsob vývoja, ktorý sa snaží natívnu a webovú formu kombinovať.

2.1 Natívny spôsob vývoja aplikácií

Tento spôsob je vždy zameraný na špecifickú platformu alebo zariadenie. Aplikácie vyvíjané touto formou môžu vďaka hlbšej integrácií využiť všetky funkcie operačného systému. Typickým príkladom je využitie GPS, kontaktov a všetkých senzorov telefónu.

Na svetovom trhu dominujú dvaja výrobcovia mobilného operačného systému – spoločnosť Google a jeho operačný systém Android a spoločnosť Apple s operačným systémom iOS. Vývoj aplikácií pre tieto operačné systémy je špecifický pre každý z nich. Natívne aplikácie pre operačný systém Android je možné vyvíjať v programovacom jazyku Java a od začiatku roku 2018 aj v jazyku Kotlin. Podporovaným nástrojom na vývoj pre Android bol v jeho začiatkoch vývojový nástroj Eclipse IDE. Tento nástroj bol používaný aj na vývoj desktopových a webových projektov v Jave. Neskôr vzniklo vývojové prostredie Android Studio, ktoré je navrhnuté pre vývoj mobilných aplikácií pre operačný systém Android. V tomto nástroji máme k dispozícii okrem iného aj editor grafického používateľského rozhrania, podporu emulátorov, zabudované postupy na generovanie často používaných komponentov, špecifikovaný refaktoring a rýchle opravy.

Druhým najpoužívanejším mobilným operačným systémom je iOS. Je vytvorený a vyvíjaný firmou Apple inc., exkluzívne pre jej zariadenia. Pôvodným podporovaným programovacím jazykom pre vývoj iOS aplikácií bol jazyk Objective-C. Od roku 2014 majú používatelia na výber aj nový programovací jazyk Swift. Medzi najdôležitejšie zmeny patrí napríklad automatická správa pamäte, ukončovanie príkazov bez bodkočiarky a pridanie typovej kontroly. Podporovaným vývojovým prostredím pre iOS je program Xcode [6].

Za hlavné rozdiely medzi vývojom pre iOS a Android môžeme označiť napríklad cenu distribúcie. Za distribúciu aplikácie v špecifickom obchode s mobilnými aplikáciami Google Play zaplatíme jednorazovo 25 dolárov. Ak sa rozhodneme publikovať v obchode s mobilnými aplikáciami pre iOS - App Store, musíme sa zaregistrovať do Apple Developer programu. Členstvo v tomto programe stojí 99 dolárov za jeden kalendárny rok, nezávisle od počtu publikovaných aplikácií.

Výhody natívneho vývoja aplikácií:

- Možnosť využívať všetky funkcie operačného systému ako senzory, kontakty, notifikácie, GPS a iné.
- Distribúcia aplikácií cez jeden z digitálnych obchodov.
- Použitie vývojového prostredia presne určeného pre danú platformu.
- Aplikácie vyvíjané natívne sú výkonnejšie ako tie, ktoré je potrebné kompilovať do natívneho prostredia.

Nevýhody:

- Musíme vyvinúť aplikáciu pre každú platformu, ktorej používateľov chceme osloviť (zvlášť pre Android a iOS).
- Závislosť od spoločnosti Google a Apple od programovacieho jazyka. Ak sa rozhodnú vytvoriť nový jazyk, nutnosť naučiť sa programovať v ňom.

2.2 Vývoj webových aplikácií

Webové aplikácie sú v podstate stránky, ktoré sú prispôsobené pre mobilné zariadenia. Tento druh vývoja je nezávislý od platformy a použitého mobilného zariadenia. Pre zobrazenie aplikácie na všetkých druhoch zariadení je potrebné klásť veľký dôraz na responzívny dizajn a tiež na podporu rôznych prehliadačov. Keďže logika spojená s aplikáciou sa vykonáva v prehliadači, nie je možné využívať naplno všetky funkcie operačného systému ako napríklad notifikácie a prístup ku kontaktom. Pomocou značkového jazyka HTML 5 a skriptovacieho jazyka JavaScript môžeme používať napríklad funkcie pre získavanie GPS súradníc, alebo prístup k fotoaparátu, akcelerometru alebo mikrofónu zariadenia. Pre vývoj webových aplikácií sa používajú rovnaké nástroje ako na tvorbu webových stránok. Základom dizajnu používateľského rozhrania je HTML a CSS. Na vytvorenie backendu aplikácie sa používajú jazyky JavaScript, PHP alebo iné skriptovacie jazyky. Existuje viacero nástrojov, ktoré nám umožňujú vytvárať webové aplikácie. Niektoré z nich sú platené a iné zdarma, no stačí nám aj jednoduchý textový editor.

Výhody webového vývoja aplikácií:

- jedna aplikácia pre všetky typy zariadení.
- Platíme len za hosting a doménu.
- Programovanie pomocou štandardných webových jazykov.
- Nezávislosť na platforme.

Nevýhody:

- Nemôžeme využívať všetky funkcie zariadenia ako notifikácie a kontakty.
- Nemôžeme distribuovať aplikáciu v digitálnych obchodoch jednotlivých platforiem.
- Potreba upraviť aplikáciu pre rôzne prehliadače.
- Rýchlosť aplikácie je závislá na rýchlosti internetového pripojenia.

2.3 Hybridný vývoj mobilných aplikácií

Podstata hybridného vývoja spočíva v tom, že programujeme pomocou jazykov určených na tvorbu webových stránok HTML, CSS a JavaScript. Výslednú aplikáciu pomocou natívneho púzdra, ktoré obsahuje webový prehliadač spustíme na mobilnom zariadení. Prehliadač natívneho púzdra, ktorý sa nazýva *Web View* neobsahuje žiadne ovládacie prvky, ako klasický webový prehliadač, čo spôsobuje, že aplikácia vyzerá ako natívna. Takáto aplikácia nepotrebuje internetové pripojenie, pokiaľ si ho nevyžaduje jej funkcionalita. Môžeme ju distribuovať v digitálnych obchodoch a pre rôzne operačné systémy. Existuje niekoľko frameworkov pre vývoj hybridných aplikácií. Medzi najpoužívanejšie frameworky pre programovanie mobilných aplikácií zaraďujeme PhoneGap a Xamarin [7].

PhoneGap je framework na vývoj mobilných aplikácií, ktorý vytvorila firma Nitobi. Túto firmu v roku 2011 kúpilo Adobe. Poskytuje spôsob ako vytvoriť natívnu aplikáciu pomocou webových technológií. Obsahuje jednoduché aplikačné rozhranie na spúšťanie funkcií v jazyku JavaScript, ktoré sú následne mapované do natívneho kódu. V podstate dokáže akúkoľvek natívnu funkcionalitu premeniť na webovú. Aplikačné rozhranie pozostáva hlavne z pluginov, ktoré nám umožňujú prístup ku všetkým funkciám zariadenia. Výslednú aplikáciu môžeme skompilovať do inštalačného súboru pre Android alebo iOS pomocou PhoneGap build. V rámci jedného účtu môžeme bezplatne skompilovať jednu súkromnú aplikáciu. Ak chceme

ISBN 978-80-557-1533-9 ISSN 2454-051X

vyvíjať viacero súkromných aplikácií musíme zaplatiť poplatok 12 a viac dolárov mesačne v závislosti od typu vývojárskeho účtu. [7]

Xamarin je nástroj na tvorbu multiplatformových aplikácií, nielen pre mobilné operačné systémy. Vznikol v roku 2016 a jeho vlastníkom je v súčasnosti firma Microsoft. Pomocou nástroja Xamarin môžeme v programovacom jazyku C# vytvoriť aplikáciu pre Android, iOS, Windows aj macOS. Navyše sa nejedná iba o webovú aplikáciu v natívnom Web View, ale o identickú aplikáciu ako pri natívnom vývoji. Portuje všetky knižnice systému Android a iOS do jazyka C#. Xamarin je súčasťou vývojového prostredia Visual Studio od spoločnosti Microsoft. [8]

Výhody hybridného vývoja:

- Vývoj aplikácie pomocou webových štandardov.
- Prístup ku všetkým funkciám zariadenia.
- V prípade nástroja Xamarin - výkon porovnateľný ako dosahujú natívne aplikácie.
- Možnosť distribuovať vytvorené aplikácie v špecializovaných digitálnych obchodoch.

Nevýhody:

- Výkonnosť (okrem nástroja Xamarin)

3 Mobilná aplikácia *What to Cook*

Pre porovnanie programovania natívnym, webovým a hybridným spôsobom sme sa rozhodli vytvoriť rovnakú aplikáciu v každom spomínanom type programovania. Rozhodli sme sa vytvoriť aplikáciu určenú pre domácnosti. Aplikácia poskytuje informácie o jednej zo základných ľudských potrieb a to o jedle. V súčasnosti existuje množstvo aplikácií a internetových stránok zameraných na varenie. Vyhľadávanie v databázach je však zdĺhavé a človek často nevie, čo vlastne chce. Často sa rozhodujeme podľa toho, aké ingrediencie máme k dispozícii. Vzniká tak potreba vyhľadávať recepty nie podľa názvu, ale podľa prísad. Často sa tiež chceme o recepty podeliť s priateľmi. Tiež je dobré často používané recepty uložiť a mať k nim prístup ešte rýchlejšie. Okrem spomínanej funkcionality obsahuje naša aplikácia aj pridávanie receptov a teda, aby sme sa mohli podeliť s našimi receptami a zároveň nechali ostatných používateľov ohodnotiť novo-vzniknuté recepty. Existuje niekoľko aplikácií a stránok, ktoré ponúkajú podobné funkcie, ale ani jedna neobsahuje funkcionality, ktoré obsahuje naša aplikácia.

Navrhnutá aplikácia je výborným príkladom pre porovnanie rôznych spôsobov programovania, nakoľko v nej potrebujeme používať fotoaparát (na pridávanie obrázkov receptov), databázu (ukladanie a výber receptov), prístup k sociálnym sieťam a rozdiely pri tvorbe grafickou-používateľského rozhrania.

3.1 Grafické používateľské rozhranie (GUI)

V natívnej aplikácii pre operačný systém Android sa grafické používateľské rozhranie vytvára pomocou jazyka XML. Funkcionalitu aplikácií majú na starosti triedy – aktivity. Rozloženie prvkov v okne natívnej aplikácie určuje layout element. Tento element určuje akým spôsobom sa budú jednotlivé komponenty rozhrania zobrazovať používateľovi. Je rodičovským elementom ostatných komponentov, odvodeným z triedy ViewGroup. V súčasnosti máme k dispozícii viacero druhov layoutu. Jednotlivé typy layoutov môžeme navzájom vnárať do seba.

Vzhľad vo webových aplikáciách sa vytvára pomocou jazyka HTML, CSS a knižnici Bootstrap. V porovnaní s natívnou aplikáciou všetko je uložené v jednej webovej stránke. Obsah dokumentu v HTML je definovaný na základe kontajnerových tagov, do ktorých vkladáme tagy pre ovládacie prvky rozhrania. Pozíciu, veľkosť jednotlivých elementov aj zmeny v závislosti od rozlíšenia a veľkosti obrazovky definujeme pomocou kaskádových štýlov.

Výhodou webovej aplikácie je skutočnosť, že môžeme oddeliť štylizáciu prvkov rozhrania od jeho kostry. To zabezpečuje istú dávku prehľadnosti. Natívna aplikácia pre Android má väčšinu rozhrania v jednom súbore, no môžeme ho upravovať aj pomocou Java súborov, ktoré definujú logiku aktivity.

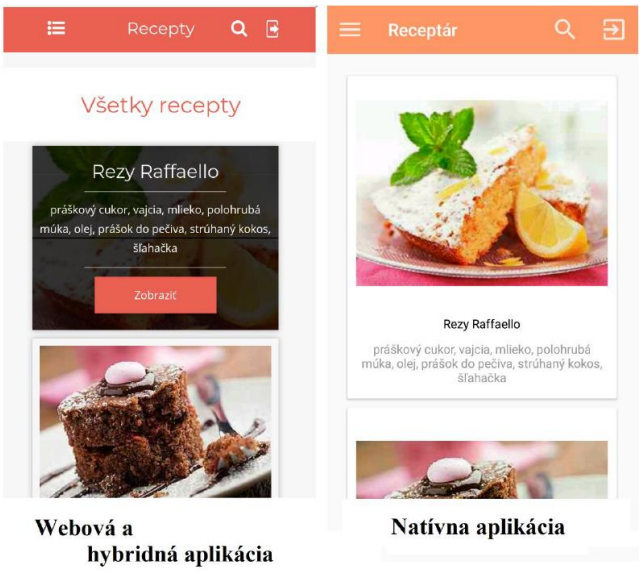
Pri vytváraní hybridnej aplikácie sme použili PhoneGap. Pri vytváraní vzhľadu sme s veľkej miery využili zdrojové súbory, ktoré sme vytvorili pri webovej mobilnej aplikácii.

3.2 Používateľské prvky

Validácia údajov z textových polí prebieha vo všetkých spôsoboch vývoja mobilných aplikácii rovnako, rozdiel je však napríklad pri zobrazovaní chybových hlásení. V natívnom prostredí máme pripravených viacero prostriedkov, ktoré si však v tom webovom musíme sami vytvoriť. Napríklad prvok *Toast*, ktorý je prispôsobený na rýchle zobrazovanie hlásení používateľovi vo webovej aplikácii k dispozícii nemáme. Prvok *ProgressBar* je v natívnom prostredí samozrejmosťou, no vo webovej aplikácii sme ho museli vytvoriť pomocou kaskádových štýlov a nadefinovať jeho animáciu. Zobrazovanie chybových hlásení pri nesprávnom formáte údajov sme v natívnej aplikácii zabezpečili pomocou funkcie *setError*, no vo webovej sme museli vytvoriť kontajner ktorý bude zobrazovať hlásenia a rušiť ich pomocou vlastnosti *visibility*. Tiež sme kvôli prehľadnosti zafarbovali obsah textových polí červenou farbou, keď mal ich obsah nechcený formát. V natívnej aplikácii nám pomohlo *SharedPreferencies* na uloženie prihlasovacích údajov. Vo webovej aplikácii sme mali k dispozícii predpripravenú metódu Firebase, na nastavenie spôsobu uloženia prihlásenia, ktorá v natívnom prostredí nie je k dispozícii [7].

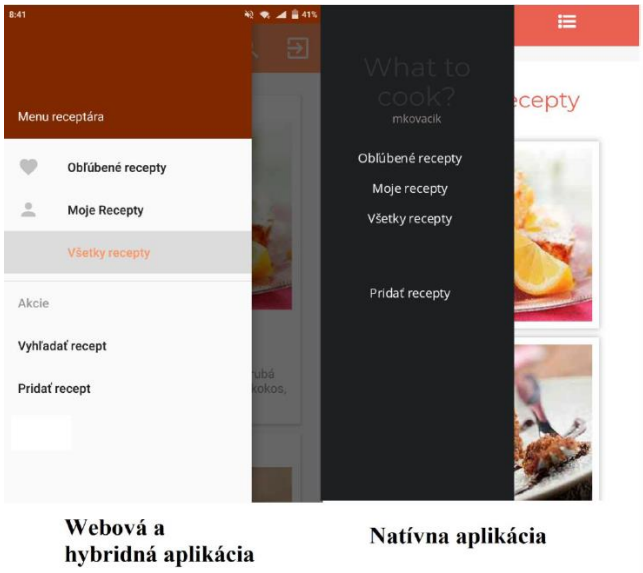
Keďže JavaScript je dynamicky typovaný programovací jazyk, logika jednotlivých okien sa programuje jednoduchšie ako v Jave. Nemusíme vytvárať triedy na uchovávanie objektov z Firebase databázy, ani meniť dátové typy premenných. V natívnom prostredí máme zas pre mnohé dôležité časti používateľského rozhrania k dispozícii pripravené triedy, ktoré sa jednoducho implementujú a prispôsobujú veľkosti displeja. Sú to napríklad triedy ActionBar a NavigationView. Ich nevýhodou oproti webovému prostrediu je slabšia podpora upravovania. Pri webových aplikáciách môžeme dizajn prvkov upravovať prakticky akokoľvek, no musíme ich od základu vytvoriť, alebo na ich zostrojenie využiť pluginy a šablóny. Štylizácia natívnych prvkov býva zvyčajne jednoduchšia, no napríklad vytvorenie galérie bolo podľa nášho názoru rýchlejšie vo webovej aplikácii. Celkovo je podľa nás v natívnej aplikácii komplikovanejšie definovať logiku okna a vo webovej aplikácii jeho dizajn. Podarilo sa nám vytvoriť veľmi podobný dizajn hlavného okna vo webovom a natívnom prostredí (Obrázok 1).

ISBN 978-80-557-1533-9 ISSN 2454-051X



Obrázok 1: Obrázok hlavného okna aplikácia *What to Cook?*

Na obrázku 2 sú zobrazené aplikácie vytvorené pomocou webového a hybridného vývoja naľavo a napravo pomocou natívneho vývoja. Obidva obrázky zobrazujú menu aplikácie.



Obrázok 2: Porovnanie menu aplikácie *What to Cook?*

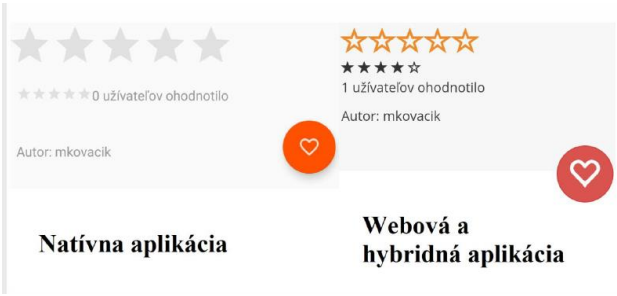
3.3 Zdieľanie údajov

Podľa nášho názoru je prenos dát medzi aktivitami v natívnej aplikácii bezpečnejší ako ich prenos medzi stránkami. Pri webovej aplikácii je potrebné klásť dôraz na bezpečnosť napríklad kvôli nebezpečenstvu SQL injekcii alebo krádeži prihlasovacích údajov. Nám však SQL útoky nehrozia, keďže používame NO SQL databázu. Proti zneužitiu údajov sa bránime SSL certifikátom, ktorý umožní naše stránky jednoznačne identifikovať. Prehliadače aj vyhľadávače našu aplikáciu s SSL certifikátom vyhodnocujú ako bezpečnú a tiež umožňuje použitie HTTPS protokolu.

3.4 Zdieľanie receptov

V natívnej aplikácii sme implementovali funkcionalitu, ktorá umožňuje zdieľať recepty cez sociálne siete Facebook, Twitter a Instagram (Obrázok 3). Tiež umožňujeme odoslať recepty cez sms, e-mail a uložiť ako poznámku do telefónu. Vo webovej aplikácii umožňujeme používateľom zdieľať recepty na sociálnej sieti Facebook alebo ich odoslať e-mailom.

Na zdieľanie dát v natívnej aplikácii úplne postačuje spustenie zámeru a operačný systém sa postará, aby sa dáta dostali do aplikácie Facebooku. To samozrejme neplatí pri použití Facebook API. Pri jeho použití síce nepotrebujeme aplikáciu Facebooku, no musíme si vygenerovať identifikátor na stránkach pre developerov. Oproti tomu pri webovej aplikácii potrebujeme nielen identifikátor aplikácie, ale aj dokument ochrany súkromia. Tu môžeme podľa nás tiež vidieť rozdiel v nutnom zabezpečení pri rôznych formách vývoja aplikácie. Natívna aplikácia pri spustení zámeru pošle dáta len aplikácii Facebooku, no cez webovú a natívnu s použitím Facebook API sa môže používateľ aj prihlasovať a povoliť jej využívať informácie z profilu. Preto je potrebné vyššie zabezpečenie, aby nedošlo k zneužitiu súkromných informácií.



Obrázok 3: Zdieľanie receptov pomocou sociálnych sietí

3.5 Testovanie aplikácií na reálnych zariadeniach

Priebežné testovanie mobilných aplikácii v programe Android Studio je možné na reálnych zariadeniach alebo emulátoroch. Projekt spustíme pomocou zeleného tlačidla Run (podobne ako v iných programovacích jazykoch) a následne si vyberieme z ponuky vytvorených emulátorov alebo ak máme pripojený mobilný telefón prostredníctvom kábla k počítaču, tak vyberieme názov nášho pripojeného mobilného zariadenia.

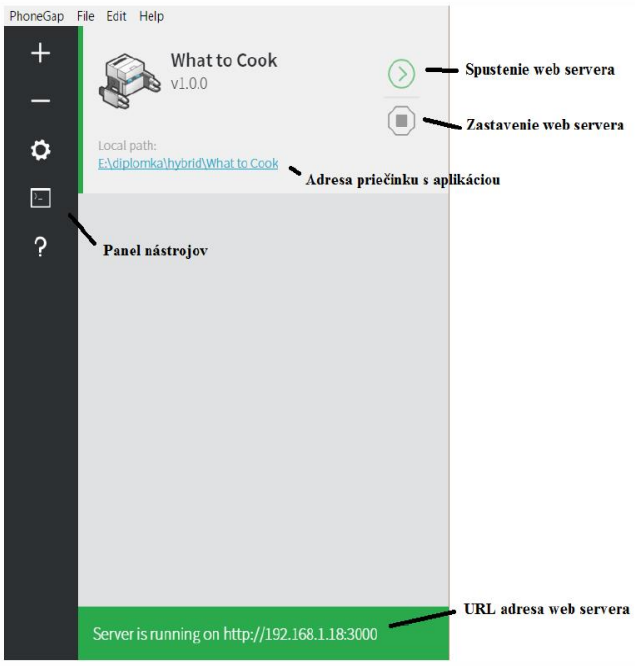
Keď chceme testovať vytvorenú aplikáciu pre operačný systém Android vytvorený pomocou PhoneGap musíme vykonať nasledujúce kroky:

1. Stiahnuť a nainštalovať desktopovú aplikáciu pre PhoneGap[1]
2. Vyplniť základné informácie o našom projekte (názov, identifikátor, adresár)
3. Vytvoriť, alebo nakopírovať do priečinka www celý adresár s webovou aplikáciou.
4. Z obchodu Google Play v mobilnom zariadení musíme stiahnuť aplikáciu PhoneGap

[1] Odkaz na stiahnutie: https://phonegap.com/getstarted/

ISBN 978-80-557-1533-9 ISSN 2454-051X

5. V desktopovej aplikácii PhoneGap spustiť projekt (aplikácia vytvorí webový server s našim projektom a zobrazí adresu tohto servera)
6. Vygenerovanú aplikáciu otestujeme vložením do prehliadača mobilného zariadenia (v tomto prípade, však bude vyzerať ako webová aplikácia). Ak chceme aby sme videli, ako bude vyzerať po prekonvertovaní do natívnej podoby, do mobilnej aplikácie PhoneGap (Obrázok 4) vložíme vygenerovanú adresu a stlačíme tlačidlo *Connect*. Aplikácia stiahne náš projekt z webového servera a spustí ho na našom zariadení.



Obrázok 4: Mobilná aplikácia PhoneGap pre testovanie vytvorených aplikácií pomocou hybridného vývoja

3.6 Generovanie spustiteľného súboru (apk)

Vytvorenie spustiteľného súboru apk vytvoreného vo vývojovom prostredí Android Studio je nenáročný proces. V záložke Build stačí vybrať možnosť Generate Signed APK a následne iba vybrať miesto uloženia.

V prípade hybridného vývoja pomocou PhoneGap si musíme otvoriť internetovú stránku PhoneGapu a spustiť PhoneGap build , ktorý umožňuje konvertovať PhoneGap projekt do súboru pre operačné systémy iOS, Android alebo Windows. Pred vygenerovaním spustiteľného súboru vyberieme možnosť, či chceme vytvoriť súkromnú aplikáciu s maximálnou veľkosťou 50 megabajtov (takúto aplikáciu môžeme na jednom účte vytvoriť iba raz), alebo vytvoríme verejnú aplikáciu, ktorej zdrojové kódy sú voľne dostupné v repozitári GitHub. Takýchto aplikácií môžeme vytvoriť neobmedzený počet. Do PhoneGap buildu musíme nahrať vytvorený kód a stlačiť tlačidlo APK, ktorý nám vygeneruje inštalačný balík aplikácie pre Android.

4 Vývoj mobilných aplikácií na školách

Žiaci sa s výučbou programovania mobilných aplikácií na slovenských školách stretávajú žiaľ zriedkavo. Niektorí žiaci si mohli vyskúšať programovanie mobilných aplikácií v rámci rôznych projektov ako IT Akadémia alebo Malá školská reforma (občianske združenie Harpúna). Na základných školách je dominantné a odporúčané prostredie MIT App Inventor – blokové programovanie. Na stredných školách sa žiaci môžu stretnúť tiež s MIT App Inventorom (učitelia preškolení v rámci projektu IT Akadémia) alebo sa môžu stretnúť s niektorou formou programovania mobilných aplikácií spomínaných v tomto článku (Malá školská reforma – natívny vývoj aplikácií).

Na univerzitách u nás, ale aj vo svete je úplne bežné, že študenti prejdú viacero spôsobov programovania mobilných aplikácií.

5 Záver

Na porovnanie rozdielov natívneho, webového a hybridného vývoja sme vyvinuli rovnakú aplikáciu každým z uvedených spôsobov vývoja. Aplikácia slúži všetkým milovníkom dobrého jedla. Umožňuje pridávanie, vyhľadávanie, prezeranie, hodnotenie a zdieľanie rôznych receptov. Natívnu aplikáciu sme vytvárali v prostredí Android Studio (v jazyku Java). Na webovú aplikáciu sme vyuţili jazyky HTML, CSS a JavasScript. Hybridná aplikácia vznikla konverziou webovej aplikácie pomocou nástroja PhoneGap.

Všetky tri spôsoby vývoja majú svoje výhody aj nevýhody. Podľa nás je z pohľadu rýchlosti aplikácie najbezpečnejším riešením natívna aplikácia, pretože hlboká integrácia s operačným systémom zabezpečuje hladký chod aplikácie a nie je obmedzená ani rýchlosťou internetu. Najlacnejším riešením z pohľadu nákladov na vývoj a distribúciu je webové riešenie, pretože platíme len za doménu a hosting. V prípade, že naša aplikácia nemá veľké nároky na výpočtový výkon, je tiež výhodné vyvíjať hybridnou formou. Je to aj najrýchlejší spôsob z pohľadu času vynaloženého na vývoj. Najlepším riešením je podľa nás vytvoriť webovú aplikáciu pomocou naportovaná cez PhoneGap, pretože zabezpečíme, že bude dostupná na každom zariadení a používateľ k nej bude mať vždy prístup.

Rozdielov medzi vývojom vo webovom a natívnom prostredí je mnoho. Nám sa podarilo popísať aspoň niektoré z nich. Mobilné technológie sa však neustále rozvíjajú a preto je aj naďalej potrebné skúmať aspekty vývoja mobilných aplikácií vo všetkých formách, ktoré poznáme.

POĎAKOVANIE

Príspevok bol spracovaný ako súčasť projektu KEGA č. 003UMB-4/2017 „Implementácia blended learningu do prípravy budúcich učiteľov matematiky“.

Príspevok vznikol za podpory grantového programu e-Školy pre budúcnosť v rámci Nadácie Orange.

BIBLIOGRAFICKÉ ODKAZY

[1] POKORNÝ, M. Kurz desatinné čísla. In *Sborník příspěvků z konference a soutěže eLearning 2018*. Hradec Králové : Gaudeamus, 2018, s. 15-19. ISBN 978-80-7435-718-3.

[2] POKORNÝ, M. Blended learning of combinatorics and data processing. In *EDUKACJA – TECHNIKA – INFORMATYKA W BUDOWANIU LEPSZEJ PRZYSZŁOŚCI*. Radom : Pułaskiego w Radomiu, 2018, s. 15-22. ISBN 978-83-7351-860-5.

[3] KLIMOVÁ, N, HORVÁTHOVÁ, D. Súťaž KODU CUP ako podporný prostriedok algoritmického myslenia. In: *DidInfo&DidactIG 2017*. Banská Bystrica: Univerzita Mateja Bela, Fakulta prírodných vied, 2017, s. 197–198. ISBN 978-80-557-1216-1.

ISBN 978-80-557-1533-9 ISSN 2454-051X

[4] ŠKRINÁROVÁ, J., KLIMOVÁ, N. Výpočtové myslenie. In: *DidInfo&DidactIG 2017*. Banská Bystrica: Univerzita Mateja Bela, Fakulta prírodných vied, 2017, s. 83–87. ISBN 978-80-557-1216-1.

[5] KLIMOVÁ, N., HORVÁTHOVÁ, D. Robot Phiro ožíva v rukách detí. In: *DIDINFO 2018,* s. 270-277. ISBN 978-80-7494-424-6.

[6] ZIFLAJ, A. Native vs Hybrid App Development [online]. [cit.5.2.2018] Dostupné z: https://www.sitepoint.com/native-vs-hybrid-app-development/

[7] KOVÁČIK, M. Rozdiely medzi natívnym, webovým a hybridným vývojom mobilných aplikácií : diplomová práca. Banská Bystrica : UMB, 2017. 87 s.

[8] KADLEC, R. Ako zdokonaľovať mobilné a webové aplikácie? Vyskúšajte službu Firebase. [online]. [cit. 15.2.2018]. Dostupné z: https://touchit.sk/akozdokonalovat-mobilne-a-webove-aplikacie-vyskusajte-sluzbu-firebase/87808.

Krátke príspevky

Short Papers

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Súťaž iBobor – kategória Nevidiaci ZŠ - prvýkrát online

Ľudmila Jašková
KDMFI FMFI UK
Mlynská dolina
84248 Bratislava
Slovensko
jaskova@fmph.uniba.sk

ABSTRACT
In this extended abstract, we describe and analyze solutions of tasks for blind students of lower secondary education in the Beaver contest. We will mention the tasks of the last year in terms of adaptation and thematic focus. We analyze students' success in solving the tasks to identify the factors affecting the difficulty of tasks.

Keywords
Blind. Bebras contest. Algorithmic thinking.

ABSTRAKT
V našom rozšírenom abstrakte sa venujeme stručnému popisu a analýze riešení úloh pre nevidiacich žiakov nižšieho sekundárneho vzdelávania v súťaži iBobor. Zmienime sa o zadaniach úloh ostatného ročníka z hľadiska prispôsobení a tematického zamerania. Analyzujeme úspešnosť žiakov pri riešení úloh s cieľom identifikovať faktory vplývajúce na obťažnosť úloh.

Kľúčové slová
Nevidiaci. iBobor. Algoritmické myslenie.

1 ÚVOD
Súťaž Informatický bobor sa na Slovensku koná každoročne od roku 2009. V ostatnom roku prebehla vo viac ako 40 krajinách[1]. Na Slovensku sa jej zúčastnilo 77 928 súťažiacich z 979 škôl[2]. Slovensko je jediná krajina, ktorá ponúka dve špeciálne kategórie pre nevidiacich žiakov sekundárneho vzdelávania. Vďaka tomu sa do súťaže môžu zapojiť títo žiaci bez asistencie iných osôb. V našom príspevku priblížime priebeh súťaže v uvedených kategóriách počas ostatného ročníka. Najprv stručne priblížime prispôsobenia pravidiel súťaže a zadaní úloh, aby sa nevidiaci žiaci mohli do súťaže zapojiť. Ďalej sa zameriame na kategóriu žiakov nižšieho sekundárneho vzdelávania (Nevidiaci ZŠ). Stručne analyzujeme úspešnosť týchto žiakov pri riešení úloh s cieľom identifikovať faktory vplývajúce na obťažnosť úloh.

2 IBOBOR PRE NEVIDIACICH
V roku 2013 sme realizovali pilotný beh súťaže pre nevidiacich žiakov nižšieho sekundárneho vzdelávania. Úlohy neboli prezentované online na webe, ale použili sme formát textového dokumentu v počítači lebo s textovým dokumentom dokázali pracovať všetci nevidiaci žiaci, s webovým prehliadačom mali ešte niektorí z nich problémy. O výsledkoch z predošlých ročníkov sme sa zmienili v publikáciách [1, 2]. Tu len zopakujeme modifikácie, ktoré sme vykonali, aby sa súťaže mohli zúčastniť nevidiaci. Vychádzali sme z odporúčaní organizácie Royal National Institute for the Blind [3] a American Printing House for the Blind [4].

Zmenili sme počet úloh, aby mali žiaci viac času. Intaktní žiaci majú na jednu úlohu v priemere 2 minúty a 40 sekúnd. Pre nevidiacich sme predĺžili priemerný čas na riešenie jednej úlohy na takmer 4,5 minúty. Takže namiesto 15 úloh (5 ľahkých, 5 stredných, 5 ťažkých) riešia nevidiaci žiaci počas 40 minút iba 9 úloh (3 ľahké, 3 stredné, 3 ťažké).

Snažíme sa použiť rovnaké úlohy ako pre intaktných žiakov. Niektoré úlohy sa dajú použiť bez akejkoľvek modifikácie. Mnohé úlohy si vyžadujú menšie alebo väčšie modifikácie nasledujúceho charakteru [5]:

- Obrázky obsahujúce relevantné informácie nahradíme textom.
- Nepoužívame referencie na farby.
- Tabuľky nahradíme zoznamom alebo ich textovým formátom.
- Nepoužívame interaktívne prvky, aby žiaci mohli zadávať riešenia iba pomocou klávesnice.
- Používame kratšie postupnosti objektov alebo prvkov, pretože nevidiaci si pri riešení úlohy musia postupnosti pamätať.

Kategóriu pre nevidiacich študentov vyššieho sekundárneho vzdelávania sme pilotne overili v roku 2016 na jednom bratislavskom gymnáziu. Vtedy sa do súťaže zapojili štyria nevidiaci žiaci prvého ročníka. Úlohy neriešili vo webovom prehliadači, ale v textovom dokumente. V roku 2017 už nevidiaci študenti v tejto kategórii riešili úlohy online, takže sa do súťaže zapojili študenti z piatich škôl v rôznych končinách Slovenska. Tento formát sa nám osvedčil, tak sme sa rozhodli, že v roku 2018 budú online súťažiť aj nevidiaci žiaci nižšieho sekundárneho vzdelávania. V tejto kategórii sa zúčastnilo súťaže 24 žiakov z piatich škôl, každá v inom meste Slovenska. Dve školy boli špeciálne pre zrakovo postihnutých a tri školy boli bežné, v ktorých žiaci študujú integrovane. Nakoľko sa súťaže v kategórii nevidiacich žiakov vyššieho sekundárneho vzdelávania zúčastnili len šiesti žiaci z jednej školy, ďalej sa zameriame na kategóriu pre nevidiacich žiakov nižšieho sekundárneho vzdelávania.

3 PRIEBEH SÚŤAŽE
Ako sme už uviedli, v kategórii Nevidiaci ZŠ súťažilo 24 ťažko zrakovo postihnutých žiakov (15 chlapcov a 9 dievčat). Úspešnými riešiteľmi (t. j. dosiahli viac ako 29 bodov zo 48) boli 8 žiaci (4 chlapci, 4 dievčatá). Najviac žiakov, až 15, bolo z 8. a 9. ročníka ZŠ, zvyšných 9 žiakov bolo z 5. až 7. ročníka ZŠ. Zo špeciálnych škôl pre zrakovo postihnutých bolo 20 žiakov a len štyria žiaci boli z troch rôznych škôl, kde študujú integrovane spolu s intaktnými rovesníkmi.

[1] https://www.bebras.org/?q=countries

[2] http://www.ibobor.sk/

ISBN 978-80-557-1533-9, ISSN 2454-051X.

Žiaci mali zadania zobrazené vo webovom prehliadači a v ňom aj zadávali odpovede voľbou prepínačov vo formulároch. Okrem toho mohli používať textový editor na zápis poznámok a postupu riešenia a mohli si kresliť obrázky perom na euroobal položený na mäkkej podložke, čím vznikal hmatateľný reliéf (obrázok 1).

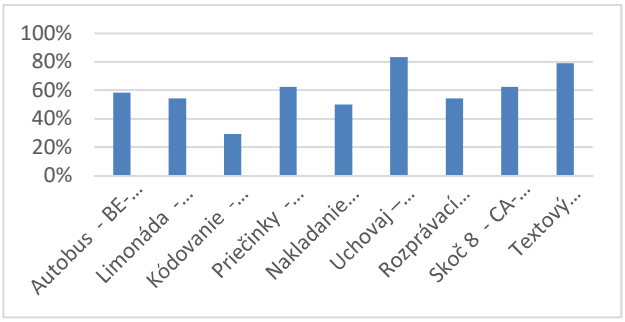
Obrázok 1: Snímok euroobalu s obrázkom, ktorý vytváral nevidiaci žiak pri riešení jednej z úloh

Úlohy pre kategóriu Nevidiaci ZŠ sme vyberali spomedzi úloh, ktoré sa plánovali použiť pre rôzne kategórie intaktných žiakov ZŠ. Keďže riešenie väčšiny úloh si vyžadovalo buď prácu s obrázkami, alebo vizuálnu predstavivosť, výber úloh nebol jednoduchý. Každá úloha si vyžadovala menšie, či väčšie úpravy. Najčastejšie bolo potrebné zmeniť interaktívnu úlohu na úlohu s výberom odpovede, obrázky nahradiť popisom alebo textom, prípadne skrátiť alebo zmeniť postupnosť prvkov. V mnohých prípadoch bolo potrebné upraviť aj úroveň náročnosti.

V rámci tohto príspevku nemáme priestor, aby sme bližšie popísali úlohy. Ich presné znenie možno nájsť na webe [6]. Uvedieme len informáciu, že prvé štyri úlohy sa týkali oblasti Reprezentácie a nástroje a druhé štyri úlohy sa týkali oblasti Algoritmické riešenie problémov. Posledná úloha **Textový kurzor** sa týkala oboch oblastí.

4 ANALÝZA ÚSPEŠNOSTI

Po analýze úspešnosti žiakov pri riešení úloh sme zistili, že kategorizácia viacerých úloh podľa náročnosti nebola optimálna.



Obrázok 2: Náročnosť úloh Q

Hodnota náročnosti Q je percentuálny podiel iných než správnych odpovedí a všetkých odpovedí. Za bežné sa považujú úlohy s hodnotou Q od 30 do 70, úlohy dosahujúce 20 až 30 sú jednoduché a pod 20 sú veľmi jednoduché. Podobne úlohy s náročnosťou 70 až 80 sú náročné a nad 80 extrémne náročné [7,8].

Podľa grafu na obrázku 2 vidíme, že úlohy **Autobus**, **Limonáda**, **Priečinky** a **Uchovaj-Pridaj** sme považovali za menej náročné ako v skutočnosti boli a naopak úloha **Rozprávací robot** bola menej náročná. Vidíme tiež, že naozaj ľahká bola iba úloha **Kódovanie** a veľmi ťažké boli úlohy **Uchovaj-Pridaj** a **Textový kurzor**. Všetky zvyšné úlohy mali strednú náročnosť. V budúcnosti by sme mali zaradiť menej takýchto úloh a viac ľahkých úloh.

Počas súťaže sme sledovali žiakov na jednej špeciálnej škole pre zrakovo postihnutých a hneď po skončení súťaže sme mali možnosť realizovať s nimi skupinové interview. V rámci pozorovaní sme zistili, že žiaci, ktorí si robili poznámky v textovom editore, boli úspešnejší. V rámci skupinovej diskusie sme zistili, že úlohy sa žiakom páčili a vyhovoval im aj online formát. Zadania úloh považovali za zrozumiteľné, okrem úlohy **Uchovaj-Pridaj**. Žiaci sa vyjadrili, že pri riešení tejto úlohy vôbec nevedeli, čo majú robiť. Za najľahšie považovali žiaci úlohy **Limonáda**, **Priečinky**, **Nakladanie zmrzlín**. Za najťažšiu považovali úlohu **Textový kurzor**.

Ako sme uviedli, polovica úloh bola zameraná na riešenie algoritmických problémov. Dôležitým faktorom ovplyvňujúcim úspešnosť žiakov bol teda predchádzajúci rozvoj ich algoritmického myslenia a programátorských zručností, čo bol ešte do nedávna v prípade nevidiacich žiakov problém kvôli dostupnosti vhodnej metodiky a vhodného softvéru.

Vychádzajúc z našich pozorovaní počas súťaže konštatujeme, že nižšia úspešnosť nemá na žiakov demotivujúci vplyv. Aj žiaci s nižšou úspešnosťou mali radosť z toho, že sa mohli súťaže zúčastniť a vďaka tomu sa niečo nové naučiť. Podľa [9] sú testy a súťaže efektívnou metódou učenia, ak študenti v pomerne krátkom časovom období dostanú spätnú väzbu a budú mať možnosť dozvedieť sa správne riešenia. Táto podmienka je v prípade súťaže iBobor splnená. Vďaka online formátu sa žiaci bodové skóre dozvedia okamžite a správne riešenia s vysvetleniami si môžu pozrieť v priebehu niekoľkých týždňov. Nápomocný je aj archív úloh z predchádzajúcich rokov. Vďaka nemu sa môžu učitelia a žiaci na nasledujúci ročník súťaže lepšie pripraviť.

BIBLIOGRAFICKÉ ODKAZY

[1] JAŠKOVÁ, Ľ.et al. Prvé skúsenosti s realizáciou súťaže iBobor pre nevidiacich žiakov druhého stupňa ZŠ. In: *Didinfo 2015*, TRAJTEĽ, Ľ. (Eds.), BB: UMB, 2014. ISBN 978-80-557-0852-2

[2] JAŠKOVÁ, Ľ., et al. Porovnanie úspešnosti nevidiacich a intaktných žiakov pri riešení úloh súťaže iBobor In: TRAJTEĽ, Ľ. (Eds.) *Didinfo 2016*, BB: UMB, 2016. ISBN 978-80-557-1082-2

[3] *Overview of exam access arrangements* http://www.rnib.org.uk/sites/default/files/Overview_of_exam_access_arrangements_May_2014.doc

[4] ALLMAN, C. B. 2009. *Test Access. Making Tests Accessible for Students with Visual Impairments: A Guide for Test Publishers, Test Developers, and State Assessment Personnel*. 4th Release, American Printing House for the Blind, Louisville, Kentucky

[5] *Vyučovanie Informatiky pre Nevidiacich*, Súťaž iBobor - kategória nevidiaci, http://vin.edi.fmph.uniba.sk/iBobor.html

[6] *iBobor - o kategórii Nevidiaci*, http://www.ibobor.sk/nevidiaci.php

[7] CHRÁSKA, M.: *Metody pedagogického výzkumu,* Grada, Praha 2008, ISBN 978-80-247-1369-4

[8] TOMCSÁNYI, P. Náročnosť úloh v súťaži informatický bobor, In: ROVAN, B. *Didinfo 2009*, BB: UMB, ISBN

[9] CAREY, B.: *How we learn*, New York : Random House, 2014.

Výučba komunikačnej bezpečnosti a kryptografie
Teaching of Communication security and cryptography

Alžbeta Kanáliková
KRIS, FEIT, UNIZA
Univerzitná 1
010 26 Žilina
Slovensko
alzbeta.kanalikova@gmail.com

ABSTRACT
In today's information time, it is very important to secure all the information that is transmitted in any way over the network, via mobile phones and other communication devices. Communication security is realized through various cryptographic algorithms. For this reason, it is necessary to educate future technical experts in this field. Cryptography and communication security require closer understanding of even more demanding algorithms. For this reason, it is necessary to search, respectively create software tools as well as video tutorials that will more clearly explain the operation of the algorithm. The article describes software tools and video tutorials used in cryptography teaching and at the same time, the article also deals with teaching through the tools mentioned.

Keywords
Cryptography, Cryptographic algorithm, Cryptool, Software tools, Video tutorial.

ABSTRAKT
V dnešnej informačnej dobe je veľmi dôležité zabezpečiť všetky informácie, ktoré sa prenášajú akýmkoľvek spôsobom po sieti, cez mobilné telefóny a iné komunikačné zariadenia. Zabezpečenie komunikácie sa realizuje prostredníctvom rôznych kryptografických algoritmov. Z tohto dôvodu je potrebné vzdelávať budúcich technických odborníkov v tejto oblasti. Kryptografia a komunikačná bezpečnosť si vyžaduje bližšie pochopenie aj náročnejších algoritmov. Z tohto dôvodu je potrebné hľadať, resp. vytvárať softvérové nástroje ale aj videa, ktoré názornejšie vysvetlia činnosť algoritmu – jeho jednotlivé kroky a zároveň ukážu aj výsledok šifrovania, resp. dešifrovania. Článok popisuje softvérové nástroje ale aj výučbové videa využívané vo výučbe kryptografie. Zároveň sa článok zaoberá aj výučbou prostredníctvom spomínaných nástrojov.

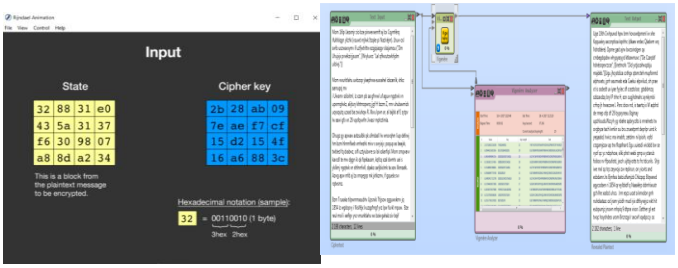
Kľúčové slová

Kryptografia, kryptografické algoritmy, Cryptool, softvérové nástroje, video.

1 ÚVOD
Výučba bakalárov v odboroch automatizácia a telekomunikácie na fakulte a Elektrotechniky a informačných technológií v Žiline si vyžaduje komplexnú prípravu v oblasti bezpečnej komunikácie. Študenti sú vedení aby poznali základné kryptografické algoritmy a v praxi vedeli nastaviť bezpečnú komunikáciu. Jedným z viacerých predmetov z tejto oblasti je predmet komunikačná bezpečnosť. Tento predmet je realizovaný prostredníctvom prednášok, cvičení a praktických laboratórnych cvičení. Študenti prechádzajú od základov klasickej kryptografie, k modernej kryptografií. Výučba je postavená nielen na informácií, že daný algoritmus existuje a kde sa používa, ale aj na znalosti matematických princípov algoritmov. Študent by mal vedieť na jednoduchých príkladoch šifru použiť (zašifrovať a dešifrovať správu), resp. vedieť postup jednotlivých operácií. Z tohto dôvodu je vo výučbe dôležité uplatňovať hlavné didaktické zásady: názornosti, vedeckosti a zásadu prepojenia s praxou. [1] [2].

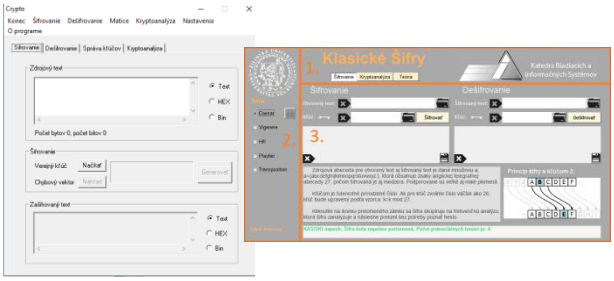
2 Softvérové a video pomôcky vo výučbe komunikačnej bezpečnosti
Pri výučbe predmetu komunikačná bezpečnosť sa používa viacero softvérových nástrojov. Medzi najpoužívanejšie nástroje patrí e-learningový nástroj *Cryptool*, *Matlab* a ďalšie vytvorené nástroje pre výučbu hlavne v bakalárskych a diplomových prácach.

2.1 E-learningový nástroj Cryptool
CrypTool je *opensource* e-learningový projekt, ktorý obsahuje bezplatný e-learningový softvér. Úlohou tohto projektu je predstaviť kryptografické a kryptoanalytické koncepty [3]. *CrypTo*ol implementuje viac ako 400 algoritmov [4]. Má viacero implementácií: *Cryptool 1*, *Cryptool 2*, pre vývoj v programovacom jazyku JAVA sa používa verzia - *JCryptool*.



Obrázok 1: Použitie nástrojov Cryptool 1, 2

2.2 Vytvorené softvérové nástroje a video
Pri výučbe sa využívajú softvérové nástroje, vytvorené v rámci bakalárskych a diplomových prác, napr. aplikácia – *Crypto*, alebo aplikácia pre klasickú kryptografiou a kryptoanalýzu (Obrázok 2).



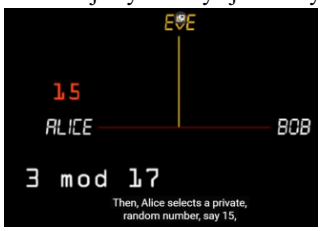
Obrázok 2: Softvérové nástroje vytvorené študentmi

Video, resp. video *tutorialy* vysvetľujú veľmi dobre matematicky koncept kryptografických algoritmov. Veľmi vhodné sú videa z Khanovej školy, alebo prednášky odborníkov.

3 Realizácia výučby

Výučba predmetu komunikačná bezpečnosť prebieha v 3. ročníku bakalárskeho štúdia v odboroch automatizácia a telekomunikácie na fakulte Elektrotechniky a informačných technológií. Realizuje sa prostredníctvom prednášok, cvičení a laboratórnych cvičení.

Prednášky sú realizované v 3 hodinových blokoch klasickým spôsobom formou prezentácií. Veľmi vhodným doplnením sú videá a prednášky, ktoré pomáhajú vysvetliť hlavne matematické princípy (Obrázok 3 Video Diffie Hellman algoritmus). Na niektorých prednáškach je využívaný aj názorný softvér.

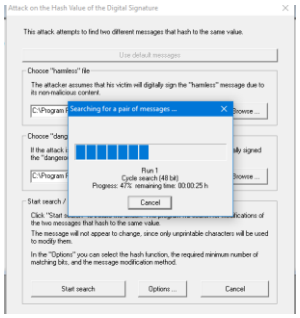


Obrázok 3: Video Diffie Hellman – Khanová škola

Cvičenie vedú samotní študenti usmerňovaní učiteľom. Každý študent má pripravenú konkrétnu tému - referát na vybraný kryptografický algoritmus, alebo inú tému z oblasti kryptografie. Študent má svojim spolužiakom názorne, ale aj matematicky vysvetliť základný princíp algoritmu (šifrovanie, dešifrovanie, kľúčový mechanizmus). Zároveň má doplniť nové poznatky, napríklad útoky na danú šifru, jej použitie a pod. Úlohou študenta, ktorý vedie cvičenie je zapojiť spolužiakov prostredníctvom pripravených úloh na praktické riešenie.

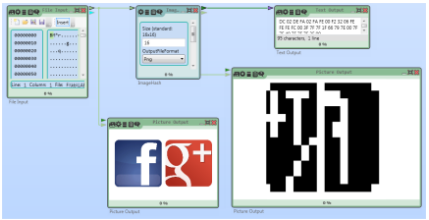
Laboratórne cvičenia tvoria tretiu, najpraktickejšiu časť predmetu komunikačná bezpečnosť. Študenti dostanú zadanie laboratórneho cvičenia podľa ktorého postupujú. Zadanie je rozdelené do niekoľkých bodov a každý bod špecifikuje softvér, postup a úlohu ktorú má študent splniť. Základným softvérovým nástrojom je nástroj *Cryptool* vo svojich verziách 1 a 2. Študent šifruje, dešifruje v danej šifre, alebo realizuje formálne útoky.

-



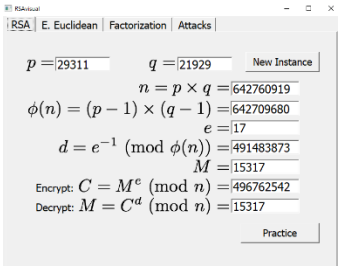
Obrázok 4 Modelovanie útoku na digitálny podpis v *Cryptool 1*

V nástroji *Cryptool 2* je možné vytvárať kryptografické schémy algoritmov.



Obrázok 5 Schéma v *Cryptool 2* na vytvorenie hashu obrázku

Na Obrázku 5 je viditeľná schéma na vytvorenie hashu z obrázku. Podobne sú využívané aj ostatné nástroje, napr. pri laboratórnom cvičení, ktoré sa venuje asymetrickej kryptografií sa používa jednoduchá aplikácia *WinRSA* z univerzity Michigene, [6].



Obrázok 6 RSA algoritmus v nástroji *WinRSA*

Štruktúra laboratórnych cvičení je vedená tak, aby si študenti prakticky vyskúšali jednotlivé algoritmy a vedeli využiť v praxi.

4 Záver

V dnešnej informačnej dobe, pri zavádzaní IOT – Internetu vecí je veľmi potrebné vedieť nastaviť bezpečnosť dát a siete. Z tohto dôvodu, budúci odborníci, inžinieri by mali dobre ovládať problematiku kryptografie a bezpečnosti. Výučba kryptografie nie je jednoduchá a vyžaduje poznanie matematiky a informatiky. Pri výučbe je veľmi vhodný každý nástroj (softvér, video), ktorý pomôže k lepšiemu a názornejšiemu vysvetleniu kryptografických princípov na strane učiteľa a k lepšiemu pochopeniu na strane študenta. Úspechy správnej výučby, ako aj svoje vedomosti študenti neskôr potvrdia vo svojej odbornej praxi.

BIBLIOGRAFICKÉ ODKAZY

[1] DOSTÁL, Jiří. Učební pomůcky a zásada názornosti. Olomouc: Votobia, 2008. ISBN 978-80-7220-310-9.

[2] MALACH, Josef. Základy didaktiky. Ostrava: Ostravská univerzita v Ostravě, 2003. ISBN 80-7042-266-1. Kapitola Zásady procesu vyučování - učení se.

[3] Esslinger, Bernhard (22 September 2016). "CrypTool: An Open-Source E-Learning Project for Cryptography and Cryptanalysis" (PDF). Gesellschaft fuer Informatik, Crypto Day at SAP. This presentation delivers an overview. University of Siegen. https://www.cryptool.org/images/ctp/documents/CrypTool-Project_Crypto-Day_Walldorf_2016-09_v09.pdf

[4] Algorithms: Functions in different CrypTool versions Within a table all cryptographic functions implemented are listed. This list can be adjusted using categories, search phrases and by specifying the CT versions.

[5] The best way, IT security magazine, HAKING, 2011 no. 3 [online]. Dostupné na: https://www.cryptool.org/images/ctp/documents/ct_hackin9_en.pdf [cit. 2019-02-12].

[6] Dr. Ching-Kuang Shene, Cryptography Visualization Tools (under construction). Dostupné na: http://pages.mtu.edu/~shene/NSF-4/index.html [cit. 2019-02-13].

Využitie didaktických pomôcok v mimoškolskej činnosti
The Use of Tangible Learning Aids in the Extracurricular Activity

Nika Klimová
Univerzita Konštantína Filozofa
Tr. A. Hlinku 1
949 74 Nitra
Slovensko
nika.klimova@ukf.sk

Patrik Voštinár
Univerzita Mateja Bela
Tajovského 40
974 01 Banská Bystrica
Slovensko
patrik.vostinar@umb.sk

Jarmila Škrinárová
Univerzita Mateja Bela
Tajovského 40
974 01 Banská Bystrica
Slovensko
jarmila.skrinarova@umb.sk

Dana Horváthová
Univerzita Mateja Bela
Tajovského 40
974 01 Banská Bystrica
Slovensko
dana.horvathova@umb.sk

ABSTRACT
The contribution mainly deals with the robotics which is used at the extracurricular activity for pupils from lower-secondary schools in Banská Bystrica. There are future Computer Science teachers and students of Applied Informatics involved as lecturers. The aim is to give our future Computer Science teachers and students of Applied Computer Science opportunity to teach and gain the experience with kids especially in order to teach them programming. The project lasts the second academic year and is very popular among people from the department and kids from primary schools as well. As turned out, based on our experience, tangible learning aids are key elements when motivating and teaching programming and Computer Science as well. The contribution describes our experience with the following: Scottie Go!, mBot Ranger, Sparkfun moto:bit, BBC micro:bit IoT, App Inventor, Codeybot, Phiro Pro Unplugged.

Keywords
Learning aid. Extracurricular activity. Scottie Go. MBot Ranger. BBC micro:bit. App Inventor. Codeybot.

ABSTRAKT
Článok popisuje najmä robotiku na informatickom krúžku pre deti z druhého stupňa základných škôl. Budúci učitelia informatiky a študenti aplikovanej informatiky sú na krúžku lektormi sprevádzajúci žiakov. Hlavným cieľom je poskytnúť budúcim učiteľom a študentom aplikovanej informatiky možnosť vzdelávať a nadobúdať skúsenosti so žiakmi zamerané hlavne na programovanie. Projekt trvá už druhý akademický rok (od školského roku 2017/2018) a je populárny nielen medzi členmi katedry informatiky, FPV, UMB, ale aj medzi samotnými žiakmi zo základných škôl. Na základe našich skúseností sa ukázalo, že materiálne didaktické pomôcky sú kľúčovými elementmi pri vyučovaní programovania, ale takisto informatiky vo všeobecnosti. Článok sa zaoberá a popisuje nasledovné didaktické pomôcky: Scottie Go!, mBot Ranger, Sparkfun moto:bit, AlphaBot2, BBC micro:bit IoT, App Inventor, Codeybot, Phiro Pro Unplugged.

Kľúčové slová
Učebná pomôcka. Mimoškolská činnosť. Scottie Go. MBot Ranger. BBC micro:bit. App Inventor. Codeybot.

1 ÚVOD
Učebná pomôcka sa definuje ako didaktický nástroj riadenia vyučovacieho procesu a regulácie samotného učenia [1]. Ide zároveň o prostriedok, ktorý je určený na dokonalejšie, rýchlejšie a komplexnejšie osvojenie učiva s využitím zásady názornosti. Pokiaľ ide o materiálnu, tzv. hmatateľnú učebnú pomôcku, príkladom v informatike môže byť samotný robot, tá zahrňuje nielen názornú funkciu, ale i informatívnu, formatívnu, inštrumentálnu, motivačnú, systematizujúcu či racionálnu. Význam učebnej pomôcky vo výchovno-vzdelávacom procese, ale aj v mimoškolskej činnosti, je značne opodstatnený.

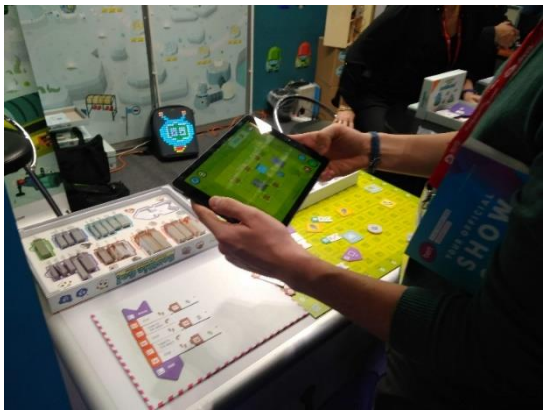
2 INFORMATICKÝ KRÚŽOK
Informatický krúžok na Katedre informatiky, Fakulte prírodných vied, Univerzity Mateja Bela v Banskej Bystrici, začal fungovať od šk. roku 2017/2018 [2]. Vtedy sme na základe grantu od Komenského inštitútu zakúpili Airblock Drone, ktorý je programovateľný a uspôsobený detskému používateľovi. Na základe týchto skúseností, ktoré boli jednoznačne pozitívne, sa nám osvedčilo priniesť na informatický krúžok rôznych robotov. Zistili sme, že žiaci majú záujem programovať, najmä vtedy, keď svoj výsledok môžu chytiť, napríklad v prípade LEGO robotov.

Keďže krúžok bol primárne určený na to, aby predstavoval bezpečné prostredie, kde môžu budúci učitelia informatiky zažiť svoje vlastné vyučovanie, a tak sa zdokonaľovať vo svojej budúcej profesii, rozhodli sme sa, že budeme pokračovať v získavaní robotov a technického príslušenstva. Jednak túto myšlienku ocenili budúci učitelia, jednak naša katedra si myslí, že keď budeme mať široké spektrum učebných pomôcok, aj priestor na sebarealizáciu nielen budúcich učiteľov, ale aj samotných žiakov, sa zvýši. Na základe týchto ohlasov sme sa v šk. roku 2018/2019 zapojili do projektov a obdržali granty, na základe ktorých sme okrem iného kúpili:

- v rámci Nadácie Orange eŠkoly pre budúcnosť 2018/2019:
  - 3x tablet Lenovo Tab 4 8palcový,
  - 1x STEM robot Codeybot,
  - príslušenstvo pre micro:bit (IoT príslušenstvo - expansion board, programovateľné ovládača (gamepad), crowtail base shield, sparkfun moto:bit, obaly na microbit)
  - 1x programovacia stolová hra Scottie Go! EDU,
- grantová agentúra KEGA a projekt KEGA č. 009KU-4/2017 „Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní“ (2018-2019):
  - 2x Phiro Pro Unplugged,
  - 1x mBot Ranger -Transformable STEM Educational
  - 2x AlphaBot2
  - príslušenstvo pre micro:bit (breadboards, prepojovacie káble)
  - 2x programovacia stolová hra Scottie Go! EDU,

2.1 Scottie Go! EDU
Edukačná verzia hry Scottie Go! predstavuje spojitosť s mobilnou aplikáciou Scottie Go!, ktorú sme využili na tabletoch. Od

ostatných programovacích aplikácií sa líši v tom, že má fyzické skladacie blokové príkazy, ktoré sa podobajú na štýl puzzle. Úloha v tvare hracej plochy sa objaví na tablete, príkazy si žiak vlastnoručne poskladá napr. na stole, tabletom zoskenuje kód a na základe toho sa mu vykoná program v aplikácii. Poľský výrobca vyrába verziu aj v slovenskom jazyku. Zahŕňa 91 úloh, materiály pre učiteľov a 179 hmatateľných blokov [3]. Prezentácia tejto hry bola zaradená aj na najväčšej svetovej konferencii na svete v oblasti vzdelávania a technológií Bett 2019 (obr. 1).



Obrázok 1 Scottie Go! na konferencii Bett 2019

Na informatickom krúžku sme využili implementáciu tejto stolovej hry a 2 tabletov pre 2 žiakov 6. ročníka ZŠ. Keďže ju nemáme dlho, žiaci zatiaľ prechádzajú len prvý svet.

2.2 Phiro Pro Unplugged

Programovateľný robot pochádzajúci z Indie a známy pod názvom Phiro Pro Unplugged je určený najmä pre 2. stupeň ZŠ a s nadstavbou aj pre SŠ [4]. Umožňuje viacero programovacích módov, z ktorých, na základe našich skúseností, si žiaci vyberajú hlavne mód tzv. swish kariet (obr. 2). Na základe metodík vytvorených počas uplynulého školského roku pracujeme so žiakmi na aktivitách Detekcia tváre s Pocket Code či Ovládanie robota pomocou gyroskopu.

Obrázok 2 Phiro Pro – ukážka programovania.

2.3 Codeybot

Codeybot s aplikáciou mBlockly App umožňuje rozvíjať algoritmické myslenie v blokovom móde. Vyznačuje sa tým, že má tvár, a teda dokáže „vyjadrovať emócie“ [5]. Okrem toho dokáže aj tancovať, pretože má 5 prednastavených tancov. Robot je trojkolkou, ktorá sa pohybuje na rovnakom princípe ako segway – je samobalančný. Dokáže reagovať na zvukové povely, pokiaľ sú vyslovené anglicky. Z našej skúsenosti sa o codeybot zaujímali hlavne dievčatá, nazdávame sa, že dôvodom sú funkcie ako tanec alebo znázorňovanie tváre.

2.4 mBot Ranger

Ako predchádzajúci Codeybot, aj mBot Ranger pochádza z dielne Makeblock. Ide o pokročilejšiu formu programovacieho robota mBot [6]. mBot produkty sú akreditované fínskou agentúrou pre vzdelávacie štandardy. Je určený pre STEM metódu, teda žiaci si robota môžu postaviť aj sami a programovať ho v blokovom prostredí.

2.5 App Inventor

App Inventoru venujeme značnú časť informatického krúžka hlavne v tomto letnom semestri. App Inventor umožňuje programovanie mobilných aplikácií pre operačný systém Android. Programuje sa pomocou blokov kódov. Pri programovaní využívame zakúpené tablety, ale žiaci si nosia aj vlastné mobilné telefóny. Programujú na počítačoch a cez QR si v tabletoch a mobilných telefónoch spúšťajú program.

2.6 Programovacie micro:bit autíčka

Robot s mikrokontrolérom moto:bit sme skonštruovali do formy auta. Dá sa programovať v prostredí Microsoft MakeCode. My sme využili aj gamepad, ktorý komunikuje s moto:bitom a na základe toho ho vieme diaľkovo ovládať. Gamepad musia žiaci naprogramovať a pomocou rádia odoslať do programovateľného autíčka.

2.7 BBC micro:bit a IoT príslušenstvo

BBC micro:bit je už známym mikrokontrolérom pre vyučovanie programovania. Na základe príslušenstva IoT však dokážeme riešiť zaujímavejšie problémy. Zároveň táto sada zaujala hlavne starších žiakov, ktorí už majú skúsenosti s elektronikou. Do tejto aktivity sa zapájajú najmä budúci učitelia informatiky a techniky.

POĎAKOVANIE

Príspevok bol spracovaný ako súčasť projektu KEGA č. 009KU-4/2017 „Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní“ a tiež vďaka programu e-Školy pre budúcnosť Nadácie Orange.

BIBLIOGRAFICKÉ ODKAZY

[1] KOŽUCHOVÁ, M. et al. *Elektronická učebnica didaktiky technickej výchovy.* [online]. Bratislava : Univerzita Komenského, 2011. Dostupné na: http://utv.ki.ku.sk/ ISBN 978-80-223-3031-2.

[2] KLIMOVÁ, N. et al. *Čo ak vám katedra informatiky umožní realizovať vaše sny?* [online]. In: Didinfo 2018.

[3] Bett. *Scottie Go! – the first game in the world for the youngest learners of coding [online]. Dostupné na:* https://www.bettshow.com/bett-press-releases/scottie-go-the-first-game-in-the-world-for-the-youngest-learners-of-coding

[4] KLIMOVÁ, N. HORVÁTHOVÁ, D. *Robot Phiro ožíva v rukách detí.* [online]. In: Didinfo 2018.

[5] Makeblock. *MBot Ranger*. [online]. Dostupné na: https://www.makeblock.com/steam-kits/mbot-ranger

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN ..... ISSN .....

Prieskum o výučbe informatiky na 1. stupni ZŠ na Slovensku v kontexte súťaže iBobor

Survey on Teaching Informatics at Primary Schools in Slovakia in the Context of the iBobor Competition

Lucia Budinská
KDMFI FMFI UK
Mlynská dolina 1
84248 Bratislava
SR
lucia.budinska@fmph.uniba.sk

Karolína Mayerová
KDMFI FMFI UK
Mlynská dolina 1
84248 Bratislava
SR
mayerova@fmph.uniba.sk

Monika Tomcsányiová
KDMFI FMFI UK
Mlynská dolina 1
84248 Bratislava
SR
tomcsanyiova@fmph.uniba.sk

ABSTRACT
In this article, we present the results from the questionnaire sent to the first-grade teachers who registered their pupils in the Slovak Bebras contest (iBobor) in 2018. The aim of the survey was to find out who teaches Informatics, in which grades and what their lessons involves. The results presented in this article represent our endeavour, as task authors, to better understand the current state of informatics education at elementary school and thus create better contest tasks.

Keywords
Survey. Elementary school. Bebras contest.

ABSTRAKT
V článku predstavujeme prehľad výsledkov z dotazníka, ktorý sme poslali učiteľom z prvého stupňa, ktorí prihlásili svojich žiakov do súťaže iBobor v roku 2018. Zisťovali sme, kto informatiku učí, v akom rozsahu a s akým obsahom. Našou snahou, ako autorov úloh, bolo lepšie porozumieť aktuálnemu stavu výučby informatiky na prvom stupni ZŠ a tak vytvárať lepšie súťažne úlohy.

Kľúčové slová
Prieskum. Prvý stupeň ZŠ. Súťaž iBobor.

1 ÚVOD
Ako organizátori súťaže iBobor, každý rok analyzujeme dosiahnuté výsledky žiakov v súťaži [1,2] a kontrolujeme tak napr. odhad obtiažnosti úloh. Tohtoročná analýza výsledkov žiakov 1. stupňa poukázala na pomerne slabé vedomosti z niektorých oblastí. Rozhodli sme sa preto zistiť, aké témy, s akými materiálmi a v akom rozsahu preberajú učitelia na prvom stupni.

2 DOTAZNÍK
Zaujímalo nás, kto učí informatiku na 1. stupni, aké materiály sa využívajú na výučbu, aké témy sa preberajú a podobne. Vytvorili sme preto online dotazník, ktorý sa zameriaval práve na tieto otázky a rozposlali sme ho učiteľom, ktorí prihlásili do súťaže žiakov z 1. stupňa, čo znamená kategórie Drobci (2. a 3. ročník), Bobríci (4. a 5. ročník). Dokopy ich bolo 754, z toho sa nám vrátilo 270 odpovedí. Dotazník sme vytvorili s pomerne málo otázkami, aby neodradil učiteľov od vyplnenia.

V nasledujúcej časti uvádzame spracované výsledky z dotazníka a zhrnutie z nich.

3 Výsledky
Výsledky uvedené v tejto kapitole neboli spracované štatisticky, ale ide o údaje popisnej štatistiky. Každá z nasledujúcich podkapitol reprezentuje výsledky jednej otázky v dotazníku.

3.1 Kto učí
V jednej z prvých otázok sme sa pýtali, kto učí informatiku na 1. stupni ich školy. Odpovede boli rôzne. Pozitívne hodnotíme, že takmer v 80% išlo o učiteľov 1. stupňa, ktorí buď malo alebo nemali kvalifikáciu pre výučbu informatiky (predpokladáme, že sa vo svojom vysokoškolskom vzdelávaní stretli s didaktikou informatiky pre 1. stupeň ZŠ, respektíve absolvovali ĎVUi [3] alebo iné doplňujúce vzdelávanie). V 23% išlo o učiteľov z 2. stupňa, výnimočne vychovávateľku či učiteľa SŠ.

3.2 V ktorých ročníkoch
Od zavedenia inovovaného Štátneho vzdelávacieho programu [4] prestala byť informatika povinná v druhom ročníku základnej školy a na prvom stupni sa tak má učiť iba v treťom a štvrtom ročníku. Keďže z kontaktov s učiteľmi vieme, že na niektorých školách vyučujú informatiku aj v nižších ročníkoch, chceli sme zistiť, ako sú tieto školy zastúpené v súťaži.

Pre každý ročník mali učitelia možnosť označiť jednu z odpovedí: *neučíme, 1 hod/týždenne, 2 hod/týždenne, iná časová dotácia* (napríklad 1 hodina/týždenne počas jedného polroka a podobne). Na 27 školách z 270 odpovedí sa informatika učí už v prvom ročníku, v druhom ročníku učí informatiku 87 škôl. Zaujímavé je, že v treťom ročníku neučia informatiku 2 školy a v štvrtom ročníku jedna. Všetky tieto tri školy vyplnili iba jeden ročník, v ktorom sa u nich učí informatika, nevieme ale povedať, prečo. Objavili sa aj školy, v ktorých sa učia na prvom stupni 4 hodiny informatiky, dokonca aj školy, v ktorých je 6 hodín (1-1-2-2) informatiky.

	1. ročník	2. ročník	3. ročník	4. ročník
neučia	243	183	2	1
1 hod/týždenne	23	83	266	267
2 hod/týždenne	0	0	2	2
iná časová dotácia	4	4	0	0

3.3 Učebné materiály
Ďalšia otázka, ktorá nás zaujímala, bola otázka „Aké materiály učitelia používajú na vyučovanie informatiky na 1. stupni?“ V dotazníku sme túto otázku položili ako otvorenú, t. j. učiteľ mal

slovami uviesť názov materiálov, resp. odkaz na materiály, ktoré používa. Viacerí učitelia uviedli aj viac ako jeden zdroj materiálov. Po spracovaní odpovedí sme zistili, že školy najčastejšie používajú pracovné zošity z informatiky od vydavateľstva AITEC (82 odpovedí). Druhý najčastejšie využívaný zdroj, ktorý uvádzali, je platený prístup na portál Akadémia Alexandra (38 učiteľov). Učebnicu Tvorivá informatika, ktorá je síce schválenou učebnicou, ale pre žiakov na druhom stupni, využíva už na prvom stupni 36 učiteľov. Vlastné materiály si vytvára 35 učiteľov. 21 učiteľov uviedlo, že používajú „rôzne webové stránky". Knihu Informatická výchova uviedlo 17 učiteľov, portál zborovna.sk 14 a ovce.sk 11 respondentov. Menej ako 10 učiteľov uviedlo webové stránky: Infovekáčik, Scratch, Baltík, či materiály z Ďalšieho vzdelávania učiteľov, pozri [3]. Z týchto materiálov sa nám ako najzaujímavejšia zdá stránka p. učiteľky Szlamkovej, pozri [5], ktorá obsahuje veľa námetov a aj interaktívnych prostredí, ktoré si žiaci môžu priamo vyskúšať.

3.4 Softvér

Keďže sme predpokladali, že veľa učiteľov využíva vlastné materiály, respektíve viacero zdrojov, pýtali sme sa ich aj na softvér, v ktorom učia. Niektoré úlohy zo súťaže iBobor sa totiž zameriavajú aj na digitálnu gramotnosť. Najviac nás zaujímalo, aký je najčastejšie používaný grafický editor a aké programovacie prostredia učitelia so žiakmi používajú. Z grafických editorov bol s 90% na prvom mieste Skicár, resp. Paint. Takmer 68% učiteľov používa RNA a asi 63% Tuxpaint. Za zmienku stojí ešte Logomotion, ktoré používa 43% respondentov. Približne 10% respondentov označilo Pivot-a a 3D-Skicár.

Na otázku *Aké nástroje využívate na výučbu oblasti Algoritmické riešenie úloh?* takmer 47% učiteľov vybralo odpoveď Baltík, takmer 40% mali Imagine a EasyLogo, okolo 30% mali aktivity bez použitia počítača a online edukačné hry. Takmer 30% mal Scratch a okolo 10% GalaxyCodr a robotické stavebnice.

Tretia otázka sa týkala využívania archívu súťaže iBobor, prípadne prostredia Boborovo. Asi 23% opýtaných ho na vyučovaní nepoužíva. Ostatní opýtaní ho používajú s rôznou frekvenciou. Viacerí učitelia uviedli, že archív používajú hlavne pred súťažou. Veľmi príjemne nás prekvapili odpovede, v ktorých niektorí učitelia uviedli, že ho využívajú v pravidelných intervaloch (napr. raz za týždeň, či raz za mesiac).

3.5 Témy

Keďže v súťaži iBobor sa úlohy snažíme prispôsobovať tomu, čo sa žiaci na slovenskej informatike učia (samozrejme, s najväčším dôrazom na logické a algoritmické myslenie), zaujímalo nás, ktorým témam z iŠVP sa učitelia venujú a kedy (keďže každej z tém sa môžu učitelia venovať vo viacerých ročníkoch, otázka ponúkala možnosť zvoliť viac odpovedí). Celkové výsledky sú v grafe na obrázku 1. Písanie a úprava textu, práca s obrázkami a práca s počítačom sa učí na všetkých školách. Najviac škôl neučí šifrovanie a kódovanie a tabuľky a štruktúry, z oboch týchto oblastí sa v súťaži iBobor vyskytuje viacero úloh. Mailová komunikácia, ktorá je súčasťou iŠVP, sa neučí na 51 školách. Mnohí potom do voliteľnej odpovede uviedli ako dôvod, že nemajú možnosť oficiálne vytvárať žiakom mailové kontá. Viacerí preto uviedli alternatívne spôsoby elektronickej komunikácie (školské elektornické žiacke knižky a pod.)

4 ZÁVER a diskusia

Na záver by sme chceli zdôrazniť, že prieskum sa netýkal učiteľov z celého Slovenska, ale len učiteľov, ktorí sa zapojili do súťaže iBobor. Zaujímavý bol paradox, kedy viacerí učitelia zvolili možnosť, že nevyučujú tému *programovanie, algoritmické myslenie*, a zároveň napísali, že učia v Baltíkovi alebo EasyLogu. Ponúka sa teda otázka, nakoľko rozumejú, resp. nerozumejú obsahu ŠVP. Druhou zaujímavosťou bolo, že 3 zo 4 uvedených vychovávateliek, ktoré učia informatiku, vyučujú aj oblasť Algoritmické riešenie problémov.

BIBLIOGRAFICKÉ ODKAZY

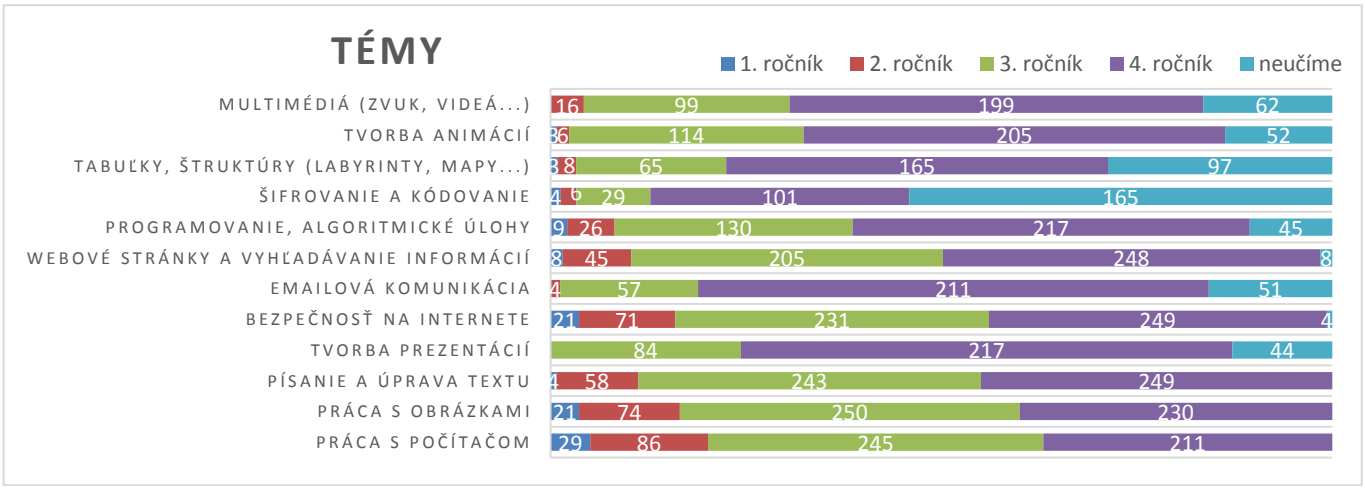
[1] BUDINSKÁ, L., MAYEROVÁ, K., VESELOVSKÁ, M. Bebras task analysis in category little beavers in Slovakia. In *International Conference on Informatics in Schools:* (pp. 91-101). Springer, Cham. 2017

[2] TOMCSÁNYIOVÁ, M., TOMCSÁNYI, P. Analýza riešení úloh súťaže iBobor v kategórii Benjamíni v šk. roku 2012/13. In Didinfo 2013. Banská Bystrica : Univerzita Mateja Bela.

[3] ĎVUI. [online]. [vid. 20. 2. 2019]. http://www.statpedu.sk /sk/o-organizacii/projekty/projekt-dvui/

[4] *iŠVP.* [online]. [vid. 20. 2. 2019]. Dostupné na: http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_nsv_2014.pdf

[5] Szlamková, E. [online]. [vid. 20. 2. 2019]. Dostupné na: http://www.ucitelka.wz.sk/prvy-stupen.html



Obrázok 1 - Graf s odpoveďami na otázku "V ktorom ročníku sa venujete tejto téme?"

Programovanie trochu inak s použitím mikrokontroléra Arduino

Little bit different programming with microcontroller Arduino

Miroslav Melicherčík
Katedra informatiky
FPV UMB Tajovského 40
974 01 Banská Bystrica
Slovensko
miroslav.melichercik@umb.sk

Michal Vagač
Katedra informatiky
FPV UMB Tajovského 40
974 01 Banská Bystrica
Slovensko
michal.vagac@umb.sk

Patrik Voštinár
Katedra informatiky
FPV UMB Tajovského 40
974 01 Banská Bystrica
Slovensko
patrik.vostinar@umb.sk

Michal Povinský
Katedra informatiky
FPV UMB Tajovského 40
974 01 Banská Bystrica
Slovensko
michal.povinsky@umb.sk

ABSTRACT

Education in field of algorithm design and programming skills is basic and inevitable part of all computer science oriented study programs. However, it is usually focused on development of software products in a form of desktop, web or mobile application. Our goal is to motivate students through microcontroller programming, which gives them deeper insight into the world of hardware and leads to better understanding of principles of computing technologies. In this article, we present realized workshop for computer science students at our department. Within them students had opportunity to work in groups, design and build some interesting projects based on Arduino platform.

Keywords

Programming. Microcontroller. Arduino. Education. Eduscrum.

ABSTRAKT

Vzdelávanie v oblasti algoritmizácie a programátorských zručností je základnou a neoddeliteľnou súčasťou všetkých informaticky orientovaných študijných programov. Toto vzdelávanie však spravidla býva orientované na vývoj softvérových produktov v podobe aplikácií pre desktop, web alebo mobilné zariadenia. Našim cieľom je motivovať študentov pomocou programovania mikrokontrolérov, čo im umožní hlbší pohľad do sveta hardvéru a následne lepšiemu pochopeniu samotných princípov fungovania výpočtovej techniky. V príspevku prezentujeme realizáciu workshopov pre študentov informatiky na našom pracovisku. V rámci nich mali študenti možnosť v skupinách navrhnúť a vytvoriť zaujímavé projekty na báze platformy Arduino.

Kľúčové slová

Programovanie. Mikrokontrolér. Arduino. Vzdelávanie. Eduscrum.

1. ÚVOD

Mladá generácia prichádza do styku s informačnými technológiami už od detského veku a preto k ním pristupujú zväčša ako používatelia a konzumenti. Toto v nich môže vyvolávať dojem, že tieto technológie tu boli odjakživa a sú samozrejmosťou. Opak je však pravdou a len hlbší pohľad na ich fungovanie im umožní pochopiť rozdiel medzi ich používateľom a tvorcom. Našou snahou je rozšíriť pohľad študentov tak, aby na tieto technológie pozerali nielen ako ich používatelia a konzumenti, ale aj ako ich potenciálni tvorcovia a návrhári.

Na Katedre informatiky, Fakulty prírodných vied, Univerzity Mateja Bela sme zrealizovali dva workshopy s názvom *Naprogramuj si vlastnú vec* pre študentov bakalárskeho stupňa študijného programu Aplikovaná informatika. Ich cieľom bolo umožniť študentom nadobudnúť základné zručnosti v zostavovaní zariadení na báze mikrokontroléra Arduino, ako aj prehĺbiť ich záujem o samotné programovanie. Za týmto účelom bolo úlohou študentov navrhnúť a zrealizovať zaujímavý projekt s využitím základnej sady mikrokontroléra Arduino.

2. PRIEBEH WORKSHOPOV

Samotná realizácia workshopov bola rozdelená na dve časti. Prvá časť spočívala v oboznámení sa študentov s obsahom základnej sady mikrokontroléra Arduino obsahujúcej rôzne senzory, zobrazovacie a iné komponenty. Spoločne s hardvérom mali k dispozícii aj návodné úlohy pre ich zapojenie a programovania v prostredí Arduino IDE [1]. Workshopy boli organizované pomocou inovatívnej vyučovacej metódy eduScrum.

2.1 Metóda eduScrum

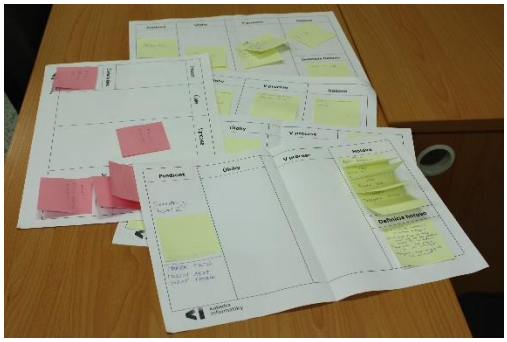
Jedná sa o výučbovú metódu, ktorá má pôvod v anglickom spôsobe vývoja Scrum. Jej aplikácie je možné nájsť v rôznych oblastiach vzdelávania [2,3]. Táto metóda reflektuje požiadavky praxe IT firiem a prispieva k rozvoju soft-skills (komunikácia v tíme, kooperácia, kreativita a kritické myslenie) u študentov. Metóda umožňuje študentom vytvárať a plánovať úlohy a následne sledovať ich postup. Na rozdiel od konvenčných metód je úloha pedagóga trochu odlišná. Pedagóg určuje riešený problém a dôvody k jeho riešeniu, naopak samotný postup riešenia je v rukách študentov, ktorí vďaka spolupráci musia sami nájsť riešenie. Toto prispieva k zvyšovaniu ich vnútornej motivácie, osobnému rastu a lepším výsledkom. Kľúčovými aspektmi rozvíjanými touto metódou sú kreativita, kolaborácia, komunikácia a kritické myslenie [4].

2.1.1 Prvé stretnutie

Počas prvého stretnutia boli študenti rozdelení do skupín, v ktorých bolo ich úlohou vytvoriť návrh zariadenia a jeho funkcionalít. V metóde eduScrum je síce odporúčaný ideálny počet členov skupiny 4, avšak nakoľko našim cieľom bolo, aby mal každý z tímu možnosť participovať na projekte ako pri jeho hardvérovej tak aj softvérovej realizácií. To bolo dôvodom vytvorenia skupín pozostávajúcich len z troch členov. Metóda eduScrum využíva na dekompozíciu problému tzv. scrumovú nástenku znázornenú na obrázku 1. Na scrum nástenkách pripravili študenti sadu jednotlivých úloh a zadaní, ktoré bude potrebné splniť a tiež

definovať jednoznačné kritérium, kedy je možné úlohu považovať za splnenú. Nástenka obsahuje stĺpce:

- **predmet** – na ktorom sa vyučuje v rámci eduScrum metódy,
- **úlohy** – zoznam menších úloh, na ktorých sa ešte nezačalo pracovať,
- **v procese** – úlohy na ktorých sa pracuje, ale ešte nie sú ukončené,
- **hotovo** – dokončené úlohy,
- **definícia hotovo** – študenti si sami určia, kedy sa dá považovať nejaká úloha za hotovú.



Obrázok 1: Ukážka scrum nástenky

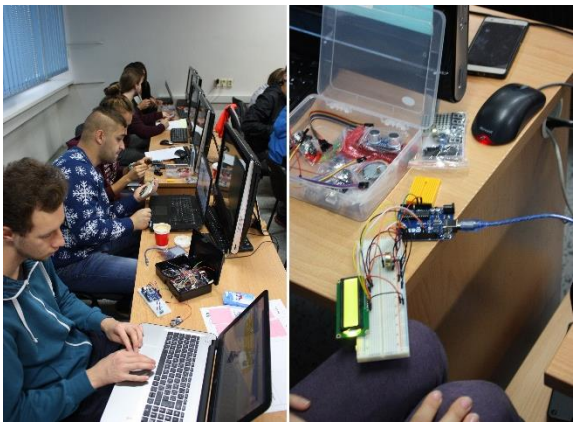
Priebeh prvého stretnutia pozostával s nasledujúcich častí:

- rozdelenie do 3 členných tímov (učitelia vybrali vedúcich tímov – tzv. Scrum Master-ov, ktorí si sami zvolili členov tímu),
- vymyslenie nápadu (brainstorming) v rámci každého tímu,
- dekompozícia problému (rozdelenie na menšie časti) a samoštúdium práce s Arduinom.

2.1.2 Druhé stretnutie

Počas druhého stretnutia prebiehalo samotné napĺňanie pridelených úloh v skupinách a postupné dokončovanie zadaných úloh.

Ukážky z priebehu práce je možné vidieť na obrázku 2.



Obrázok 2: Ukážky z priebehu workshopu

Priebeh druhého stretnutia pozostával z nasledujúcich častí:

- vytváranie projektov na workshopoch,
- demo hodina (každý tím predstavil svoje riešenie učiteľovi a zvyšným tímom, na prezentáciu mal každý tím max. 5 minút),
- retrospektíva (zhodnotenie riešenia učiteľom a zhodnotenie individuálne v rámci každého tímu).

3. ZHODNOTENIE WORKSHOPOV

Počas realizácie workshopov sa študentom podarilo postaviť a naprogramovať niekoľko efektných zariadení na báze mikrokontroléra Arduino. Tieto boli v závere odprezentované a zhodnotené. Na obrázku 3 je možné vidieť niekoľko ukážok.

Obrázok 3: Ukážky výsledných zariadení

Po skončení workshopov ich mali študenti možnosť zhodnotiť formou krátkeho dotazníka. Zaujímalo nás, či už študenti mali možnosť programovať podobné zariadenia, či už v oblasti robotiky alebo Internetu vecí. Či bol workshop pre nich zaujímavý a prínosný a či by sa podobného podujatia zúčastnili opäť. Väčšina študentov vyjadrila pozitívny názor na uvedené otázky.

ZÁVER

V príspevku opisujeme realizáciu workshopov pre študentov Katedry informatiky FPV UMB, ktorých cieľom bolo zvýšiť ich zručnosti v zapájaní hardvérových komponentov stavebníc na základe mikrokontroléra Arduino a ich programovaní. Sekundárnym cieľom bolo zvýšiť ich motiváciu k programovaniu ako takému a zlepšiť soft-skills študentov.

POĎAKOVANIE

Realizácia týchto workshopov bola možná vďaka podpore iniciatívy Meet and Code a podpore projektu KEGA 011UMB-4/2017.

BIBLIOGRAFICKÉ ODKAZY

[1] Arduino IDE. https://www.arduino.cc/en/main/software

[2] CARDOSO, M., DE CASTRO, A. V., ROCHA, A. Integration of virtual programming lab in a process of teaching programming EduScrum based. In *2018 13th Iberian Conference on Information Systems and Technologies* (CISTI). IEEE, 2018. p. 1-6. ISBN 978-9-8998-4348-6.

[3] FERREIRA, E.P., MENDONCA, J., NICOLA, S. Eduscrum Methodology in Mathematical Engineering Education. In *11th International Technology, Education and Development Conference*. Valencia : Iated, 2017, p. 2997-3000. ISBN 978-84-617-8491-2.

[4] eduScrum. http://eduscrum.nl/en/about-eduscrum

Workshopy

Workshops

Vzbuďme v študentoch chuť programovať! Aktivity s BBC micro:bit

Awake the Student's Enthusiasm for Programming! Activities with BBC micro:bit

Magdaléna Bellayová
Gymnázium, Golianova 68, Nitra
Golianova 68
949 01, Nitra
Slovensko
magdalena.bellayova@studentstc.sk

Martin Cápay
Katedra informatiky FPV UKF v Nitre
Trieda A. Hlinku 1
949 01, Nitra
Slovensko
mcapay@ukf.sk

ABSTRACT
Physical computing allows students to develop concrete, tangible products. The great benefit of "getting physical" is a holistic view of computer science which encourages creativity, promotes learning by doing and engaging the whole mind and body. The workshop provides the basics and benefits of integrating physical computing in education and its programming with block-based language and MicroPython using a microcontroller BBC micro:bit. We think that using BBC micro:bit is a suitable decision for those who want to learn to program with a powerful device. The main objective of the workshop is to show a set of tasks presenting the concept of sequence, variable, random number and conditions using physical computing in such a way that pupils need to change the sitting activity by playing activity or hands-on activity realized besides. We will use the basic commands and statements, discuss the methodical approach of how we start to teach physical computing at different students age.

Keywords
micro:bit, programing, collaborative work

ABSTRAKT
Programovanie hardvéru umožňuje študentom vyvíjať konkrétne, hmatateľné produkty. Veľkou výhodou „fyzického zapojenia" je holistický pohľad na informatiku, ktorý podporuje kreativitu, podporuje učenie tým, že robí a zapája celú myseľ a telo. Obsahom workshopu budú základy integrácie fyzickej výpočtovej techniky do vzdelávania a jej programovanie s blokovým jazykom a MicroPython-om pomocou mikrokontroléra BBC micro:bit. Myslíme si, že použitie BBC micro:bit je vhodným rozhodnutím pre tých, ktorí sa chcú naučiť programovať s malým ale výkonným zariadením. Hlavným cieľom workshopu je ukázať súbor úloh predstavujúcich koncepciu sekvencie, premenných, náhodných čísel a podmienok využívajúcich programovateľný hardvér tak, aby sa striedala činnosť žiakov pri počítači aj mimo neho. Budeme používať len základné príkazy programovacieho jazyka a zároveň diskutovať o vhodnosti implementácie mikrokontroléra do vzdelávanie u študentov rôzneho veku.

Kľúčové slová
micro:bit, programovanie, kolaboratívna práca

1 ÚVOD
Čo keby ste mali riešenie na oživenie vyučovania programovania? Riešenie, ktoré svieti, reaguje na podnety, alebo dokonca posiela správy? Máme pre vás pripravenú ochutnávku práce s programovateľným mikropočítačom BBC micro:bit, s prezentáciou praktických tipov na rôzne aktivity pre študentov.

Programovanie mikropočítačov so senzormi je v posledných rokoch, veľmi populárne. Mnohé programovateľné zariadenia nie sú však úplne použiteľné na hodinách informatiky na ZŠ, a to najmä kvôli nutnosti komplikovaného zapojenia hardvérových prvkov. S príchodom mikropočítača BBC micro:bit, ktorý bol špeciálne vyvinutý pre edukačné účely, sa však situácia výrazne zmenila. Micro:bit je malý programovateľný mikrokontrolér, ktorý má priamo na obvodovej doske integrovaný procesor, pamäť, displej s diódami, dve programovateľné tlačidlá, senzor intenzity svetla, akcelerometer, magnetometer, bluetooth anténu a vstupno-výstupné piny. Pomocou krokosvoriek je možné pripojiť prídavné zariadenia, rozširujúce jeho funkcionalitu. Je programovateľný cez blokový jazyk, MicroPython alebo Javascript.

Micro:bit je silný nástroj v oblasti „physical computing", kde sa programujú malé mikropočítače s rôznymi senzormi. Žiak, programátor, tam nepracuje iba so vstupmi a výstupmi na obrazovke, ale rieši reálne problémy. Programy už nie sú len texty a čísla. Programy zaznamenávajú dáta získané zo svojho vlastného okolia (teplota, vlhkosť, pohyb) a vhodným spôsobom reagujú. Využívanie programovateľného hardvéru umožňuje prepájať informatiku s inými predmetmi napr. s umením, matematikou, či fyzikou. Žiakov tak vo výsledku môžeme motivovať k programovaniu prostredníctvom riešenia hmatateľných projektov s využitím propagovateľného hardvéru so senzormi (meteostanica, sledovacie zariadenie, zavlažovacia stanica, detektor pohybu a pod.).

Na workshope si účastníci vyskúšajú overené aktivity, ktoré sú nastavené tak, aby od účastníkov v niektorej fáze riešenia problému vyžadovali istú formu spolupráce a tímovej práce. Našim cieľom je, aby sa striedala ich činnosť pri počítači a mimo neho, aby sa vyžadovala kolaboratívna práca a aby programovanie bolo aj zábavou. Aby ich činnosť bola v istom momente považovaná skôr za hru ako vzdelávaciu aktivitu. Naprogramujeme si hraciu kocku, hru zlepšujúcu matematické vedomosti, budeme hľadať svoju dvojičku a zakódujeme hudbu a možno stihneme navrhnúť aj kolaboratívneho smajlíka [1][2].

BIBLIOGRAFICKÉ ODKAZY
[1] CÁPAY M., BELLAYOVÁ M. Aktivity s BBC micro:bit podporujúce kolaboratívnu prácu žiakov ZŠ. Didinfo 2019.

[2] CÁPAY M., KLIMOVÁ N. Dokáže programovanie hardvéru zaujať študentov? Didinfo 2019.

Osvedčené úlohy na krúžku robotiky pre LEGO Mindstorms EV3

Angelika Haneszová
Ústav informatiky PF UPJŠ
Jesenná 5
041 54 Košice
Slovensko
angelika.haneszova@upjs.sk

ABSTRACT
During the workshop we show and try out programs for controlling LEGO robots - proven and tried-and-tested tasks in practice. We will program tasks to control the ultrasonic, color, and gyroscopic sensors, optimize and improve the solutions, and also we will show other options for extending concrete tasks.

Keywords
LEGO Mindstorms EV3 Robotic Kit. Robotics for Kids. Didactics of Informatics.

ABSTRAKT
Počas workshopu ukážeme a vyskúšame programy pre ovládanie LEGO robotov - overené a vyskúšané úlohy v praxi. Naprogramujeme úlohy na ovládanie ultrasonického, farebného a gyroslopického senzora, vyladíme a ukážeme ďalšie možnosti rozšírenia konkrétnych zadaní.

Kľúčové slová
Robotická Stavebnica Lego Mindstorms EV3. Detská Robotika. Didaktika informatiky.

ÚVOD
Robotická stavebnica LEGO Mindstorms Education EV3 je výborným a osvedčeným nástrojom pre žiakov na objavovanie poznatkov a zručností o konštrukcii vlastných modelov a programovania v detskom programátorskom prostredí. Práca s LEGO stavebnicou rozvíja manuálne a technické zručnosti, bádateľské spôsobilosti a taktiež aj dimenzie informatického myslenia a aj schopnosť tímovej spolupráce a komunikácie.

1 PRIEBEH WORKSHOPU

Úlohy uvedené na workshope budú zamerané na objavovanie a experimentovanie s LEGO robotickými súčiastkami, ktoré smerujú žiakov k systematickej analýze a návrhu programov pre ovládanie robotických modelov. Kladieme dôraz na rôznorodosť konštrukčných a programátorských riešení, tým motivujeme žiakov ku kreatívnej a samostatnej tvorbe. Programátorské úlohy v detskom grafickom programátorskom prostredí počas workshopu budeme riešiť pomocou vopred pripravených robotov zo stavebnice LEGO Mindstorms, ale budeme mať k dispozícii aj stavebnice na doplnenie konštrukčných riešení. Účastníci budú mať možnosť pracovať v dvojčlenných tímoch.

Plánované úlohy boli overené na krúžku robotiky, ktoré sa zrealizovali v rámci projektu IT Akadémia – vzdelávanie pre 21. storčie v Košiciach.

Uvedieme aj ďalšiu sadu úloh pre začiatočníkov, ktoré môžu slúžiť učiteľom ako pomôcka počas vyučovania LEGO robotiky alebo počas krúžkovej činnosti.

Predstavíme aj metodiky, ktoré sa vytvárajú v rámci projektu a sú venované programovaniu robotov tak, aby žiaci pochopili ako fungujú autonómne fungujúce roboty, ako sa ovládajú na základe senzorických meraní, aby vedeli navrhnúť v ktorých oblastiach života by bolo užitočné a účelné ich použiť.

2 PLÁN UKÁŽKOVÝCH ÚLOH

2.1 Úloha o strážcovi zámku.
Naprogramujeme robota, ktorý sa prechádza okolo hradu vytvoreného zo štvorcovej krabice s konštantným otáčaním.
Potom vylepšíme a otestujeme otáčanie v rohu zámku a narovnanie počas priameho pohybu okolo stien použitím gyroskopického senzora.
Ukážeme aj rôzne možnosti rozšírenia úlohy, napríklad keď hrad má obdlžníkový alebo nepravidelný základ. Analyzujeme použitie vhodných senzorov pre navigáciu robota, prediskutujeme možnosti nastavení parametrov.

2.2 Úloha o rybolove.
V ohraničenom jazere (izolačnou páskou ohraničená oblasť) robot rybár má loviť ryby, detegovať voľne nalepené jednofarebné tvary v oblasti jazera.
Postupne rozšírime úlohu, pridáme spočítanie nájdených rybiek a počet vypíšeme na display, pridáme rybky viacerých farieb, pridáme možnosť riadenia robota rybára pomocou tlačidiel, atď.

Uvedieme aj rôzne žiacke riešenia na konkrétne úlohy, námety pre ďalšie rozšírenie úloh a taktiež odporúčania na zostavenie sady konštrukčných a programátorských úloh pre žiakov.

Cieľom workshopu je poskytnúť záujemcom námety na úlohy, ktoré sú vhodné na prácu so žiakmi základných škôl hlavne v neformálnom vzdelávaní, ale dajú sa použiť aj v rámci vyučovania na hodinách informatiky. Počas praktických ukážok vytvoríme priestor aj na zdielanie skúseností a nápadov v danej tematike.

Tvorba medzipredmetových projektov pomocou mikropočítača BBC micro:bit

Designing interdisciplinary projects with the BBC micro:bit

Marek Mansell
FIIT STU BA
Ilkovičova 2
842 16, Bratislava 4
Slovensko
marek.mansell@gmail.com

ABSTRACT

Usage of hardware platforms in education enables creation of interdisciplinary projects, which demonstrate the practical applications of hardware components in common electronic devices and their inner workings. An important aspect of these activities is increased motivation, stimulated not only by natural interest of students in interactive electronic devices, but also by combining hardware and programming with other subjects in STEAM approach. We find the educational platform BBC micro:bit combined with the graphical programming environment MakeCode and programming language MicroPython to be a suitable choice for the creation of interdisciplinary projects on primary and secondary schools. The workshop will focus at connecting external sensors and actors to the BBC micro:bit and their usage in standard subjects of primary and secondary schools. We will introduce a set a of interdisciplinary projects which combine programming of hardware with knowledge from other fields.

Keywords

micro:bit, programming, interdisciplinary projects, STEAM

ABSTRAKT

Využívanie hardvéru vo výučbe umožňuje tvorbu medzipredmetových projektov, ktoré názorne demonštrujú využitie elektronických zariadení v každodennom živote a ich vnútorné fungovanie. Nemálo dôležitým aspektom týchto aktivít je zvýšená motiváciu žiakov, či už z dôvodu prirodzeného záujmu o elektronické pomôcky, ktoré so žiakmi interagujú, alebo z dôvodu prepojenia hardvérových projektov s inými predmetmi pomocou prístupu STEAM. Zariadenie BBC micro:bit v kombinácii s programovacím blokovým jazykom MakeCode a jazykom MicroPython je vhodnou voľbou pre tvorbu takýchto projektov na základných a stredných školách. Workshop bude zameraný na pripájanie externých senzorov a aktorov k zariadeniu BBC micro:bit a ich využití v štandardných predmetoch základných a stredných škôl. Predstavený bude súbor projektov, ktoré prepájajú programovanie hardvéru so znalosťami z iných vedných disciplín.

Kľúčové slová

micro:bit, programovanie, medzipredmetové projekty, STEAM

ÚVOD

Na workshope si predstavíme koncept tvorby jedinečných medzipredmetových projektov pomocou zariadenia BBC micro:bit a naplno využijeme technické aj didaktické možnosti tohto zariadenia.

Prvým predmetom, s ktorým prepojíme informatiku je fyzika - koncept fungovania jednoduchých elektrických obvodov, zapojenie LED diód, rezistorov a potenciometrov, deliču napätia, zmeranie vybitia tužkovej batérie. Ďalej prejdeme k biológii, kde budeme merať vlhkosť v kvetináči a spínať mini čerpadielko. Prejdeme aj ku geografii - meranie salinity vody a jej vplyv na morské prúdy či domáca meteorologická stanica so senzormi dažďa, svetla a teploty.

Experimentovať budeme aj s ľudským telom - dokážeme micro:bitom zistiť, či je elektricky vodivé? A keďže dnes je už všetko bezdrôtové, premeníme si micro:bit na sieťovú kartu počítača, pomocou ktorej budeme na diaľku merať a zobrazovať na počítači hodnoty z ďalších micro:bit zariadení. Nezabudneme ani na nevidiacich, ktorým pomôžeme špeciálnou elektronickou pomôckou na nalievanie nápoju do šálky. Mladých špiónov zas môžu zaujať elektrické "míny", nástražné drôty či domáci alarm so senzorom pohybu (presne taký, aký sa používa v klasických alarmoch). Nakoľko dôležitým aspektom hardvérových projektov je aj kreatívne a tvorivé myslenie, aktivity integrujú prístup STEAM.

Okrem získavania interdisciplinárnych znalostí aktivity taktiež zvyšujú u žiakov digitálnu gramotnosť a demonštrujú využitie technológií v každodennom živote. Veľký dôraz bude kladený na analógiu medzi tvorenými projektami a praktickým využitím danej technológie v bežne používaných elektronických zariadeniach.

Na workshope budeme pracovať v online blokovom prostredí MakeCode a v Python prostredí Mu. Aby sme sa mohli naplno zamerať na tvorbu a význam medzipredmetových projektov, odporúčame, aby účastníci už mali predchádzajúce skúsenosti so základmi práce v týchto prostrediach. V prípade, že nemáte, odporúčame najprv absolvovať workshop "Vzbuďme v študentoch chuť programovať! Aktivity s BBC micro:bit" od M. Bellayovej a M. Cápay, kde sa naučíte ako zariadenie programovať. [1][2].

BIBLIOGRAFICKÉ ODKAZY

[1] SPY O.Z. Metodické materiály pre BBC micro:bit https://www.ucimeshardverom.sk/materialy/

[2] MANSELL M. Platforma micro:bit v nižšom sekundárnom vzdelávaní. ŠVK FMFI UK 2018.

DIDINFO 2019 Univerzita Mateja Bela, Banská Bystrica
ISBN 978-80-557-1533-9, ISSN 2454-051X.

Využitie Raspberry Pi na hodinách informatiky
Using Raspberry Pi in Computer Science Lessons

Zuzana Tkáčová

Prírodovedecká fakulta UPJŠ
Jesenná 5
040 01 Košice
Slovensko
zuzana.tkacova1@upjs.sk

Pedagogická fakulta UKF
Dražovská cesta 4
949 74 Nitra
Slovensko
zuzana.tkacova@ukf.sk

ANOTÁCIA

Raspberry Pi Foundation [1] mala úmysel vytvoriť flexibilný, spoľahlivý, ľahko prístupný a malý počítač ako učebnú pomôcku, ale aj ako hračku pre ľudí, ktorých koníčkom sú informačné technológie (IT). Benefitom pôvodného Raspberry Pi bola nízka maloobchodná cena, ako aj veľký potenciál vo využití pri nízkonákladových projektoch, čo prilákalo nielen študentov ale mediálnu tlač, ako aj nadšencov v oblasti IT. Pôvodný model Raspberry Pi 1 prešiel odvtedy niekoľkými generačnými obmenami – cez Raspberry Pi 2, Raspberry Pi 3 až po Raspberry Pi Zero, ktorý slúži na konkrétne industriálne využitie alebo po upravení ako produkt tretej strany, preto disponuje menším výkonom a na konštrukciu je použité menšie množstvo materiálu, vďaka čomu je Raspberry Pi Zero v súčasnosti najlacnejším zo serie Pi.

Raspberry Pi je mikroprocesorová stavebnica vytvorená primárne pre vzdelávacie účely, ale jej popularita rastie aj v praxi. Praktická skúsenosť s mikroprocesorovou stavebnicou je užitočná pre ďalšie štúdium, aj pre prax v IT a v iných odboroch. Mikroprocesorová stavebnica Raspberry Pi je dobrým fyzickým modelom dnešných počítačov. Preto bol Raspberry Pi 3 model B+ zvolený aj ako základná technická pomôcka pre výučbu informatiky na gymnáziách v rámci Národného projektu IT Akadémia – vzdelávanie pre 21. storočie.

Raspberry Pi 3 model B+ disponuje štvorjadrovým 64-bitovým procesorom (Broadcom BCM2837B0 ARM Cortex A53) s frekvenciou 1.4 GHz,1GB LPDDR2-900 SDRAM (900MHz)a MicroSDHC slotom (max. 32GB). Na jednej doske obsahuje taktiež 802.11ac Wireless LAN, Bluetooth 4.2 (vrátane Bluetooth Low Energy), 1000Mbps Ethernetový port, 4x USB 2.0, HDMI výstup a Broadcom VideoCore IV @ 400 MHz / 300 MHz, OpenGL ES 2.0 (24 GFLOPS), MPEG-2 a VC-1 (s licenciou), 1080p30 H.264 / MPEG-4 AVC dekodér a kodér s vysokým profilom. K dispozícii je aj 26 portov GPIO v štandardnej 40-pinovej konfigurácii Pi. Dodávaný je s operačným systémom Raspbian - upravená distribúcia Linuxu založená na Debiane.

V rámci Národného projektu IT Akadémia – Vzdelávanie pre 21. storočie je priame využitie Raspberry Pi plánované na 6 vyučovacích hodinách, kde sa žiaci budú môcť najprv oboznámiť s jeho základným hardvérovým a softvérovým vybavením. Následne spoznajú možnosti programovania hudby v prostredí Sonic Pi [2]. Dôležitou súčasťou práce je programovanie hardvérových komponentov v Pythone – najprv práca s Raspberry Pi kamerou na záznam fotografií a videa, potom ukážka knižnice GPIO na obsluhu vstupných a výstupných pinov. Formou jednoduchého datalogera si môžu vyskúšať aj výstup dát do súboru a ich čítanie zo súboru. Posledná pripravená metodika predstaví Raspberry Pi ako nástroj pre budovanie jednoduchej sieťovej infraštruktúry. Takto spracované metodické materiály môžu ukázať učiteľom možnosti využitia Raspberry Pi ako nástroja na podporu výučby predovšetkým v tematických okruhoch informatiky (ISCED 3a) [3] Princípy fungovania IKT, Komunikácia prostredníctvom IKT, ale aj Postupy, riešenie problémov, algoritmické myslenie.

V rámci workshopu ukážeme zapojenie Raspberry Pi a prostredie operačného systému Raspbian. Predstavíme si prostredie Sonic Pi a niektoré ukážky programovania v Pythone.

POĎAKOVANIE

Tento článok bol vytvorený v rámci národného projektu IT Akadémia – vzdelávanie pre 21. storočie, ktorý sa realizuje vďaka podpore z Európskeho sociálneho fondu a Európskeho fondu regionálneho rozvoja v rámci Operačného programu Ľudské zdroje.

BIBLIOGRAFICKÉ ODKAZY

[1] RASPBERRY PI (oficiálna stránka). https://www.raspberrypi.org

[2] SONIC PI. https://sonic-pi.net

[3] ŠTÁTNY VZDELÁVACÍ PROGRAM, *Informatika – vyššie stredné vzdelávanie.* [online]. Bratislava: Štátny pedagogický ústav, 2014 [cit. 2019-01-30]. 18 s. http://www.statpedu.sk/files/articles/dokumenty/inovovany-statny-vzdelavaci-program/informatika_g_4_5_r.pdf

Kódování obrázků s Micro:bitem
Image coding with Micro:bit

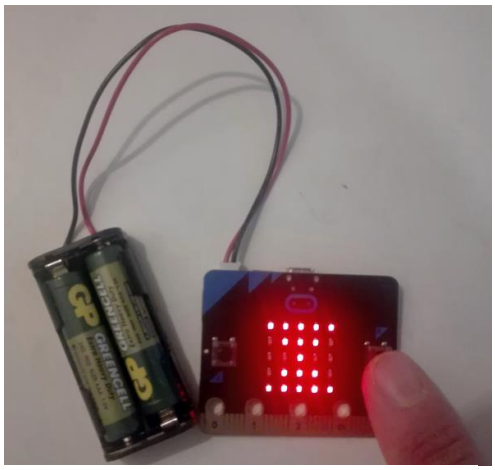
Jiří Vaníček
Jihočeská univerzita
Jeronýmova 10
37115 České Budějovice
Česko
vanicek@pf.jcu.cz

ABSTRACT
We do not intend to program Micro:bits at this workshop. We will only use them as attractive educational tool for unplugged activities of simple image coding and transfer. We will try several types of coding and play on an artificial Internet, the same as we have tested with 5-th grade pupils. Maybe we'll get to send encrypted messages.

Keywords
Unplugged informatics, image data coding, Micro:bit.

ABSTRAKT
Na tomto workshopu nebudeme Microbity programovat, jen je využijeme jako atraktivní motivační pomůcku pro unplugged aktivitu, jednoduché kódování obrázků a jejich přenos. Vyzkoušíme různé kódování a zahrajeme si na umělý internet, stejně jako jsme to my zkoušeli se žáky 5. ročníku ZŠ. Možná se dostaneme až k posílání zašifrovaných zpráv.

Klíčová slova
Informatika bez počítače, kódování obrázkových dat, Micro:bit.



Obrázek 1

WORKSHOP
Programovatelná jednočipová deska Micro:bit je určena k výuce programování a základů robotiky pro začátečníky. Kromě jeho požití ve výuce programování v blokovém prostředí Makecode nebo v JavaScriptu lze využít jeho zabudované senzory, konkrétně dvě dotyková tlačítka, a matici 5 x 5 červených LED diod, k aktivitám se žáky, při nichž se neprogramuje, ale rozvíjí poznání z jiných oblastí informatiky. Pokud se při těchto aktivitách neprogramuje, je možné je využít i u mladších žáků, na 1. stupni ZŠ. Takovéto aktivity, při nichž Micro:bit slouží spíše jako motivační pomůcka a usnadňuje žákovi kontrolu poskytováním zpětné vazby, ukazuje tento workshop.

Kódování a dekódování obrázkových dat bývá ve výuce často realizováno na matici bodů, které jsou označovány jako rozsvícené nebo zhasnuté. Namísto čtverečkovaného papíru a tužky je tu možné použít Micro:bit, jeho matici LED diod a tlačítka A, B. Učitel může předem naprogramovat nebo využít software, který takovou činnost simuluje.

Tento software obsahuje 2 hry na úvodní obrazovce svítící pod písmeny L, P. Spouští se tlačítky, úvodní obrazovka se opět zobrazí po restartu zařízení.

Hra P - kreslení obrázku jako stopy po pohybu robota Karla: blikající bod, na začátku stojící natočen směrem na sever, na stisk tlačítka A vykoná KROK, na stisk B vykoná VPRAVO VBOK. Současný stisk tlačítek A+B přepne robota do kreslicího režimu, kdy za sebou zanechává čáru a tím kreslí obrázek. Zatřesení Micro:bitem se hra spustí znovu.

Hra L – kreslení obrázku po řádcích. Kurzor běhá po řádcích zleva doprava, na stisknutí tlačítka poskočí na další místo. Tlačítko A generuje svítící bod, tlačítko B zhasnutý bod. Stisk A+B spustí hru znovu.

Realizovatelné výukové aktivity (jejichž ukázky proběhnou na workshopu) :

1. Podle hry L nakresli svůj vlastní obrázek
2. Podle hry L vytvoř obrázek dle zadání
3. Podle hry P zakresli svůj vlastní obrázek
4. Nakresli obrázek podle dodaného kódu
5. Nakresli svůj obrázek, zakóduj jej, kód si vyměň s někým jiným a jeho obrázek dekóduj
6. Vymysli jiný způsob kódování obrázku

Workshop: Uvádzame vedu do života s robotikou LEGO® Education WeDo 2.0

Workshop: Bringing Science Alive with the Robotics LEGO® Education WeDo 2.0

Mária Večeríková
EDUXE Slovensko s.r.o.
M.C.Sklodowskej 2
851 04 Bratislava
Slovensko
eduxe@eduxe.sk

Jana Ferencová
EDUXE Slovensko s.r.o.
M.C.Sklodowskej 2
851 04 Bratislava
Slovensko
eduxe@eduxe.sk

ABSTRACT
In this paper is described a workshop that focuses on simple modeling and programming of robots for pupils of school age from the age of seven. It uses the LEGO® Education WeDo 2.0 aid which is also the second generation of WeDo robotics.

Keywords
LEGO® Education WeDo 2.0. Robotics. Programming. EDUXE.

ABSTRAKT
V tomto článku opíšeme workshop, ktorý je zameraný na jednoduché modelovanie a programovanie robotov pre žiakov školského veku od siedmich rokov. Používa sa v ňom didaktická pomôcka LEGO® Education WeDo 2.0, ktorá je zároveň druhou generáciou robotiky WeDo.

Kľúčové slová
LEGO® Education WeDo 2. Robotika. Programovanie. EDUXE.

ÚVOD

LEGO® Education WeDo 2.0 prináša realitu jednoduchého modelovania a programovania robotov do nižších tried základných škôl . LEGO® Education WeDo 2.0 nielen inšpiruje a podporuje žiacku predstavivosť v oblasti vedy a techniky, ale prináša tiež rozvoj kritického myslenia, schopností riešiť problémy a v neposlednom rade podporuje komunikáciu a spoluprácu žiakov v triede. Kombináciou konštrukčných dielov LEGO s počítačovým rozhraním, senzormi a motormi, ovládanými užívateľsky príjemným programovacím prostredím, spoločne so žiackymi projektami a metodickými materiálmi pre učiteľov zvyšuje záujem žiakov o vedu a techniku.

1 OBSAH WORKSHOPU

Vzdelávacie prostredie s WeDo 2.0 otvára množstvo otázok a poskytuje priestor k ich skúmaniu a hľadaniu odpovedí.

Vzdelávacie hodnoty:

- Skúmanie, modelovanie, riešenie problému
- Simulácia reálneho vedeckého výskumu
- Základy programovania
- Spolupráca s prezentáciou výsledkov
- Kritické myslenie pri riešení problémov

Softvér (aplikácia) WeDo 2.0 v sebe integruje programovacie prostredie, užívateľský manuál, žiacke projekty a nástroj k tvorbe dokumentácie z projektov. Softvér je vhodný pre prácu v triede, existujúce lekcie, ktoré sú zaradené do štandardov, môžu sa jednoducho integrovať do učebných osnov. Programovacie prostredie je užívateľsky príjemné, intuitívne, ednoduché, technicky ľahko ovládateľné pre oživenie žiackych LEGO modelov.

Projekty sú vyvíjané učiteľmi, témy sú vytvárané tak, aby žiaci pochopili povahu vedy. WeDo 2.0 povzbudzuje žiakov k tomu, aby používali aspekty vedeckého objavovania pri riešení skutočných STEM problémov. STEM je vzdelávací koncept, v ktorom sa prirodzene spájajú štyri vedecké oblasti Science (prírodné vedy), Technology (technika), Engineering (technológie) a Mathematics (matematika).

Použitím LEGO kociek, senzorov a motorov môžu žiaci využiť toto riešenie na zapálenie ich tvorivosti, rozvíjanie kritického myslenia, preskúmanie kariérnych možností a jednoduché získavanie skúseností so STEM.

Podľa vášho želania si môžete vybrať operačný systém. Podporované platformy IOS, Chrome, Windows, Mac, Android. Okrem aplikácie LEGO® Education WeDo 2.0 môžeme pracovať aj v kódovacích jazykoch Scratch, Open Roberta a Tynker.

LEGO® Education WeDo 2.0 poskytuje jednoduchý koncept prostredníctvom ktorého môže učiteľ oživiť výučbu základov programovania a robotiky vo svojej triede. Na workshope sa účastníci zoznámia s týmto tvorivým vzdelávacím konceptom, pričom budú priamo pracovať s prostredím cez overené zadané úlohy.

BIBLIOGRAFICKÉ ODKAZY

[1] EDUXE. Ponuka LEGO® Education pre mladší školský vek. https://www.eduxe.sk/mladsi-skolsky-vek

[2] LEGO® Education WeDo 2.0 https://www.LEGOeducation.com/downloads

[3] LEGO® Education WeDo 2.0 https://www.LEGOeducation.com/lessons

[4] LEGOeducation.com/wedo2systemrequirements

[5] LEGOeducation.com/compatibility

Ukážka robotov z informatického krúžku pre žiakov 2. stupňa ZŠ

Samples of Robots from the Extracurricular Activity of Lower-secondary Schools

Patrik Voštinár
Univerzita Mateja Bela
Tajovského 40
741 01 Banská Bystrica
Slovensko
patrik.vostinar@umb.sk

Nika Klimová
Univerzita Konštantína Filozofa
Tr. A. Hlinku 1
949 74 Nitra
Slovensko
nika.klimova@ukf.sk

Jarmila Škrinárová
Univerzita Mateja Bela
Tajovského 40
741 01 Banská Bystrica
Slovensko
jarmila.skrinarova@umb.sk

Dana Horváthová
Univerzita Mateja Bela
Tajovského 40
741 01 Banská Bystrica
Slovensko
dana.horvathova@umb.sk

ABSTRACT

Workshop shows teaching aids used for education in the field of programming. We present samples of robots and tasks which are used at the extracurricular activity. Robots and microcontrollers which are presented are: mBot Ranger, Raspberry Pi, Arduino, Scottie Go, Sparkfun moto:bit, Gamepad with BBC micro:bit, CodeyBot with Apple iPad, Phiro Pro. There will be some programming activities which you can try.

Keywords

Extracurricular activity. Robotics. Phiro Pro. BBC micro:bit. MBot Ranger. Arduino. Scottie Go.

ABSTRAKT

Workshop prezentuje učebné pomôcky a ukážkové úlohy používané k vyučovaniu programovania. Ukážeme časť robotov, ktorých využívame počas mimoškolskej činnosti. Didaktické robotické pomôcky a mikrokontroléry, s ktorými budete mať možnosť pracovať, sú: mBot Ranger, Scottie Go, Sparkfun moto:bit, Gamepad with BBC micro:bit, CodeyBot s Apple iPad, Phiro Pro, micro:bit IoT sada. Na workshope nájdete niekoľko aktívit spojené s programovaním, ktoré si budete môcť vyskúšať.

Kľúčové slová

Mimoškolská činnosť. Robotika. Phiro Pro. BBC micro:bit. MBot Ranger. Scottie Go.

1 ÚVOD

Workshop je postavený na základe učebných pomôcok využívaných na programovanie, ktoré boli zakúpené primárne na Informatický krúžok na Katedre informatiky, Fakulte prírodných vied, Univerzity Mateja Bela v Banskej Bystrici. Cieľom workshopu je naučiť učiteľov základnú prácu s týmito učebnými pomôckami.

2 Aktivity

Okrem samotnej realizácie krúžku sa snažíme o zapojenie sa do projektov, na základe ktorých máme finančný obnos použiteľný pre účely informatického krúžku [1]. Ide najmä o **Nadáciu Orange – eŠkoly pre budúcnosť** 2018/2019:

- 3x tablet Lenovo Tab 4 8palcový
- 1x tablet Apple iPad
- 1x STEM robot Codeybot
- 1x programovacia stolová hra Scottie Go!
- príslušenstvo pre micro:bit (IoT príslušenstvo - expansion board,programovateľný ovládač (gamepad), crowtail base shield, sparkfun moto:bit, obal na microbit)

Ale aj **grantová agentúra KEGA a projekt KEGA č. 009KU-4/2017 „Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní“ (2018-2019):**

- 2x Phiro Pro Unplugged,
- 1x mBot Ranger -Transformable STEM Educational
- 2x AlphaBot2
- príslušenstvo pre micro:bit (breadboards, prepojovacie káble)
- 2x programovacia stolová hra Scottie Go! EDU,

Či **iniciatíva Meet and Code a občianske združenie SPy (2018):**

- 20 x edukačná doska Micro:bit + príslušenstvo (LED pásiky, reproduktory, krokosvorkové káble)

Workshop bude mať formu tzv. stánkov, pri ktorých budú autori workshopu, lektori (budúci učitelia informatiky v kombinácii a študenti aplikovanej informatiky) a takisto aj 3-4 žiaci. Budú predvádzať jednotlivé robotické didaktické pomôcky a mikrokontroléry tak, ako ich používame na Informatickom krúžku. Účastníci budú mať možnosť si vyskúšať nasledujúce aktivity:

1. STEM aktivity s mBot Ranger,
2. Programovanie Scottie Go! EDU na tablete,
3. Zostavujeme auto so Sparkfun moto:bit,
4. Naprogramuj si Gamepad s BBC micro:bit,
5. Ovládame CodeyBot s Apple iPad,
6. Phiro Pro a swish karty,
7. Zapoj si micro:bit IoT sadu.

POĎAKOVANIE

Príspevok bol spracovaný ako súčasť projektu KEGA č. 009KU-4/2017 „Inovatívne metodiky v predmete informatika v sekundárnom vzdelávaní“ a tiež vďaka programu e-Školy pre budúcnosť Nadácie Orange.

BIBLIOGRAFICKÉ ODKAZY

[1] FPV UMB . *Informatický krúžok.* http://www.fpv.umb.sk/katedry/katedra-informatiky/informaticky-kruzok-umb-pre-ziakov-zs/informacie-o-kruzku.html

Organizátori

Organisers

Katedra informatiky je pevnou súčasťou Fakulty prírodných vied Univerzity Mateja Bela. Má priznané právo udeľovať titul „bakalár“ a akademický titul „magister“ absolventom dennej formy štúdia a externej formy štúdia v študijnom programe Aplikovaná informatika a aj absolventom dennej formy štúdia v študijnom programe Učiteľstvo informatiky v kombináciách, ktoré sú zriadené podľa zákonných predpisov a uskutočňované na Fakulte prírodných vied Univerzity Mateja Bela. Katedra informatiky má priznané právo udeľovať akademický titul „doktor pedagogiky“ (v skratke „PaedDr.“) a akademický titul „doktor prírodných vied“ (v skratke „RNDr.“) absolventom študijných programov, ktorí získali titul „magister“, po vykonaní rigoróznej skúšky v študijnom odbore, v ktorom získali vysokoškolské vzdelanie alebo v príbuznom študijnom odbore uskutočňovanom na Fakulte prírodných vied Univerzity Mateja Bela. Katedra informatiky vykonáva aj výučbu mnohých celofakultných predmetov a celouniverzitných predmetov, ktoré sú buď povinnými, povinne voliteľnými alebo výberovými predmetmi na katedrách FPV UMB v Banskej Bystrici, resp. na iných katedrách; aj v rámci spoločného základu.

Zameranie edukačnej, vedeckej a výskumnej činnosti Katedry informatiky FPV UMB sa odvíja od hlavnej výskumnej línie fakulty, profilácie katedry a od záujmu jednotlivých členov katedry v symbióze s potrebou obohacovať predmety v študijných programoch o najnovšie poznatky vedy a výskumu v odbore. Hlavnými smermi činnosti katedry sú:

- Teória vyučovania informatiky.
- Softvérové inžinierstvo.
- Umelá inteligencia.
- Aplikovaná informatika.

Katedru informatiky charakterizuje:

- vysoký dopyt po absolventoch našich odborov,
- široké spektrum uplatnenia absolventov v učiteľskej i odbornej praxi,
- kontakt so školami i s firmami z oblasti IT už počas štúdia,
- viacročná tradícia v príprave učiteľov a odborníkov pre oblasť informatiky,
- ústretové výučbové prostredie,
- tím skúsených akademických a vedeckých pracovníkov katedry,
- previazanosť teoretických a praktických poznatkov,
- primerane vybavené laboratóriá,
- možnosť absolvovať časť odborného štúdia na zahraničnej partnerskej univerzite.

Katedra je medzinárodne uznávaným a akceptovaným pracoviskom. Má kontakty s rôznymi domácimi aj zahraničnými pracoviskami a organizáciami, s ktorými dochádza k pravidelnej výmene učiteľov, študentov i ďalšej spolupráci v oblasti vedy a vzdelávania: Technická univerzita Ostrava, Metropolitní univerzita a ČVUT Praha, Masarykova univrzita Brno, Česká republika; Petroleum-Gas University, Ploieşti, Rumunsko; "Prof. Dr. Assen Zlatarov" University, Burgas, Bulharsko; Oulu University of Applied Sciences, Oulu, Fínsko; Atilim University, Ankara a Fatih University, Istambul, Turecko; Utrecht University, Utrecht, Holandsko; Liepaja University, Liepaja, Lotyšsko; Corvinus University of Budapest, Maďarsko; Technische Universität Chemnitz, Nemecko a ďalšie.

Na katedře najdete lidi se dvojím odborným zaměřením. Jednak jsou zda, jak název napovídá, matematici. Ti se zaměřují především na výuku matematiky aplikované v ostatních disciplínách. Jedná se především o diskrétní matematiku, lineární algebru a statistiku. Druhou skupinou jsou informatici. Ti se podílí na dalším rozvoji ICT gramotnosti studentů bez ohledu na jejich profilaci. U budoucích učitelů se zaměřujeme na zvládnutí vybraných technologií, aby pak dále měli dobrý základ pro rozvoj počítačem podporované výuky ve svých specializacích. Pro budoucí učitele informatiky pak katedra zajišťuje část odborných předmětů, jako jsou kódování a šifrování či matematika pro informatiky. Informatická část katedry se pak soustředí v rámci přípravy těchto učitelů na didaktiku informatiky, programování a počítačem podporovanou výuku. Je to pracoviště, pod nímž jsou akreditovány všechny studijní obory připravující budoucí učitele informatiky na univerzitě – jak v bakalářském, tak navazujícím magisterském stupni. Část kolegů se specializuje na GIS a jejich využití.

Členové katedry se také zapojují do komunitního života v Jednotě školských informatiků a v Jednotě českých matematiků a fyziků. Spolupodílíme se na organizaci Bobříka informatiky a Matematické olympiády. Katedra se významně podílí i na řešení různých projektů na fakultě. Mezi nejvýznamnější projekty z poslední doby patří:

- Podpora rozvíjení informatického myšlení,
- Podpora rozvoje digitální gramotnosti,
- Učitelem moderně a odborně,
- Zvýšení kvality vzdělávání žáků, rozvoje klíčových kompetencí, oblastí vzdělávání a gramotností,
- Prameny spojují krajiny a lidi.

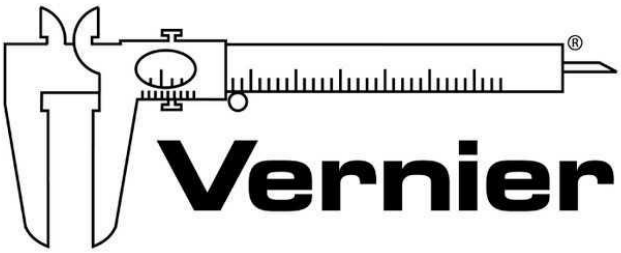
Informatická část dlouhodobě spolupracuje s Národním ústavem pro vzdělávání a má i několik dlouhodobých partnerů v zahraničí. Patří mezi ně Univerzita Mateje Bela v Banské Bystrici, se kterou spolupracujeme právě na pořádání konference Didinfo. V kontaktu jsme také s kolegy z Univerzity P. J. Šafárika v Košiciach, Univerzity Komenského v Bratislave, Vilniaus universitetas nebo Carl von Ossietzky Universität Oldenburg. Naším dlouhodbým cílem je spolupracovat na společných výzkumných projektech, což se bohužel zatím úplně nedaří vzhledem k nízkému počtu členů informatické sekce.

Sponzori a mediální partneri

Sponsors and media partners

PC REVUE

Quark
Magazín o vede a technike


Vernier

B/S/H/

EDUXE

it akadémia

Názov: **DidInfo 2019**
Editori: Ing. Dana Horváthová, PhD., RNDr. Alžbeta Michalíková, PhD.,
doc. Ing. Jarmila Škrinárová, PhD., PaedDr. Patrik Voštinár, PhD.
Vydanie: 1. vydanie
Vydavateľ: Univerzita Mateja Bela, Fakulta prírodných vied v Banskej Bystrici, Slovensko
Rok: 2019
Rozsah: 231 strán
Formát: elektronický zborník
ISBN: 978-80-557-1533-9
ISSN: 2454-051X

Title: **Proceedings of conference DidInfo 2019**
Editors: Ing. Dana Horváthová, PhD., RNDr. Alžbeta Michalíková, PhD.,
doc. Ing. Jarmila Škrinárová, PhD., PaedDr. Patrik Voštinár, PhD.
Edition: First edition
Publisher: Matej Bel University, Faculty of Natural Sciences in Banská Bystrica, Slovakia
Year: 2019
Pages: 231
Format: electronic proceedings
ISBN: 978-80-557-1533-9
ISSN: 2454-051X