Univerza na Primorskem

Fakulteta za matematiko, naravoslovje in informacijske tehnologije

Al Kinkopf

Izdelava aplikacije Igrajmo matematiko

Zaključna naloga

Koper, september 2012

Mentor: doc. dr. Peter Rogelj
Somentor: doc.dr. Branko Kavšek

Zahvala

Rad bi se zahvalil mentorju in somentorju za ves trud, ki sta ga vložila v pregledovanje pojmov in idej prisotnih v tej nalogi. Rad bi se zahvalil tudi Roku Ziherlu za pomoč pri oblikovanju grafičnega vmesnika. Zahvalil bi se tudi Tanji Sinković za pomoč pri iskanju slovničnih in tiskarskih napak.

KAZALO

KAZALO TABEL

KAZALO SLIK

Povzetek

V nalogi je predstavljeno načrtovanje aplikacije “Igrajmo matematiko”, namenjeno osnovnim šolam. Predstavljena je večina faz razvoja z izjemo vzdrževanja. Začetna poglavja opisujejo načrtovanje sistema, zadnja poglavja pa implementacijo in testiranje. Skozi celotno nalogo bom podrobneje opisal programske procese, in sicer: definicija problema, študija izvedljivosti, analiza in definiranje zahtev, načrtovanje sistema oziroma programa, izvedba ter testiranje. Prevzem in izdaja ter obratovanje in vzdrževanje ne bodo opisani, saj kupec aplikacije ne obstaja. Implementacija je izvedena s programskim jezikom Java. Vsa uporabljena programska oprema je brezplačna. Končni izdelek bo aplikacija, ki bo omogočala istočasno učenje matematike in igranje oziroma zabavo. Aplikacija bo nosila ime “Igrajmo matematiko” in bo namenjena osnovnošolcem do petega razreda ter učiteljem oziroma staršem. Z njo bomo učencem matematiko predstavili na drugačen način, saj je zastavljena kot zabava oziroma sprostitev. Vsebovala bo tako enoigralski kot večigralski način, pri čemer se bodo učenci pri igranju lahko medsebojno družili. Vse igre, ki jih bo vsebovala aplikacija, bodo zasnovane na reševanju preprostih matematičnih računov. Staršem in učiteljem bomo ponudili možnost sestavljanja nalog tako, da bodo lahko sami odločali, kaj bodo učenci oziroma njihovi otroci reševali.

Ključne besede: matematika, matematične operacije, matematične enačbe, aplikacija, učenje, osnovnošolci, zabava, igra, Java.

Abstract

This thesis presents the development of an application called “Igrajmo matematiko”, which is intended to be used in elementary schools. All stages of development are presented, with exception of maintenance. The initial chapters describe the planning of the system and the few last chapters describe the implementation and testing [2] . Throughout the whole dissertation there are further described software phases: the definition of the problem, the feasibility study, the analysis and definition of requirement, software planning, implementation and testing. Acquisition, issue, operation and maintenance of program are not described because the customer does not exist. The components of the application are written in Java programming language. All software used for the developing is free. The final product is an application which allows learning the basis of mathematics in a fun way, therefore school and fun are combined at the same time. The application, which is named »Igrajmo matematiko«, is designed for elementary school pupils up to fifth grade, for teachers and for parents. With it we will present mathematics in a different, funny and alternative way for children. It contains single-player and multiplayer way, so that pupils can socialize while studying. All contained games are based on solving simple mathematic tasks and calculations. Parents and teachers have the possibility of the assembling tasks, therefore they can decide, what children need to study.

Keywords: math, matematchial operataions, matemathical equations, application, study, elementary school pupils, fun, game, Java.

1. Uvod

Matematika je kraljica vseh znanosti. Z njo se srečamo že v zgodnjem otroštvu, ko razmišljamo, katere lego kocke spadajo skupaj. Spremlja nas celo življenje pri najrazličnejših vsakdanjih opravilih, ki jih počnemo naučeno in samoumevno, pri čemer se podzavestno uporabljenega znanja s področja matematike ne zavedamo.

Nekaterim gre matematike boljše, drugim slabše. To velja zlasti za dijake in osnovnošolce, saj je eden izmed krivcev za slabo voljo in mogoče celo za neprespane noči. Ker se z njo srečamo že kot otroci, bi bilo dobro, da jo spoznamo kot nekaj zabavnega in zanimivega, zaradi česar bi nam kasneje učenje utegnilo biti lažje in zanimivejše.V današnji družbi informacijske tehnologije postajajo vse pomembnejša.Že od otroštva nas namreč na vsakem koraku spremljajo računalniki. Če bi torej združili učenje osnov računalnika ter učenje matematike, bi lahko dosegli razumevanje dveh pomembnih znanosti hkrati [8] .

Cilj te naloge je narediti program oziroma aplikacijo, ki bi osnovnošolce učila matematiko na neformalen in njim bolj zabaven način. To se razlikuje od formalne uporabe učbenika in zvezka. Poleg končnega izdelka bomo v nalogi predstavili postopke oziroma faze programskega procesa, ki vodijo do produkta. Predstavili bomo programske procese, definicijo problema, študijo izvedljivosti, analizo in definiranje zahtev, načrtovanje sistema oziroma programa, izvedbo ter testiranje [10] . Preskočili bomo le prevzem in izdajo ter obratovanje in vzdrževanje, saj kupec apikacije ne obstaja. Aplikacija bi bila namenjena istočasnemu spoznavanju računalnika, njegovih osnovnih funkcij in osnovnih matematičnih pravil oziroma operacij. Poleg tega bi bila v osnovi zasnovana kot računalniška igra. Namenjena bi bila tako učencem kot njihovim staršem, ter tudi učiteljem matematike.

V preostalem delu zaključne naloge sledi v drugem poglavju opisana podrobno definicija problema, nato pa opis študije izvedljivosti. Po študiji izvedljivosti sledijo opisi vseh komponent aplikacije, kjer za lažjo predstavitev uporabljamo tudi diagrame. Opisu komponent sledita poglavji o načrtovanju in izvedbi, nato pa še samo testiranje programa, v katerem smo opisali najdene napake in rešitve. V zaključku sledi kratek povzetek zaključne naloge ter ideja o nadaljnem delu.

2. Definicija problema

Problem pri reševanju težjih matematičnih nalog večkrat predstavlja naše predznanje osnov matematike.

Menimo, da bi morali ob vstopu v osnovno šolo ustvariti pozitiven odnos do matematike, ker se z njo kot znanstveno vedo in njenim neposrednim učenjem srečamo že v prvem razredu. Predšolski otroci niso vajeni učenja in reševanja računov v zvezek, saj se pred tem niso srečali z učnimi situacijami. Problem, ki ga opisujemo, predstavlja učenje matematike na zabaven način. Ker se matematike v šoli učimo s pomočjo učiteljev, le-ti precej pripomorejo k odnosu, ki ga razvijemo do te znanstevene vede. Zato smo mnenja, da lahko prav učitelji pripomorejo k temu, da bi se ob matematiki zabavali. To bi lahko dosegli z raznimi igrami, ki temeljijo na matematiki ali pa z alternativnim prikazom nevsakdanjih matematičnih primerov na tablo. Številke bi lahko predstavili na bolj grafičen ali risan način, saj imajo otroci, sploh v teh letih, zelo radi risanje. Menimo, da primanjkuje računalniških aplikacij, ki bi otrokom pomagale pri učenju matematike, še posebaj takšnih, ki vsebujejo neformalen pristop k učenju, kot je igra. Pri tem je potrebno narediti sistem, ki bo dovolj preprost, da ga bodo znali uporabniki uporabljati na pravilen način. Istočasno naj bi bil dovolj zabaven in poučen, da bi z njim dosegli zastavljene cilje, kot so učenje matematike, računalniških osnov in zabave hkrati. Med zadanimi cilji je najpomembnejši narediti aplikacijo, ki bo vsebovala več manjših pod-aplikacij oziroma funkcij. Ena izmed funkcij je igra, v kateri uporabnik rešuje matematične enačbe na zabaven način, s čimer dosežemo, da se uporabnik ob učenju zabava.

Težava, ki se pri tem lahko pojavi, je izmisliti si najbolj preprost in zabaven način učenja. Druga funkcija je vrsta večuporabniške izkušnje, pri kateri ponudimo dvema uporabnikoma istočasno igranje. Uporabnika med seboj tekmujeta v reševanju raznovrstnih matematičnih enačb. Težava, ki lahko nastopi, je narediti vmesnik, ki dovoljuje istočasno igranje dveh uporabnikov na enem računalniku oziroma ekranu. Poleg tega obstaja možnost nepravilne sinhronizacije dogodkov, saj bi bilo v igro zanimivo vključiti tudi spremenljivko oziroma dejavnik časa. S tekmovalno igro bi dosegli, da bi se uporabniki matematike v aplikaciji učili tudi iz razloga, da premagajo nasprotnika. Zadnja funkcija ponuja možnost reševanja domačih nalog. Funkcijo smo si zamislili tako, da bi lahko učitelj ali starš na splet naložil nalogo, aplikacija bi jo nato prebrala oziroma sprejela s spleta in jo prikazala uporabniku, ki bi jo imel možnost rešiti ter jo nazaj posredovati učitelju ali staršu. Težavo pri tej funkciji lahko predstavlja pošiljanje rešene naloge učitelju ali staršu nazaj.
Prav tako pa je potrebno učitelje oziroma uporabnike, ki bodo naloge naložili na splet, usposobiti za delo s predstavljeno aplikacijo.

3. Študija izvedljivosti

Pri izvedljivosti projekta imamo dve večji oviri. Prva ovira so denarna sredstva, ki pa je v tem primeru nična, saj gre za univerzitetni projekt. Druga ovira je rok izdelave. Torej je poleg pravočasnega zaključka projekta glavni cilj še izpeljava le-tega brez dodatnih stroškov ob lastnemu delu. Odločitvi, da bomo projekt realizirali brez danih sredstev sledi, da bo vsa uporabljena programska oprema brezplačna [1] .

Ker gre za aplikacijo, sestavljeno iz več podaplikacij, ki bodo kasneje skupaj predstavljale celoto, je potrebno projekt razdeliti na več ločenih komponent. Prva komponenta aplikacije bo grafični vmesnik, ki bo deloval kot vrsta okvirja za ostale podaplikacije oziroma igre našega zaključnega izdelka. Po izdelavi grafičnega vmesnika bo možno posamezne podaplikacije razvijati in testirati ločeno, dokler ne bomo prišli do željenega rezultata. Ko bomo to dosegli, bomo lahko vse skupaj združili. Posledično bomo lahko vzporedno ravijali neodvisne komponente in s tem skrajšali čas razvoja. Vendar pa v našem konkretnem primeru ta vidik ni toliko pomemben, saj ekipo sestavlja le en član. Glavni razlog razdelitve projekta na manjše komponente je v tem primeru lažje obvladovanje posameznih aplikacij ter lažje testiranje posameznih komponent.
Zaključni rok projekta je sredina septembra 2012, kar pomeni, da smo časovno omejeni na približno štiri mesece. V danem času je potrebno zaključiti pet komponent našega projekta.
Prva komponenta, s katero se bomo ukvarjali, je torej grafični vmesnik [3] . Ko bo grafični vmesnik narejen, se lahko lotimo še ostalih štirih samostojnih komponent oziroma iger znotraj naše aplikacije. Med temi komponentami so tri enako zahtevne enoigralske igre.
Četrta komponenta je večigralski način ene izmed enoigralskih iger, zato menimo, da bomo ob implementaciji večigralske igre brez večjih težav implementirali še enoigralsko igro.
Za izdelavo grafičnega vmesnika bomo porabili približno tri tedne. Nato bomo imeli za vsako izmed komponent približno dva meseca časa, saj jih bomo razvijali istočasno. V tem primeru jih bom razvijal sam in bom tako imel za vsako približno dva do tri tedne za izdelavo in testiranje ter seveda odkrivanja napak. Menimo, da bo največ odkritih napak pri implementaciji večigralske igre, saj se tam izvaja največ niti hkrati. Časovni roki so postavljeni okvirno, projekt pa mora biti končan vsaj tri tedne pred iztekom roka zaradi raznovrstnih testiranj in odkivanj napak.
Zakonov z našim projektom v nobenem primeru ne bomo kršili, saj ne gre za zlonamerno aplikacijo, temveč za interno uporabo znotraj šole ali doma, s katero nimamo namena pridobiti protipravnega premoženja. Zaradi dobre prenosljivosti in podpore bomo za izdelavo projekta uporabili programski jezik Java, ki je zelo priljubljen in podprt na vseh popularnih operacijskih sistemih.

Poleg že navedenega pa bomo Javo uporabili zaradi same tehnične podkovanosti ter brezplačnih knjižic, ki jih Java ponuja. Za samo uporabo aplikacije se bo lahko tako uporabljal tudi brezplačni in odprtokoden operacijski sistem Linux. Najdemo ga namreč v različnih distribucijah, izmed katerih bo lažje najti tisto, ki je pisana na kožo najrazličnejšim uporabnikom.
Vsa tehnologija in protokoli, ki se bodo uporabljali pri implementaciji aplikacije, so prosto dostopni, brezplačni in odprtokodni.

4. Analiza in definiranje zahtev

V tem poglavju bomo predstavili analizo in definicijo zahtev za aplikacijo “Igrajmo matematiko”. Ta del navadno opravi sistemski analitik v sodelovanju z naročnikom ali uporabnikom. Ker pa gre v našem primeru za univerzitetni projekt, naročnik ne obstaja oziroma smo naročniki mi sami. Končni uporabnik aplikacije bo učenec osnovne šole in učitelj ali starš [11] . Ker imamo možnost direktnega kontakta s končnim uporabnikom, lahko lažje izvedemo realne potrebe in želje.

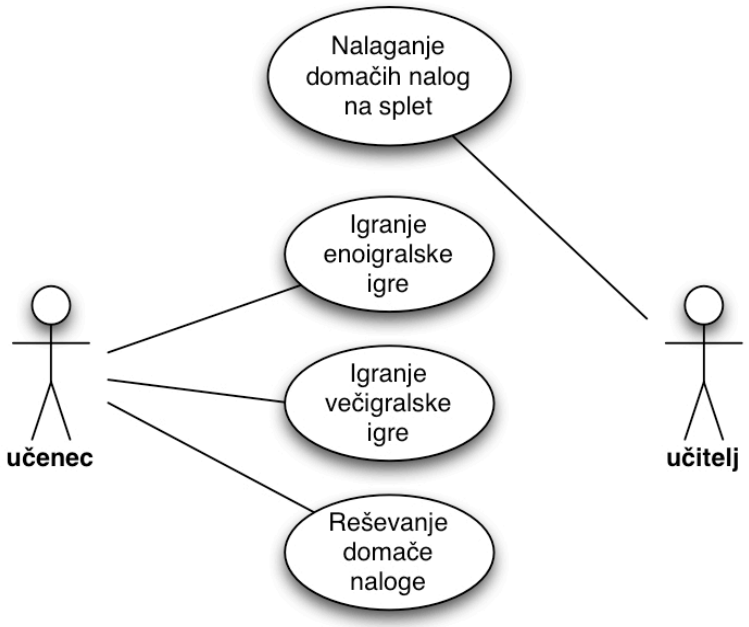
4.1 Namen aplikacije

Aplikacija se razvija za uporabo v odprtem krogu ljudi, ki si želijo učenja matematike. Glavni namen aplikacije je pokazati, da je matematika zanimiva in tudi zabavna, če se je lotimo na pravilen način. Same zahteve tako izhajajo iz potreb po boljšem in zabavnejšem učenju matematike. Zahteve so naslednje:

-reševanje matematičnih enačb z možnostjo nastavljive časovne omejitve in težavnosti,
-hitrostno tekmovanje v reševanju matematičnih enačb z drugim igralcem,
-reševanje nalog na osnovi matematike in pošiljanje rešitev preko spleta,
-barvitost aplikacije primerna osnovnošolskim otrokom,
-uporaba zvoka za določene aktivnosti v aplikaciji.

Z razvitjem aplikacije bo uporabnik razvil pozitiven odnos do matematike, spoznal njene osnove in pravila ter osnove računalnika.

4.2 Primeri uporabe



Slika 1: Diagram primera uporabe aplikacije

Diagram prikazan na sliki 1 prikazuje način uporabe aplikacije [4] . Zaženemo jo z dvojnim klikom na ikono. Temu sledi prikaz glavnega menija, v katerem imamo na izbiro dve enoigralski in eno večigralsko oziroma dvoigralsko igro. Vse igre temeljijo na reševenju matematičnih nalog oziroma enačb. Poleg treh navedenih iger lahko uporabnik izbere še reševanje domače naloge.

4.3 Enoigralski način

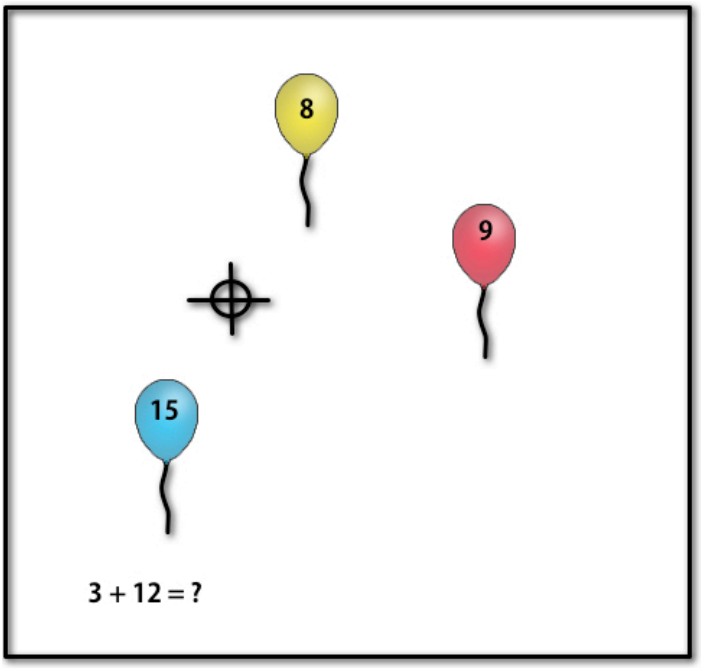
Enoigralski način predstavljajo igre, ki so namenjene samo enemu igralcu. Sledijo opisi treh enoigralskih iger, ki jih vsebuje aplikacija.

4.3.1 Poči pravi balon

Opis igre:

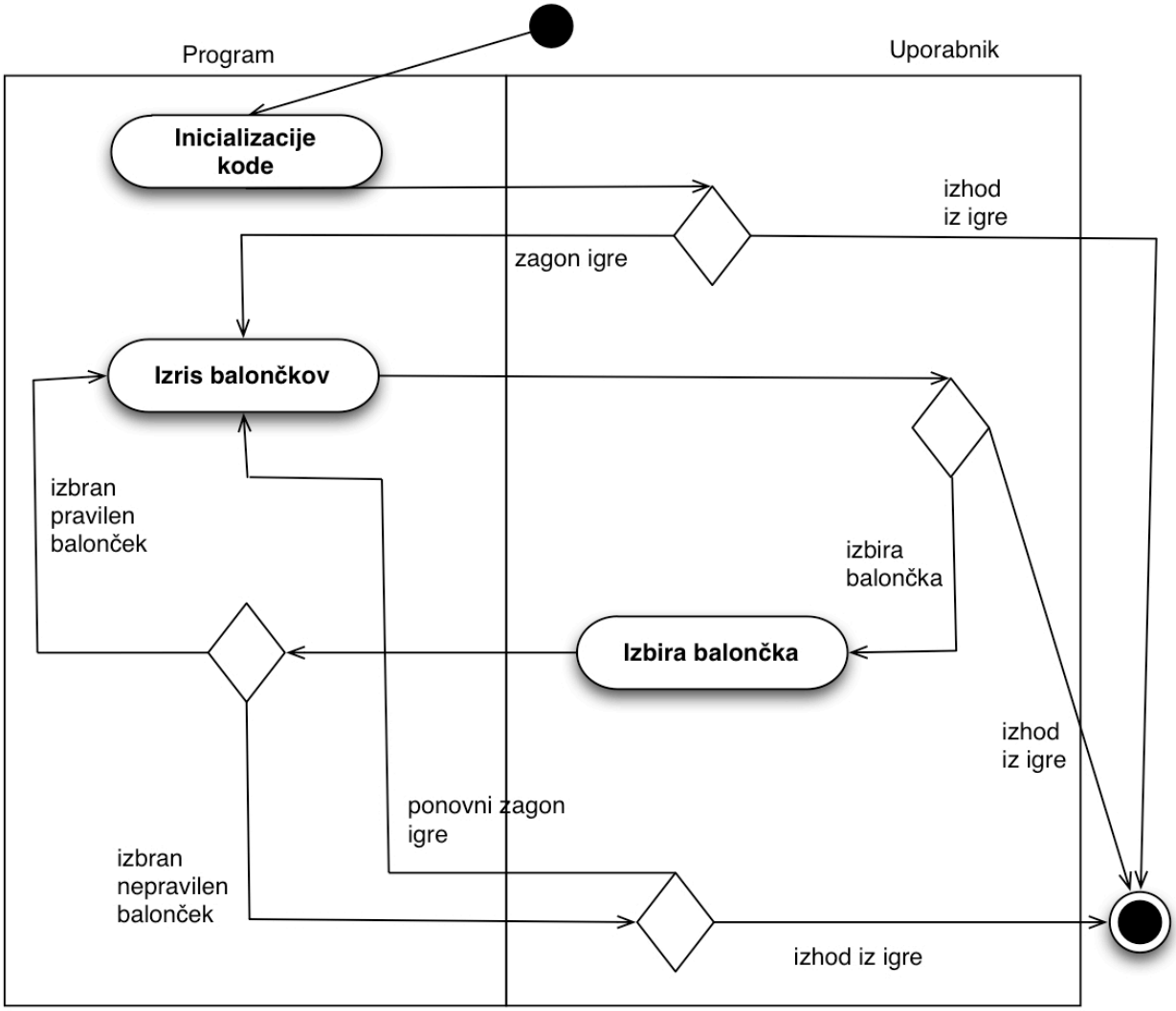
Uporabnik aplikacije "Igrajmo matematiko" ima na izbiro dve enoigralski igri, ki temeljita na reševanju matematičnih enačb. Prva igra, ki jo ima uporabnik na izbiro, se imenuje "Poči pravi balon". V igri mora uporabnik izmed raznih balonov, ki se mu prikazujejo po ekranu, izbrati tistega s pravilno rešitvijo.

Kadar igralec zažene igro, se mu na spodnjem delu ekrana oziroma grafičnega vmesnika prikaže enačba v obliki "4 + 6 = ?" kot je prikazano na sliki 2. Enačbe lahko vsebujejo operacijo seštevanja, odštevanja, množenja ali deljenja. Ob pojavu enačbe se na ekranu istočasno pojavijo tudi trije baloni, ki vsebujejo tri možne rešitve za dano enačbo. Kadar igralec izbere pravilen balon, se njegove točke zvišajo oziroma rezultat poveča za eno točko, istočasno pa se na dnu ekrana pojavi nova enačba ter novi baloni z rešitvami. Za njih je značilno, da se premikajo od dna proti vrhu z različnimi hitrostmi (včasi gredo tudi vsi istočasno), pri tem pa mora igralec klikniti na pravilen balon, preden le-ta doseže vrh. Če balon s pravilno rešitvijo doseže vrh preden ga igralec klikne, izgubi igro. Število balonov se povečuje glede na število doseženih točk.
Kadar igralec izgubi, lahko igro preprosto ponovno zažene s klikom na gumb *restart*. Kadarkoli pa lahko igralec igro zapusti in se vrne v osnovni meni aplikacije s klikom na tipko *izhod*.



Slika 2: Skica igre "Poči pravi balon"

Diagram aktivnost za igro “Poči pravi balon” je prikazan na sliki 3 [6] :



Slika 3: Diagram aktivnosti igre “Poči pravi balon”

Potek diagrama aktivnosti za igro “Poči pravi balon” (* predstavljajo alternative) :

1.) Uporabnik zažene igro.
2.) Izrišejo se balončki z rešitvami.
3.1.) Uporabnik izbere pravilen balonček.

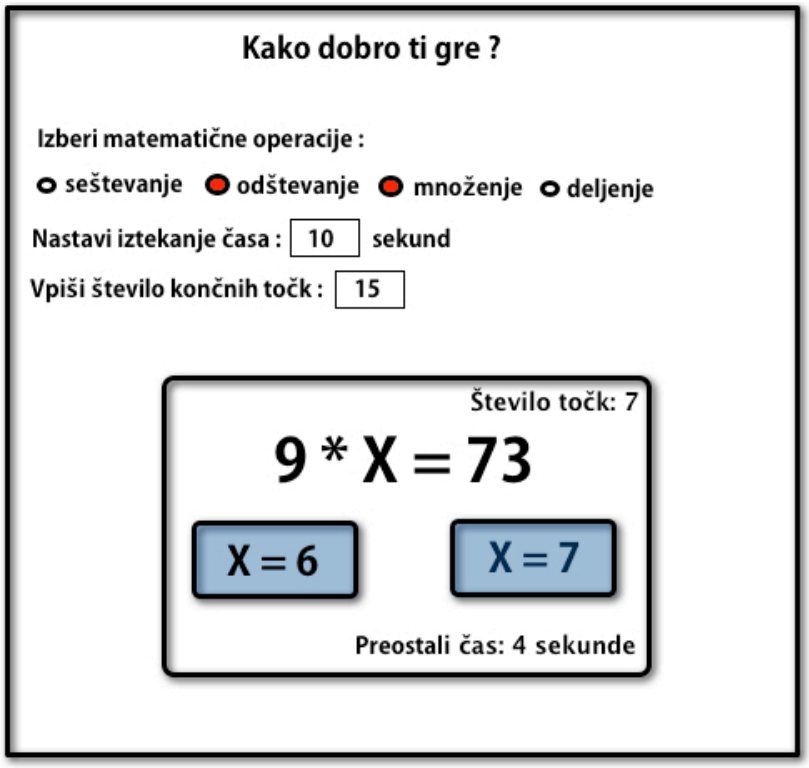
*3.2.) Alternativa: uporabnik ponovno zažene igro (1.).
*3.3.) Alternativa: uporabnik zapusti igro.
*3.4.) Alternativa: uporabnik je izbral napačen balonček.
*3.4.1) Uporabnik ponovno zažene igro (1.).
*3.4.2) Alternativa: uporabnik zapusti igro.

4.) Uporabniku se ponovno izrišejo novi balončki.

4.3.2 Kako dobro ti gre?

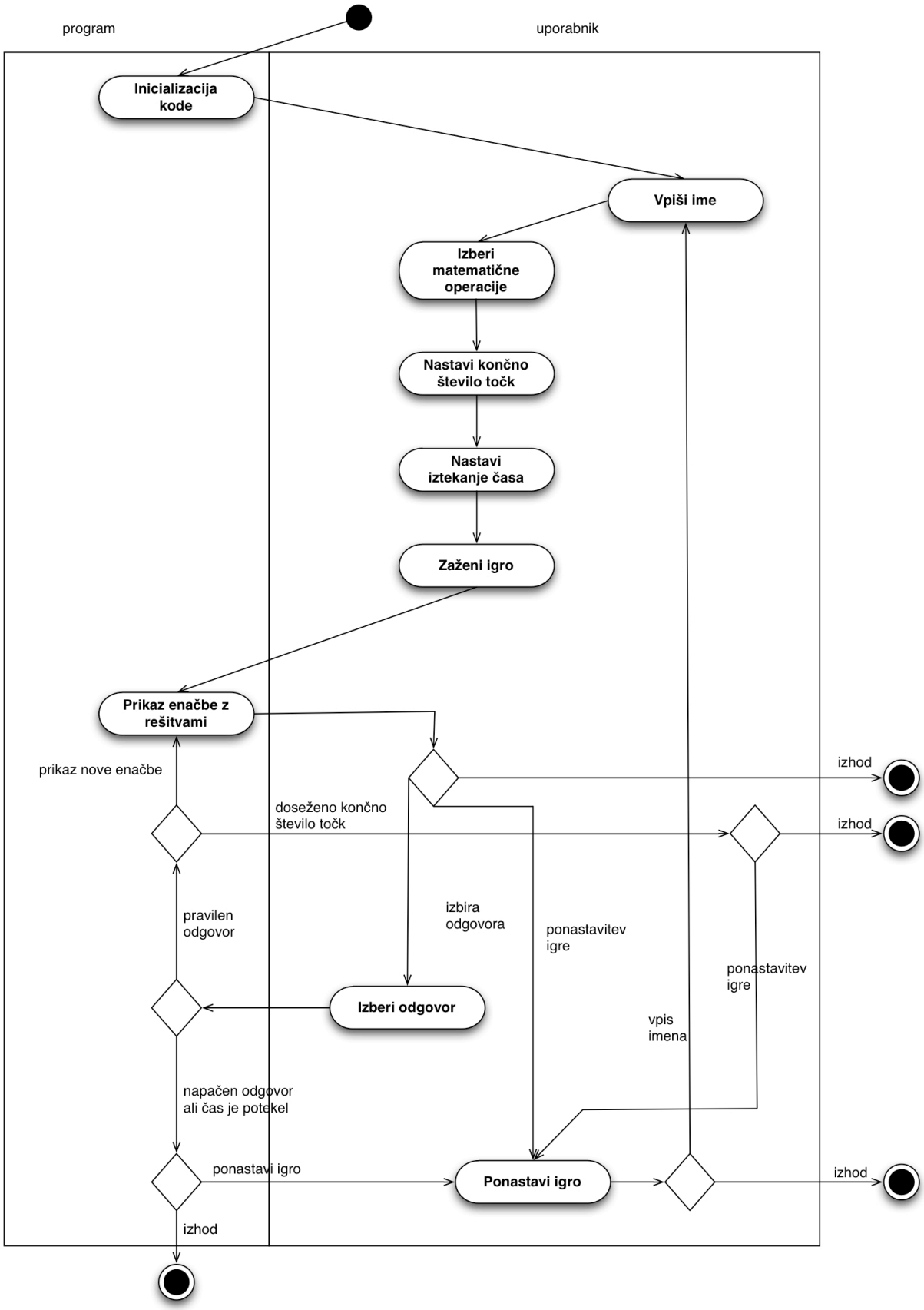
Opis igre:

V enoigralskem načinu vsebuje aplikacija poleg igre “Poči pravi balon”, še eno igro.
Gre za igro “Kako dobro ti gre”, v kateri gralec izbira med štirimi matematičnimi operacijami - množenjem, seštevanjem, deljenjem in odštevanjem. Izbere lahko eno, dve, tri ali štiri matematične operacije, s katerimi se želi spopadati v igri. Natančneje, če igralec izbere deljenje in množenje, bo moral kasneje v igri reševati matematične enačbe, temelječe na množenju ali deljenju.
Za vsako enačbo, ki se izpiše, primer: 4 + x = 10, ima igralec na izbiro dve možni reštivi za spremenljivko x kot je prikazano na sliki 4. V primeru, da izbere pravilno rešitev, dobi točko. Temu sledi prikaz nove enačbe. Poleg matematičnih operacij si uporabnik lahko sam izbere tudi končno število točk ter čas, ki ga ima na voljo za rešitev matematične enačbe. Če igralcu ne uspe izbrati pravilne rešitve v danem času, izgubi. V nasprotnem primeru se mu prikaže nova enačba, čas, ki ga ima na voljo za rešitev enačbe pa se resetira. Če igralec v danem času izbere napačno rešitev, se igra konča. Po končani igri ima igralec na izbiro dve možnosti. Lahko znova začne igro, kjer si sam nastavi matematične operacije, iztekanje časa ter končno število točk ali pa zapusti igro ter se tako vrne v glavni meni aplikacije. Kot je že omenjeno, ima igralec možnost nastavitve končnega števila točk. Na primer, če si uporabnik nastavi končno število točk na deset, se po desetih pravilno rešenih enačbah igra zaključi in je tako zmagovalec uporabnik. Ko se igra zaključi, tj. ko igralec izgubi ali zmaga, se igra sama resetira, kar pomeni, da lahko znova izbere matematične operacije, čas iztekanja ter končno število točk. Po določitvi vsega navedenega klikne gumb *start* in igra se začne. Kadar se uporabnik odloči zapustiti igro “Kako dobro ti gre”, klikne na gumb *izhod*, ki se nahaja v samem grafičnem vmesniku, ter se tako vrne v osnovni meni aplikacije.



Slika 4: Skica igre “Kako dobro ti gre?”

Diagram aktivnost za igro “Kako dobro ti gre?” prikazan s sliko 5 :



Slika 5: Diagram aktivnosti igre “Kako dobro ti gre?”

Potek diagrama aktivnosti za igro “Kako dobro ti gre?” (* predstavljajo alternative) :

1.) Uporabnik vpiše ime.
2.) Uporabnik izbere matematične operacije.
3.) Uporabnik nastavi končno število točk.
4.) Uporabnik nastavi iztekanje časa.
5.) Uporabnik zažene igro.
6.) Program prikaže enačbe z možnim rešitvami.
7.1.) Uporabnik izbere odgovor.
*7.2.) Alternativa: uporabnik zapusti igro.

*7.3.) Alternativa: uporabnik ponastavi igro (1.).
*7.3.1.) Uporabnik se vrne na začetek kjer mora vpisati ime (1.).
*7.3.2.) Alternativa: uporabnik zapusti igro.

8.1.) Uporabnik je izbral pravilen odgovor.

*8.2.) Alternativa: uporabnik je izbral napačen odgovor ali mu je potekel čas.
*8.2.1.) Uporabnik ponastavi igro (7.3).
*8.2.2.) Alternativa: uporabnik zapusti igro.

9.1.) Uporabniku se prikaže nova enačba z možnimi rešitvami.

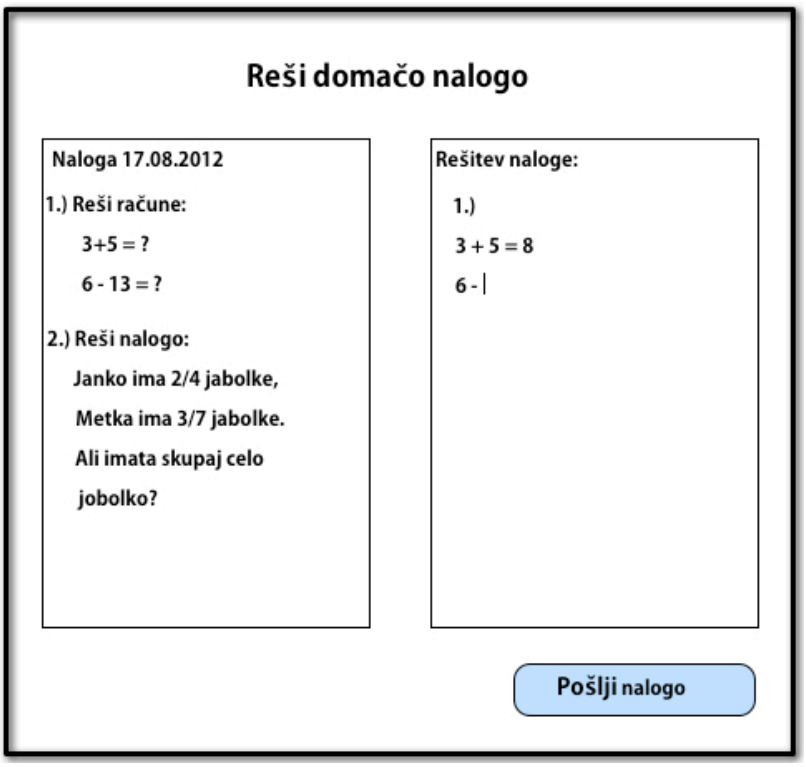
*9.2.) Alternativa: uporabnik je dosegel končno število točk.
*9.2.1.) Uporabnik ponastavi igro (7.3).
*9.2.2.) Alternativa: uporabnik zapusti igro.

4.3.3 Reši nalogo

Opis igre:

V načinu “Reši nalogo” ponujamo uporabniku možnost reševanja nalog tako kot prikazuje slika 6. Uporanik rešuje naloge, ki so bile predhodno naložene na splet. Kadar se uporabnik odloči za reševanje nalog, mora prvo vnesti URL naslov, na katerem se nahaja naloga v obliki *datoteka.txt,* pri čemer je ime datoteke poljubno, končnica pa mora biti .txt. URL vpiše v namensko okence ter pritisne na gumb za pridobitev naloge.

Ko uporabnik to naredi, se na levi strani zaslona prikaže besedilo naloge, na desni strani pa ima možnost vpisa rešitve. Ko preneha z reševanjem naloge, izpolni še osnovne podatke, kot so ime, elektronska pošta ter geslo elektronske pošte. Okence se nahaja na spodnjem delu ekrana. Elektronska pošta deluje preko SMTP strežnika, trenutno pa je nastavljena na Gmail zaradi popularnosti [17] . Po izpolnitvi vseh potrebnih podatkov za pošiljanje pošte uporabnik klikne na gumb *pošlji nalogo*. Naloga je nato poslana na elektronski naslov osebe, ki jo je napisala. Elektronski naslov mora oseba, ki napiše nalogo, vpisati v prvo vrstico *datoteke.txt*.
Kadar želi uporabnik začeti z reševanjem druge naloge, mora klikniti na gumb za ponastavitev. Tako dobi možnost reševanja druge naloge z vpisom njenega URL naslova.
Za vrnitev v osnovni meni aplikacije “Igrajmo matematiko” mora uporabnik klikniti gumb za izhod.



Slika 6: Skica igre “Reši nalogo”

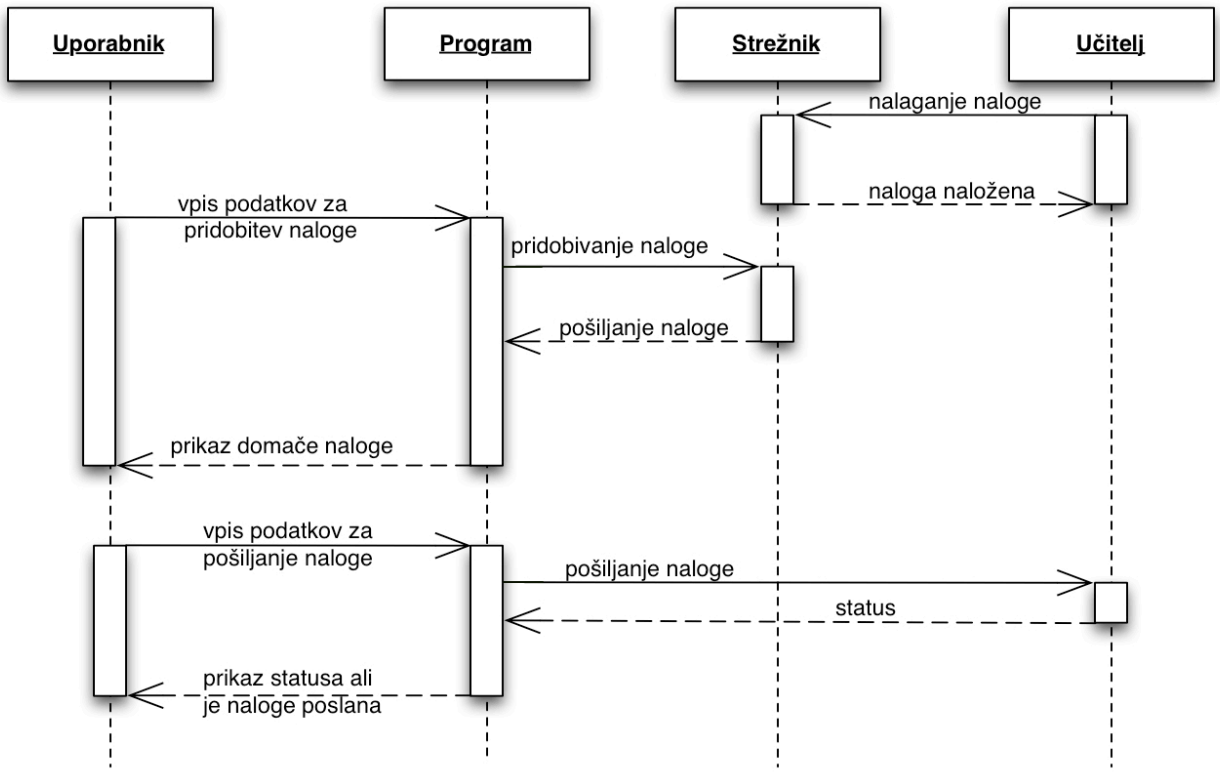
Opis nalaganja naloge na splet:

Domačo nalogo, ki jo mora učenec rešiti, mora učitelj ali starš najprej sestaviti. To pomeni, da najprej napiše besedilo naloge in ga nato shrani v *datoteko.txt,* kjer je ime datoteke poljubno, končnica pa mora biti .txt. Primer besedila *datoteke.txt* je prikazan na sliki 7.
Ko je naloga sestavljena, jo mora učitelj ali starš shraniti oziroma naložiti na splet. Shrani jo na namenski strežnik (FTP strežnik), ki podpira nalaganje datotek tipa *.txt*. Strežnik oziroma mesto, kjer se nahaja domača naloga, mora biti prosto dostopno, kar pomeni, da ne sme biti zavarovano z geslom. Istočasno pa mora biti omogočen prenos *datoteke.txt*, ki vsebuje domačo nalogo.
Ko je naloga naložena na splet, učitelj posreduje učencu URL naslov, kjer se naloga nahaja. Učenec nato URL naslov vpiše v namesko okence, ki se nahaja v igri "Reši nalogo". S tem se naloga prenese in učenec lahko prične z reševanjem.

Slika 7: Primer sestave *datoteke.txt,* ki jo učitelj ali starš naloži na splet

Zgornja slika prikazuje pravilno sestavo *datoteke.txt*, ki jo starš ali učitelj naloži na splet.
Rdeči kvadratek predstavlja prvo vrstico *datoteke.txt* in vsebuje elektronski naslov, na katerega bo kasneje uporabnik poslal rešeno domačo nalogo [13] . Moder kvadratek predstavlja besedilo domače naloge. Le-to se začne z drugo vrstico, ni pa omejeno v širino in ne v dolžino. Zaradi lepšega prikaza se uporabniku priporoča, da ne prekorači širine besedila domače naloge za več kot 60 znakov, saj se bo moral uporabnik v nasprotnem primeru premikati levo in desno po površini za prikazovanje domače naloge.

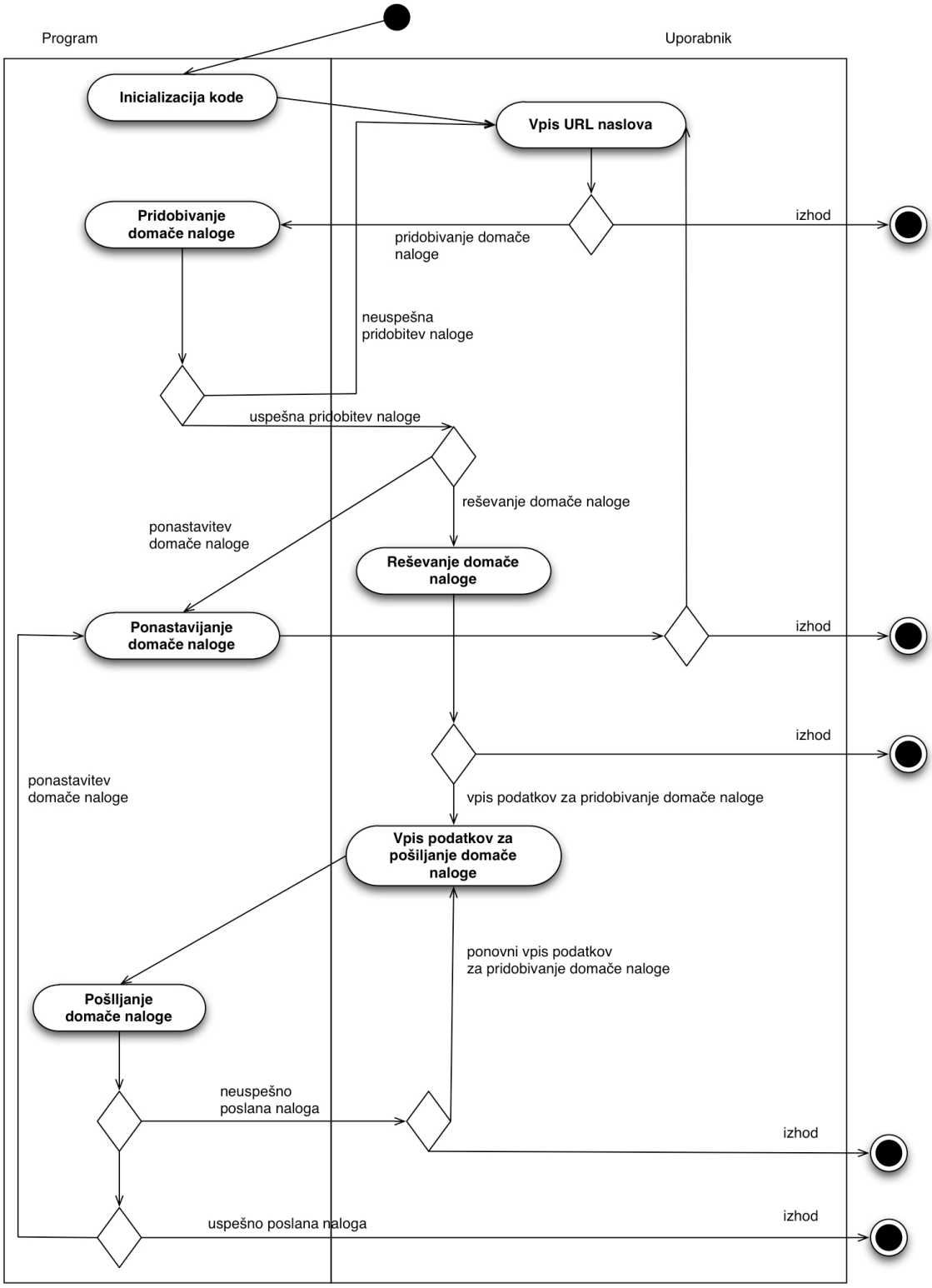
Sekvenčni diagram za igro “Reši nalogo” :



Slika 8: Sekvenčni diagram igre “Reši nalogo”

Iz diagrama priakzanega na sliki 8 je razvidno, da mora učitelj ali starš, preden uporabnik preko aplikacije pridobi domačo nalogo, le-to prej naložiti na splet [9] . Ko je naloga naložena na splet, jo lahko uporabnik preko aplikacije pridobi s strežnika, na katerega je bila naloga naložena.
Ko uporabnik reši nalogo, jo z vpisom podatkov, potrebnih za pošiljanje domače naloge, pošlje nazaj učitelju ali staršu. Uporabnik dobi potrdilo o tem, da je bila naloga poslana, v nasprotnem primeru dobi obvestilo, da naloga ni bila poslana.

Diagram aktivnost za igro “Reši nalogo” prikazan na sliki 9:



Slika 9: Diagram aktivnosti igre “Reši nalogo”

Potek diagrama aktivnosti za igro “Reši nalogo” (* predstavljajo alternative) :

1.) Uporabnik vpiše URL naslov za pridobitev domače naloge.
2.1.) Program začne s pridobivanjem domače naloge.
*2.2.) Alternativa: uporabnik zapusti igro.
3.1.) Program uspešno pridobi domačo nalogo.
*3.2.) Alternativa: program ne pridobi domače naloge in se vrne na vpis URL naslova.
4.1.) Uporabnik začne reševati domačo nalogo.

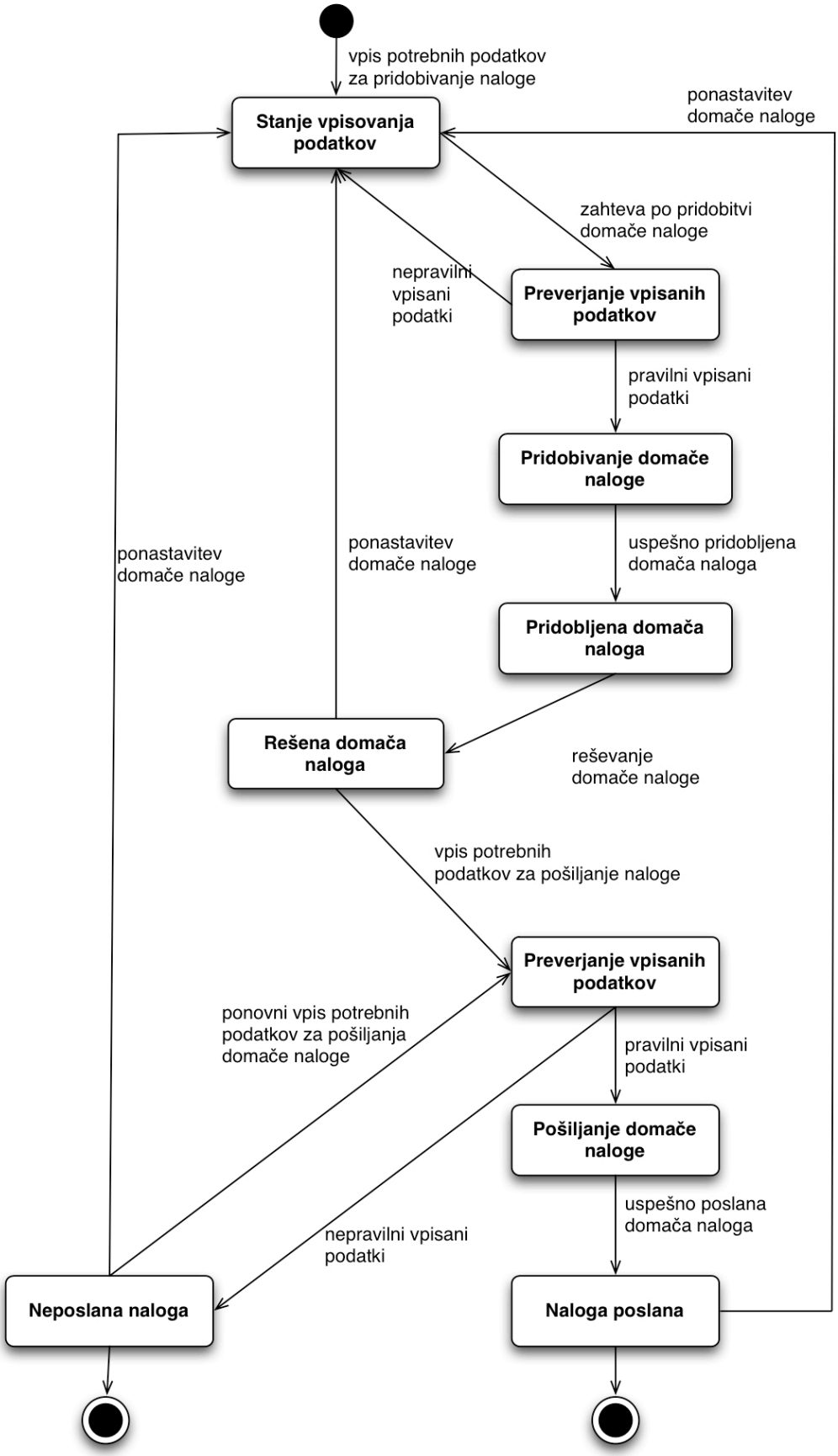
*4.2.) Alternativa: uporabnik ponastavi domačo nalogo.
*4.2.1.) Uporabnik se vrne na vpis URL naslova.
*4.2.2.) Alternativa: uporabnik zapusti igro.

5.1.) Uporabnik vpiše podatke za pošiljanje domače naloge.
*5.2.) Alternativa: uporabnik zapusti igro.
6.) Program začne pošiljati domačo nalogo.
7.1) Program uspešno pošlje domačo nalogo.

*7.2) Alternativa: program ne pošlje domače naloge.
*7.2.1.) Uporabnik ponovno vpiše podatke za pošiljanje domače naloge.
*7.2.2.) Alternativa: uporabnik zapusti igro.

8.1.) Uporabnik zapusti igro.
*8.2.) Alternativa: uporabnik ponastavi domačo nalogo (4.2.).

Diagram stanj za igro “Reši nalogo” prikazan na sliki 10:



Slika 10: Diagram stanj igre “Reši nalogo”

Diagram stanj prikazuje vsa možna stanja programa med izvajanjem in možna prehajanja med stanji kot posledico določene aktivnosti oziroma dogodka, ki ga sproži uporabnik [19] . Za lažje razumevanje delovanja naloge “Reši nalogo” smo se odločili opisati potek diagrama stanj.

Potek diagrama stanj za igro “Reši nalogo” (* predstavljajo alternative) :

1.) Uporabnik vpiše potrebne podatke za pridobitev domače naloge.
2.) Sledi postopek preverjanja vpisanih podatkov.
3.1.) V primeru, da so bili vpisani podatki pravilni, sledi stanje pridobivanja domače naloge.
*3.2.) Alternativa: v primeru, da vpisani podatki niso bili pravilni, vrne uporabnika ponovno na vpisovanje podatkov.
4.) Ko uporabnik pridobi domačo nalogo, preide v stanje pridobljene domače naloge.
5.) Rešena domača naloga je stanje, ki sledi pridobljeni domači nalogi. V tem stanju je uporabnik končal z reševanjem.
6.1.) Kadar hoče uporabnik poslati rešeno nalogo učitelju, mora vpisati podatke, potrebne za pošiljanje naloge preko spleta. Kadar se vpisani podatki preverjajo, je igra v stanju preverjanja podatkov.
*6.2.) Alternativa: v primeru, da uporabnik ne želi poslati domače naloge, lahko začne z reševanjem nove domače naloge s ponastavitvijo in se tako vrne v prvotno stanje igre.
7.1) Če je uporabnik vpisal pravilne podatke, sledi postopek pošiljanja domače naloge.
*7.2.) Alternativa: nepravilni podatki pripeljejo do stanja neposlane domače naloge, iz katerega lahko uporabnik ponovno vpiše podatke, ponastavi nalogo ali zapusti igro.
8.) Končno stanje igre predstavlja poslana domača naloga, ki ji sledi izhod iz igre.

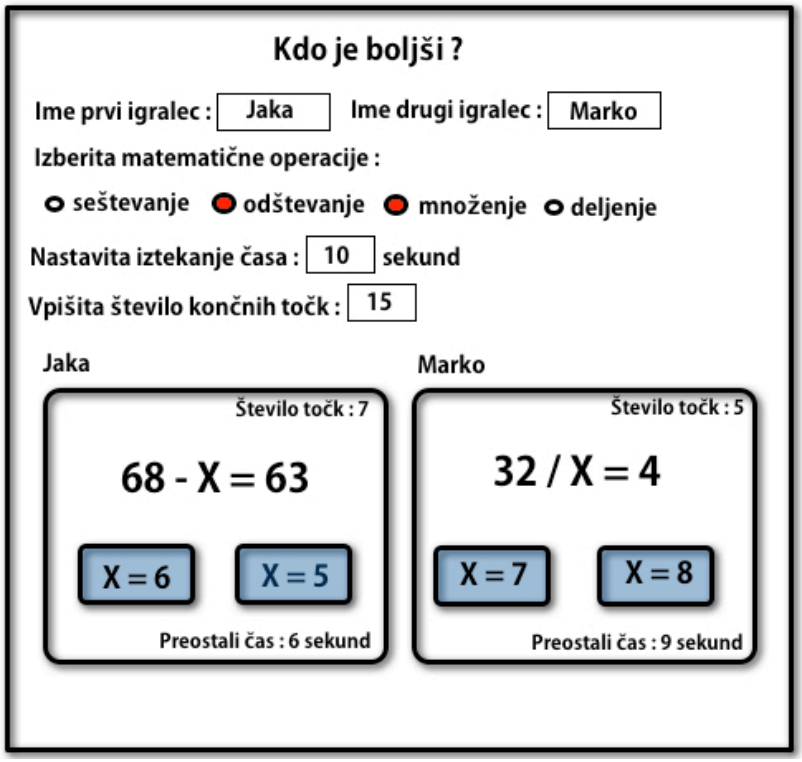
4.4 Večigralski način

V večigralskem načinu smo uporabniku ponudili igranje z nasprostnikom. Sledi opis igre, v kateri lahko dva igralca med seboj tekmujeta v hitrostnem reševanju matematičnih enačb.

4.4.1 Izzivaj prijatelja

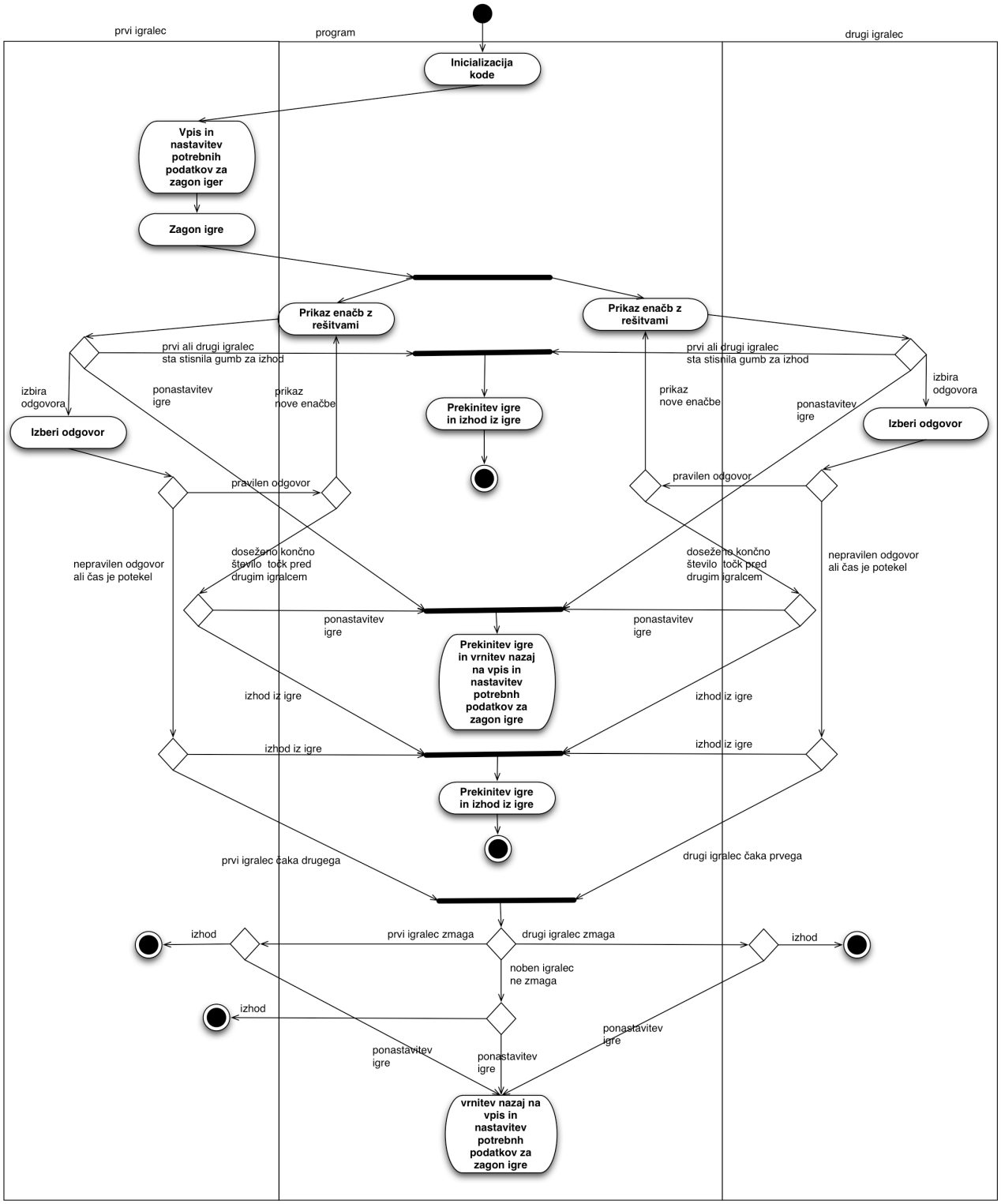
Opis igre:

Aplikacija “Igrajmo matematiko” ponuja tudi igro, ki teče v večigralskem načinu. Pri igri “Izzivaj prijatelja” dva uporabnika tekmujeta v hitrosti reševanja matematičnih enačb. Igra temelji na že prej opisani igri “Kako dobro ti gre”, le da je v tem primeru implementirana tako, da lahko igrata dva igralca hkrati, kot prikazuje slika 11. Zmaga tisti igralec, ki prej doseže končno število možnih točk. V primeru, da se ne doseže končnega števila točk zaradi napake pri reševanju, zmaga tisti igralec, ki ima več točk.



Slika 11: Skica igre “Izzivaj prijatelja”

Diagram aktivnosti za igro “Izzivaj prijatelja” prikazan na sliki 12:



Slika 12: Diagram aktivnosti igre “Izzivaj prijatelja”

Potek diagrama aktivnosti za igro “Izzivaj prijatelja” (* predstavljajo alternative) :

1.) Uporabnik vpiše potrebne podatke in izbere nastavitve za zagon igre.
2.) Uporabnik zažene igro.
3.) Program prikaže obema igralcema različne enačbe z možnimi rešitvami.
4.1.) Igralca izbirata odgovor.

*4.2.) Alternativa: eden izmed igralcev zapusti igro (igra se zaključi za oba).
*4.3.) Alternativa: eden izmed igralcev ponastavi igro (igra se ponastavi za oba) (1.).

5.1.) Eden izmed igralcev je izbral pravilen odgovor.

*5.2.) Alternativa: eden izmed igralcev je izbral napačen odgovor ali mu je potekel čas.
*5.2.1.) Eden izmed igralcev zapusti igro (igra se zaključi za oba).
*5.3.2.) Alternativa: eden izmed igralcev ponastavi igro (igra se ponastavi za oba) (1.).
*5.3.3.) Alternativa: igralec, ki je odgovoril napačno, počaka, da drugi preneha igrati igro.
*5.3.3.1.) Drugi igralec je napačno odgovoril ali mu je potekel čas in ima manj doseženih točk (zmaga prvi igralec).
*5.3.3.1.1.) Eden izmed igralcev zapusti igro (igra se zaključi za oba).
*5.3.3.1.2.) Alternativa: eden izmed igralcev ponastavi igro (igra se ponastavi za oba) (1.).
*5.3.3.2) Alternativa: drugi igralec je napačno odgovoril ali mu je potekel čas in ima več doseženih točk (Zmaga prvi igralec).
*5.3.3.2.1.) Eden izmed igralcev zapusti igro (igra se zaključi za oba).
*5.3.3.2.2.) Alternativa: eden izmed igralcev ponastavi igro (igra se ponastavi za oba) (1.).
*5.3.3.3) Alternativa: drugi igralec je dosegel končno število točk in zmagal.
*5.3.3.3.1.) Eden izmed igralcev zapusti igro (igra se zaključi za oba).
*5.3.3.3.2.) Alternativa: eden izmed igralcev ponastavi igro (igra se ponastavi za oba) (1.).

6.1.) Igralcu, ki je izbral pravilen odgovor, se prikaže nova enačba z možnimi rešitvami (3.).
*6.2.) Alternativa: igralec, ki je izbral pravilen odgovor, je dosegel končno število točk pred drugim igralcem in s tem postal zmagovalec.
*6.2.1.) Eden izmed igralcev zapusti igro (igra se zaključi za oba).
*6.2.2.) Alternativa: eden izmed igralcev ponastavi igro (igra se ponastavi za oba) (1.).

4.5 Omejitve

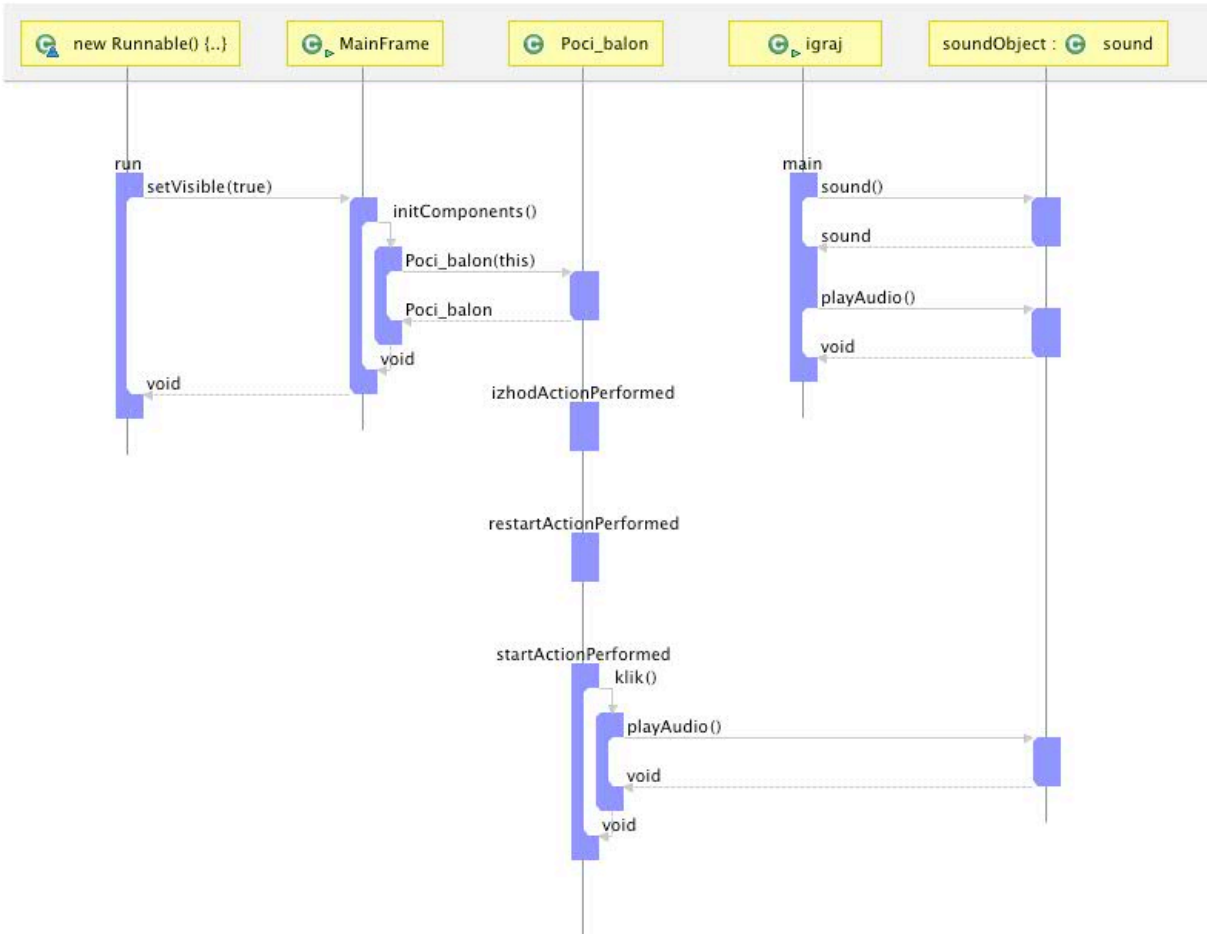
Pri uporabi aplikacije bi utegnili imeti omejitve z resolucijo. Za pravilen prikaz aplikacije mora uporabnik imeti monitor, ki podpira resolucijo 1280x800 ali več.
Med omejitve bi lahko spadala tudi Java kot programski jezik, saj poganja aplikacijo [14] . Vendar pa je ta prosto dostopna na spletu in je programski jezik, ki ga podpira večina operacijskih sistemov. Tako je vse, kar uporabnik mora narediti, prenesti pravo različico Jave in jo namestiti.
Za uporabo funkcije “Reši nalogo”, ki jo ponuja aplikacija “Igrajmo matematiko”, potrebuje uporabnik dostop do spleta, saj lahko samo tako prenese nalogo in jo kasneje tudi pošlje nazaj rešeno.

5. Načrtovanje aplikacije

Kot je bilo že omenjeno, gre pri aplikaciji “Igrajmo matematiko” za univerzitetni projekt, pri katerem nimamo dodeljenih finančnih sredstev. Za implementacijo se zato uporabljajo izključno trenutna zastonjska orodja in programska oprema. V tem primeru imamo na voljo več primernih programskih oprem, ki uporabljajo različna orodja. Temu primerno je potrebno dobro premisliti, katero programsko opremo bomo uporabili pri določeni fazi programskega procesa. Pred uporabo določene programske opreme je priporočeno, da se prej pozanimamo o dobrih oziroma slabih lastnosti, ki jih sama programska oprema nosi. Ugotovitev nepravilnosti programske opreme v pozni fazi projekta lahko pomeni velike časovne zamike. Za projekte lahko nepravilna programska oprema pomeni tudi povišane stroške in v skrajnosti tudi neuspešnost projekta.

5.1 Aplikacija

V aplikaciji “Igrajmo matematiko” imamo glavno komponento ozrioma razred z imenom *MainFrame* kateri prikazuje vse ostale igre oziroma razrede. Sekvenčni diagram prikazan s sliko 13 prikazuje glavne aktivnosti, pri zagonu igre “Poči pravi balon”.

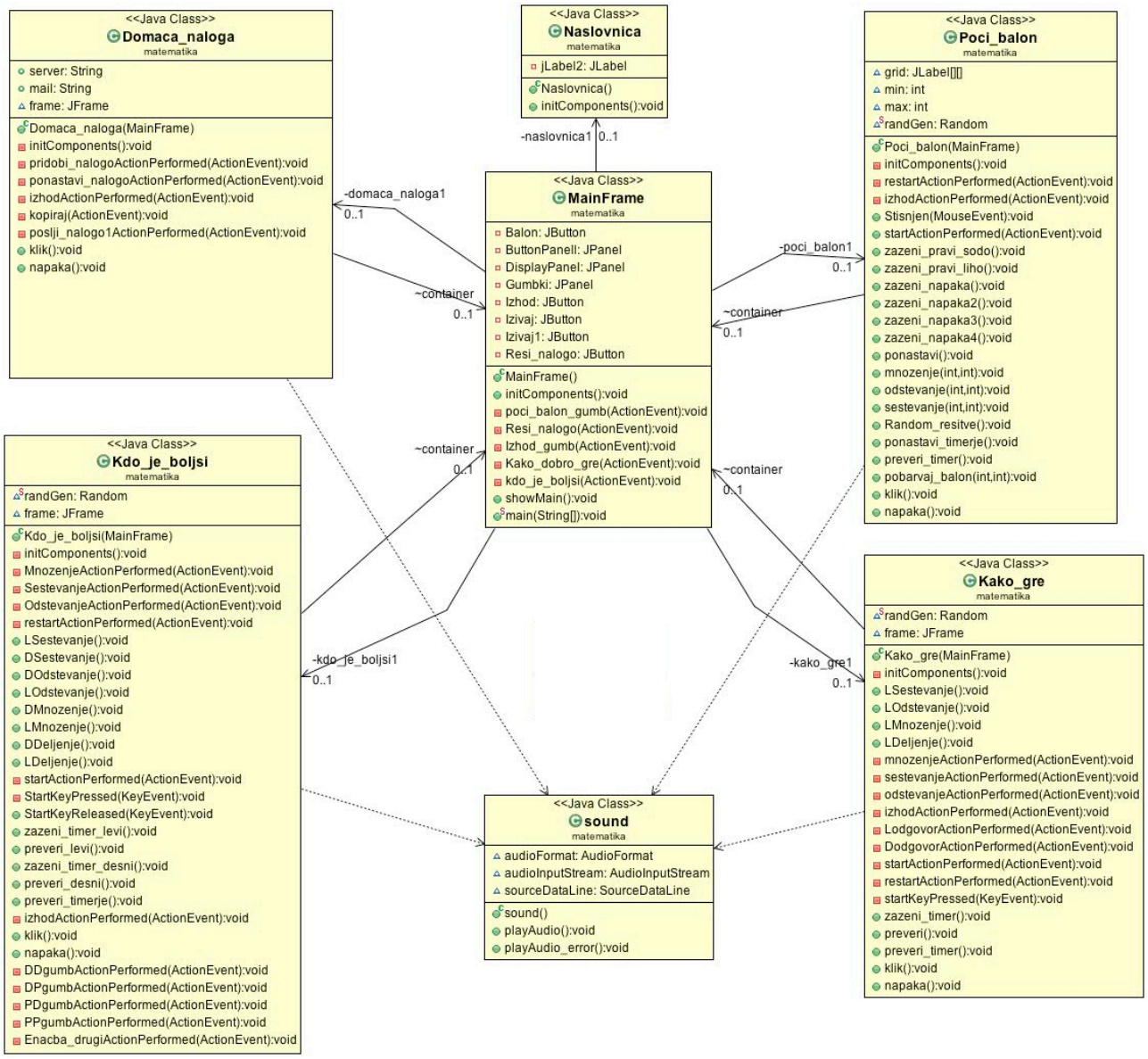


Slika 13: Sekvenčni diagram aplikacije ob igri “Poči pravi balon”

Sekvenčni diagram prikazuje komunikacijo med objekti v smislu zaporedja sporočil [18] . Ko zaženemo igro, se kot prva komponenta zažene razred *MainFrame*. V primeru, ki ga opisuje zgornji sekvenčni diagram, se je uporabnik odločil za igro "Poči pravi balon". *MainFrame* tako kliče razred *Poci_balon,* ki nam prikaže grafični vmesnik igre. V igri "Poči prvi balon" imamo na izbiro zagon igre, ponastavitev igre in izhod iz igre. Vsaka izbira oziroma odločitev uporabnika povzroči zvok, ki, da uporabniku vedeti, da je s klikom na gumb sprožil določeno aktivnost. V našem primeru je uporabnik kliknil na gumb za zagon igre in je tako razred *Poci_balon* poklical komponento aplikacije, ki je zadolžena za igranje zvoka. Vse ostale tri komponente naše aplikacije delujejo po istem principu. V ozadju vedno teče *MainFrame,* iz katere lahko dostopamo do vseh štirih glavnih komponent, ki jih predstavljajo funkcije oziroma igre. Ko enkrat kličemo eno izmed štirih komponent, lahko klicana komponenta komunicira z razredom, ki je zadolžen za zvok oziroma z *MainFrame* komponento. Komponente, ki vsebujejo igre, niso med seboj povezane, posledično pa kakršne koli spremembe v eni izmed komponent nebi vplivale na druge. V primeru, da zapustimo igro, se nam prikaže komponenta z imenom *Naslovnica*, ki predstavlja prvo stran aplikacije. Istočasno je *Naslovnica* tudi prva komponenta, ki jo kliče *MainFrame* ob zagonu igre.

5.2 Razredni diagram

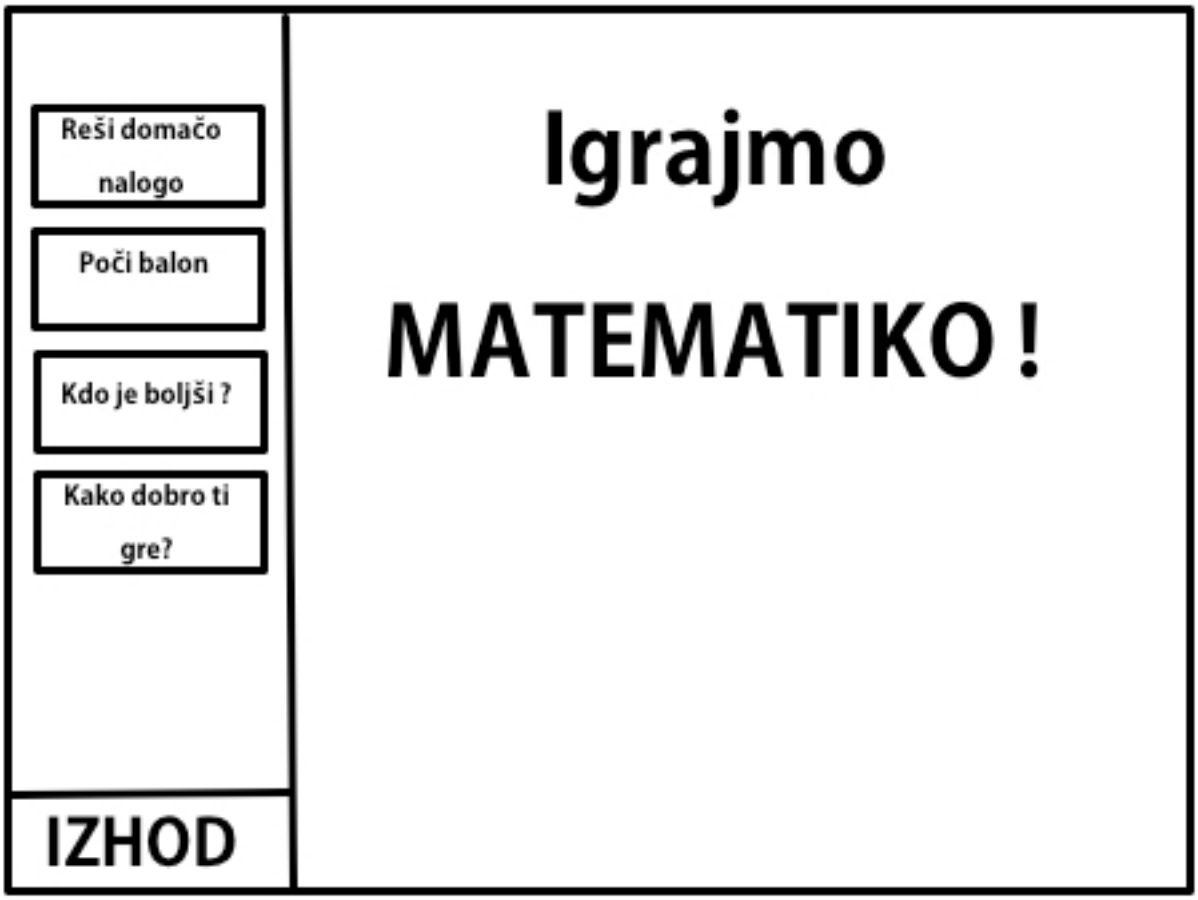
Razredni diagram opisuje strukturo sistema oziroma programa s prikazom razredov, atributov in relacij med atributi [20] . Na sliki 14 je prikazan razredni diagram za našo aplikacijo. Vsebuje osem razredov. *MainFrame* je glavni razred in vsebuje osem zasebnih atributov, pet zasebnih operacij in štiri javne. *Poci_balon* je razred, ki je zadolžen za igro "Poči pravi balon". Vsebuje enajst zasebnih atributov, štiri zavarovane atribute in petinštirideset javnih atributov. Poleg navedenega števile atributov vsebuje še tri zasebne operacije in devetnajst javnih operacij.
Za enoigralsko igro "Kako dobro ti gre?" stoji v ozadju razred *Kako_gre*, ki vsebuje petindvajset zasebnih atributov, dva zavarovana atributa, sedemnajst javnih atributov ter deset javnih in zasebnih operacij. "Izzivaj prijatelja" je edina večigralska igra aplikacije. Razred zadolžen za pravilno delovanje le-te je razred *Kdo_je_boljsi*. Razred, ki skrbi za pravilno delovanje večigralske igre, ima največje število atributov in operacij predvsem zaradi pravilne sinhronizacije dogodkov preverjenja dogajanj obeh igralcev. *Kdo_je_boljs*i vsebuje triinštirideset zasebnih atributov, dva zavarovana in dvajset javnih. Poleg atributov vsebuje še petnajst zasebnih operacij in osemnajst javnih.
Za zadnjo igro aplikacije imenovano "Reši nalogo" skrbi razred *Domaca_naloga,* ki vsebuje skupno šestindvajset atributov, od tega triindvajset zasebnih, en zavarovan in dva javna. Vsebuje pa tudi šest zasebnih operacij in tri javne. Poleg navedenih razredov, ki skrbijo za delovanje iger, imamo še tri zazrede, in sicer razred *Naslovnica,* ki skrbi za prikaz prve strani aplikacije, vsebuje pa en zasebni atribut in dve javni operaciji. Za zvok skrbita razreda *sound* in *igraj*. Razred *sound* ima tri javne operacije in tri zavarovane atribute, razred *igraj* pa ima dve javni operaciji.
Kadar hoče kateri izmed razredov zadolženih za igro, predvajati zvok, pokliče razred *sound*, ki je zadolžen za predvajanje zvoka.



Slika 14: Razredni diagram aplikacije “Igrajmo matematiko”

5.3 Grafični vmesnik (prva stran)

Kot smo že na začetku zaključne naloge omenili, bo aplikacija “Igrajmo matematiko” namenjana otrokom oziroma osnovnošolcem. Zaradi tega smo se potrudili narediti čim bolj prepost in samointuitiven grafični vmesnik. Sledi skica uvodne strani grafičnega vmesnika ter kratek opis kako grafični vmesnik deluje.



Slika 15: Skica uvodne strani grafičnega vmesnika aplikacije “Igrajmo matematiko”

Iz slike 15 je razvidno, da ima uporabnik na levi strani uvodne strani možnost dostopati do štirih iger s klikom na gumb. Na spodnjem levem delu opazimo gumb z napisom *izhod* kateri služi samemu izhodu iz igre. Kadar aplikacijo zaženo je razred *Naslovnica* zadolžen za prikaz grafike oziroma risbe, namenjene za uvodno stran. Na kvadratnem polju, kjer se nahaja napis “Igrajmo MATEMATIKO”, se ob kliku na gumb za igranje igre prikaže igra. Ko uporabnik izbere igro, zamenja razred *Naslovnica* za tisti razred igre, ki jo je uporabnik izbral. Med celotnim igranjem so uporabniku še vedno vidni štirje gumbi na levi strani saj se menjava le igralno polje. Uporabnik lahko zapusti igro bodisi tako, da med igro klikne na gumb, ki predstavlja povezavo do igre, ali pa klikne na gumb za izhod, ki se nahaja v sami igri in se tako vrne nazaj na uvodno, kjer se na igralnem polju spet pojavi risba, za katero je zadolžen razred *Naslovnica*.

5.4 Programski jezik

Programska oprema, je ključnega pomena za uspešno implementacijo programa. Slaba izbira programske opreme oziroma orodij lahko pomeni težave v končnih fazah implementacije. Poleg programske opreme so pomembne tudi izkušnje osebe oziroma skupine, ki izbira in uporablja orodja za implementacijo.

5.4.1 Java

Java programski jezik je izbran zaradi možne implementacije na vse popularne operacijske sisteme, kot so naprimer Windows, Linux, OS X in ostale. Poleg zelo že navedenega pa smo Javo izbrali tudi zaradi samega znanja jezika saj je to programski jezik, ki ga najbolje obvladamo.
Delo z Javo je enostavno, ker ima odlično dokumentacijo in na izbiro veliko zbirko knjižic, za katero bi se lahko skoraj reklo, da je neskončna. Ker delamo preprosto namizno aplikacijo, menimo, da bo Java brez težav kos nalogi.

6. Izvedba

V pogalvju, ki sledi, je predstavljena praktična implementacija aplikacije “Igrajmo matematiko”. Izvedba je ena najzahtevnejših faz, saj se pri njej pogosto pojavijo napake, napravljene v prejšnjih fazah. Slaba analiza problema in zahtev lahko pomeni slab konec za projekt.
Zelo pomembno je, da so bile v predhodnih fazah podane realne ocene in pričakovanja. Slaba definicija zahtev v povezavi s slabo oceno potrebnih stroškov lahko pomeni tudi propad projekt ozrioma najhujših primerih celo propad podjejta.

6.1 Programska orodja

Uporabljena programska orodja za implementacijo aplikacije so prostodostopna na spletu. Prav tako so razširjena za vse najbolj popularne operacijske sisteme, kot so naprimer Linux, Windows, OS X in tako naprej.

6.1.1 Eclipse

Aplikacijo “Igrajmo matematiko” smo razvili v programskem orodju Eclipse IDE. To je razvojno programersko okolje, ki podpira več programskih jezikom med katerimi je seveda tudi Java. Programejem omogoča vodenje tako kompleksnih do zelo preprostih programskih projektov. Vsebuje tudi dokumentacijo o Javi in orodja za odpravljanje napak, sledenje poteku programa, sprotno označevanje napak itn. Pri tem naj poudarimo, da je Eclipse prosta programska oprema, ki jo lahko brez težav prenesemo s spleta. Za implementacijo naše aplikacije smo uporabili Eclipse Indigo, ki je osma različica Eclipsa izdana junija 2011. Poleg implementacije programa smo uporabili Eclipse tudi za risanje sekvenčnega in razrednega diagrama, saj omogoča veliko različnih “plug-inov”, med katerimi najdemo tudi orodja za risanje raznih diagramov.
Vse, kar smo morali narediti, je bilo najti določen diagram, ki ponuja, kar smo iskali. Program mora biti brezplačen, saj gre, kot smo omenili na začetku, za projekt brez finančnih sredstev.

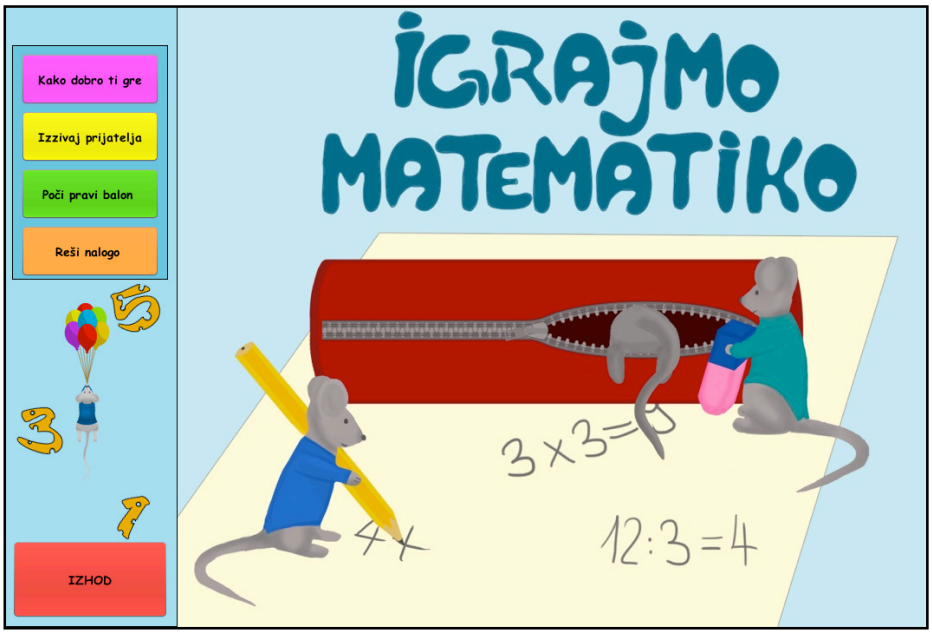
6.1.2 NetBeans

Za izdelavo grafičnega vmesnika smo uporabili NetBeans, ki je večjezično razvojno programsko okolje podjetja Sun Microsystems [16] . Uporabili smo ga zaradi ponudbe široke palete orodij. V našem primeru smo uporabili različico 7.0.1, ki je tako kot Eclipse, prosta programska oprema, na voljo najpopularnejšim operacijskim sistemom.

Celotna programska koda z uporabljenimi slikami za oblikovanje aplikacije se nahaja na zgoščenki, ki je priložena.

6.2 Izvedna grafičnega vmesnika

V koraku izvedbe grafičnega vmesnika smo implementirali grafični vmesnik, ki ga predstavljamo z zaslonskimi posnetki. Pri izdelavi vmesnika smo upoštevali funkcijske zahteve glede barvitosti in privlačnosti aplikacije kot je prikazano na sliki 16.



Slika 16: Zaslonski posnetek uvodne strani aplikacije “Igrajmo matematiko”

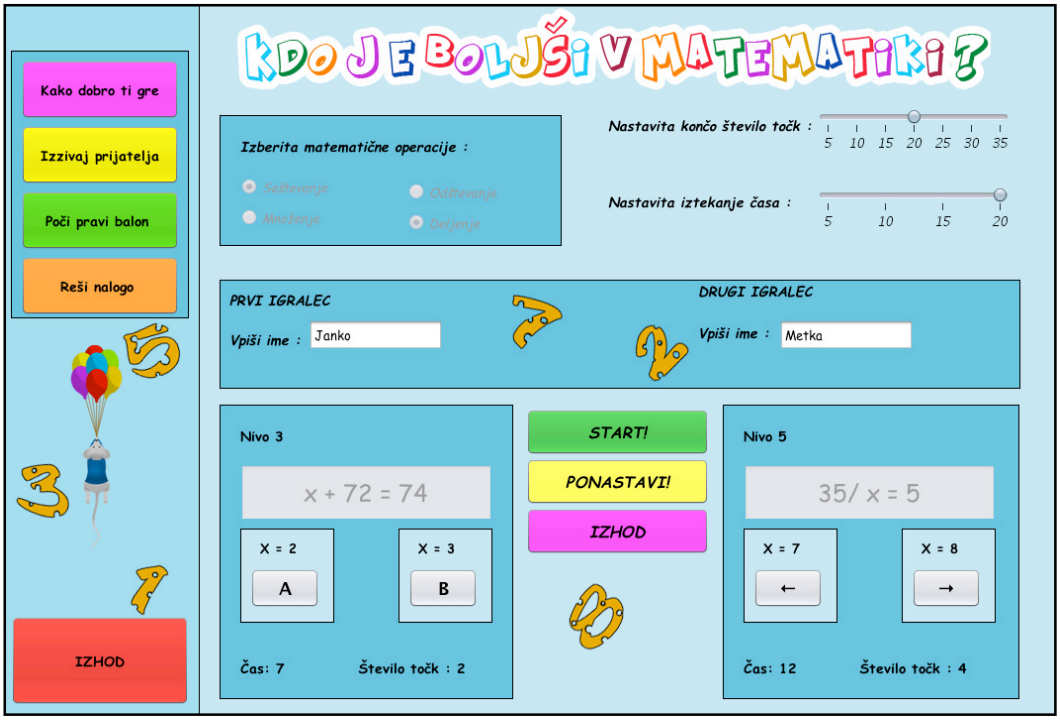
Slika 17 prikazuje celozaslonski posnetek enoigralske igre “Kako dobro ti gre?”, v kateri imamo uporabnika, ki si je izbral matematični operaciji množenja in seštevanja. Nastavil je končno število točk na dvajset ter iztekanje časa na petnajst sekund. Iz slike lahko tudi ugotovimo, da je uporabnik pravilno odgovoril na dve enačbi.

Slika 17: Zaslonski posnetek igre “Kako dobro ti gre?”

Slika 18 je prikaz celozaslonskega posnetka igre “Poči pravi balon” na kateri je igralec trenutno dosegel 8 točk.

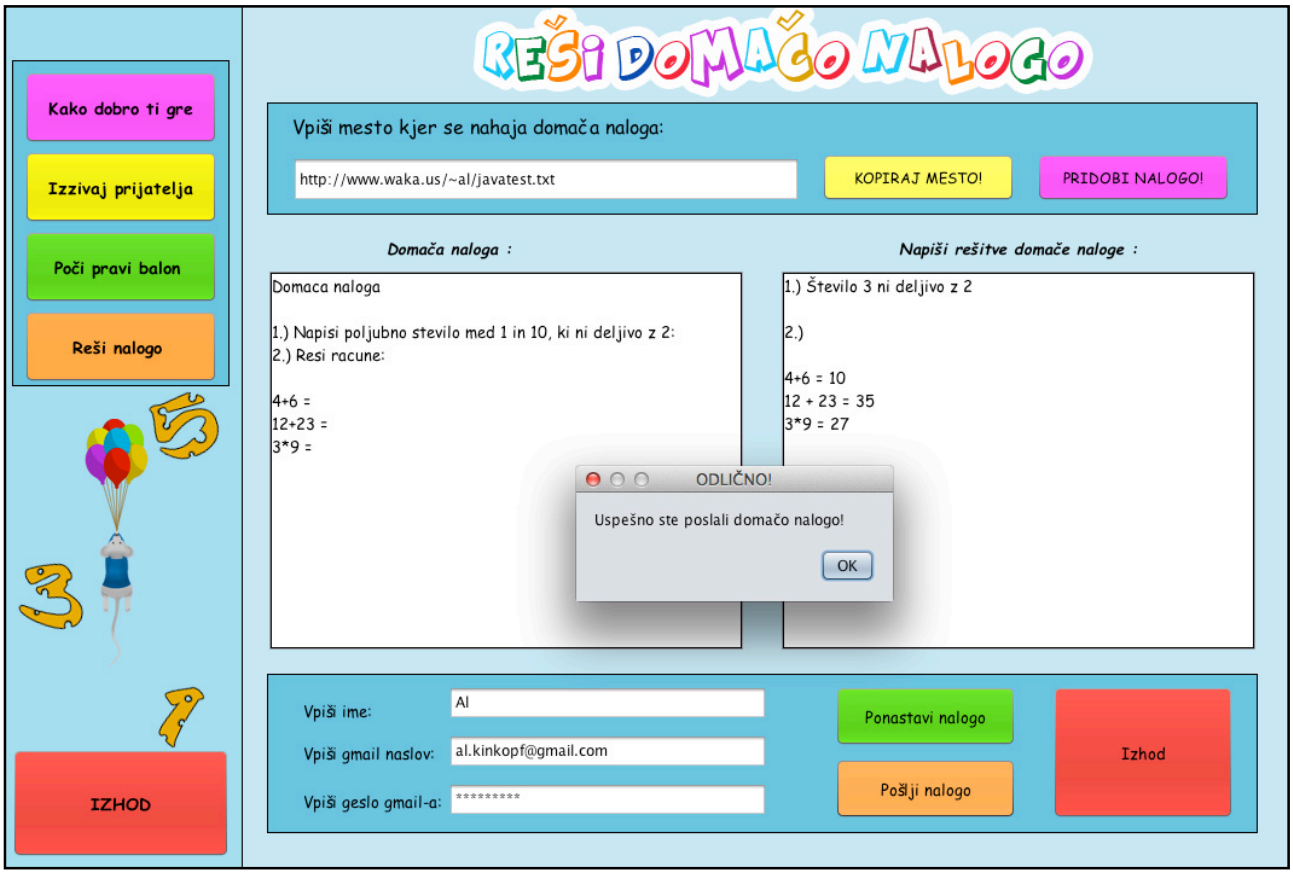
Slika 18: Zaslonski posnetek igre “Poči pravi balon”

Slika 19 prikazuje igro “Izzivaj prijatelja” v kateri sta se pomerila igralca z uporabniškimi imeni Janko in Metka. Uporabnika sta izbrala matematični operaciji seštevanja in deljenja. Končno število točk sta nastavila na dvajset, iztekanje časa pa prav tako na dvajset sekund. Trenutno vodi igro uporabnik Metka saj ima dve točki več kot Janko.



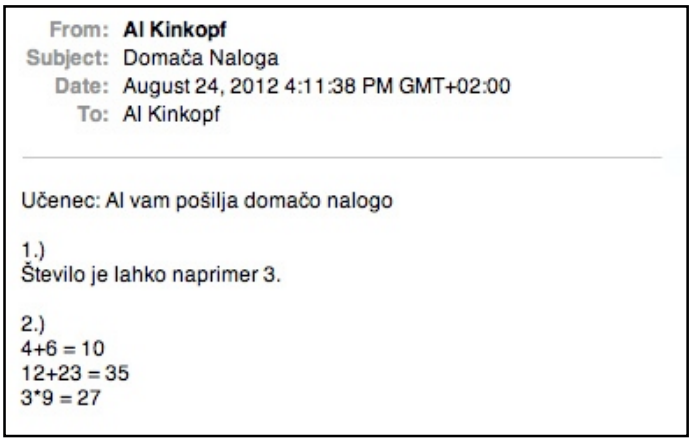
Slika 19: Zaslonski posnetek igre “Izzivaj prijatelja”

Slika 10 je celozaslonski posnetek, ki prikazuje igro “Reši nalogo”. Iz slike je razvidno, da je uporabnik prenesel nalogo iz URL naslova *http://www.waka.us/~al/javatest.txt*. Opazimo lahko, da je uspelo uporabniku tudi pravilno rešiti nalogo ter jo posredovati nazaj učitelju ali staršu.



Slika 20: Zaslonski posnetek igre “Reši nalogo”

Na sliki 21 je prikazan primer, kako dobi učitelj oziroma starš rešeno domačo nalogo na elektronski naslov enkrat, ko mu ju jo učenec poslal.



Slika 21: Primer prejete domače naloge na elektronski naslov

7. Testiranje

Testiranje je zadnji programski proces, ki se opravi pred predajo izdelka. Pri testiranju gre za proces izvajanja programa z namenom odkriti napako in jo popraviti pred izdajo končnemu uporabniku. Oseba, ki testira program, se imenuje preizkuševalec. Namen preizkuševalca je porušiti pravilno delovanje programa. Obstajata dve vrsti preizkuševalcev [12] . Testiranje lahko opravi razvijalec programa, ki je bil prisoten pri vseh fazah programskega procesa, ali oseba, ki je bila prisotna pri vseh fazah programskega procesa, razen načrtovanja in izvedbe. Načeloma je najboljši primer testiranja, ko program oziroma aplikacijo testira neodvisna skupina, spremlja pa jih razvijalec programa. Slednji tako natančno vidi, kje in kdaj se pojavijo napake. V našem primeru bo aplikacijo testiral razvijalec programa.

7.1 Načrt testiranja

Najprej bomo testirali zagon aplikacije in sam prikaz le-te. Ugotovili bomo ali, so vsi elementi znotraj igre pravilno postavljeni (gumbi, plošče, določene nastavitve ipd.). Testirali bomo vsako komponento posebej, in sicer uvodno stran ter vsako igro.

Na samem začetku bomo testirali enostavne elemente, ki jih je lažje popraviti in katerih popravki zahtevajo manj časa. Preverili bomo pisavo, ali je v vseh igrah oziroma komponentah enaka in ali je velikost pisave povsod oziroma na istih elementih grafičnega vmesnika enaka. Nato bomo preverili zvoke, ki so posledica uporabnikovih dejavnosti, kot je na primer klik na gumb.Vsaka uporabnikova dejavnost oziroma vsak klik na gumb mora biti pospremljen s pravilnim zvokom. Ko bomo preverili zvok in pisavo, se bomo lotili preverjanja gumbov, in sicer, ali vsak gumb opravlja svojo nalogo, kot je naprimer zagon igre, izhod iz igre, prehajanja med igrami in tako naprej. Skozi celotno testiranje bomo pozorni na grafični vmesnik. Opazovali bomo, ali se obnaša po pričakovanjih. Po končanem preverjanju osnovnih stvari se bomo lotili vsake igre posebej. Pri igri “Reši nalogo” bomo pozorni na to, ali igra pravilno sprejema domačo nalogo s spleta. Preverili bomo ali učitelj ozrioma starš dobi nalogo na elektronski naslov, ko jo učenec pošlje in ali je oblika oziroma vsebina rešene domače naloge pravilna. Za konec bomo ugotovili ali igra pravilno ponastavlja vse parametre v primeru, da hoče uporabnik reševati novo nalogo.
Pri enoigralskih igrah “Poči balon” in “Kako dobro ti gre?” bomo pozorni na pravilno sinhronizacijo dogodkov. Ugotovili bomo, ali igra ve, kdaj je uporabnik izbral pravilen odgovor in kdaj napačen in ali pravilno ponastavlja iztekanje časa. Pri igri “Kako dobro ti gre?” bomo pozorni na to, ali igra pravilno sprejme nastavitve uporabnika, kot so izbira matematičnih operacij, iztekanje časa in končno število točk.

V zadnji testirani igri “Izzivaj prijatelja” bomo predvsem pozorni na pravilno sinhronizacijo dogodkov med obema uporabnikoma. Ugotoviti želimo, ali igra pravilno prepoznava zmagovalca in poraženca ter, tudi kdaj ni zmagal noben uporabnik. Tako kot pri enoigralski igri “Kako dobro ti gre?”, bomo tudi tukaj preverili, ali igra pravilno sprejema nastavitve uporabnikov.
Za sam konec testiranja bomo še preverili, ali smo upoštevali in izpolnili vse funkcijske zahteve.

7.1.1 Seznam testiranja

S tabelo 1 prikazujemo elemente testiranja ter pričakovane rezultate.

Elementi testiranja	Pričakovan rezultat
postavitev grafičnih elementov	Pravilna postavitev vseh grafičnih elementov.
zvok	Vsako aktivnost uporabnika spremlja zvok.
pisava	Vsa uporabljena pisava v aplikaciji je Comic Sans.
povezave gumbov	Pravilne dogodki ob kliku na gumb.
uvodna stran aplikacije	Pravilno prehajanje v igre.
igra “Reši nalogo”	Pravilno prejemanje in pošiljanje naloge, pravilen prikaz vsebine domače naloge, na splošno pravilno delovanje in prikaz igre.
igra “Poči balon”	Pravilno delovanje igre, usklajenost balončkov in enačbe, pravilno prikazovanje balončkov, pravilno stopnjevanje težavnsoti.
igra “Kako dobro ti gre?”	Pravilno sprejemanje uporabnikovih nastavitev, pravilno obnavljanje iztekanja časa, pravilno delovanje igre.
igra “Izzivaj prijatelja”	Pravilno sprejemanje uporabnikovih nastavitev, pravilno obnavljanje iztekanja časa, pravilno spremljanje dogodkov obeh igralcev, pravilno delovanje iger na sploh.

Tabela 1: Seznam elelemtov za testirat

7.1.2 Rezutati testiranja

Tabela 2 predstavlja rezultate testiranja.

Elementi testiranja	Uspešnost testiranja	Preizkuševalec / datum / sistem
postavitev grafičnih elementov	uspešno	Al Kinkopf / 30.8.2012 / OS X Lion
zvok	uspešno	Al Kinkopf / 30.8.2012 / OS X Lion
pisava	uspešno	Al Kinkopf / 30.8.2012 / OS X Lion
povezave gumbov	neuspešno	Al Kinkopf / 30.8.2012 / OS X Lion
uvodna stran aplikacije	neuspešno	Al Kinkopf / 30.8.2012 / OS X Lion
igra “Reši nalogo”	uspešno	Al Kinkopf / 31.8.2012 / OS X Lion
igra “Poči balon”	uspešno	Al Kinkopf / 31.8.2012 / OS X Lion
igra “Kako dobro ti gre?”	uspešno	Al Kinkopf / 31.8.2012 / OS X Lion
igra “Izzivaj prijatelja”	uspešno	Al Kinkopf / 31.8.2012 / OS X Lion

Tabela 2: Rezultati testiranja

7.2 Najdene napake in popravki

7.2.1 Testiranje zvoka in gumbov

Najprej smo testirali zvok gumbov, da ugotovimo, ali vsak gumb predvaja namenski zvok. Testiranje je bilo uspešno, saj smo ugotovili, da nima vsak gumb ob kliku zvoka. Napako smo uspešno odpravili, saj smo dodali zvok vsakemu gumbu.

7.2.2 Pregled pisave

Testiranje pisave smo uspešno opravili, saj smo ugotovili pomanjkljivosti pri različni uporabi pisave znotraj aplikacije, odkrili pa smo tudi »tiskarske škrate«. Napako smo odpravili z uporabo *Comic Sans* pisave za vse napise znotraj aplikacije razen naslovov iger, saj menimo, da je ta pisava še najbolj primerna za aplikacijo namenjeno osnovnošolskim otrokom. Poleg tega je razširjena na vse popularne operacijske sisteme. »Tiskarske škrate« smo preprosto popravili.

7.2.3 Testiranje uvodne strani

Pri testiranju uvodne strani nismo ugotovili nobene napake. Grafično je bila stran brez napak, povezave gumbov so delovale po pričakovanjih, izhod iz aplikacije pa se je izvedel brez dodatnih zapletov.

7.2.4 Testiranje igre “Reši nalogo”

Pri testiranju igre smo testirali prenos domače naloge s spleta, ki je deloval po pričakovanjih. Prikaz domače naloge in reševanje domače naloge je tudi potekalo tako, kot smo zadevo predvideli. Pošilanje domače naloge je bilo uspešno.
Napako smo ugotovili pri ponastavitvi domače naloge, ko smo želeli ponovno prenesti drugo domačo nalogo s spleta in jo začeti reševati. Ko smo stisnili na gumb za ponastavitev domače naloge, sta se okenci za vpis URL naslova domače naloge in gumb za kopiranje domače in pridobitev domače naloge izbrisala oziroma izginila s zaslona kot prikazuje slika 22.

Slika 22: Prikaz napake najdene v igri “Reši nalogo”

Napako smo uspešno odpravili z menjavo postavitve elementov na modri plošči, na kateri se nahajata gumba za kopiranje in pridobivanje domače naloge ter okence za vpis URL naslova domače naloge. Postavitev smo menjali iz *free na null* [5] .

Popravljena koda ploščice, ki vsebuje elemente, ki so izginili ob kliku na gumb “Ponastavitev naloge” :

```
Panel_vpis_naloge.setLayout(null);
```

7.2.5 Testiranje igre “Kako dobro ti gre?”

Igra “Kako dobro ti gre?” je bila uspešno testirana, saj smo odkrili napako prikazovanja nivoja, v kateri se nahaja uporabnik. Šlo je za napako, kjer je oznaka, ki je bila zadolžena za prikazovanje nivoja, imela možnost prikazovanja največ pet znakov. To pomeni, da ko je uporabnik zagnal igro, mu je program pravilno izpisal “Nivo:”, vendar ko je uporabnik pravilno odgovoril na prvo enačbo, je, namesto da bi pisalo “Nivo: 1”, pisalo še zmeraj “Nivo:”. Napako smo odpravili s povečanjem števila možnih znakcev v oznaki za nivo. Navedena napaka je bila sicer edina najdena napaka. Igra je delovala po pričakovanjih. Vpis imena, nastavitev časa in končnega števila točk ter izbira matematičnih operacij delujejo normalno. Čas se je pravilno iztekal in se v primeru pravilnega odgovora pravilno ponovno zagnal. Točke so se seštevale pravilno, igra pa se je končala v primeru, ko je uporabnik dosegel končno število točk. Nobene napake nismo ugotovili pri prikazovanju oziroma izbiranju pravilnih odgovorov. Ko smo izbrali pravilen odgovor, ga je prepoznalo kot pravilnega, v nasprotnem primeru je igra ugotovila, da smo se zmotili.

7.2.6 Testiranje igre “Izzivaj prijatelja”

V igri “Izzivaj prijatelja” smo skoraj takoj naleteli na prvo napako. Napaka je bila v tem, da eden izmed uporabnikov, ko je dosegel končno število točk, ni zmagal, ampak je lahko naprej igral.
Po natančnejšem vpogledu v programsko kodo smo ugotovili napako in jo popravili.
Napaka je bila zelo preprosta, vendar je nimo zlahka našli. Šlo je namreč za »tiskarskega škrata« oziroma površno napako, kjer smo spremenljivko, ki je hranila nastavitev končnega števila možnih točk, zamenjali z drugo spremenljivko zelo podobnega imena.
Druga napaka, ki smo jo odkrili, je bila, da ko se je en uporabnik zmotil in je imel manj točk kot drugi, igra tega ni ugotovila in tako drugi igralec z več točkami ni nikoli zmagal. Do napake je prišlo zato, ker ko je en uporabnik pravilno odgovoril, smo preverili samo, če je dosegel končno število točk. Istočasno smo pozabili preveriti, če je ta čas, ko smo mi pravilno odgovorili, drugi igralec že izgubil igro. Rešitev problema je bila nov dodan pogoj v kodo. Primer pseudo kode pogoja [7] :

```
if ((levi_igralec answer right) && (levi igralec points > desni igralec points)){
  levi igralec wins; }
```

Poleg navedenih napak nismo zasledili drugih. Igra deluje po pričakovanjih. Vpis imen, nastavitev časa in končnega števila točk ter izbira matematičnih operacij delujejo. Čas se je pravilno iztekal za oba igralca in v primeru pravilnega odgovora se je čas pravilno resetiral. Točke so se seštevale pravilno. Ko je uporabnik izbrali pravilni odgovor, ga je prepoznalo kot pravilnega, v nasprotnem primeru pa je igra ugotovila napako.

7.2.7 Testiranje igre “Poči pravi balon”

V igri “Poči pravi balon” smo odkrili napako, tako da lahko rečemo, da je bilo testiranje uspešno. Napaka se je pojavila, ko smo hoteli igro resetirati oziroma ponovno zagnati in, ko smo igro zapustili, ter se kasneje vrnili, da bi jo ponovno igrali. Napako je predstavljalo napačno prikazovanje balončkov z rešitvami, saj se balončki niso začeli pojavljati na spodnjem delu igralne površine ampak na sredini oziroma tam, kjer smo predhodno končali prejšnjo igro kot prikazuje slika 23. Poleg nepravilne postavitve balončkov smo opazili tudi zapise na ekranu izven balončkov.

Slika 23: Prikaz napake najdene v igri “Poči pravi balon”

Napaka je bila v tem, da ko smo ponastavili ali zapustili igro, smo pozabili ponastaviti vrednosti postavitve balončkov na začetne. Zato se je pojavila napaka, da so imeli balončki, ko smo hoteli ponovno zagnati igro, nastavitve postavitve še iz prejšnje igre.
Napako smo uspešno odpravili.

7.3 Preverjanje skladnosti s funkcijskimi zahtevami

Ob koncu testiranja je zelo pomembno preveriti, ali smo dosegli vse zastavljene cilje oziroma ali smo zadovoljili vse funkcijske zahteve aplikacije. Izdelali smo aplikacijo z igro, ki ponuja možnost reševanja matematičnih enačb in možnost nastavitve časovnih omejitev in težavnosti.
Aplikacija ponuja večigralsko igro, v kateri se lahko med seboj pomerita dva igralca v hitrostnem reševanju matematičnih enačb. “Reši nalogo” je igra, ki ponuja uporabniku sprejemanje domačih nalog preko spleta ter pošiljanje rešenih nalog nazaj.

Za apliakacijo “Igrajmo matematiko” smo se potrudili, da bi bila čim bolj barvita in da bi vsebovala otrokom primerno oblikovane igre. Sicer je odvisno od posameznika, ali se mu bo zdela aplikacija dovolj barvita in zabavna za osnovnošolce, vendar mi mislimo, da smo tudi to funkcijsko zahtevo opravili. Aplikacija uporablja ob določenih dogodkih tud, ki je posledica aktivnosti uporabnika. S to zadnjo opravljeno zahtevo smo ugodili vsem funkcijskim zahtevam. Za konec lahko povemo, da program deluje dovolj dobro, da gre lahko v izdajo.

8. Zaključek in nadaljno delo

Preko načrtovanja in implementacije aplikacije smo lahko na praktičnem primeru videli, kako poteka razvoj programske opreme. Ugotovili smo, da je za razvoj in nadaljno načrtovanje programskega produkta zelo pomembna analiza zahtev. V tej fazi je zelo pomembno sodelovanje razvijalca programa ter končnega uporabnika, ker lahko samo naročnik pove, kaj želi kot končni produkt, saj on najbolje pozna vsebino problema. Pomembno je tudi to, da znamo ločiti med potrebami in željami uporabnika. Iz analize zahtev izhaja celotno načrtovanje in implementacija, katerima sledita testiranje in validacija. Tako lahko brez težav sklepamo, da je analiza zahtev ena najpomembnejših faz načrtovanja, če ne celo najpomembnejša. Končni produkt predstavlja zaključen program, ki omogoča učenje matematike na zabaven način. Primeren je za osnovnošolske učence začetnih razredov. Ena od morebiti zanimivih reči, ki bi jih lahko imeplementirali v prihodnje, je podobna aplikacija z možnostjo namestitve na pametni mobilni telefon. S tem imamo v mislih to, da bi lahko ponudili uporabnikom večigralsko igro, kjer bi se pomerili v hitrostnem reševanju, enačb tako da bi se med seboj povezali z mobiteli v omrežje. Druga zamisel, ki jo imamo, je možnost implementacije aplikacije, ki ne bi vsebovala samo zbirke matematičnih nalog, ampak bi vsebovalo tematiko vseh predmetov, kot na primer zgodovina, geografija in tako naprej. Aplikacijo bi lahko naredili uporabnikom bolj privlačno z možnostjo prikazovanja statusa uporabnika, v smislu, da bi uporabnik, ki je rešil več nalog, imel status bolj “pridnega” učenca. Ob zaključku našega projekta si želimo, da bi produkt dobro služil naročniku, saj bo s tem naše delo upravičeno.

Slovarček

Slovarček vsebuje nekatere pojme, ki se pojavljajo v nalogi [15] .

Java	Programski jezik.
programski jezik *(angl. programming language)*	Umetni jezik, ki se uporablja za pisanje računalniških programov.
operacijski sistem *(angl. operating system)*	Skupek sistemskih programov, ki omogoča delovanje računalniškega sistema.
računalniški program *(angl. computer program)*	Zaporedje ukazov, ki jih lahko izvede računalnik.
URL *(Uniform Resource Loca- tors)*	Enoličnik krajevnik vira.
SMTP *(Simple mail transfer protocol)*	Standard za prenos elektronske pošte po internet.
aplikacija *(angl. application)*	Računalniški program.
FTP *(File Transfer Protocol)*	Protokol za prenos datotek.
UML *(Unified Modeling Language)*	Standard diagramskega modeliranja.
diagram aktivnost *(angl. activity diagram)*	Diagram poteka izvajanja korakov sistema.
diagram stanj *(angl. state machine diagram)*	Diagram, ki prikazuje stanja in možno prehajanje med stanji kot posledico dogodkov v sistemu.
sekvenčni diagram *(angl. sequence diagram)*	Diagram, ki prikazuje komunikacijo med objekti v smislu sekvence sporočil.
razredni diagram *(angl. class diagram)*	Diagram, ki opisuje strukturo sistema s prikazom razredov, atributov in relacij med razredi.
IDE (*angl. integrated development environment*)	Integrirano razvojno okolje.

Literatura

[1] Mike Cohn, Agile Estimating and Planning, (2005).

[2] Roger pressman, Software Engineering: A Practitioner's Approach , (2004).

[3] Tom DeMarco, Timothy Lister, Peopleware: Productive Projects and Teams, (1999).

[4] Alistair Cockburn, Writing Effective Use Cases, (2000).

[5] Uroš Mesojedec, Borut Fabjan, Java 2, temelji programiranja, (2004)

[6] Martin Fowler, UML Distilled: A Brief Guide to the Standard Object Modeling Lan- guage, (2003).

[7] Erich Gamma, Richard Helm, Ralph Johnson, John M. Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, (1994).

[8] Jim Highsmith, Agile Project Management: Creating Innovative Products, (2004).

[9] Caig Larman, Applying UML and Patterns: An Introduction to Object-Oriented Ana-lysis and Design and Iterative Development, (2003).

[10] Robert C. Martin, Agile Software Development, Principles, Patterns, and Practices, (2002).

[11] Steve McConnell, Code Complete: A Practical Handbook of Software Construction, (2004).

[12] Steve McConnell, Rapid Development, (2003).

[13] Leonard Richardson, Sam Ruby, David Heinemeier Hansson, Restful Web Services, (2007).

[14] Karl Wiegers, Software Requirements, (2003).

Viri

[15] http://islovar.org/, islovar, Slovar informatike, (2011).

[16] http://netbeans.org/, NetBeans the smarter way to code, Open source, Oracle.

[17] http://www.oracle.com/technetwork/java/javamail/index.html, Java API used to receive and send email via SMTP, POP3 and IMAP, Oracle.

[18] http://en.wikipedia.org/wiki/Sequence_diagram,UML diagram, interaction diagram.

[19] http://en.wikipedia.org/wiki/State_diagram, UML diagram, state diagram.

[20] http://en.wikipedia.org/wiki/Class_diagram, UML diagram, structure diagram

Priloga

Zgoščenka s celotno programsko kodo in uporabljenimi slikami za oblikovanje aplikacije je priložena zraven.