VZPODBUJANJE UČENJA PROGRAMIRANJA S POMOČJO PROBLEMSKEGA PRISTOPA

Melita Kompolšek
Elektrotehniško - računalniška strokovna šola in gimnazija Ljubljana
melita.kompolsek@vegova.si

Povzetek

Računalnike uporabljamo za pomoč pri reševanju problemov. Toda preden lahko problem rešimo, ga moramo razumeti in najti načine, na katere ga lahko rešimo. To nam omogoča računalniško mišljenje. Programiranje je torej več kot le kodiranje, saj od programerja zahteva računalniško mišljenje, ki vključuje reševanje problemov z uporabo računalniških konceptov, kot sta abstrakcija in delitev problemov na manjše probleme. Poleg tega je programiranje veščina, ki jo pridobimo z veliko vaje. Da bi dijaki razvili spretnosti programiranja, potrebujejo predvsem veliko utrjevanja.

V prispevku je analiziran vpliv problemskih in motivacijskih nalog na motivacijo dijakov za reševanje nalog. V ta namen je bila izdelana spletna stran s 25 nalogami, ki bodo med seboj povezane v zgodbo. Dijak bo prejel naključne podatke, za katere bo moral sestaviti program, ki bo skladno z navodili naloge generiral rešitev. Rešitev bo vnesel na spletno stran, ki ga bo v primeru pravilne rešitve preusmerila na naslednjo nalogo. Ker se bo moral uporabnik za reševanje nalog prijaviti v sistem, bo mogoče preko spletne strani beležiti podatke posameznega dijaka o njegovem reševanju nalog. Poleg tega je v prispevku raziskano tudi, s katerimi težavami se spopadajo učitelji pri sestavljanju takih vrst nalog.

Abstract

FOSTERING PROGRAMMING WITH PROBLEM-BASED LEARNING

We use computers to help solve problems. But before we can solve a problem, we need to understand it, and find ways to solve it. This is called computational thinking. Programming is more than just coding, as it requires computational thinking from the programmer, which involves solving problems using computer concepts such as abstraction and dividing problems into smaller problems. In addition, programming is a skill that we acquire through a lot of practice. Above all, in order for students to develop programming skills, they need a lot of consolidation.

The study analyses the impact of problem-based and motivational tasks on student motivation to solve them. For this purpose, a web page with 25 tasks linked together in a story was created. The student receives input random data for which he has to compile a program that will generate a solution according to instructions. The solution is then entered in a web page and if it is correct, the user is redirected to the next task. As the user has to log into the system to solve tasks, this allows us to collect information on his task solving. In addition, this study also investigates the difficulties teachers came across while preparing these types of tasks.

Ključne besede

Programiranje, učenje programiranja, problemsko učenje, poigritev, učenje s primeri, spletna stran za učenje

Key words

Programming, learning programming, problem learning, gamification, learning with examples, e-learning

UVOD

Računalnike uporabljamo za pomoč pri reševanju problemov. Toda preden lahko problem rešimo, ga moramo razumeti in najti načine, na katere ga lahko rešimo. To nam omogoča računalniško mišljenje. Programiranje je torej več kot le kodiranje, saj od programerja zahteva računalniško mišljenje, ki vključuje reševanje problemov z uporabo računalniških konceptov, kot so abstrakcija in delitev problemov na manjše probleme. Poleg tega je programiranje veščina, ki jo pridobimo z veliko vaje. Da bi dijaki razvili spretnosti programiranja, potrebujejo predvsem veliko utrjevanja. To je eden od razlogov, zakaj njihovo zanimanje za učenje programiranja lahko upade. Drugi razlog za to, da dijaki opustijo programiranje, so lahko ponavljajoče se napake v njihovih programih ter težave pri odkrivanju in odpravljanju le-teh. Tudi to lahko dijaki v veliki meri odpravijo le z utrjevanjem. Pri načrtovanju poučevanja programiranja je torej zelo pomembno upoštevati dejavnike, ki spodbujajo učno motivacijo dijakov.

Eden od pristopov za spodbujanje učenja in utrjevanja programiranja je problemsko učenje. Gre za celostni pristop, s katerim dosegamo učinkovito in smiselno učenje, v sklopu katerega morajo dijaki problem razumeti, razviti načrt za reševanje in ga preizkusiti. Problemsko učenje velja v računalništvu za ključno kompetenco, saj na tem področju bolj kot le tehnično usmerjene aktivnosti potrebujemo dobre sposobnosti reševanja problemov. Sposobnost reševanja problemov pa je glavni primanjkljaj začetnikov.

UČENJE IN POUČEVANJE PROGRAMIRANJA

Programiranje je veščina, ki se je ne naučimo, temveč jo pridobimo z veliko vaje. Da bi dijaki razvili dobre spretnosti programiranja in pridobili izkušnje pri odpravljanju napak, potrebujejo predvsem veliko vaje [7]. Vendar pa dijaki zlahka izgubijo navdušenje in zanimanje za učenje programiranja, še posebej, če pri samostojnem utrjevanju doživljajo ponavljajoče se napake. Večina dijakov se ob učenju programiranja, zlasti v prvem letu, sooča s težavami, ob tem pa se počasi zmanjšuje tudi njihova motivacija za učenje [1]. Razočaranje se pri dijakih pojavi, ko začutijo primanjkljaj spretnosti za razvoj pravilnega računalniškega programa in nepoznavanje konceptov programiranja [16]. Pri načrtovanju poučevanja programiranja je tako nujno potrebno pozornost nameniti dejavnikom, ki vplivajo na učno motivacijo dijakov [3]. Proces učenja je dinamičen, zato na pridobivanje in širjenje znanja vplivajo različni dejavniki [8]. Poleg individualnih razlik, učne motivacije in sposobnosti dijakov lahko na tako učenje vplivajo tudi dejavniki, kot so infrastruktura, socialni pritisk in učni pristopi [9].

Pedagoška teorija je osredotočena predvsem na poučevanje dijakov [2], zato je pomembno nasloviti vrste mentalnih modelov, ki jih imajo dijaki ter proučiti, kako se ti modeli spreminjajo [15]. Izvedena je bila tudi študija [13] v kateri so avtorji proučevali poučevanje dijakov v okviru uvodnega tečaja programiranja in ugotovili, da je najbolj zanesljiv napovedovalec njihovega uspeha ocena. Drugi pomembni dejavniki so še motivacija, zaupanje v svoje zmožnosti, pozitivni čustveni odzivi in iskanje dijakovih skupnih ciljev. Eden od načinov za doseganje takšnih ciljev je uporaba atraktivnih učnih orodij, ki morajo biti privlačno oblikovana in zgrajena z upoštevanjem pedagoških spoznanj.

Tradicionalno vzgojno načelo podpira koncept prikritega učenja, pri katerem se dijake spodbuja k zabavi, ob kateri pa se hkrati tudi nezavedno učijo [14]. Učitelji poskušajo takšen način učenja

uvesti v šole, in sicer tako, da dosegajo učne cilje s pomočjo netradicionalnih orodij. Ta način učenja, v primerjavi s formalnim pristopom poučevanja in ocenjevanja, pri katerem dijaki in učitelji ustvarjajo skupne učne cilje, skrbi, da posamezni dijak razvije svoje učne cilje.

Večina današnjih otrok igra digitalne igre ali pa uporablja socialna omrežja. Pogosto tudi sodelujejo v aktivnostih, ki nagrajujejo njihovo predanost in trud. Zato so začeli učitelji, da bi zagotovili dobre učne rezultate, iskati učne cilje v učenju, z uporabo okolij, ki so dijakom blizu in v igralnih tehnologijah. Cilj takšnega učenja je, da se dijaki med igranjem poglobijo v sam problem v takšni meri, da pozabijo na samo učenje in se programskih konceptov naučijo nezavedno in nenamerno. Ta pristop temelji na predpostavki, da je najboljše učenje interaktivno, kjer je vsaka naslednja naloga malenkost težja od dijakove trenutne ravni znanja. Konstruktivistični pristop se pogosto uporablja za zagotovitev doseganja učnih ciljev, saj daje prednost učenju iz izkušenj, problemskem in projektnemu učenju [10]. Pri takšnih vrstah učenja lahko dijaki opazujejo delo sovrstnikov in z njimi sodelujejo, da bi tako izboljšali svoje sposobnosti. Takšno učenje vključuje proces delanja napak, preizkušanja in sposobnost dijakov, da z uporabo svojih preteklih in sedanjih izkušenj svoje znanje nadgradijo. Primerno oblikovano učno okolje lahko poveča notranjo motivacijo, radovednost, domišljijo in dijakom predstavlja izziv [12]. Takšno okolje lahko dijakom ponudi praktične primere nalog iz vsakdanjega življenja, s pomočjo katerih lahko eksperimentirajo in se učijo iz lastnih napak.

Pri poučevanju programiranja je eden glavnih izzivov učitelja, kako pritegniti dijake k takemu učenju in utrjevanju programiranja, ki jim bo omogočilo, da postanejo ustvarjalci in oblikovalci tehnologije, ne le njeni uporabniki. Za to potrebujemo motivacijske pristope in orodja, ki bi bila primerna za začetnike in bi zagotovila kar najučinkovitejše učne izkušnje za vse dijake.

PROBLEMSKO UČENJE

Problemsko učenje je celostni pristop za doseganje učinkovitega in smiselnega učenja. Dijaki morajo za uspešno reševanje problemov problem razumeti, zanj razviti načrt in ta načrt tudi preizkusiti. Sami morajo torej analizirati strategije, za katere mislijo, da bodo rešile problem, posledično pa bodo pri tem razvili kreativne rešitve in dosegli učinkovito učenje. Iz tega razloga je problemsko učenje kot sredstvo za spodbujanje učenja razširjeno na mn*ogih področjih, kot so znanost [11], matematika [5], oblikovanje [4]* in računalništvo. Problemsko učenje v računalništvu, ki vključuje sestavljanje računalniških programov za rešitev problema, velja za ključno kompetenco izobraževanja na računalniškem področju, saj v računalništvu bolj kot zgolj tehnično usmerjene aktivnosti potrebujemo predvsem dobre sposo*bnosti reševanja problemov [6]*. Ravno dobre sposobnosti reševanja problemov pa so glavna težava in primanjkljaj začetnikov. Začetniki so večinoma osredotočeni zgolj na površinsko znanje, primanjkuje jim tudi modelov za reševanje in zato ne razvijejo potrebnih spretnosti z*a rešitev problema [13]*. Reševanje učnih problemov pri programiranju je za začetnike težko še posebej, ker morajo razumeti in znati uporabljati koncepte za reševanje problemov, kot so zanke, pogoji, rekurzija itd. Spodbujanje razvoja sposobnosti dijakov za problemsko reševanje je torej ena glavnih tematik izobraževanja na področju računalništva.

Problemsko učenje je bilo najprej in najširše uporabljeno na področju medicine. Natančneje je ta pristop usmerjal dijake k ugotavljanju smiselnosti zbranih podatkov. Problemski pristop kot eden izmed pedagoških pristopov pa se uporablja na več področjih, ne le na področju medicine.

Učenje na podlagi problemov je koncept poučevanja, ki usmerja učenje na osnovi nekega problema, ki ga po navadi definira učitelj, da bi z njegovo pomočjo dijaki dosegli neke učne

cilje. Izhaja iz kognitivne in konstruktivistične teorije in zagovarja dejstvo, da dijaki s povezovanjem izobraževalnega gradiva in svojih življenjskih izkušenj pridobijo več. Konstruktivizem poziva k učnim konceptom, ki so izkustveni, aktivni, sodelovalni in ki razvijajo spretnosti reševanja problemov. Cilj je, da dijak ne samo pasivno absorbira in ponavlja informacije, temveč da aktivno uporablja gradivo, pri tem pa sodeluje z vrstniki in uporablja svoje pretekle izkušnje in sposobnosti reševanja problemov. Končni cilj je torej razvoj sposobnosti kritičnega mišljenja.

Dijak mora biti pri problemskem učenju aktivni udeleženec v učnem procesu. Pri tem je pomembno, da lahko dijaki razvijajo svoj tok misli in rešitev, ne da bi morali pri tem strogo slediti predpisanemu postopku, ki si ga je zamislil učitelj. Dijaki morajo iskati rešitve in razvijati poti do cilja svobodno, na svoj način. Seveda pa tudi pri tem procesu obstaja določena struktura, le da je ta nekoliko ohlapnejša in dijaku vendarle omogoča, da lahko ob pomoči učitelja raziskuje različne smeri reševanja problema. Obstaja več pedagoških pristopov, ki omogočajo ta način povezovanja, eden izmed njih je problemsko učenje.

SPLETNA STRAN HEKERSKI NAPADI

Spletna stran Hekerski napadi je dostopna na https://www.vegova.si/e-sola/hekerski-napad/login.php. Izdelana je bila s pomočjo jezika HTML (angl. Hyper Text Markup Language) in oblikovana s pomočjo jezika CSS (angl. Cascading Style Sheets). Ker se je za sodelovanje na spletni strani potrebno prijaviti in ker spletna stran preverja pravilnost uporabnikovih odgovorov, je bil za to implementacijo uporabljen jezik PHP (angl. Hypertext Preprocessor). S pomočjo phpMyAdmin je bila ustvarjena baza podatkov o uporabnikih. Za dostop strani do podatkov uporabnika in za pošiljanje posameznih podatkov spletni strani pa so bili uporabljeni stavki v jeziku SQL (angl. Structured Query Language).

S pomočjo spletne strani in njenih nalog želimo, da se dijaki vživijo v vlogo programerja, ki mora rešiti svet pred hekerskimi napadi. Pri tem pa mora pri vsakem napadu rešiti enega od problemov. To stori tako, da lokalno na svojem računalniku sestavi program, ki mu vrne pravilni rezultat glede na naključno generirane unikatne podatke, ki so mu bili dodeljeni pri nalogi. Če uporabnik nalogo reši pravilno, se izvede nov hekerski napad, ki ga mora dijak uspešno preprečiti. Pri reševanju nalog se dijak spoprijema z različnimi izzivi, ki od njega zahtevajo poznavanje osnovnih struktur programiranja, kot so pogojni stavki, zanke, tabele, funkcije in nizi.

Cilj igre je pravilna rešitev vseh 25 nalog oz. hekerskih napadov in s tem utrditi znanje s področja programiranja. Posredni cilj je torej prepričati dijake, da vztrajajo pri reševanju nalog za utrjevanje, in sicer skozi koncept problemskega učenja in koncept poigritve. Ta dva koncepta sta bila izbrana, ker je v literaturi mogoče zaslediti, da na dijake vplivata bolj motivacijsko kot pa klasičen način poučevanja programiranja. Poleg tega so bile naloge sestavljene tako, da se dijak ob njihovem reševanju prelevi v neko drugo osebo, ki rešuje privlačne probleme namesto klasičnih programerskih nalog, ki po navadi predstavljajo reševanje golih matematičnih problemov in so za dijake dostikrat nezanimivi.

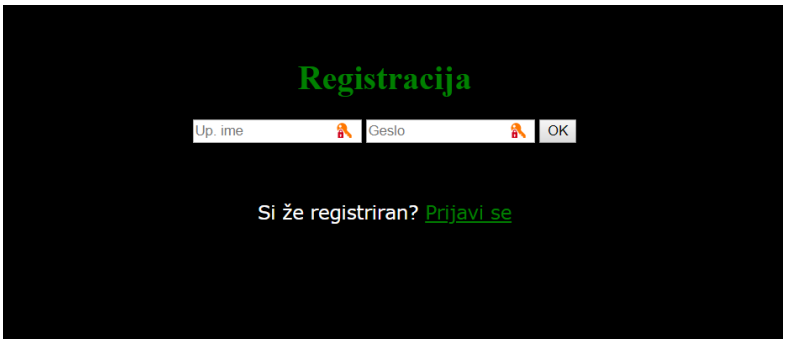
Namen in uporaba spletne strani

Spletna stran *Hekerski napad* je namenjena dijakom 2. letnika strokovne šole, smer tehnik računalništva, in tudi vsem drugim začetnikom v programiranju, ki želijo utrditi svoje znanje. Naloge s spletne strani lahko uporabimo za utrjevanje znanja posameznega sklopa pri pouku

programiranja ali za domače delo, lahko pa služijo tudi kot utrjevanje znanja ob koncu poučevanja predmeta osnove programiranja oz. pred začetkom izvajanja predmeta zahtevnejšega programiranja.

Z reševanjem nalog na tej spletni strani pa dijaki poleg algoritmičnega mišljenja, urjenja pri reševanju problemov in utrjevanja programskih struktur pridobijo tudi druge veščine, pomembne za njihov razvoj. Zgradba strani in nalog jih namreč spodbujajo k ustvarjalnemu načinu razmišljanja, k razmišljanju o tem, kako čim hitreje in čim lažje rešiti določeni problem. Pri vsem tem sta v ospredju predvsem zgodba in dizajn spletne strani, medtem ko je pravi namen strani in nalog uporabnikom prikrit.

Ob dostopu do spletne strani se mora uporabnik najprej registrirati (Slika 1). V ta namen mora določiti še neuporabljeno uporabniško ime in geslo.



Slika 1: Registracija

Ob uspešni prijavi se uporabniku za vsako od 25 nalog generirajo naključni podatki (Slika 2 in Slika 3), na podlagi teh podatkov in programov za rešitev posamezne naloge pa se mu določijo še pravilne rešitve.

	ID	username	password	naloga
Uredi Kopiraj Izbriši	3	Anonymous	7079c72c21415131774625ba1d64f4b0	1
Uredi Kopiraj Izbriši	4	PinkiePie	0fbd581b94286c8313dbbb209383d87b	1
Uredi Kopiraj Izbriši	5	Bitquark	d4685ed464121bcf31a0f06d6fa6ac49	1

Slika 2: Baza uporabnikov

	ID10	podatki10_spr	podatki10_PZ	podatki10_zakoliko	resitev10	naloga10_usersFKEY	aktivna	stevec	oddaje_cas
Uredi Kopiraj Izbriši	3	{'z', 'v', 'v', 'z', 'x', 'y', 'x', 'y', 'w', 'x',...	{'P', 'Z', 'P', 'P', 'Z', 'Z', 'P', 'P', 'Z', 'Z',...	{4, 1, 7, 3, 4, 9, 9, 6, 4, 2, 7, 8, 9, 8, 4, 7, 9...	119	3	0	0	
Uredi Kopiraj Izbriši	4	{'v', 'z', 'w', 'z', 'y', 'z', 'z', 'w', 'x', 'y',...	{'Z', 'P', 'P', 'Z', 'Z', 'P', 'Z', 'P', 'P',...	{5, 2, 1, 1, 6, 1, 3, 5, 5, 1, 9, 2, 5, 7, 3, 4, 7...	130	4	0	0	
Uredi Kopiraj Izbriši	5	{'z', 'z', 'v', 'v', 'x', 'y', 'x', 'w', 'y', 'w',...	{'P', 'Z', 'P', 'P', 'P', 'P', 'Z', 'Z', 'Z', 'Z',...	{0, 2, 6, 6, 0, 2, 5, 7, 0, 7, 5, 8, 1, 1, 9, 0, 1...	132	5	0	0	

Slika 3: Baza ene naloge

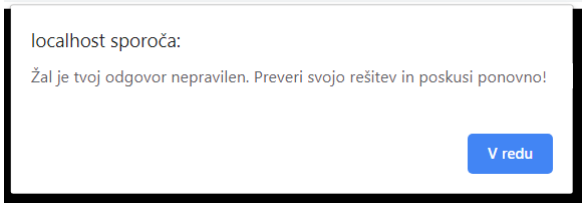
Ko je registracija zaključena, spletna stran uporabnika preusmeri na začetno stran, kjer je predstavljena uvodna zgodba (Slika 4).

Slika 4: Začetna stran z uvodno zgodbo

S klikom na gumb "začni" se odpre prvi hekerski napad (Slika 5), ki opiše problem, poda neke naključne podatke in od uporabnika zahteva pravilen odgovor. Enaka zgradba se pojavlja tudi v preostalih nalogah.

Slika 5: Primer naloge

Če uporabnik v polje za odgovor vnese napačno vrednost, se mu izpiše opozorilo (Slika 6), sicer pa ga stran preusmeri na naslednjo nalogo.



Slika 6: Obvestilo o napačni rešitvi

Ko uspešno preprečimo vseh 25 hekerskih napadov, se pojavi končna stran (Slika 7) z obvestilom: *»Čestitke! Hekerski napad na sistem si uspešno preprečil in poskrbel, da so hekerji iz Severne Jekorije razkriti in kaznovani!«* S tem se uporabnikovi izzivi končajo.

Slika 7: Končna stran

Naloge

Na spletni strani se nahaja 25 nalog za utrjevanje programiranja, ki simulirajo hekerske napade. Naloge si logično sledijo in se navezujejo na začetno zgodbo, tako da se uporabnik zlahka vživi v vlogo rešitelja sveta pred hekerskimi napadi. Vsak napad zahteva rešitev nove naloge. Vsaka naloga ima svoje ime in določen koncept, ki ga obravnava.

Varnostno geslo	Štetje (prištevanje in odštevanje), pogojni stavki, zanke

Iskanje ključa šifriranih podatkov	Preštevanje različnih objektov, pogojni stavki, zanke, nizi, tabele
GPS	Pretvarjanje med številskimi zapisi, deljenje, deljenje z ostankom, pogojni stavki, zanke, tabele
Poišči geslo	Štetje, iskanje nekega znaka v nizu, koncept stanja, pogojni stavki, zanke, nizi
Dešifriraj geslo	Računanje vsote, pogojni stavki, zanke, nizi
Pretvarjanje v binarni zapis	Štetje, pretvarjanje med številskimi zapisi, deljenje, ostanek, pogojni stavki, gnezdene zanke, tabele
Analiza besedil glede na iskano besedo	Štetje, pogojni stavki, zanke, nizi
Analiza besedil glede na dolžino besed	Koncept štetja z vmesno ponastavitvijo števca, pogojni stavki, zanke, nizi
Napad na ključe za podpisovanje certifikatov	Iskanje največjega in najmanjšega elementa, gnezdene zanke, dvodimenzionalne tabele
Napad na pomnilnik	Odločitveni stavki, zanke, tabele
Pošiljanje sporočila guvernerju	Funkcije
Lokacije državnih skrivališč	Dvodimenzionalne tabele, iskanje elementa
Kodiranje po RLE-metodi	Primerjanje dveh znakov v nizu, štetje enakih zaporednih znakov niza
Napad na telefonske številke	Tabele nizov, iskanje znakov v nizih, dvodimenzionalne tabele

Izbris besed v zaupnih dokumentih	Iskanje podniza v nizu, štetje
Dodane besede v zaupnih dokumentih	Tabele nizov, iskanje črk v nizih, štetje
Leksikografsko urejanje besed	Urejanje elementov v tabeli
Izbrisane črke	Funkcije, iskanje znakov v nizu
Napad na števila	Razstavljanje števil na števke, odločitveni stavki, pogoji
Napad na identifikacijske številke dokazov	Razstavljanje števil na števke, funkcije
Branje zakodiranega števila	Branje elementa z določenega mesta v tabeli, dvodimenzionalne tabele
Določitev novih identifikacijskih števil	Iskanje po tabeli, tabele
Napad na računske operacije	Štetje različnih elementov v nizih
Mapa Fibonacci	Iskanje člena zaporedja
Koordinate napadalcev	Sestavljanje pogojev, uporaba koordinatnega sistema

Tabela 1: Naloge in koncepti

ZAKLJUČEK

Programiranje je za dijake težko predvsem zato, ker jih izpostavlja računalniškemu mišljenju, s tem pa reševanju problemov z uporabo računalniških konceptov, ki so za dijake praviloma zelo zahtevni. Za učenje programiranja ni recepta oz. jasnih korakov, ki bi vsakega dijaka

pripeljali do tega, da bi znal programirati. Z gotovostjo pa lahko trdimo, da je potrebo znanje programiranja utrjevati.

V sklopu raziskave smo izdelali spletno stran s 25 nalogami, povezanimi v zgodbo in uspešno izvedli reševanje teh nalog na srednji računalniški šoli, v 2. letniku, saj se prav takrat dijaki seznanijo z osnovami programiranja. Vseh 25 pripravljenih nalog je rešilo kar 92,2 % (anketiranih je bilo 77 dijakov). To potrjuje, da so pozitivni učinki problemskega poučevanja pri utrjevanju vidni že na majhnem vzorcu. S problemskim postopom dijake namreč lahko bolj vzpodbujamo k učenju in vztrajnosti, kar smo prepoznali tudi iz odgovorov anketiranih. Dijaki so navajali, da so se jim zdele naloge zanimive in zabavne ter da jim je bila všeč zgodba, kar je vsekakor pokazatelj, da problemske naloge in spletna stran dijake vzpodbujajo k utrjevanju programiranja. Te naloge so primerne tako za uporabo pri pouku kot tudi za dodatno delo v obliki domačih nalog.

VIRI IN LITERATURA

[1] AZMI, Shahdatunnaim, IAHAD, Noorminshah A., AHMAD, Norasnita: Gamification in online collaborative learning for programming courses: a literature review. ARPN Journal of Engineering and Applied Sciences, 2015, 10(23), 1–3.

[2] JAKOŠ, Franc, VEBER, Domen: Learning Basic Programing Skills With Educational Games: A Case of Primary Schools in Slovenia. Journal of Educational Computing, 2016, 55(5), 673–698.

[3] JENKINS, Tony: The motivation of students of programming. In Proceedings of ITiCSE 2001: The 6th annual conference on innovation and technology in computer science education, 2001, 53–56.

[4] JERMANN, Patrick, DILLENBOURG, Pierre: Group mirrors to support interaction regulation in collaborative problem solving. Computers & Education, 2008, 51(1), 279–296.

[5] JONASSEN, David: Designing research-based instruction for story problems. Educational Psychology Review, 2003, 15(3), 267–296.

[6] KAY, Judy, BARG, Michael, FEKETE, Alan, KINGSTON, Jeffrey: Problem-based learning for foundation computer science courses. Computer Science Education, 2000, 10, 109–128.

[7] LAM, Maria S. W., CHAN, Eric Y. K., LEE, Victor C. S., YU, Y. T.: Designing an automatic debugging assistant for improving the learning of computer programming. Lecture Notes in Computer Science, 2008, 5169, 359–370.

[8] LAU, Wilfred. W. F., YUEN, Allan. H. K.: Exploring the effects of gender and learning styles on computer programming performance: Implications for programming pedagogy. British Journal of Educational Technology, 2009, 40(4), 696–712.

[9] LAW, Kris M. Y., SANDNES, Frode Eika, JIAN, Hua, HUANG, Yo: A comparative study of learning motivation among engineering students in South East Asia and beyond. International Journal of Engineering Education, 2009, 25(1), 144–151.

[10] LEWIS, Linda, WILLIAMS, Carol: Experiential learning: Past and present. New Directions for Adult and Continuing Education, 1994 62, 5–16.

[11] LINN, Marcia C., CLARK, Douglas, SLOTTA, James D.: Wise design for knowledge integration. Science Education, 2003, 87(4), 517–538.

[12] MARAGOS, Konstantinos, GRIGORIADOU, Maria: Designing an educational online multiplayer. Proceedings of the Informatics Education Europe II Conference IEEII, 2007, 322–331.

[13] ROBINS, Anthony, ROUNTREE, Janet, ROUNTREE, Nathan: Learning and teaching programming: a review and discussion. Computer Science Education, 2003, 13(2), 137–172.

[14] SHARP, Laura: Stealth learning: Unexpected learning opportunities through games. Journal of Instructional Research, 2012, 87, 42–48.

[15] WINSLOW, Leon: Programming pedagogy – A psychological overview. SIGCSE Bulletin, 1996, 28, 17–22.

[16] XINOGALOS, Stelios: Designing and deploying programming courses: Strategies, tools, difficulties and pedagogy. Education and Information Technologies, 2014, 1–30.