Izrada alata za učenje programiranja

Peran, Ivana

Master's thesis / Diplomski rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Science / Sveučilište u Zagrebu, Prirodoslovno-matematički fakultet**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:217:707412

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-04-24**

Repository / Repozitorij:

Repository of the Faculty of Science - University of Zagreb

SVEUČILIŠTE U ZAGREBU

PRIRODOSLOVNO-MATEMATIČKI FAKULTET

MATEMATIČKI ODSJEK

Ivana Peran

IZRADA ALATA ZA UČENJE PROGRAMIRANJA

Diplomski rad

Voditelj rada:

doc. dr. sc. Goranka Nogo

Zagreb, 2019.

Ovaj diplomski rad obranjen je dana _____ pred ispitnim povjerenstvom u sastavu:

1. _____, predsjednik
2. _____, član
3. _____, član

Povjerenstvo je rad ocijenilo ocjenom _____ .

Potpisi članova povjerenstva:

1. _____
2. _____
3. _____

Sadržaj

Sadržaj iv
Uvod 5
Analiza postojećih alata za učenje programiranja 6
1.1 Tynker 6
1.2 Waterbear 7
1.3 RoboMind 8
1.4 CodeCombat 9
Detalji implementacije 11
Varijable 14
3.1 Razina 1 – Operacija pridruživanja i promjena vrijednosti varijable 14
3.2 Razina 2 – Aritmetičke operacije s dvije varijable 21
3.3 Razina 3 – Rješavanje problemskog zadatka 25
Naredba unosa 26
Naredba grananja 31
Naredba ponavljanja 35
6.1 Razina 1 – Naredba ponavljanja s određenim brojem ponavljanja - Motivacija 35
6.2 Razina 2 – Naredba ponavljanja s određenim brojem ponavljanja 40
6.3 Razina 3 – Naredba ponavljanja s logičkim uvjetom 41
Zaključak 44
Bibliografija 46

Uvod

U ovom diplomskom radu bit će detaljno opisan originalni alat za učenje programiranja i njegova izrada. Alat je izrađen kao podrška pri učenju programiranja. Ovaj softver prvenstveno je namijenjen učenicima, ali ga mogu koristiti i svi zainteresirani za učenje. Stoga, u nastavku ćemo osobu koja se njime služi oslovljavati s riječi 'korisnik'. Ideja ovog alata je na slikovit način prikazati i približiti korisniku razne koncepte programiranja s kojima se susreće početnik. Tu prije svega mislimo na koncepte petlje i ponavljanja. Alat je organiziran kao igra koja se sastoji od više razina, a korisnik prelazi na sljedeću razinu rješavanjem zadataka. Alat nije namijenjen za učenje nekog određenog programskog jezika, već se u njemu koristi jednostavni pseudokod s intuitivnim i lako pamtljivim naredbama. Naglasak je postavljen na razvijanju algoritamskog načina razmišljanja.

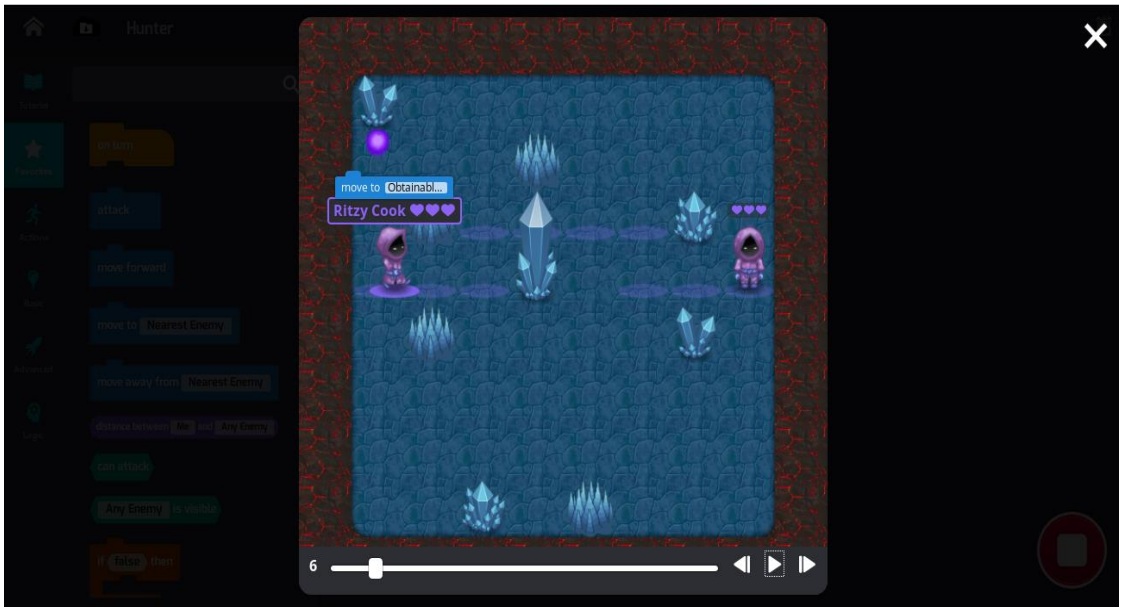
U prvom poglavlju ukratko će se analizirati neki postojeći alati za učenje programiranja. U drugom će biti objašnjeni i pojedini detalji vezani uz samu izradu alata. Zatim, u sljedećim poglavljima opisivat će se izrađeni alat od prve do posljednje razine.

Poglavlje 1
Analiza postojećih alata za učenje programiranja

U ovom poglavlju kratko ćemo se osvrnuti na neke od postojećih alata za učenje programiranja. Prvenstveno smo promatrali alate namijenjene početnicima, a posebno djeci. Uz jedan od najpoznatijih blokovskih programskih jezika – Scratch, razvili su se brojni drugi blokovski programski jezici. Također, kako bi se programiranje približilo djeci, pojavljuju se i mnogi softveri koji povlače paralelu između upravljanja videoigrom i pisanja programskog koda. Neki blokovski programski jezici i igre za učenje programiranja opisani su u nastavku.

1.1 Tynker

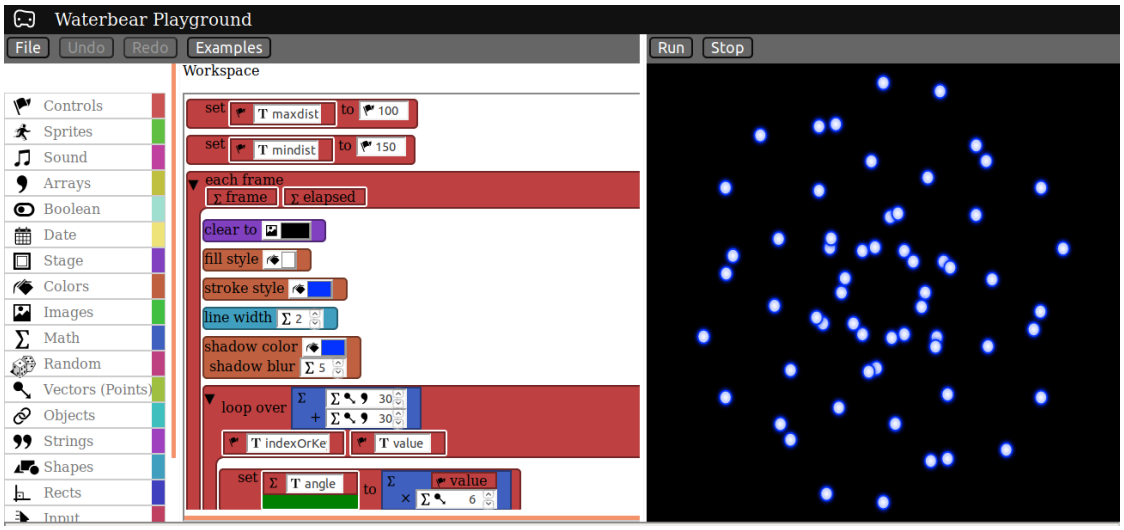
Tynker je alat za učenje programiranja koji je prvenstveno namijenjen djeci. Zahvaljujući *Creative Common Licence* koju ima poznati blokovski programski jezik Scratch, Tynker preuzima brojne njegove elemente. Programi se pišu slaganjem i spajanjem blokova programskog koda te mogu imati grafičko sučelje. Također, izgled razvojnog okruženja podosta podsjeća na razvojno okruženje Scratcha. Kao najveću razliku između Tynkera i Scratcha potrebno je istaknuti činjenicu da je Tynker komercijalni alat. Uz to, za razliku od Scratcha koji nema opciju planskog i vođenog učenja, Tynker nudi već gotove predloške za korištenje u nastavi što pokazuje cilj proizvođača da se Tynker implementira u formalno obrazovanje. Kao dodatak Tynker nudi razne igre koje dijelove programskog koda prikazuju kao animacije.



Slika 1. igra u alatu Tynker (Hunter)

1.2 Waterbear

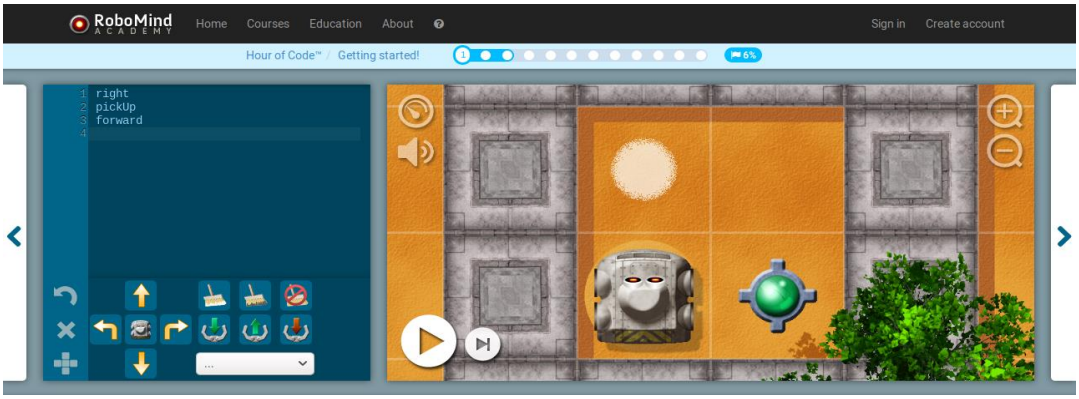
Waterbear je vizualni programski jezik namijenjen početnicima, a posebno djeci. Za slaganje koda koristi metodu povlačenja i ispuštanja (engl. *drag and drop*) blokova na kojima se često nalaze praznine u koje je moguće upisivati parametre. U usporedbi s već spomenutim programskim jezikom *Scratch,* blokovi u ovom programskom jeziku mogu izgledati podosta kompliciranо djeci koja se prvi put susreću s programiranjem. Uz to, mnoštvo naredbi lako zbuni korisnika početnika. S pozitivne strane, Waterbear omogućava učenje iz postojećih primjera.



Slika 2. Waterbear - primjer gotovog programa

1.3 RoboMind

RoboMind je besplatno okruženje za učenje programiranja koje koristi vlastiti programski jezik. Alat je zaštićen Apache licencom (verzija 2.0). Ovaj programski jezik ima izuzetno jednostavnu sintaksu koja korisniku omogućava brzo i lako učenje osnova programiranja. Izvršavanje programskog koda je povezano s animacijom u kojoj je prikazan robot. Robot izvršava naredbe zadanim redoslijedom. Ovakav način učenja je izuzetno efektivan jer je korisniku jako lako uočiti vlastite pogreške. Jedini nedostatak koji možemo istaknuti jest da se na ovaj način korisnika početnika, prije nego što savlada korištenje samih osnova programiranja, direktno uvodi u objektno-orijentirano programiranje.



Slika 3. RoboMind – prva razina

1.4 CodeCombat

CodeCombat je videoigra koja služi za učenje programiranja. Ovaj alat djelomično je besplatan, izrađen je od mnoštva projekata otvorenog koda. Polazi se od učenja algoritamskog načina razmišljanja i nekih osnovnih naredbi. Slično kao kod alata RoboMind, kod je popraćen animacijom u kojoj je zorno prikazano njegovo izvršavanje. Unutar samog alata dodane su brojne opcije koje ga čine zanimljivijim korisniku (biranje glavnog lika igre, novo oružje i sl.). Na nekim razinama se korisniku daju dodatni bodovi ako napiše odgovarajući kod koristeći najmanji mogući broj linija. Odmicanjem igre, zadaci koje korisnik mora izvršiti se otežavaju i korisnik mora primjenjivati znanje koje je stekao tijekom cijele igre. Smatramo da, od svih navedenih alata, CodeCombat pruža najefikasniji način učenja i provjere naučenog, a sve to se odvija na jako zanimljiv način, uz videoigru koja može zainteresirati mlađe i starije korisnike.

Slika 4. CodeCombat – prva razina

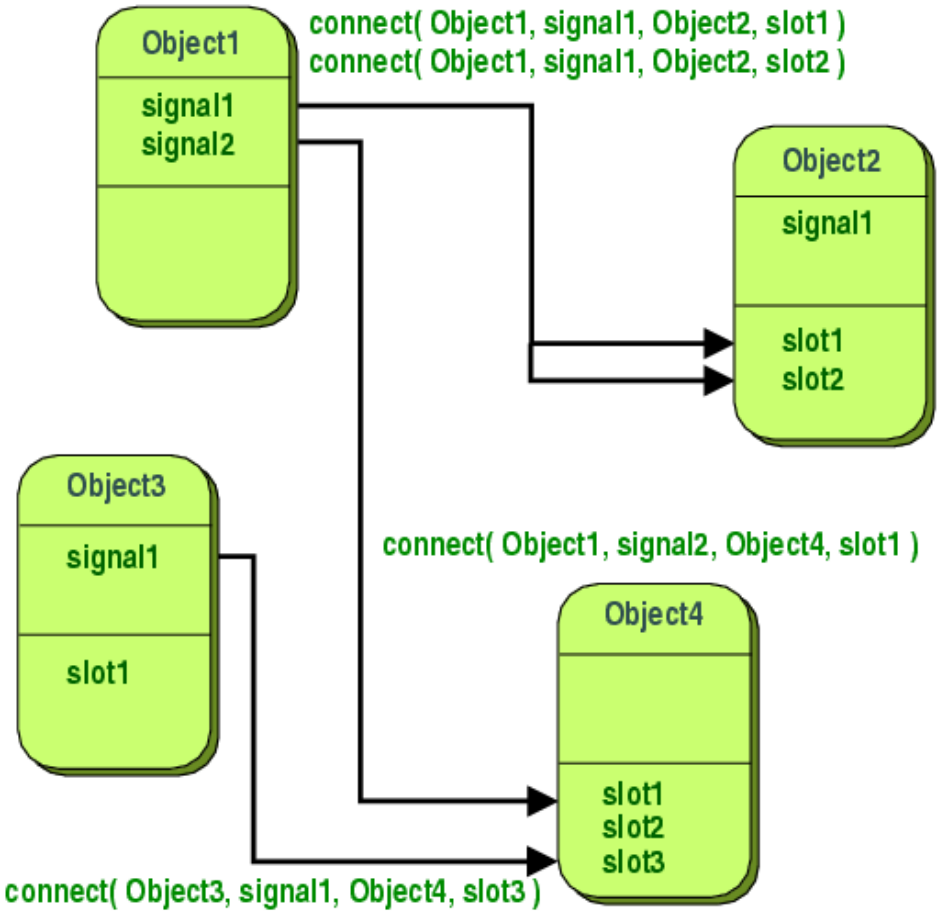
Poglavlje 2

Detalji implementacije

Za implementaciju našeg softvera bilo je potrebno odabrati alate koji će omogućiti efikasno stvaranje grafičkog sučelja i prilagođavanje tog sučelja interakciji s korisnikom. Za izradu softvera korišten je programski jezik C++. Kao dobar izbor za samo grafičko korisničko sučelje pokazala se biblioteka Qt.

Naime, Qt je višeplatformski okvir za razvoj aplikacija. Sama Qt biblioteka izrađena je u programskom jeziku C++. Uz funkcionalnosti programskog jezika C++, Qt dodaje neka nova svojstva kao što su *signals and slots*. Upravo ovo svojstvo se pokazalo kao izuzetno korisno pri izradi alata.

Signals and slots su jedna od karakteristika kojom se Qt posebno ističe. Na ovaj način omogućena je direktna komunikacija između objekata što je izuzetno korisno kod objekata grafičkog sučelja. Jedan signal moguće je povezati s više priključaka, ali je i više signala moguće povezati s jednim priključkom. Shematski prikaz nalazi se na slici broj 5.



Slika 5. *Signals and slots* primjer (https://doc.qt.io/qt-5/signalsandslots.html)

Kao primjer upotrebe iz ovog rada možemo uzeti provjeru napisanog pseudokoda. Naime, čest zadatak korisnika je napisati pseudokod u predodređeni uređivački dio. Nakon što napiše pseudokod korisnik klikne na gumb s natpisom „OK”. Tada objekt klase *Button* šalje *signal* pod nazivom *clicked()*. Poslani *signal* prima *slot* pod nazivom *valueTheCode()* koji je ujedno i funkcija za provjeravanje točnosti napisanog pseudokoda. Za ilustraciju, na slici 6. prikazan je isječak koda koji se nalazi u pozadini provjere točnosti pseudokoda pri početku 3. poglavlja:

```
editor = new CodeEditor(this);
editor->move(10, 200);
editor->resize(500, 400);
editor->show();

highlighter = new MySyntaxHighlighter(editor->document());

OKButton = new Button("OK");
OKButton->setPos(400, 620);
scene->addItem(OKButton);

HINTButton = new Button("HINT");
HINTButton->setPos(20, 620);
scene->addItem(HINTButton);

arrowButton = new Button("←");
arrowButton->setPos(520, 500);
arrowButton->setScale(0.7);
scene->addItem(arrowButton);
connect(arrowButton,SIGNAL(clicked()),editor,SLOT(addOP()));


connect(OKButton, SIGNAL(clicked()), this, SLOT(valueTheCode()));
```

Slika 6. Isječak koda iz 3. poglavlja

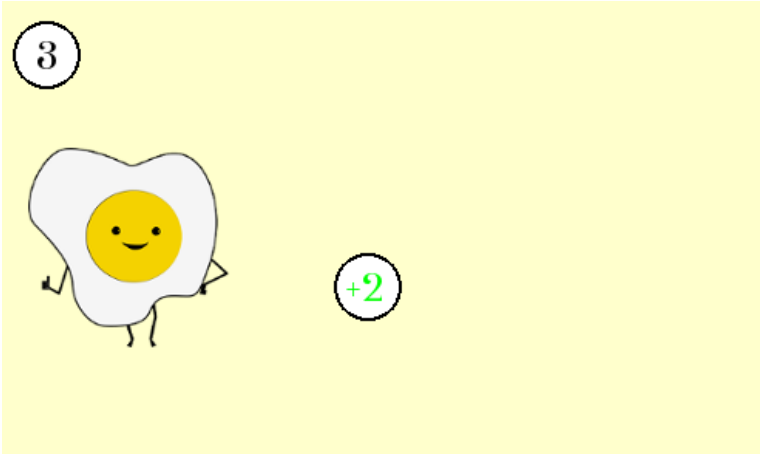
Sve slike koje su dodane na grafičko sučelje izrađene su u GIMP-u (*GNU Image Manipulation Program),* besplatnom programu za uređivanje slika
Alat je testiran na računalu HP Compaq Presario CQ58 – 148SG Notebook s 4 GB RAM-a i grafičkom karticom baziranom na Intel HD Graphics 2000 chipu.

Poglavlje 3
Varijable

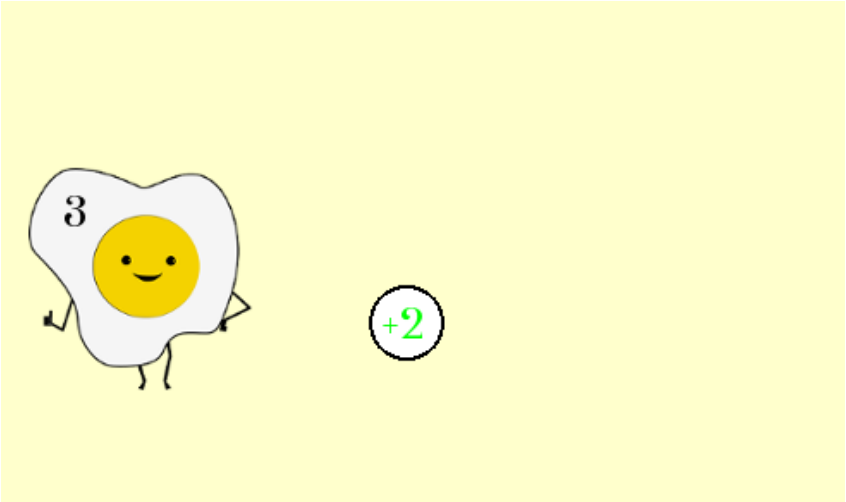
3.1 Razina 1 – Operacija pridruživanja i promjena vrijednosti varijable

Prva razina se sastoji od dva dijela. U prvom dijelu je prikazana animacija s pripadnim pseudokodom koji ju opisuje. Nakon toga, učenici će, vodeći se primjerom animacije, ispuniti zahtjeve zadatka te popuniti dijelove pseudokoda koji nedostaju. Dobiveni pseudokod opisivat će korake rješavanja zadatka. U drugom dijelu ove razine ponovno je potrebno rješenje zadatka opisati pseudokodom, ali ovog puta učenik samostalno piše cijeli pseudokod.

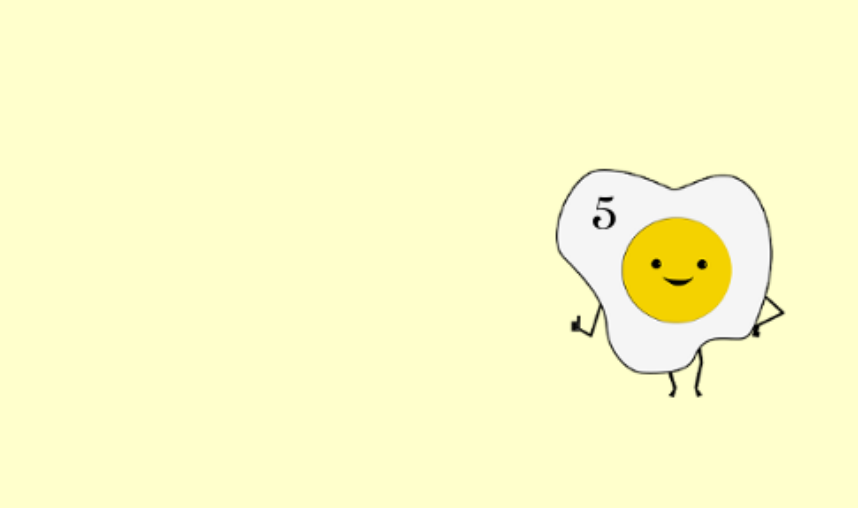
Prvi dio drugog nivoa započinje kratkom animacijom koja se nalazi u gornjem lijevom dijelu prozora igre. Klikom na gumb s porukom "START" pokreće se animacija. U animaciji prikazan je lik igre (jaje) te iznad njega kružić na kojem je zapisan broj 3. Kružić s brojem 3 polako pada na lika. Nakon što kružić padne na lika, kružić nestaje, a na liku je zapisan broj 3. S desne strane lika nalazi se kružić u kojem je zapisano: "+2". Lik igre se kreće prema kružiću i "skupi" ga. Nakon toga se broj koji je zapisan na liku poveća za 2, odnosno po završetku animacije na liku je zapisan broj 5.



Slika 7. Početno stanje animacije



Slika 8. Broj 3 je na liku



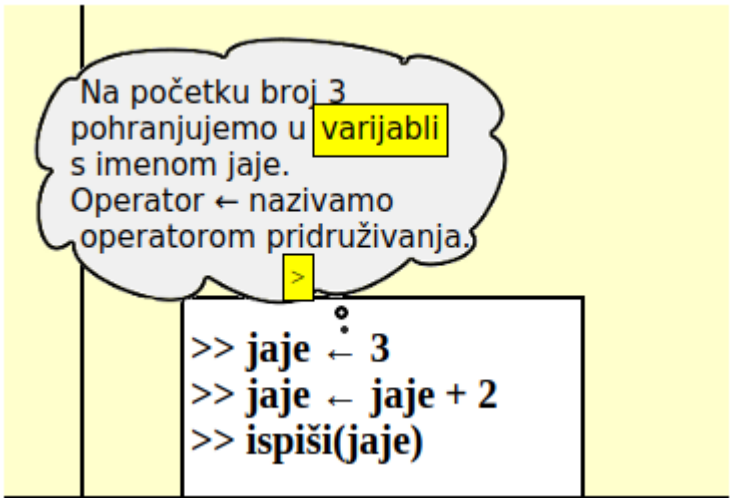
Slika 9. Broj 5 je na liku

U gornjem desnom dijelu prozora nalazi se pseudokod koji opisuje danu animaciju. Pseudokod je sljedeći:

>> *jaje* ←*3*
>> *jaje* ←*jaje* + 2
>> *ispiši(jaje)*

Pokraj svakog retka pseudokoda pojavljuju se oblačići s objašnjenjem. U prvom oblačiću piše "Na početku broj 3 pohranjujemo u varijabli s imenom *jaje*. Operator ← nazivamo operatorom pridruživanja." Riječ "varijabli" u ovom slučaju je tekst koji se nalazi na gumbu. Klikom na taj gumb otvara se dijaloški okvir u kojem se dodatno pojašnjava pojam varijable. Zapisan je sljedeći tekst:
"Podatke možemo pohranjivati u memoriji računala. Memoriju računala možemo zamisliti kao mnoštvo pretinaca u kojima se čuvaju podatci. Ime varijable predstavlja ime pretinca, a podatak koji pridružujemo varijabli pohranjujemo u pretinac označen njenim imenom."
U drugom retku, pokraj izraza „*jaje* + 2" zapisano je: "Vrijednost varijable povećamo za 2", a pokraj znaka ←: "Novu vrijednost pridružujemo varijabli *jaje*."

U trećem retku je oblačić s tekstom "Naredba *ispiši()* ispisuje na ekran vrijednost pohranjenu u varijabli *jaje*."

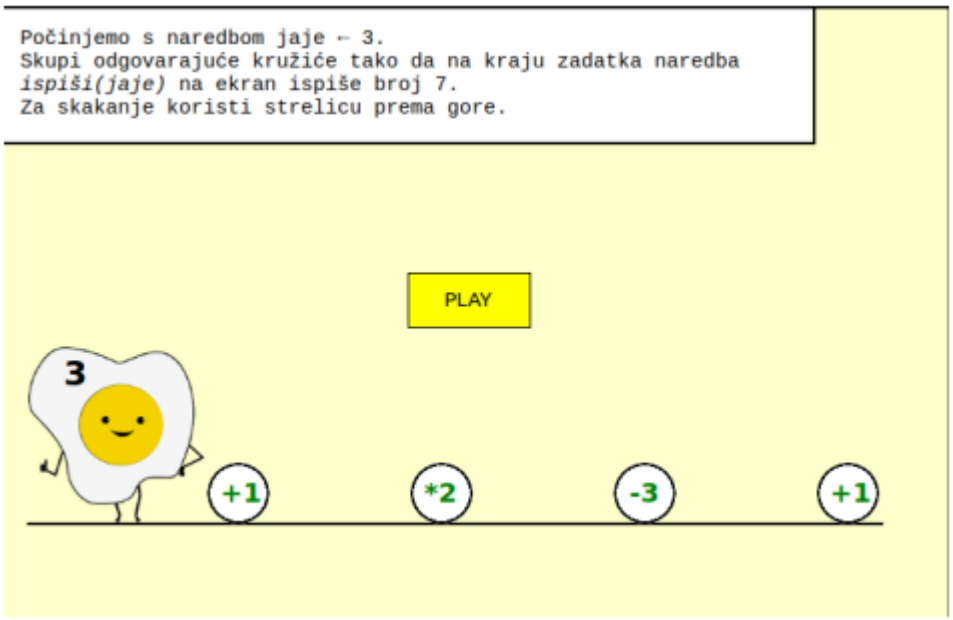


Slika 10. Pseudokod uz animaciju

Ispod toga nalazi se zadatak kojeg je potrebno riješiti vodeći se gornjim primjerom. U lijevom dijelu prozora nalazi se prvi dio zadatka sa zahtjevom koji je zadan u sljedećem tekstu:

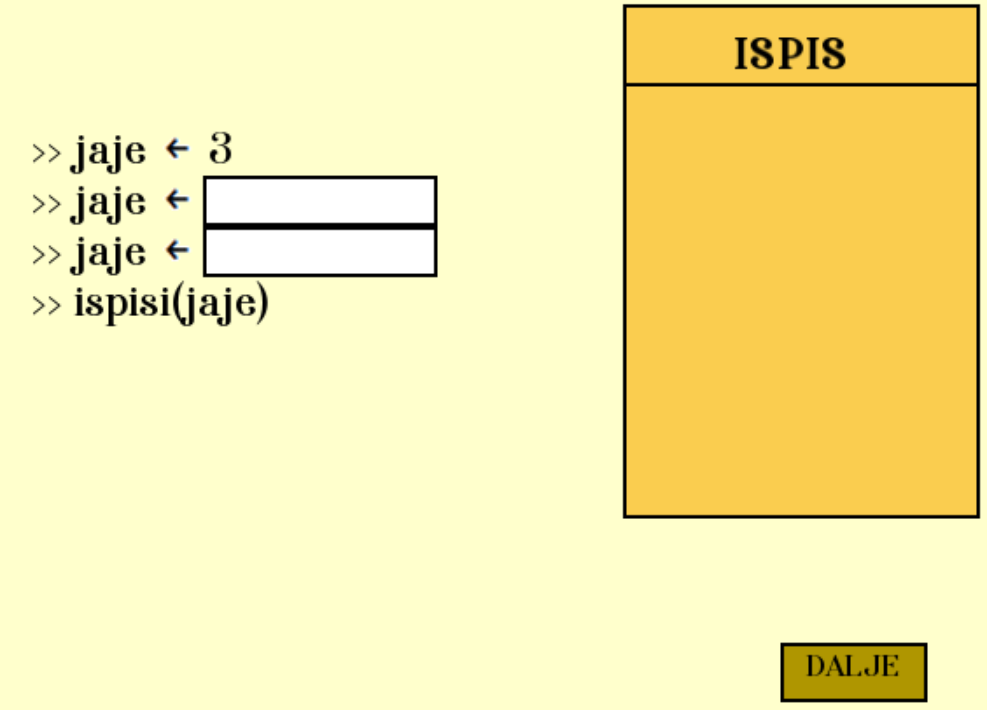
"Počinjemo s naredbom *jaje* ←3. Skupi odgovarajuće kružiće tako da na kraju zadatka naredba *ispiši(jaje)* na ekran ispiše broj 7. Jaje pomičemo sa strelicama *lijevo* i *desno*, a skače se strelicom prema *gore.*"

Ispod teksta prikazan je lik igre na kojem je zapisan broj 3, a desno od njega su kružići s raznim nizovima znakova koji predstavljaju računske operacije. Kako bismo postigli da vrijednost pohranjena u varijabli *jaje* bude broj 7, prvo je potrebno "skupiti" kružić na kojem je zapisano "*2", a nakon toga kružić na kojem je zapisano "+1". Sve ostale kružiće potrebno je preskočiti. Kad lik dođe desno od svih kružića izvršava se naredba *ispiši(jaje)*, ako broj koji se ispiše na ekran nije 7, ovaj dio igre potrebno je odigrati ponovno. Jednom kad naredba *ispiši(jaje)* ispiše ispravan rezultat, korisnik može prijeći na sljedeći dio zadatka. U tom slučaju, pojavljuje se gumb s tekstom "PONOVI", pa korisnik u bilo kojem trenutku može ponoviti ovaj dio igre.



Slika 11. Igra u prvom dijelu razine

U drugom dijelu zadatka potrebno je popuniti nedostatke u pseudokodu koji opisuje izvršavanje prethodnog zadatka. Na slici je prikazan pseudokod, a dijelove koji nedostaju potrebno je upisati u označena područja.

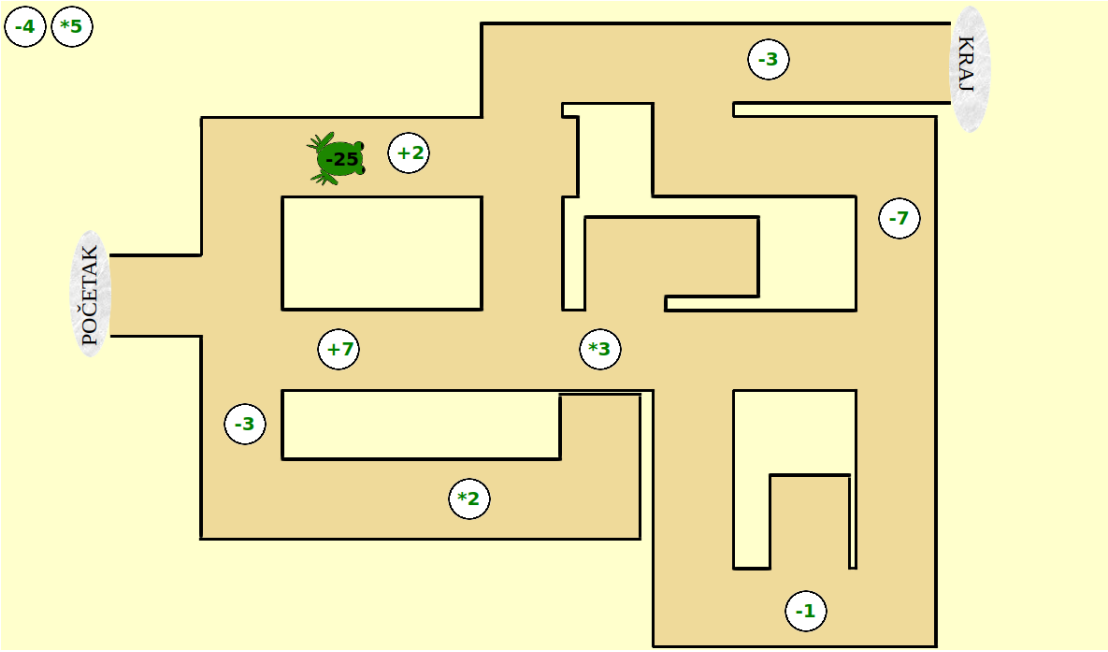


Slika 12. Pseudokod u prvom dijelu razine

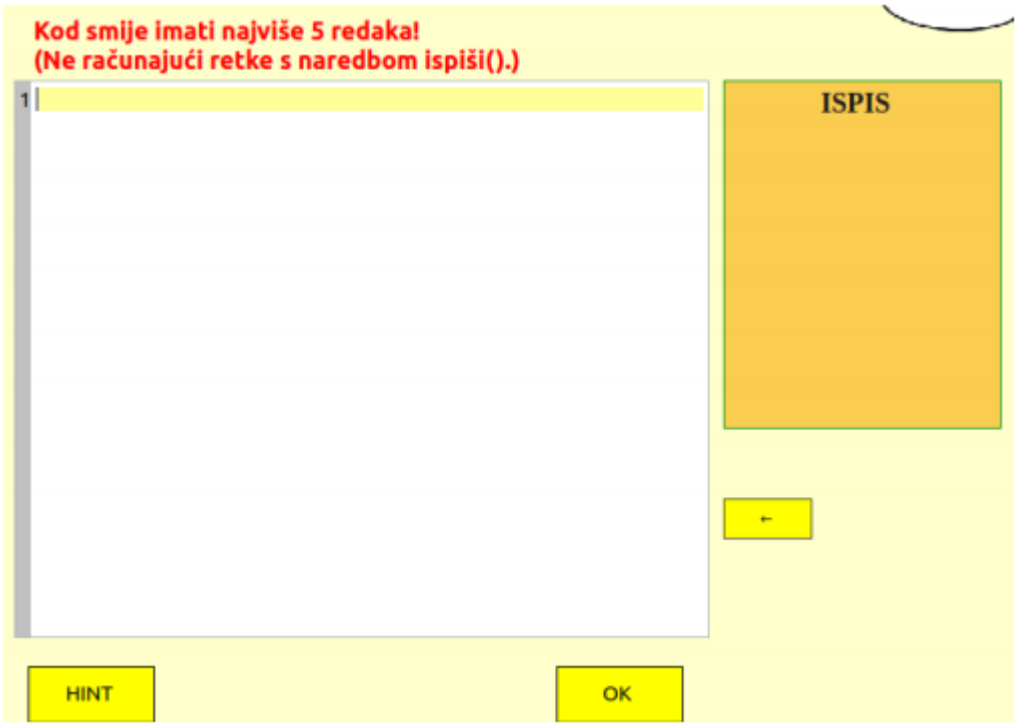
U drugom retku potrebno je upisati *"jaje* * 2" ili " 2 * *jaje*", a u trećem "*jaje* + 1" ili "1 + *jaje"*. Nakon što su sva polja ispravno popunjena u području za ispis ispiše se broj 7. Ovime završava prvi dio druge razine. Klikom na gumb "DALJE" korisnik prelazi na drugi dio ove razine.

Glavni lik drugog dijela razine je žaba. Žaba se nalazi na ulazu u labirint. Zadatak korisnika je dovesti žabu do kraja labirinta, a nakon toga pseudokodom zapisati korake igre u predviđeni prostor za to. Naime, na samom početku u žabu je upisan broj "-1", a u labirintu se nalaze kružići s nizovima znakova koji predstavljaju računske operacije. Kao u prethodnim primjerima, skupljanjem kružića vrijednost koja je upisana u lik igre se mijenja ovisno o računskim operacijama na kružićima koje lik skupi. Dakle, vrijednost koja će biti zapisana na liku po izlasku iz labirinta ovisi o putu koji igrač izabere te kružićima koje "skupi" na putu. Kružići koje lik skupi na svom putu pojavljuju se u gornjem lijevom dijelu ekrana, s lijeva na desno, redoslijedom sakupljanja. Kao što je naznačeno iznad prostora za pisanje pseudokoda, operator pridruživanja ←pojavljuje se

na ekranu pritiskom na odgovarajući gumb koji se nalazi pokraj prostora za pisanje pseudokoda. Također, pored prostora za pisanje pseudokoda nalazi se gumb s natpisom "HINT". Klikom na gumb pojavi se dijaloški okvir s porukom:
"Koristite naredbu *ispiši(ime_varijable)* kako biste provjerili koja je vrijednost trenutno pohranjena u varijabli". Također, pokraj prostora za pisanje postavljen je gumb s natpisom "OK". Klikom na njega provjerava se ispravnost napisanog pseudokoda. Ako je pseudokod ispravan, moguće je kliknuti na gumb s natpisom "DALJE" koji vodi na sljedeći nivo. Ako pseudokod nije ispravan, pojavi se crvena točkica kod prvog problematičnog retka te korisnik mora prepraviti napisani kod.



Slika 13. Igra u drugom dijelu razine



Slika 14. Prostor za pseudokod u drugom dijelu razine

U ovom dijelu može se uočiti kratka promjena glavnog lika igre iz jaja u žabu. Razlog za ovu promjenu je ukazivanje na činjenicu da su imena varijabli proizvoljna. Nadalje, promjena imena varijable daje korisniku više slobode pri pisanju pseudokoda.

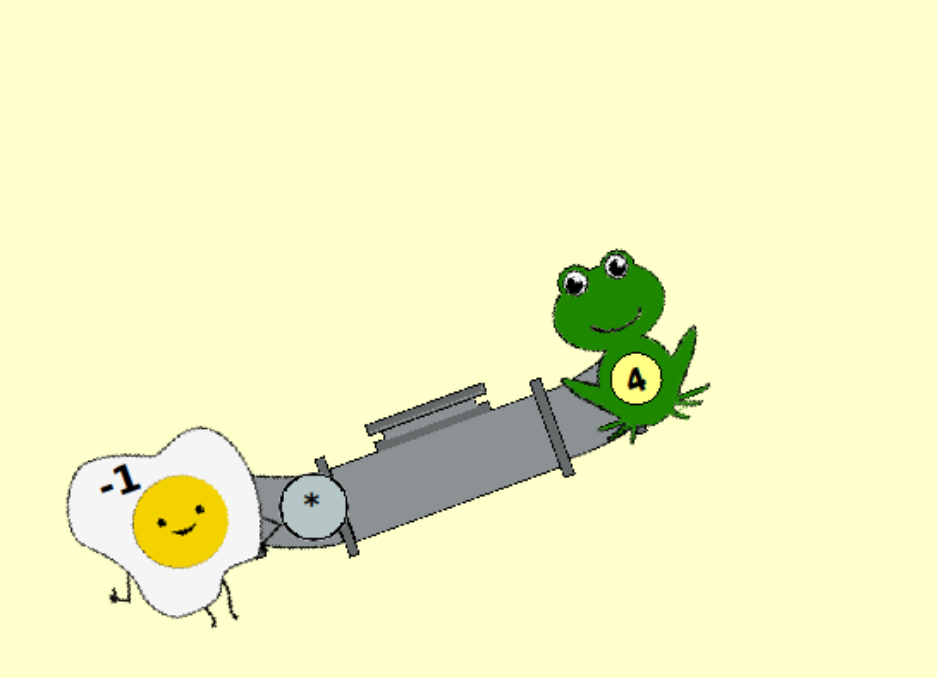
Po završetku prvog dijela nivoa korisnik je u stanju prepoznati varijablu te ju koristiti tako da popunjava dijelove programskog koda. Međutim, u drugom dijelu nivoa korisnik samostalno odabire ime varijable te formira pseudokod kojim opisuje vlastite korake igre.

3.2 Razina 2 – Aritmetičke operacije s dvije varijable

Slično kao druga, treća razina započinje kratkom animacijom te pripadnim pseudokodom koji opisuje događaje u animaciji. Klikom miša na gumb s natpisom „START” pokreće

se animacija. Animacija simulira izvršavanje aritmetičkih operacija s vrijednostima pohranjenim u varijablama te pridruživanje dobivene vrijednosti jednoj od tih dviju varijabli. U ovom dijelu igre koristimo oba lika s kojima smo se već susreli: jaje i žabu. Kao i do sada, oba lika predstavljaju varijable, a vrijednosti napisane na njima predstavljaju vrijednosti pohranjene u varijablama.

U animaciji prikazano je nešto nalik zaobljenoj cijevi s otvorom prema gore. Na lijevoj strani cijevi nalazi se lik jaja, a na desnoj lik žabe. Na početku animacije kružić s natpisom „-1" pada na jaje, a kružić s natpisom „4" pada na žabu. Nakon toga na jaju piše „-1", a na žabi „4". Uskoro, s vrha ekrana pada prvi kružić. Na kružiću je zapisano „*". Cijev se pomiče i „hvata" kružić, odnosno, kružić upada u otvor cijevi. Nakon toga cijev se naginje u lijevu stranu, kružić s natpisom „*" dodiruje lik jaja te se vrijednost zapisana na njemu mijenja iz „-1" u „-4", a kružić nestaje. Dakle, upadanje kružića s natpisom „*" u otvor cijevi predstavlja množenje broja zapisanog na liku jaja s brojem zapisanim na liku žabe. Kad kružić s operacijom dodirne lik jaja, rezultat operacije se pohranjuje u varijabli koju predstavlja lik jaja.
Animacija se nastavlja i s vrha ekrana pada novi kružić, ovaj put s natpisom „+". Cijev se naginje u desnu stranu, pa kružić s operacijom dodiruje lik žabe te se na njemu pojavljuje vrijednost „0".



Slika 15. Isječak iz animacije

Desno od animacije, nalazi se prostor sa pseudokodom koji ju opisuje. Pseudokod je sljedeći:

>> *j* ←-1
>> *z* ←4
>> *j* ←*j* * *z*
>> *z*←*j* + *z*
>> *ispiši(*j)
>> *ispiši*(z)

Lako je uočiti da je ime varijable koju predstavlja jaje, ovog puta slovo „j”, a ime varijable koju predstavlja žaba sada je slovo „z”. Pokraj trećeg retka nalazi s oblačić s pojašnjenjem:

„Rezultat množenja brojeva -1 i 4 pohranjuje se u varijablu *j*, pa je nakon ovog retka vrijednost varijable *j* jednaka -4.” Još jedan oblačić s pojašnjenjem nalazi se pored četvrtog retka, a u njemu piše:
„Rezultat zbrajanja brojeva -4 i 4 pohranjuje se u varijablu *z*, pa je nakon ovog retka vrijednost varijable *z* jednaka 0.”

Klikom miša na gumb s natpisom „OK” korisnik prelazi u sljedeći dio treće razine.
Ponovno je prikazana cijev s likovima iz prvog dijela zadatka, ali ovog puta korisnik ju može pomicati tipkama *lijevo* i *desno*. Na ekranu je zapisana uputa: „Uhvati sve aritmetičke operacije!”
Dakle, zadatak korisnika je uhvatiti sve kružiće koji padaju. Prije početka igre na liku jaja se pojavi broj 2, a na liku žabe se pojavi broj -5. Kružići koje korisnik „uhvati” pojavljuju se u donjem dijelu ekrana, a ispod njih je mala slika lika čija se vrijednost promijenila tom operacijom. Na taj način korisnici lako mogu rekonstruirati tijek igre.

Na ekranu, desno od igre nalazi se prostor za pisanje pseudokoda koji ju opisuje. Korisnicima su vidljive sličice „uhvaćenih” operacija i tijekom pisanja pseudokoda. Klikom na gumb s natpisom „HINT” pojavljuje se dijaloški okvir sa sljedećom uputom:
„Naredbom *ispiši(*ime_varijable1 + ime_varijable2*)* ispiše se rezultat zbroja dvije vrijednosti. Na isti način provjeravamo rezultat i drugih aritmetičkih operacija.” U dijaloškom okviru koji se pojavio moguće je kliknuti na gumb sa znakom „<” kojim se prikaže poruka za pomoć iz prethodne razine.

Nakon što završi s pisanjem koda, korisnik klikne na gumb „OK”, nakon toga se provjeri napisani kod, ako je kod ispravan, korisnik može kliknuti na gumb „DALJE” te prijeći na sljedeći nivo.

U drugom dijelu druge razine (labirint) i u trećoj razini pojavljuju se negativni cijeli brojevi što je dio nastavnih sadržaja 6. razreda osnovne škole. Stoga, ako se ovaj softver koristi u nastavi s učenicima 5. razreda, potrebno je dodatno pojašnjenje nastavnika za ove razine.

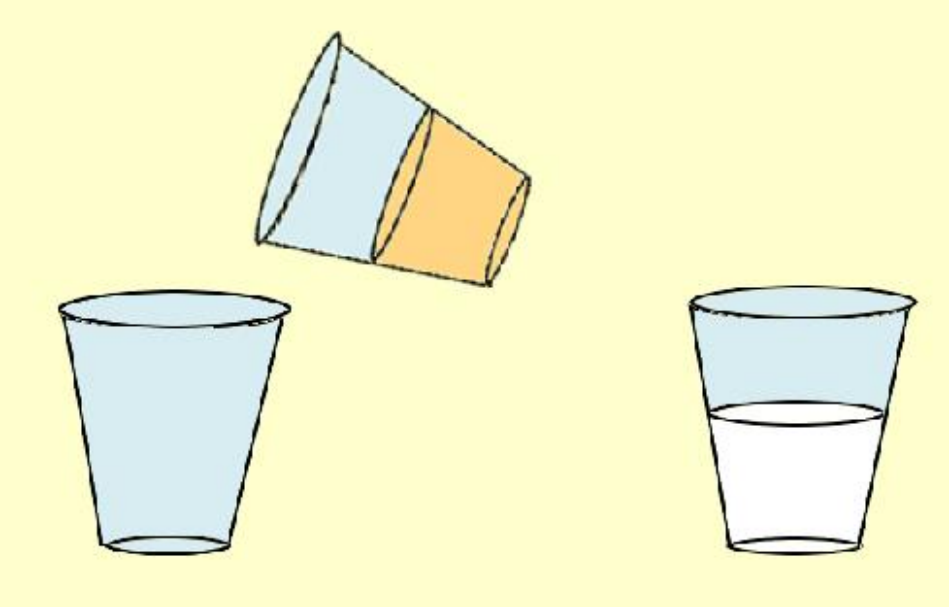
3.3 Razina 3 – Rješavanje problemskog zadatka

Korisnici će rješavati problemski zadatak. U zadatku će biti potrebno zamijeniti vrijednosti pohranjene u dvjema varijablama.

Zadatak je zadan sljedećim tekstom: „U sredini šume nalazi se jezero. U varijabli *žabe* pohranjen je broj žaba koje svakodnevno prelaze to jezero. U varijabli *lopoči* pohranjen je broj lopoča u jezeru. Napiši program koji će zamijeniti vrijednosti pohranjene u ovim dvjema varijablama."

U ovom zadatku korisniku je ponuđena nova uputa koja se pojavljuje klikom na gumb s natpisom „HINT". Unutar upute prikazana je animacija s 3 čaše. U dvije rubne čaše nalaze se tekućine različitih boja, a srednja čaša je prazna. Koristeći srednju čašu kao pomoć pri prelijevanju, dvije rubne čaše zamijene svoj sadržaj.

Također, ispod animacije nalazi se tekst: „Kako bismo zamijenili sadržaj dvije varijable, koristimo treću, pomoćnu varijablu."

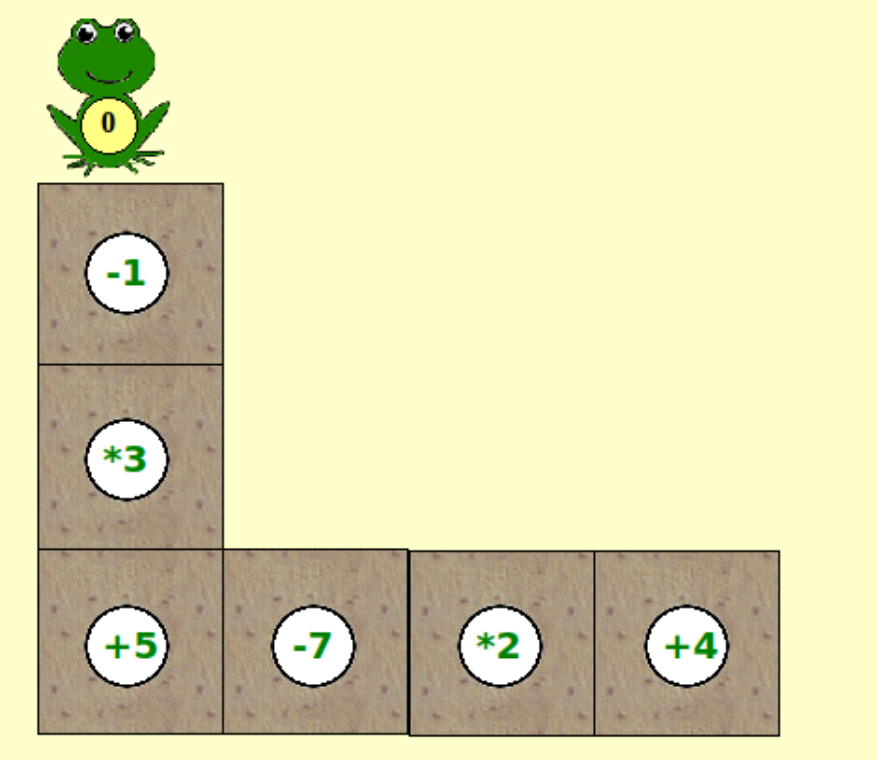


Slika 16. Isječak iz animacije s čašama

Poglavlje 4
Naredba unosa

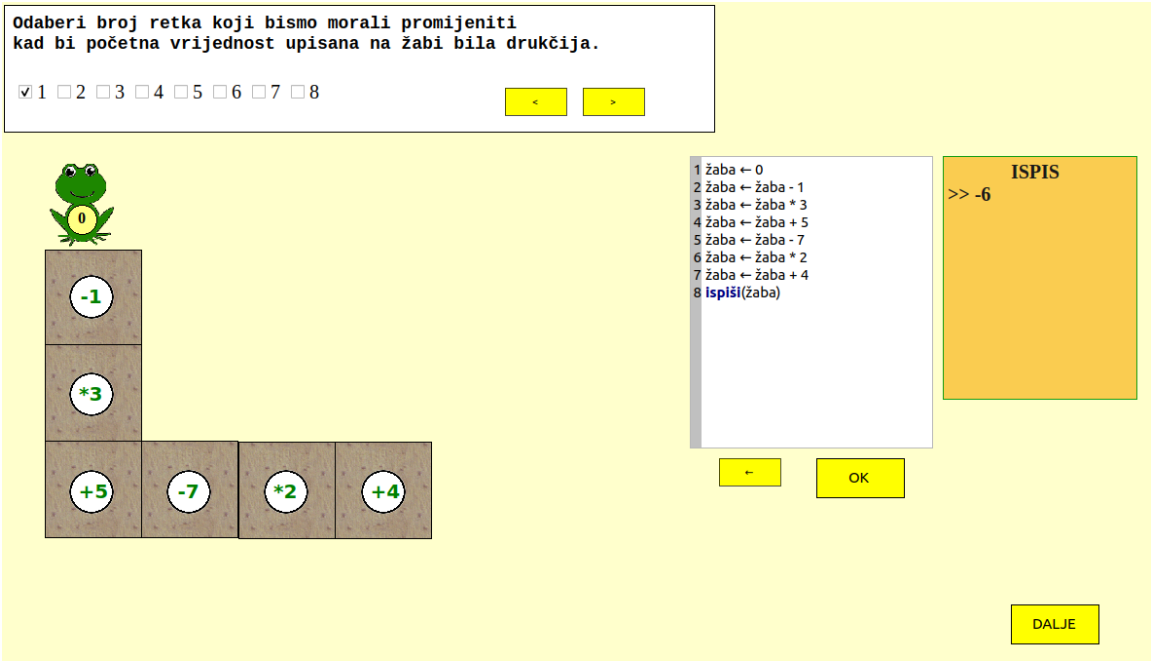
Već u prvoj razini korisnici su se imali prilike susresti s naredbom *ispiši(),* pomoću koje se ispisuju razni nizovi znakova na korisničko sučelje. U drugoj razini korisnik će se susresti s naredbom *upiši()*. Naredba *upiši()* je naredba unosa, točnije, pri izvršavanju programa, naredba *upiši()* omogućuje korisniku unošenje podatka. Uneseni podatak se obično pohrani u varijablu.
Ovo poglavlje imat će samo jednu razinu.

Na korisničkom sučelju nalazi se staza koja se sastoji od 6 polja, a na svakom polju nalazi se kružić s računskom operacijom. Lik žabe prolazi stazom i "skuplja" kružiće. Prije prolaska stazom u žabi je upisan broj „0". Kao i do sada, skupljanjem kružića s računskim operacijama mijenja se broj upisan u žabi. Klikom na gumb „Pokreni" pokreće se animacija u kojoj je prikazano kako žaba prolazi stazom, skuplja kružiće, a vrijednost zapisana na žabi se mijenja. Uz animaciju prikazan je i kod koji ju opisuje. Iznad animacije pojavi se tekst: „Vrijednost na žabi mijenja se ovisno o kružićima koje skupi dok prolazi stazom. Konačni rezultat ovisi o računskim operacijama na kružićima, ali i o broju koji je upisan na liku žabe prije prolaska stazom."



Slika 17. Početni dio animacije

Klikom na gumb „>” pojavi se nastavak teksta: „Odaberi broj retka koji bismo morali promijeniti kad bi početna vrijednost upisana na žabi bila drukčija.”



Slika 18. Zadatak

Ispod teksta ponuđeni su brojevi redaka, a zadatak korisnika je odabrati točan odgovor (ili odgovore).

Jedini redak koji je potrebno promijeniti je redak u kojem se nalazi naredba pridruživanja:

>> *žaba* ←0

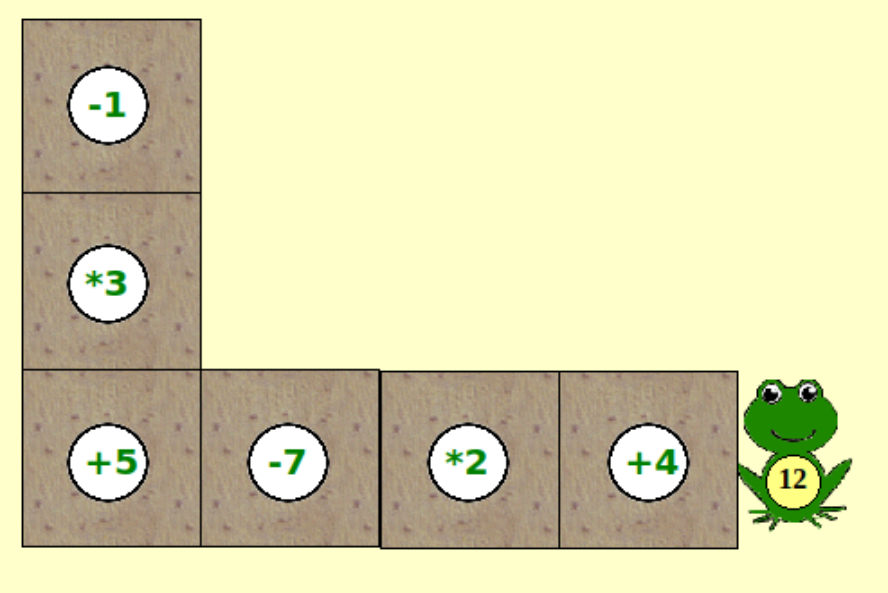
Nakon što korisnik točno odgovori na postavljeno pitanje pojavljuje se tekst s objašnjenjem naredbe *upiši()*. Zatim se pojavljuje tekst novog zadatka. U ovom zadatku korisnik treba upotrijebiti naredbu *upiši()* tako da dobiveni program računa vrijednost koja će pisati na žabi nakon što ona prođe stazom za bilo koji upisani broj.

Naredba *upiši()* služi korisniku programa za unos proizvoljnog cijelog broja.
Npr.
x ← upiši()
ispiši(x)
Pokretanjem programa otvara se dijaloški okvir. Korisnik programa unosi broj 5
koji se pohrani u varijablu *x* . Stoga, ispis je sljedeći:
>> 5

Slika 19. Objašnjenje naredbe upiši

```
1 žaba ← upiši()
2 žaba ← žaba - 1
3 žaba ← žaba * 3
4 žaba ← žaba + 5
5 žaba ← žaba - 7
6 žaba ← žaba * 2
7 žaba ← žaba + 4
8 ispiši(žaba)
```

Slika 20. Rješenje



Slika 21. Kraj animacije za početnu vrijednost 3

Poglavlje 5

Naredba grananja

Kao i kod naredbe unosa, kod naredbe grananja također, radi jednostavnosti, imamo samo jednu razinu. I u ovom primjeru korisnik ima priliku povezati pseudokod s prikazanom animacijom. Prikazana je cijev kod koje se na desnom kraju nalazi lik žabe, a na lijevom se nalazi lik jaja. Na likovima su upisani brojevi. Brojevi upisani na likovima će se mijenjati tako što će ih upisivati korisnik, a cijev se naginje na lijevu ili desnu stranu, ovisno o brojevima koji su na likovima prema sljedećem pravilu:

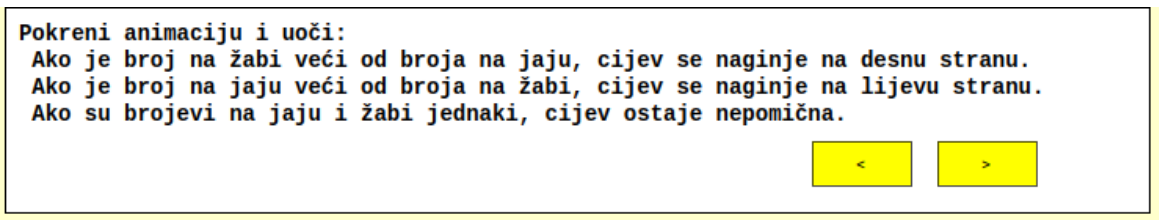
Ako je broj na žabi veći od broja na jaju, cijev se naginje na desnu stranu.

Ako je broj na jaju veći od broja na žabi, cijev se naginje na lijevu stranu.

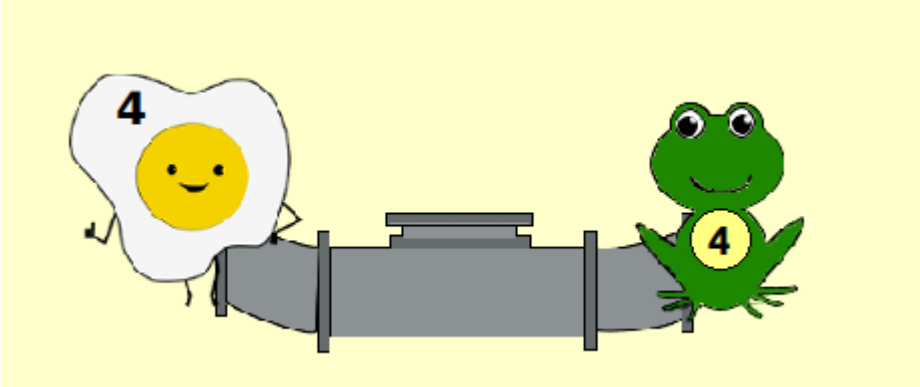
Ako su brojevi na jaju i žabi jednaki, cijev ostaje nepomična.

Ukratko, cijev se naginje na stranu lika u kojem je napisan veći broj. Ako su brojevi na likovima jednaki, cijev ostaje nepomična.

Sve spomenuto je objašnjeno u početnom tekstu koji se nalazi iznad animacije. Klikom na gumb s natpisom „Pokreni”, animacija se pokreće i odvija prema navedenim pravilima.



Slika 22. Početni tekst

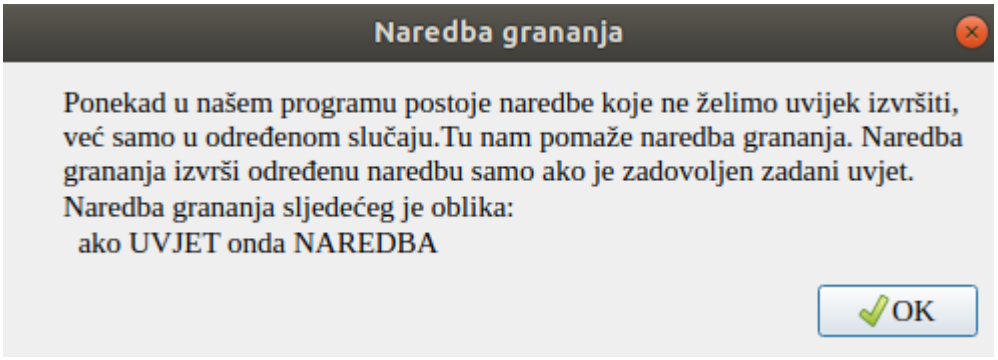


Slika 23. Isječak iz animacije

Nakon animacije pojavljuje se tekst: „Opišimo animaciju pomoću pseudokoda. Za upisivanje brojeva na likove koristit ćemo naredbu *upiši()*. Kad se cijev treba nagnuti na lijevu stranu, ispisat ćemo tekst „Lijevo”, a kad se mora nagnuti na desnu, ispisat ćemo tekst „Desno”, kad se ne naginje ispisat ćemo „Ravno”.”

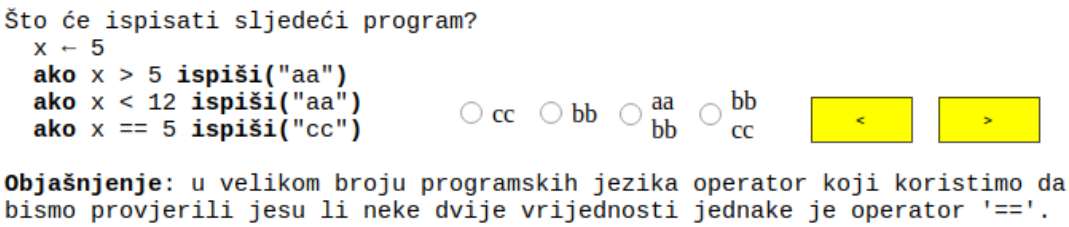
Klikom na gumb „>” pojavi se nastavak teksta: „Uočimo da u zadatku imamo određene uvjete, odnosno tek kad je pojedini uvjet ispunjen cijev se naginje (ili ostaje nepomična). Da bismo provjerili točnost uvjeta i izvršili odgovarajući dio koda koristimo naredbu grananja.”

Tekst „naredbu grananja” zapisan je na gumbu, Klikom na gumb otvara se dijaloški okvir s objašnjenjem naredbe grananja.



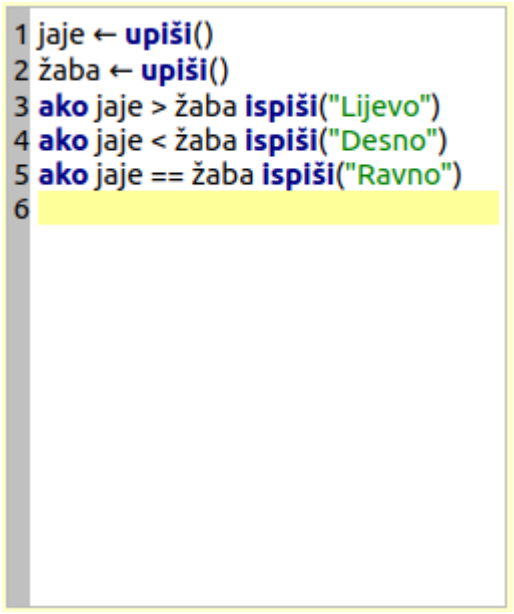
Slika 24. Dijaloški okvir s opisom naredbe grananja

U nastavku, korisnik rješava kratki kviz. U svakom zadatku kviza prikazan je pseudokod u kojem se koristi naredba grananja i korisnik mora označiti točan odgovor, tj. mora odrediti što će se ispisati. Kroz sam kviz objašnjeni su operatori usporedbe, a nakon kviza pojavljuje se gumb s natpisom „HINT". Klikom na taj gumb pojavi se kratki podsjetnik s operatorima usporedbe.



Slika 25. Prvi zadatak kviza

Nakon kviza ponovno se pojavi tekst zadatka u kojem se od korisnika traži da pseudokodom opiše animaciju.



Slika 26. Rješenje zadatka

Poglavlje 6
Naredba ponavljanja

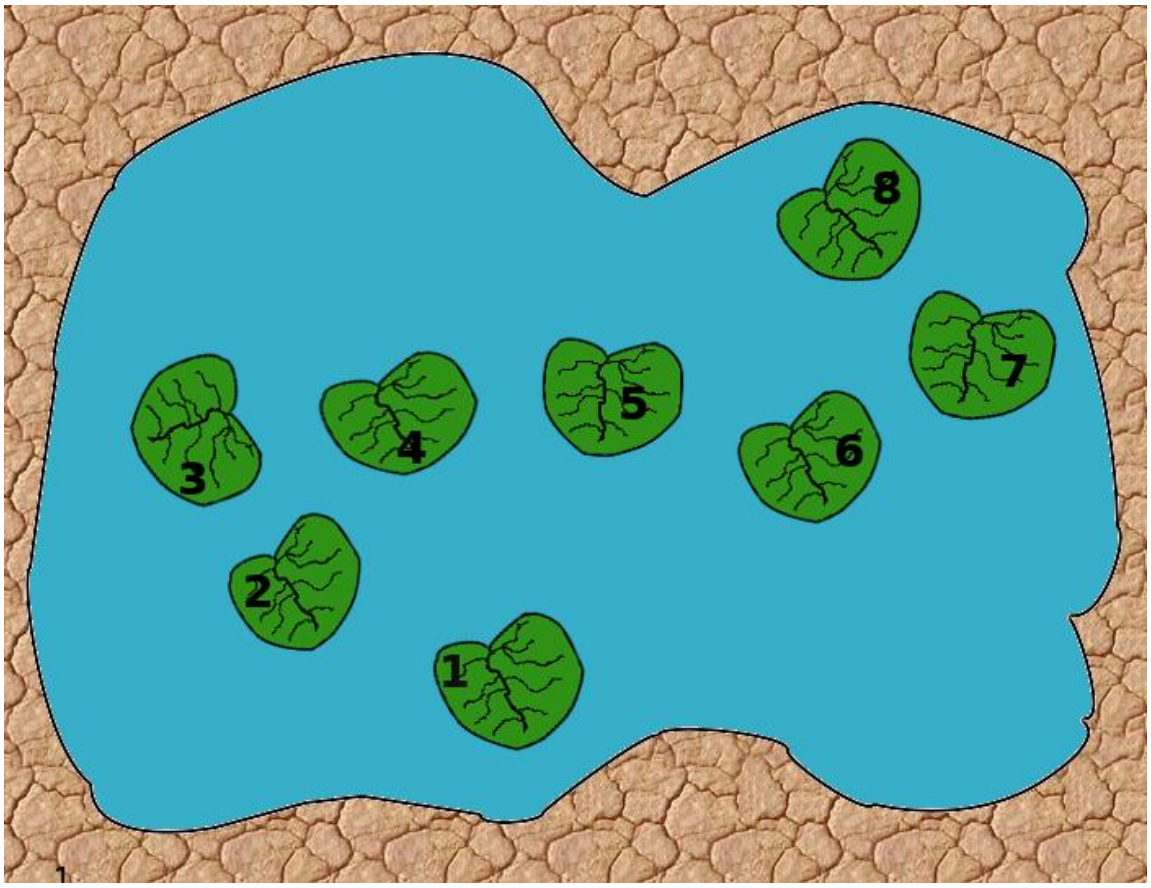
6.1 Razina 1 – Naredba ponavljanja s određenim brojem ponavljanja - Motivacija

Prva razina u ovom poglavlju počinje motivacijskim primjerom. Naime, izuzetno je važno da korisnik uoči potrebu za korištenjem naredbe ponavljanja, kako bi bio motiviran za njezino korištenje. Ideja je zadati korisniku zadatak čije rješenje zahtijeva ponavljanje iste naredbe više puta. Nakon toga korisniku je prikazan lakši i brži način izvršavanja istog zadatka koristeći naredbu ponavljanja.

Prikazano je jezero s lopočima koji čine put s jedne na drugu stranu. Kraj jezera nalazi se žaba koja treba prijeći jezero tako da skače s lopoča na lopoč. Lopoči su numerirani brojevima od 1 do 8.

Prikazan je tekst zadatka: „Žaba želi prijeći na drugu stranu jezera. Koristeći pseudokod pomozi žabi da prijeđe jezero.

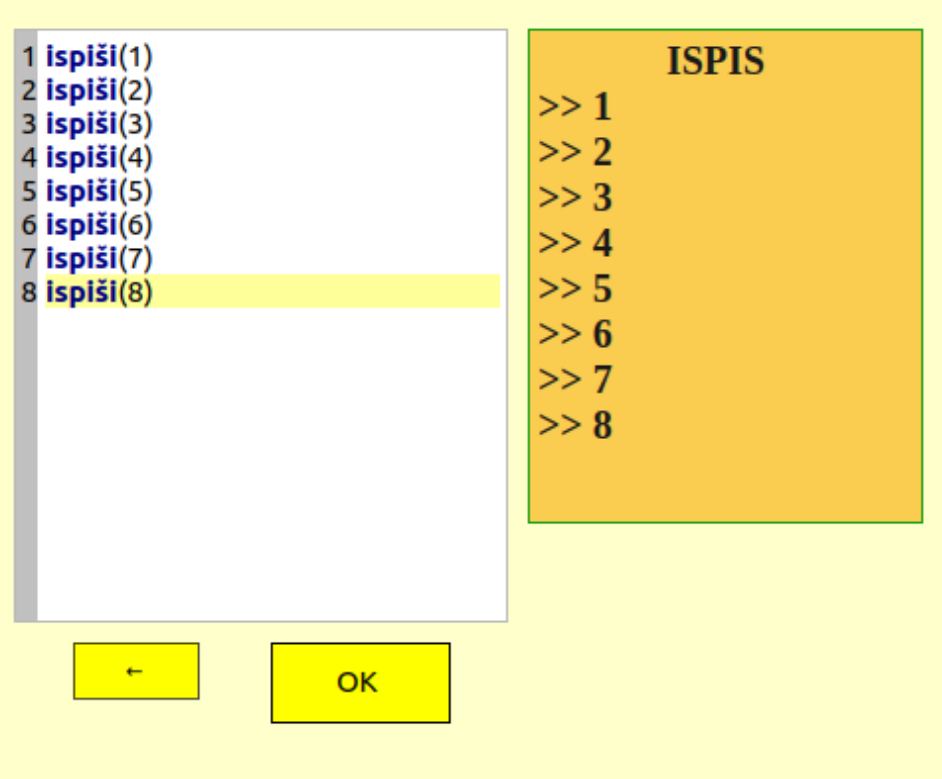
UPUTA: Svaki put kad ispišeš broj lopoča na ekran, žaba skoči na taj lopoč. Žaba ne može preskakati lopoče. Zadatak je potrebno izvršiti s minimalnim brojem redaka (naredbi).



Slika 27. Jezero s lopočima

Na desnoj strani nalazi se prostor za pisanje pseudokoda. Nakon što korisnik završi s pisanjem pseudokoda može kliknuti na gumb s natpisom "OK". Potom se provjerava točnost pseudokoda te se on izvršava (ako je sintaksa točna) i ispisuje na ekran odgovarajuće vrijednosti. Ako upisani pseudokod nije ispravan, ispisuje se odgovarajuća poruka.

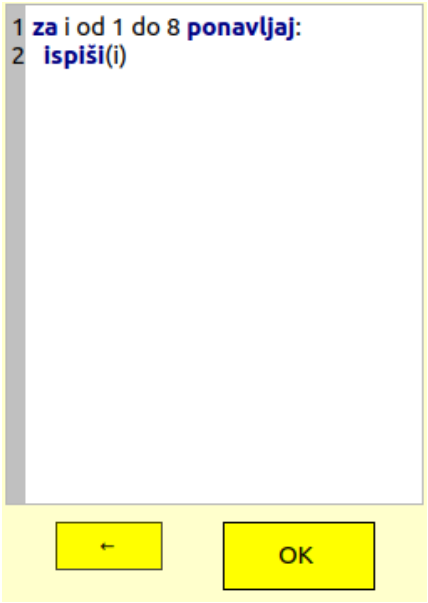
Nakon što se izvrši ispravno napisan kod, žaba skače po lopočima sukladno ispisu. Kad žaba dođe na zadnji lopoč, sama skoči na kraj jezera i zadatak je izvršen.



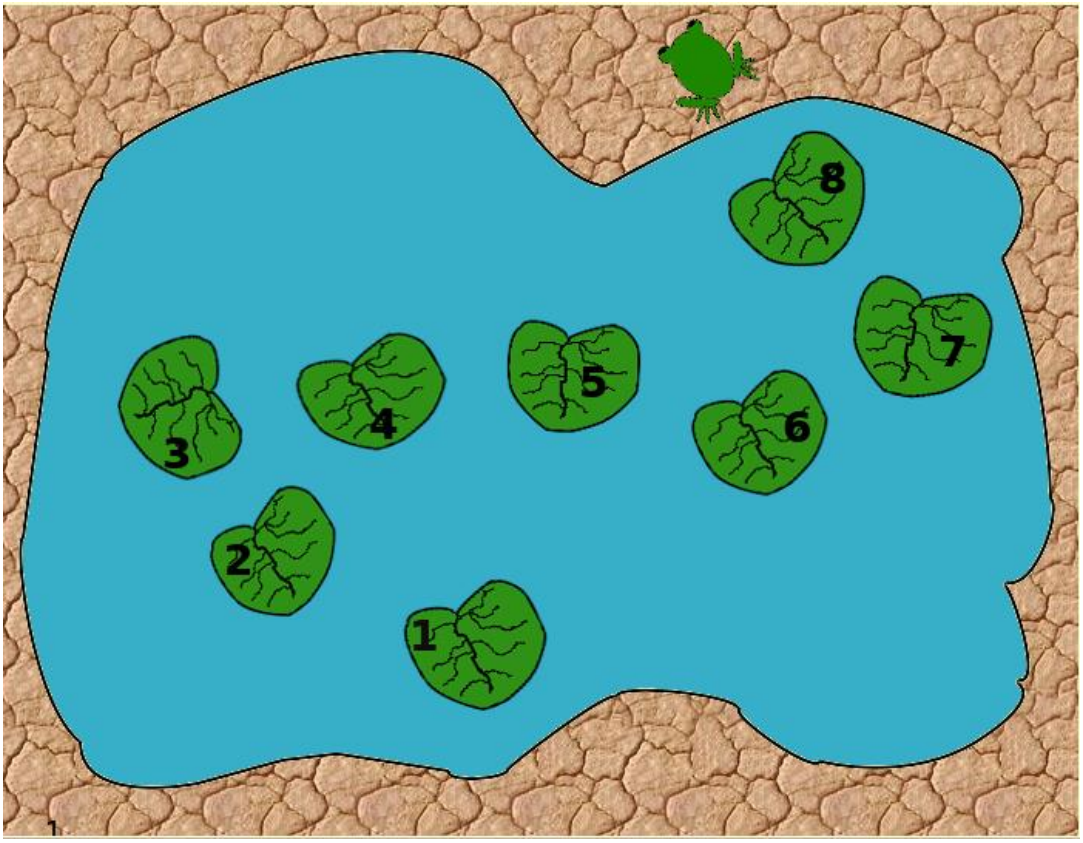
Slika 28. Rješenje zadatka

Na ekranu se pojavi gumb s natpisom “DALJE”.

Klikom na natpis dalje pojavi se opet isti zadatak, ali s već napisanim rješenjem. Ovog puta rješenje je zapisano korištenjem naredbe ponavljanja.



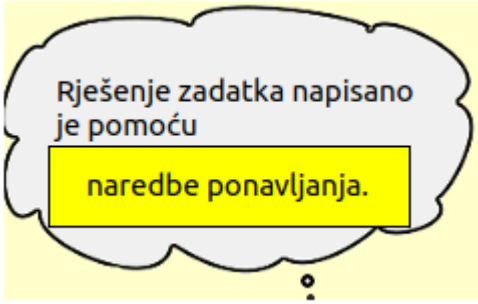
Slika 29. Unaprijed napisano rješenje



Slika 30. Žaba na drugoj strani jezera

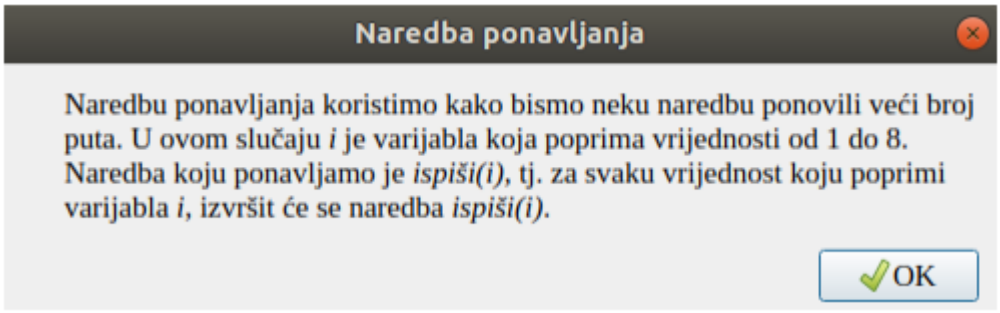
Klikom na gumb „OK“ provjerava se pseudokod napisan na slici 29. i pokreće se animacija. Na slici 30. prikazana je završetak animacije, tj. žaba je uspješno prešla na drugu stranu jezera.

Iznad pseudokoda nalazi se oblačić u kojem piše: „Rješenje zadatka zapisano je pomoću naredbe ponavljanja.”



Slika 31. Oblačić s porukom

Dio teksta „naredbe ponavljanja" zapravo je gumb. Klikom na gumb otvara se dijaloški okvir s objašnjenjem pojma.



Slika 32. Dijaloški okvir s objašnjenjem naredbe ponavljanja

6.2 Razina 2 – Naredba ponavljanja s određenim brojem ponavljanja

U drugoj razini primarni cilj je navesti korisnika da koristi naredbu ponavljanja pri rješavanju problema. Problem je postavljen na sličan način kao u prvoj razini. Opet, zadatak korisnika je prevesti žabu na drugu stranu jezera. U ovom slučaju žaba se kreće tako da preskače po jedan lopoč. Zadatak je nešto izazovniji od uvodnog primjera jer je za njegovo rješavanje, osim naredbe ponavljanja, potrebno iskoristiti i naredbu grananja.

Tekst zadatka je sljedeći: „Kao u prethodnom zadatku, žaba želi prijeći jezero. Ovog puta, pri svakom skoku žaba preskoči jedan lopoč. Koristeći naredbu ponavljanja prevedi žabu preko jezera."

Izgled grafičkog sučelja isti je kao u prethodnoj razini, ali zadatak je malo modificiran.
Korisnik rješenje zadatka utipkava u prostor za pseudokod. Klikom na gumb „OK" počinje provjera točnosti rješenja.
Ako rješenje nije točno, na ekranu se ispiše odgovarajuća poruka. Ako je rješenje ispravno, okvir prostora za ispis poprimi zelenu boju, a žaba počne preskakati lopoče.
Nakon što se izvrši animacija, na ekranu se pojavi gumb s natpisom „DALJE" koji vodi na sljedeću razinu.

6.3 Razina 3 – Naredba ponavljanja s logičkim uvjetom

Treća razina ima kratki uvodni dio u kojem je objašnjena upotreba naredbe ponavljanja s logičkim uvjetom. Korisnik je do sada već upoznat s pojmom logičkog uvjeta koji je sastavni dio poglavlja o grananju.
Prikazan je sljedeći tekst: „Što ako ne znamo unaprijed broj ponavljanja, već je naredbu potrebno izvršavati do određenog događaja?"

Ispod teksta nalazi se gumb sa znakom „>". Klikom na taj gumb pojavljuje se novi tekst: „Npr. koristimo naredbu ***upiši()*** kako bismo upisali neki prirodan broj na jaje. Potrebno je smanjivati vrijednost tog broja za 1, sve dok na jajetu ne bude ispisan broj 0."

Ponovo, klikom na gumb „>" pojavljuje se novi tekst: „U ovom slučaju koristimo naredbu ponavljanja s logičkim uvjetom."
Ispod teksta nalazi se prostor za pisanje pseudokoda, a napisan je sljedeći kod:

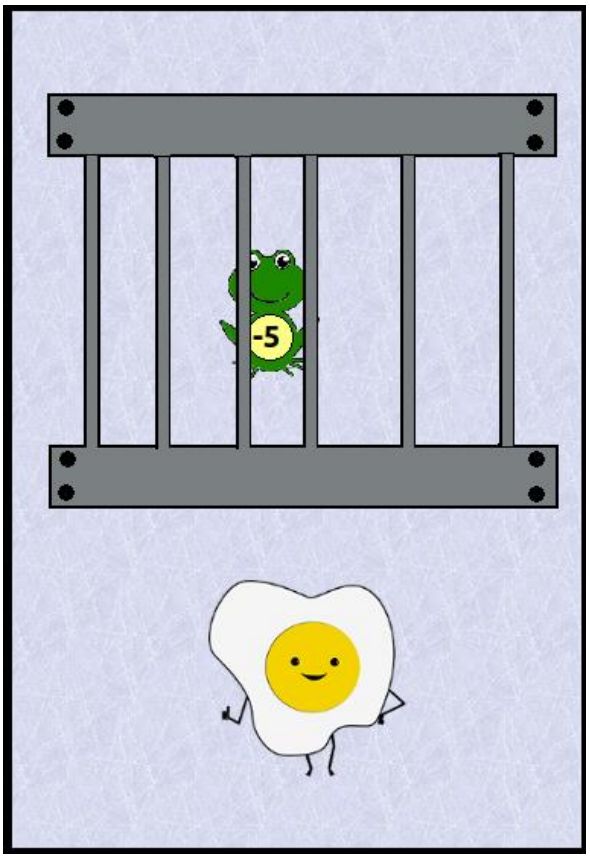
```
>> upiši(n)
>> dok je n > 0 ponavljaj n = n – 1
```

>> ispiši(n)

Klikom na gumb „OK” pseudokod se izvrši te možemo promatrati kako se mijenja broj na jajetu i kako se zaustavlja na nuli.

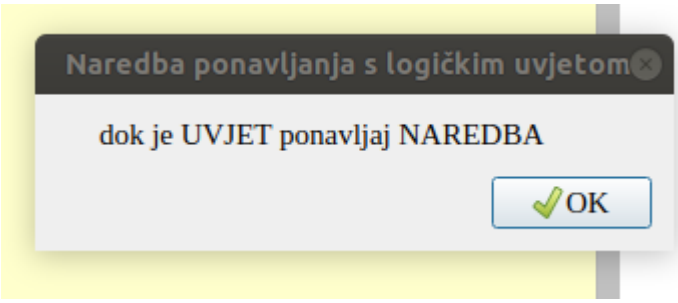
Nakon toga, pojavi se gumb s natpisom „DALJE”. Klikom na gumb pojavi se novi zadatak. Za rješavanje novog zadatka korisnik mora primijeniti naredbu ponavljanja s logičkim uvjetom.
U ovom zadatku potrebno je osloboditi žabu iz kaveza.
Tekst glasi: „Žaba je zarobljena i trebamo ju osloboditi iz kaveza! Kad se na žabi pojavi broj 100 kavez nestaje. Svaki put kad jaje skoči, udari glavom od kavez te se broj na žabi promijeni. Da bi jaje skočilo potrebno je ispisati „Skok”.”

Ispod teksta prikazan je lik žabe zarobljen u kavezu, a ispod kaveza je jaje koje ju treba osloboditi. Kao što je već navedeno u tekstu zadatka, kad jaje skoči, broj na žabi se promijeni. Ovaj put broj se mijenja nasumično pa zadatak neće biti moguće riješiti naredbom ponavljanja s unaprijed određenim brojem ponavljanja.



Slika 33. Početak animacije

Kraj prostora za pisanje pseudokoda, osim standardnih gumba s natpisima „OK” i „←” i na ovoj razini nalazi se gumb s natpisom „HINT”. Klikom na njega pojavljuje se dijaloški okvir s podsjetnikom na zapis naredbe ponavljanja.



Slika 33. Podsjetnik na zapis naredbe ponavljanja

Zaključak

U sklopu ovog diplomskog rada izrađen je alat za učenje programiranja namijenjen početnicima a prvenstveno učenicima 5. ili 6. razreda osnovne škole. Tijekom izrade bilo je potrebno suočiti se s raznim izazovima kako bi se ostvarila zamišljena ideja. Novost za autoricu prvenstveno je bila korištenje biblioteke Qt za izradu grafičkog sučelja i prilagođavanje na njezine funkcionalnosti. Već spomenuto svojstvo biblioteke Qt – *signals and slots* zahtijevalo je malu promjenu u algoritamskom načinu razmišljanja, ali se uskoro pokazalo kao veoma intuitivan alat.

Uz samo programiranje, važno je bilo uzeti u obzir mogućnost korištenja ovog alata na nastavi predmeta Informatike u petom razredu osnovne škole. Iz tog razloga pri pisanju ovog rada koristili smo se kurikulumom nastavnih predmeta matematika i informatika iz 2018. godine. Prema nacionalnom kurikulumu nastavnog predmeta informatika varijable, naredba unosa i naredba ponavljana dio su nastavnih sadržaja petog razreda osnovne škole, a naredba grananja dio sadržaja šestog razreda osnovne škole. Iz tog razloga, u ovom alatu dan je samo najjednostavniji koncept naredbe grananja.

U nastavku zaključnog dijela rada, kritički ćemo se osvrnuti na izrađeni alat. U svakom softveru uvijek postoji mjesta za nadogradnju i poboljšanje korisničkog iskustva, pa tako i u ovom. Za početak, poboljšanjem vještine korištenja programa za uređivanje slika moguće je znatno poboljšati izgled vizualnih elemenata, a samim time i opći dojam izgleda softvera. Nadalje, da bi se svim korisnicima pružilo što bolje iskustvo, jedna od ideja je napraviti dvije verzije softvera od kojih bi jedna bila za opću uporabu, a druga isključivo za korištenje u nastavi. U verziji za općenitu uporabu bilo bi moguće razdvojiti poglavlja igre, tako da sam korisnik odabire dio koji želi naučiti, bez da prolazi kroz sve razine.

S pozitivne strane, za razliku od brojnih postojećih alata za učenje programiranja, ovaj alat ne koristi blokovski način programiranja već korisnik mora samostalno pisati pseudokod, što stvara dobre osnove za daljnje učenje nekog određenog programskog jezika. Princip rješavanja zadataka u različitim poglavljima je podosta sličan, a opet se

razlikuje u raznim detaljima, što ga čini laganim za razumjeti, ali ujedno se izbjegava monotonost.

Na koncu, smatramo da je cilj koji je zamišljen pri početku izrade ovog alata uspješno ostvaren. Odnosno, izrađen je alat koji na jednostavan i slikovit način korisniku približava osnovne koncepte programiranja.

Bibliografija

[1] M. Babić, N. Bubica, S. Leko, Z. Dimovski, M. Stančić, N. Mihočka, I. Ružić, B. Vejnović, *#mojportal5: udžbenik informatike u petom razredu osnovne škole,* Školska knjiga, Zagreb, 2018.

[2] M. Babić, N. Bubica, S. Leko, Z. Dimovski, M. Stančić, N. Mihočka, I. Ružić, B. Vejnović, *#mojportal6: udžbenik informatike u šestom razredu osnovne škole,* Školska knjiga, Zagreb, 2018.

[3] *KURIKULUM NASTAVNOGA PREDMETA MATEMATIKA ZA OSNOVNE ŠKOLE I GIMNAZIJE U REPUBLICI HRVATSKOJ*, Narodne novine br. 7/2019, dostupno na https://narodne-novine.nn.hr/clanci/sluzbeni/2019_01_7_146.html (kolovoz 2019.).

[4] *KURIKULUM NASTAVNOGA PREDMETA INFORMATIKA ZA OSNOVNE ŠKOLE I GIMNAZIJE U REPUBLICI HRVATSKOJ*, Narodne novine br. 22/2018, dostupno na https://narodne-novine.nn.hr/clanci/sluzbeni/2018_03_22_436.html (kolovoz 2019.).

[5] Tynker: Coding for kids, About Us https://www.tynker.com/about/ (srpanj 2019.)

[6] Tynker https://en.wikipedia.org/wiki/Tynker (srpanj 2019.)

[7] Waterbear: A visual language for Javascript https://www.i-programmer.info/news/98-languages/2389-waterbear-a-visual-language-for-javacript.html (kolovoz 2019.)

[8] Waterbear, **https://waterbearlang.com/** (kolovoz 2019.)

[9] Robomind, **https://www.robomind.net/en/** (kolovoz 2019.)

[10] CodeCombat, **https://codecombat.com/** (kolovoz 2019.)

[11] About Qt, **https://wiki.qt.io/About_Qt** (rujan 2019.)

[12] Signals and slots, **https://doc.qt.io/qt-5/signalsandslots.html** (rujan 2019.)

[13] Qt 5.13, **https://doc.qt.io/qt-5/** (kolovoz 2019.)

[14] Qt 4.8, **https://doc.qt.io/archives/qt-4.8/** (kolovoz 2019.)

Sažetak

Glavni zadatak ovog diplomskog rada bio je izraditi alat za učenje programiranja. Sadržaji alata prilagođeni su učenicima 5. i 6. razreda osnovne škole, ali mogu ga koristiti svi koji žele naučiti osnovne koncepte programiranja.

Prije opisivanja izrađenog alata dan je kratak osvrt na neke od postojećih alata za učenje programiranja koji su namijenjeni sličnoj skupini korisnika. Slijedi objašnjenje najbitnijih detalja implementacije. Sam opis alata raščlanjen je na poglavlja čiji naslovi odgovaraju različitim konceptima programiranja koje korisnik treba usvojiti. Cjeline od kojih se sastoji alat su sljedeće: varijable, naredba unosa, naredba grananja i naredba ponavljanja. Tijek opisivanja ovih cjelina u radu prati tijek korisnikovog prolaska kroz cijeli alat. Na ovaj način moguće je pratiti korisnikov napredak te imati uvid u njegovu razinu znanja u svakom dijelu alata. U zaključku je dan kritički osvrt na izrađeni alat, istaknute su njegove prednosti, ali i razni dijelovi u kojima autor uočava priliku za napretkom.

Summary

The most important assigment of this thesis was creating a software which will be used to learn programming. Although this software is made for fifth and sixth grade students, it can be used by anyone who is eager to learn basic programming concepts.

Before we started to describe the software itself, there was a short review of some existing softwares which are intended to be used by the similar user group. Secondly, there was an explanation of some important implementation details. Furthermore, the software portrayal is separated to chapters which correspond to different programming concepts. These concepts are: variables, input, if-statement, loops. The course of this thesis follows the course of software user experience. This way, the reader is included in each step of the user's experience and is able to be aware of the user's current knowledge level. In conclusion, there is a critical review of this software. The autor gives her opinion about this software's advantages and improvement options.

Životopis

Rođena sam 21. lipnja 1994. godine u Šibeniku kao prvo dijete Branka i Nediljke Peran. Nakon završene jezične gimnazije u Šibeniku upisala sam preddiplomski sveučilišni studij Matematika; smjer: nastavnički na Prirodoslovno-matematičkom fakultetu Sveučilišta u Zagrebu. Diplomski sveučilišni studij Matematika i informatika; smjer: nastavnički, također na Prirodoslovno-matematičkom fakultetu u Zagrebu, upisala sam u rujnu 2017. godine.