Praktične izkušnje pri uporabi sistema Projekt Tomo pri poučevanju programiranja v srednji šoli

Ime Priimek

ŠOLA
Naslov
Kranj
ime.priimek@tu\_in\_tu.si

Matija Lokar

UL FMF
Jadranska ulica 19
Ljubljana
matija.lokar@fmf.uni-lj.si

Ime Priimek

ŠOLA
Naslov
Kranj
ime.priimek@tu\_in\_tu.si

Ime Priimek

ŠOLA
Naslov
Kranj
ime.priimek@tu\_in\_tu.si

Ime Priimek

ŠOLA
Naslov
Kranj
ime.priimek@tu\_in\_tu.si

**Povzetek**

Spletno storitev Projekt Tomo razvija Fakulteta za matematiko in fiziko Univerze v Ljubljani (UL FMF) kot pomoč pri poučevanju in učenju programiranja. Je zelo dobro sprejeta tudi s strani srednješolskih učiteljev. V prvem delu članka bomo predstavili, kako si lahko učitelji programiranja pomagajo s spletno storitvijo Projekt Tomo. Osredotočili se bomo na pripravo učnih gradiv in spremljanje napredka učencev. V drugem delu pa bodo predstavljene praktične izkušnje soavtorjev članka pri vsakdanjem delu v razredu.

**Kategorije in predmetne oznake**

D.2.4 [**Software Engineering**]: Software/Program Verification: Validation.

K.3.1 [**Computers and education**]: Computer Uses in Education: Computer-ašisted instruction.

K.3.2 [**Computers and education**]: Computer and Information Science Education: Computer science education.

**Splošni pojmi**

Algorithms, Management, Experimentation, Languages.

**Ključne besede**

Poučevanje, programiranje, spletna storitev, spremljanje napredka.

# Uvod

Poučevanje programiranja je zahteven proces. Ker je programiranje veščina, se jo učenci lahko naučijo le z veliko vaje. Pri tem neizbežno naredijo veliko napak. Velika večina začetniških napak je preprosto rešljivih, učence je treba le usmeriti v pravo smer.

Pri odkrivanju sintaktičnih napak ni večjih težav, saj sodobna orodja nudijo kar dovršeno pomoč. Večja težava je s semantičnimi napakami. Če pomoč učitelja ni takoj na voljo, to močno upočasni napredek učencev, saj ne vedo, kako naprej. Pogosto se tudi zgodi, da učenci zaradi odsotnosti pomoči nalogo rešijo narobe ali pomanjkljivo in se tega niti ne zavedajo.

Zato so dobre in hitre povratne informacije bistvene za hiter napredek. Če pomoč učitelja ni takoj na voljo, to močno upočasni napredek učencev, saj ne vedo, kako naprej. Če mora to delati učitelj za vsakega učenca posebej, mu vzame ogromno časa, ki bi ga lahko usmeril drugam. Zato učitelju zelo prav pride orodje, ki mu olajša delo in ga naredi bolj učinkovitega

Zato ni čudno, da so sistemi za avtomatsko ocenjevanje programskih nalog postali priljubljena izbira pri programskih tečajih.

Tudi pri poučevanju programiranja na Fakulteti za matematiko in fiziko smo ugotovili, da bi uporaba takega sistema lahko izboljšala naše poučevanje. Žal pa je imela večina pregledanih sistemov take ali drugačne pomanjkljivosti. Predvsem nas je motilo, da so bile praviloma povratne informacije več ali manj omejene le na to, ali se rezultat ujema s pričakovanim. Zato smo razvili nov sistem, ki se imenuje Projekt Tomo (https://www.projekt-tomo.si/). Je popolnoma odprt in na voljo vsem. Vsebuje že več kot 1000 različnih programskih vaj, ki jih je mogoče prilagoditi in ponovno uporabiti v novih tečajih. Trenutno sistem uporablja več kot 30 izobraževalnih ustanov. Med njimi je največ srednjih šol.

# Kratka predstavitev spletne storitve Projekt Tomo

Spletna storitev projekt Tomo [1] je prosto dostopna na spletu. Ob prijavi se uporabniku odpre začetna stran, na kateri so našteti vsi predmeti, ki so mu na voljo. Predmeti so razporejeni glede na šole. Če učitelj želi, da bi tudi sam imel na storitvi določen predmet, preprosto pošlje zahtevo po e-pošti.



Slika 1: Začetna stran

V zgornjem delu strani uporabnik vidi predmete, na katere je prijavljen, spodaj pa seznam predmetov, na katere se lahko še prijavi. Ti so urejeni po izobraževalnih ustanovah in projektih. Gradiva v predmetu so razdeljena na enote, ki jih imenujemo sklopi.

Vsak sklop je sestavljen iz nalog, vsaka naloga pa vsebuje eno ali več podnalog.

 

Slika 2: Naloga, predstavljena v storitvi

Izbrano nalogo učenec s klikom na ustrezno ikono prenese k sebi v obliki tekstovne datoteke. V datoteki ima zapisano vse potrebno za reševanje naloge: besedilo naloge in prostor, kamor vpiše svojo rešitev.



Slika 3: Besedilo naloge v obliki programa

Datoteko lahko odpre v poljubnem programskem okolju (npr. Python IDLE, Thonny, PyCharm …) in začne z delom. Storitev je zasnovana tako, da ne zahteva učenja novega programskega okolja.

Ko je naloga v obliki programa prenesena, jo učenec reši. Rešitve napiše v ustrezni vmesni prostor. Ob zagonu programa dobi obvestilo o uspešnosti.



Slika 4: Uspešnost reševanja naloge

Učenec se lahko loti popravkov te naloge ali pa si prenese novo nalogo in rešuje to.

Podrobneje si o uporabi sistema lahko preberete npr. v [1], [2] in [3].

Po naših izkušnjah Projekt Tomo svojo največjo moč kot pomoč učiteljem pokaže predvsem na treh področjih: pri pripravi učnega gradiva, pri spremljanju napredka učencev,in pri analiziranju njihovega dela.

# Priprava učnega gradiva

Prvi korak na poti do priprave gradiva (sklopa nalog) je ta, da dobimo status učitelja. To lahko storimo s prošnjo upravljavcem strani (ki po potrebi tudi ustvarijo nov predmet). Ko postanemo učitelj, s klikom na gumb "Dodaj sklop" ustvarimo sklop, v katerega bomo dali naloge. Naloge lahko prenesemo iz drugih predmetov ali pa jih sestavimo sami.

## Prenos gradiv iz drugih predmetov

Kakor hitro imamo v določenem predmetu status učitelja, lahko vanj prenašamo naloge iz drugih predmetov. Izberemo si nalogo, ki jo želimo prenesti. Ob kliku na gumb kopiraj se odpre okno, v katerem izberemo predmet med tistimi, kjer imamo status učitelja in sklop, v katerega se bo naloga prenesla, ter kliknemo na gumb "Prenesi". Naloga se prenese v celoti, z besedilom nalog in vsemi testnimi primeri. Gre za kopijo naloge, ki ni več povezana z nalogo, ki smo je kopirali.

Ta možnost je še posebej zanimiva za srednješolske učitelje programiranja, saj je bila v okviru projekta NAPOJ [4] zgrajena tako zbirka nalog iz učbenika Informatika [5] kot tudi zbirka nalog, primernih za dodatne vaje. Prva tako so v sklopu projektov nastale obširne zbirke originalnih nalog predvsem TODO!!!….



Slika 5: Gradivo opremljeno z videoposnetkom



Slika 6: Vprašanja v videoposnetku

Kdor ima status učitelja pri vsaj enem od predmetov, lahko naloge iz poljubnega predmeta (torej tudi iz teh dveh zbirk) prosto prenaša v svoje predmete. Tako si lahko ob pomanjkanju časa oz. idej zelo preprosto in hitro pripravi naloge na dano tematiko.

## Popravljanje obstoječih gradiv in priprava novih

Če z obstoječo nalogo učitelj ni zadovoljen in bi jo rad popravil, si na računalnik prenese datoteko za urejanje. Odpre jo v svojem najljubšem urejevalniku besedil, popravi besedilo in datoteko zažene (glej Slika 7). Ob zagonu se spremenjeno besedilo prenese na spletni strežnik in spremembe v besedilu naloge so takoj vidne na spletni strani.

 

Slika 7: Urejanje naloge

Pod besedilom naloge sledi uradna rešitev ter testni primeri, katerim morajo zadoščati rešitve (tudi uradna), da so proglašene za sprejete. Pri tem je uradna rešitev obvezna in je ne smemo izpustiti. S tem ima učitelj na prvi pogled nekaj več dela, ampak prinaša veliko prednosti. Velikokrat namreč šele pri reševanju opazimo težave s formulacijo naloge in se domislimo dobrih testnih primerov zanjo. Poleg tega pa uradno rešitev lahko pokažemo tudi učencem. Zato se splača potruditi in rešitev napisati res dobro, da se učenci ob njenem pregledu naučijo tudi lepega programiranja.

# Spremljanje napredka učencev

Za spremljanje napredka tako posameznega učenca kot celotne skupine (učenci v sklopu predmeta) poleg številskih podatkov (v obliki % ali absolutno) Tomo uporablja barvno kodiranje. Rdeča barva pomeni, koliko nalog pri sklopu učenci niso niti poskušali reševati, rumena, koliko nalog je bilo rešenih narobe in zelena delež sprejetih rešitev.

Učitelj ima pregleden vpogled tako v uspeh celega razreda pri celotnem predmetu, pri posameznem sklopu in posamezni nalogi.

Podrobnosti glede načina spremljanja si lahko ogledate v [3].



Slika 8: Uspešnost reševanja posamezne naloge - skupno in posamično

Napredek v posameznem sklopu se učitelju prikaže na strani sklopa. Tudi tu je na desni za vsako nalogo v sklopu izrisanih več tortnih diagramov, po eden za vsako podnalogo, barvno kodiranih na znan način.



Slika 9: Napredek v posameznem sklopu

Če učitelja napredek posameznega študenta zanima še podrobneje, s klikom na njegovo ime odpre razdelek s podrobnimi informacijami (Slika 10).



Slika 10: Uspešnost posameznega učenca za sklop

## Dostop do vseh rešitev

TODO!!!

<https://www.projekt-tomo.si/problem_set/1057/>

<https://www.projekt-tomo.si/problem_set/1057/results>



Slika 11: ...



Slika 12: ...



Slika 13: ...

Možnost analiz ….

# Uporaba sistema v praksi

Ena od dobrih lastnosti storitve Tomo je tudi ta, da prevzame vlogo »sitnega prfoksa«, ki vztraja, da morajo biti rešitve take, kot zahteva besedilo naloge.

Oglejmo si primer - uradno rešitev in rešitev učenca. Kot vidimo, se razlikujeta samo v eni črki: dijak je zapisal besedo "vnesi" z malo začetnico, medtem, ko je naloga zahtevala veliko. Seveda bi lahko razpravljali o tem, ali naloga zahteva točno določen pozivnik (torej, ali je v primeru navedena oblika pozivnika zahtevana), a recimo, da je temu tako.



Slika 14: Učenčeva in uradna rešitev

Tukaj spet pride do izraza dejstvo, da je Projekt Tomo zgolj orodje v rokah učitelja, ki se mora odločiti, kaj želi z nalogo doseči. Pri zgornji nalogi ima na voljo številne možnosti. Naj navedemo le štiri.

1. Nalogo pusti tako, kot je. Namen takšne naloge je navaditi učence natančnega branja navodil, zahtevkov in tega, da se jih je treba dosledno držati. Projekt Tomo nam je tukaj v veliko pomoč, saj ni treba vsakemu učencu posebej razlagati, da njegova rešitev ni dobra zaradi ene same velike črke. Sistem je pač ne sprejme kot pravilne in naloga ni rešena pravilno.
2. Spremeni test tako, da, če učenec napiše "vnesi" namesto "Vnesi", dobi povratno informacijo, da naj pazi na malo/veliko začetnico. To je nekoliko milejša oblika prejšnjega primera, kjer učenca opozorimo na pogosto napako.
3. Spremeni test tako, da je vseeno, če so uporabljene male / velike črke. To je smiselno v primeru, da je učiteljev poudarek pri nalogi drugje in je izpis postranskega pomena.
4. Spremeni test tako, da je popolnoma vseeno kakšen "pozivnik" učenec uporabi v svoji rešitvi.

In seveda - kar je najpomembneje - učiteljeva naloga je odzvati se na različne dogodke, ki se lahko med izvajanjem vaj pripetijo. In ravno to je osnovni razlog, da razvijamo Projekt Tomo: da učitelja razbremenimo preprostih opravil in mu damo dodatni čas, da se med vajami lahko pogovori z učencem. Poleg tega menimo, da na ta način omogočamo tudi poglobljeno razpravo z učenci, saj je učitelj razbremenjen vsaj dela odpravljanja rutinskih napak.

## Izkušnje na šoli 1

## Izkušnje na šoli 2

## Izkušnje na šoli 3

## Izkušnje na šoli 4

# Zaključek

Projekt Tomo učitelju na zgoraj opisane načine olajša delo, tako da lahko več pozornosti nameni pripravi gradiv in poučevanju. Prav tako pomaga študentom, ki dobijo takojšen odziv glede pravilnosti njihovih rešitev in jim tako omogoča hitrejši napredek.

Pri tem je veliko odvisno od kakovosti povratnih informacij, ki jih učencu vrnejo testni programi. Zato je pisanje testnih primerov zelo zamudno opravilo, saj zahteva veliko premišljevanja o napakah, ki jih učenci pogosto storijo, iskanja robnih primerov in šele nato lahko začnemo pisati testne primere, ki bodo vse te napake prestregli in dali učencu primeren odziv glede na napako, ki jo je naredil. Po naših izkušnjah je to zelo težko storiti naenkrat, ampak zahteva neprestano delo: učitelj pripravi testne primere, nato analizira oddane rešitve in na osnovi analize popravi testne primere in ponavlja postopek.

Tovrstna orodja bodo, kot kaže, nujna, saj je tudi v zavest širše javnosti prišlo spoznanje, da je vsaj osnovno znanje programiranja dandanes izjemno koristno. In kot kažejo izkušnje iz tujine, je trenutno glavna težava pri širši vpeljavi učenja programiranja ta, da primanjkuje dobrih učiteljev programiranja. Noben tak sistem seveda (še) ne more nadomestiti učitelja, lahko pa mu pomaga, da se laže sooči z večjimi skupinami.

Literatura

|  |  |
| --- | --- |
| [1]  | M. Pretnar, „Projekt Tomo,“ 25 9 2010. [Elektronski]. Available: https://www.projekt-tomo.si. |
| [2]  | M. Pretnar in M. Lokar, „A Low Overhead Automated Service for teaching Programming,“ v *Proceedings of the 15th Koli Calling International Conference on Computing Education Research*, Koli, Finland, 2015.  |
| [3]  | G. Jerše in M. Lokar, „Uporaba sistema za avtomatsko preverjanje nalog Projekt Tomo pri učenju programiranja,“ v *Vzgoja in izobraževanje v informacijski družbi - VIVID 2017 : zbornik referatov*, Ljubljana, 2018.  |
| [4]  | G. Anželj, A. Brodnik in M. Lokar, „NAPOJ – proti aktivni skupnosti učiteljev računalniških predmetov,“ v *Vzgoja in izobraževanje v informacijski družbi - VIVID 2017 : zbornik referatov*, Ljubljana, 2018.  |
| [5]  | G. Anželj, J. Brank, A. Brodnik, P. Bulić, M. Ciglarič, M. Đukić, L. Fürst, M. Kikelj, A. Krapež, H. Medvešek, N. Mori, M. Pančur in P. Sterle, „Računalništvo in informatika,“ 25 9 2017. [Elektronski]. Available: http://lusy.fri.uni-lj.si/ucbenik/book/index.html. |
| [6]  | M. Pretnar, „Spletna storitev za poučevanje programiranja,“ v *Vzgoja in izobraževanje v informacijski družbi - VIVID 2014* , Kranj: Fakulteta za organizacijske vede, 2014.  |
| [7]  | A. T. Corbett in J. R. Anderson, „Locus of feedback control in computer-based tutoring: impact on learning rate, achievement and attitudes,“ v *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '01).* , New York, 2001.  |
| [8]  | „Arnes AAI,“ 2017. [Elektronski]. Available: https://aai.arnes.si/. |
| [9]  | „Wikipedia,“ 25 9 2017. [Elektronski]. Available: https://en.wikipedia.org/wiki/Markdown. |
| [10]  | „CodeQ,“ 25 9 2017. [Elektronski]. Available: https://codeq.si. |
| [11]  | M. Pretnar, „GitHub,“ 2017. [Elektronski]. Available: https://github.com/matijapretnar/projekt-tomo. |
| [12]  | A. Brodnik, M. Lokar in N. Mori, „Activation of Computer Science Teachers in Slovenia,“ v *World Conference on Computers in Education (WCCE) 2017*, Dublin, Ireland, 2017.  |