

2D računalna igra u C# programskom jeziku

Žnidarec, Karlo

Undergraduate thesis / Završni rad

2016

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Josip Juraj Strossmayer University of Osijek, Faculty of Electrical Engineering, Computer Science and Information Technology Osijek / Sveučilište Josipa Jurja Strossmayera u Osijeku, Fakultet elektrotehnike, računarstva i informacijskih tehnologija Osijek**

Permanent link / Trajna poveznica: <https://um.nsk.hr/um:nbn:hr:200:637632>

Rights / Prava: [In copyright](#) / [Zaštićeno autorskim pravom.](#)

Download date / Datum preuzimanja: **2024-04-19**

Repository / Repozitorij:

[Faculty of Electrical Engineering, Computer Science and Information Technology Osijek](#)



**SVEUČILIŠTE JOSIPA JURJA STROSSMAYERA U OSIJEKU
ELEKTROTEHNIČKI FAKULTET**

Sveučilišni studij

2D RAČUNALNA IGRA U C# PROGRAMSKOM JEZIKU

Završni rad

Karlo Žnidarec

Osijek, 2016.

SADRŽAJ

1. UVOD.....	1
1.1 ZADATAK ZAVRŠNOG RADA.....	1
2. POVIJEST IGARA	2
2.1 INDUSTRIJA VIDEO IGARA.....	4
3. PREGLED KORIŠTENIH TEHNOLOGIJA.....	5
3.1 C# programski jezik	5
3.2 Microsoft Visual Studio	7
4. IZRADA ZADATKA	9
4.1 Uvod u aplikaciju	9
4.1 Križić – Kružić	12
4.2 Vješala	18
5. ZAKLJUČAK.....	25
LITERATURA	26
SAŽETAK	27
ABSTRACT	28
ŽIVOTOPIS	29

1. UVOD

Tema ovog završnog rada je računalna aplikacija koja omogućuje korisniku igranje 2D računalnih igrica. Zadatak je bio izraditi aplikaciju s grafičkim sučeljem koristeći C# programski jezik, aplikacija će omogućiti korisniku igranje 2 popularne računalne igre, „Vješala“ i „Kružić-Križić“. Korisnik iz izbornika odabire koju igru želi igrati, unosi svoje ime/imena. Aplikacija mora omogućiti praćenje rezultata uz mogućnost pohrane rezultata u datoteku kao i učitavanje prethodnih rezultata iz datoteke.

„Vješala“ i „Kružić-Križić“ su veoma poznate igre s kojima je većina ljudi upoznata, cilj ove aplikacije je implementirati ove dvije igre u jednu aplikaciju u kojoj će biti omogućeno zajedničko praćenje statistike, rezultata i tako dalje. U dogovoru s mentorom odlučeno je da će aplikacija imati ranije spomenutu mogućnost praćenja statistike koja će sadržavati ukupno vrijeme igranja korisnika, spremanje rezultata te prikaz najboljeg rezultata pojedine igre.

Kao što i sam naziv teme kaže, za izradu aplikacije je primijenjen C# programski jezik, s kojim smo bili upoznati u kolegiju „Programiranje 2“. Na kolegiju nismo u potpunosti proučili sve mogućnosti navedenog programskog jezika stoga je bilo potrebno samostalno naučiti nešto više o samom C# programskom jeziku. Pretraživanjem interneta pronađena je razna literatura koja će biti navedena na kraju ove dokumentacije. Za izradu aplikacije korišten je „Microsoft Visual Studio“, razvojno okruženje za razvoj aplikacija pisanih u raznim jezicima.

1.1 ZADATAK ZAVRŠNOG RADA

U okviru ovog završnog rada potrebno je upoznati se s C# programskim jezikom, Windows formama te radu s više formi. Potrebno je izraditi aplikaciju s grafičkim sučeljem koristeći gore spomenute tehnologije koja će omogućiti korisniku igranje 2 popularne računalne igre, "Vješala" i "Kružić-Križić". Korisnik iz izbornika odabire koju igru želi igrati, unosi svoje ime/imena. Aplikacija mora omogućiti praćenje rezultata uz mogućnost pohrane rezultata u datoteku kao i učitavanje prethodnih rezultata iz datoteke.

2. POVIJEST IGARA

Povijest računalnih igara [11] počinje krajem 1948. godine kada se pojavljuje Sprava za zabavu od katodnih cijevi (*engl. CathodeRay Tube Amusement Device*), sprava se sastojala od osam katodnih cijevi i simulirala je ispućavanje projektila na cilj, kut i brzinu ispućavanja korisnici su prilagođavali ručicama. Sljedeći korak uslijedio je 4 godine poslije navedenog patenta, riječ je o grafičkoj verziji igre križić-kružić koju je 1952. godine napravio A. S. Douglas, temeljila se na prvom obliku umjetne inteligencije, iako su postojali patenti, prva videoigra koja je bila dostupna javnosti konstruirana je tek 1958. godine.

Prva igra dostupna javnosti zvala se Tenis za dvoje (*engl. Tennis for Two*), njezin autor bio je William Higinbotham, igra je imala bočni pogled na teniski teren, a zadatak je bio prebaciti loptu preko mreže.

Godine 1961. na MIT-u nastala je igra *Spacewar*, igra se igrala u dvoje, a svaki igrač je imao svoj svemirski brod koji je mogao pucati, cilj je bio ubiti protivnika te pobjeći u crnu rupu.

Igre u osamdesetim godinama prošlog stoljeća doživljavaju veliku promjenu, a to je početak proizvodnje video konzola koje su mogle pogoniti više od jedne igre, što je dovelo do porasta broja igara i njihovih proizvođača. 1983. godine zbog prebrzog razvoja dolazi do kraha tržišta igara te su brojni proizvođači i izdavači igara bankrotirali. Krah tržišta nije dugo potrajao te se tako već 1984. tržište oporavlja i nastaju neke popularne igre kao što su *Dragon Quest*, *Super Mario Bros*, *Final Fantasy*. Ono što je važno napomenuti za osamdesete godine je razvoj osobnih računala koja su počela prestizati mogućnosti koje su pružale video konzole.

Igre u devedesetim godinama pojavljuju se nove mogućnosti za razvoj igara u obliku 3D grafike. Godine 1996. na tržištu se pojavljuje *Voodoo* grafička kartica koja je služila za iscrtavanje 3D scena, ovakvu tehnologiju među prvima koristi igra *Quake*, nakon razvoja tehnologije većina igara počela je prelaziti u tehnologiju s tri dimenzije. Devedesetih godina također je većina igara dobila mogućnost igre preko interneta.

Igre u novom stoljeću nisu doživjele značajne promjene, osim velike razlike u grafičkoj kvaliteti, dolazi do razvoja konzola koje nude nove mogućnosti, kao što su spremanje podataka na disk, pristup internetu, igranje preko interneta. Većina igara koje se razvijaju dostupne su za popularnije konzole kao i za PC. Popularnost igara je također porasla, te mlada populacija vrijeme radije provodi uz igre, nego gledajući televiziju.

Križić-Kružić

Postoje vjerovanja da su stari Egipćani 1300. godine pr. Kr. kratili vrijeme igrajući križić-kružić, također se vjeruje da su ga poznavali i stari Rimljani. Ova vrlo jednostavna igra ima čak 26.000 mogućih rješenja. Cilj igre je imati zaredom tri križića ili kružića u jednom smjeru, u anglosaksonskim zemljama takvu igru nazivaju „*tic-tac-toe*”. Prvo računalo koje je igralo ovu stratešku igru samo sa sobom izradili su američki studenti 1980-ih, dotično računalo izloženo je u Muzeju prirodnih znanosti u Bostonu. Ova igra se osim s računalom češće igra s prijateljem, no zabilježene su i igre protiv kokoši koja je u Americi igrala protiv ljudskih protivnika te u 3 godina „karijere“ izgubila samo pet partija.

1952. godine A.S. Douglas je napravio grafičku verziju igre „Križić-Kružić“ sa prvom umjetnom inteligencijom.

Vješala

Korijeni igre „Vješala“ su nepoznati, ali vjeruje se da je igra nastala u Viktorijansko doba. Igra se spominje u knjizi Alice Bertha Gomme naziva „*Traditional Games*“ 1894. godine pod nazivom „Ptice, zvijeri i ribe“. Cilj igre je jednostavan, jedan igrač postavi crtice od kojih svaka predstavlja jedno slovo u riječi koju je igrač postavio, drugi igrač tada pokušava pogoditi koje slovo sadrži riječ, za svaki netočan prijedlog prvi mu igrač dodaje dio tijela čovjeka kojeg će u potpunosti objesiti na vješala ukoliko igrač dva nastavi predlagati netočna slova.

2.1 INDUSTRIJA VIDEO IGARA

Industrija računalnih igara gotovo neprimjetno zarađuje milijarde dolara, posebno su popularne mobilne igre. Analitičarska kompanija Superdata objavila je 2015. godine kako je industrija videoigara samo od digitalne distribucije zaradila 61 milijardu dolara, ukoliko na ovu cifru dodamo i fizičku prodaju igara ta cifra znatno prelazi 100 milijardi dolara.

Sve su veći trend gotovo besplatne mobilne igre, 2015. godine digitalnom su distribucijom zaradile 6,3 milijardi dolara, što je zapanjujuće s obzirom na cijenu od nekoliko dolara po igri.

Mobilne igre koriste trik takozvanih besplatnih igara (*engl. Freemium*), koje je moguće igrati bez plaćanja, ali je veoma teško, te gotovo nemoguće postići značajne rezultate bez kupovanja dodatnih opcija u samoj igri. Iako su takve igre na početku besplatne zapravo zarađuju velike novce samim mikrotransakcijama unutar same igre. Prednost mobilnih igara je što većina ljudi, posebno mlađe populacije posjeduje pametni telefon koji im omogućava igranje mobilnih igara bilo gdje, u školi, na poslu, u toaletu, takvi pametni telefoni se neprestano razvijaju, te postaju moćniji svake godine. Koliko su mobilne igre postale važne svjedoči i izjava prvog čovjeka japanskog izdavača Konamija, Hidekija Hayakawe koji je izjavio da su mobilne igre budućnost i da će se njegova kompanija fokusirati upravo na njih, umjesto na visokobudžetne popularne naslove. Iako je ova izjava zabrinula velik broj igrača koji preferiraju PC igre, nema razloga za zabrinutost, jer velike kompanije žele biti prisutne na tržištu mobilnih igara gdje se okreću veliki novci, što ne znači da neće razvijati igre za PC.

Budućnost platformi za igranje video igara je obuhvatiti virtualnu realnost, proširenu stvarnost, mobilno igranje, prenošenje sadržaja sa interneta, te modularnost u jednoj konzoli.

U Hrvatskoj postoji ovakva grana industrije, godišnji rast im je 50 posto, a samo 2015. godine domaće su tvrtke uknjižile dobit od 50 milijuna kuna. Problem ovakve industrije u Hrvatskoj je izvoz gotovo svega što proizvedu, pošto izvoze u SAD, dolazi do dvostrukog oporezivanja. Poticaja gotovo da i nema, financiraju se samostalno, iako je u planu financiranje novoosnovanih inovativnih poduzeća, takve će se poduzeća moći aplicirati na mjeru za dobivanje do 90 posto sredstava za financiranje i realizaciju projekta. Direktor Lion Game Liona, Nikica Petrušić [10] koji je privukao investiciju od 8 milijuna dolara izjavio kako mu je vrlo teško pronaći prikladne osobe za razvoj igara pošto u Hrvatskoj nema mogućnosti školovanja isključivo za rad u industriji video igara, što je praksa u drugim državama kao što je Švedska.

3. PREGLED KORIŠTENIH TEHNOLOGIJA

Prilikom razrade cjelokupnog postupka za izradu aplikacije bilo je potrebno proširiti znanje stečeno na kolegiju „Programiranje 2“ [5], znanje je nadopunjeno raznim izvorima s interneta koji su navedeni u literaturi.

3.1 C# programski jezik

C# je objektno orijentirani [1] programski jezik koji omogućava pisanje konzolnih, Windows, pa čak i Web aplikacija. Prilikom razvoja korištene su prednosti drugih jezika, uz otklanjanje njihovih mana. Nastao je u tvrtki Microsoft [4], a razvijan je od strane tima stručnjaka pod vodstvom Andresa Hejlsberg-a i Scotta Wiltamuth-a. Tržište ga upoznaje 2000. godine kada izlazi zajedno sa .NET platformom, C# je razvijan s ciljem da bude siguran, jednostavan, moderan, objektno orijentiran jezik visokih performansi. Nastao je na temelju objektnih jezika C++, Java i Visual Basic. Sintaksa programskog jezika C# većim dijelom je preuzeta iz Java također objektno orijentiranog jezika, no za razliku od Java C# je ovisan o platformi, kreiran je za izradu desktop i Internet aplikacija u .Microsoft .NET okruženju.

C# implementira [3] sve dobre odlike objektno orijentiranog programskog jezika iz C++ i Java, te u sklopu .NET platforme omogućava i korisnicima bez programerskog iskustva kreiranje vizualnih aplikacija. Zbog svoje jednostavnosti vrlo ga je lako naučiti, čak i korisnicima bez iskustva u programiranju.

Razvoj aplikacija u C# [6] jeziku jednostavniji je nego razvoj u C++, jer je sintaksa razumljivija početnicima. Kod u C# jeziku je robusniji nego onaj napisan u C++ iz razloga što C# vodi brigu da nakon što je nekom podatku dodijeljen određeni tip, on ne može biti zamijenjen nekim drugim nevezanim tipom. Drugim riječima postoje stroga pravila kojih se mora pridržavati prilikom konverzije tipova. Robusniji kod ima i svoju prednost, a to je jednostavnije otklanjanje grešaka, odnosno .NET u svakom trenutku može provjeriti kojem tipu pripada traženi podatak.

Prednost C# jezika [2] nad ostalim objektno orijentiranim programima je u tome što je jedini napravljen od temelja konkretno za .NET platformu, samim tim može biti i primarni jezik korišten za .NET verzije namijenjene drugim operativnim sustavima. Za primjer, VB.NET napravljen je tako da bude što sličniji svojim prethodnicima i stoga ne može iskoristiti .NET

biblioteku koda u potpunosti, dok je C# jezik napravljen da može iskoristi svaki dio biblioteke u .NET okruženju.

Iako C# nema ograničenja u pogledu kakve sve aplikacije možemo pisati u njemu, ipak postoje tipovi aplikacija koje se najčešće koriste, a to su Windows aplikacije, Web aplikacije i Web servisi.

Windows aplikacije su aplikacije koje preuzimaju izgled osnovnih Windows aplikacija, kao što je Office, koriste Windows Forme unutar .NET okruženja, biblioteka se sastoji od kontrola koje nam koriste prilikom izrade korisničkog sučelja, takve kontrole mogu biti dugmad, meniji, liste, tekstni okviri i drugo.

Web aplikacije koriste sistem Aktivnih Poslužiteljskih Stranica (*engl. Active Server Pages*) .NET (ASP.NET), C# jezik koristimo za izradu ASP.NET aplikacija uz pomoć Web forma, takav sistem omogućava generiranje Web sadržaja i to dinamički, dozvoljavajući personalizaciju, sigurnost i drugo.

Web servisi predstavljaju nov i uzbudljiv način izrade raznovrsnih distribucijskih aplikacija. Razmjenjuju bilo koju vrstu podataka koristeći jednostavnu sintaksu, ne vodeći računa o jeziku u kojem je napisana aplikacija.

3.2 Microsoft Visual Studio

Microsoft Visual Studio je integrirano razvojno okruženje razvijeno od strane tvrtke Microsoft. Visual Studio prisutan je od 1997. godine, te je od njegovog početka objavljeno jedanaest verzija programa, zaključno sa Visual Studio „15“ koji je objavljen u ožujku 2016. godine.

Koristi se za razvoj računalnih programa za Windows, Web stranice, Web aplikacije i Web usluge. Visual Studio sadrži urednik izvornog koda koji podržava *IntelliSense* kao i organiziranje koda. *IntelliSense* je komponenta koja predlaže nastavak koda, odnosno predviđa što korisnik želi upisati te mu nudi automatski ispis istog.

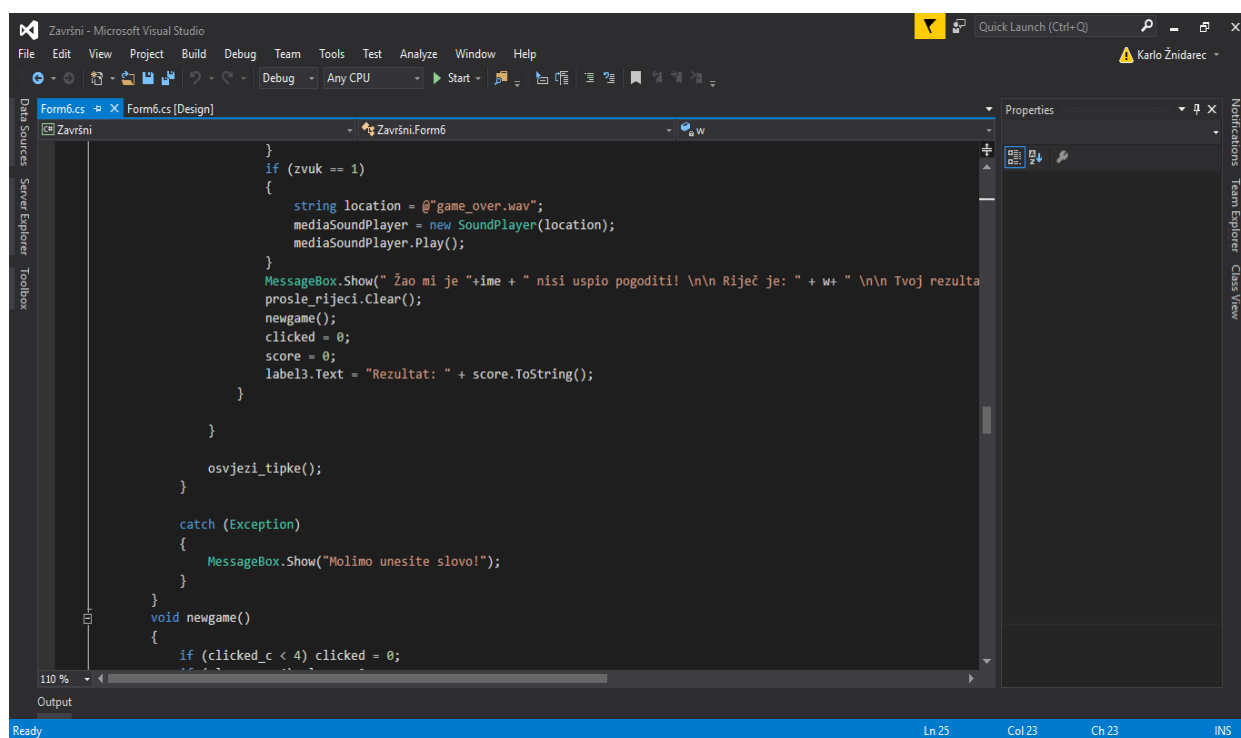
Sadrži alate kao što je dizajner oblika koji se koristi pri izradi aplikacija s grafičkim sučeljem, korištenjem ovakvog alata uvelike se olakšava programiranje aplikacije s grafičkim sučeljem jer je potrebno samo postaviti komponentu na ekran i Visual Studio automatski dodaje kod za odabranu komponentu. Visual Studio podržava različite programske jezike, moguće je dodati skoro sve moguće jezike ukoliko postoje izrađeni dodaci za iste.

Ugrađeni jezici su C, C++, C++/CLI, VB.NET, C#, F#. [9] Podrška za ostale programske jezike kao što su *M*, *Python*, *Ruby* i ostali dostupni su instalacijom jezičnih servisa koji se zasebno instaliraju kao dodatak Visual Studiju.

Umjesto da svaki puta pišemo programski kod od nule, Visual Studio nudi kodove opće namjene koji se nalazi na pravom mjestu, te se takav kod može urediti što uvelike smanjuje vrijeme potrebno za izradu projekta. Sadrži alate za prikazivanje i navigaciju elemenata projekta, bez obzira radi li se o prirodnom kodu jezika C#, nekoj slici ili možda zvučnoj datoteci.

Osim što Visual Studio olakšava pisanje aplikacija, omogućuje i jednostavniju distribuciju i prilagođavanje projekta različitim klijentima, radi lakšeg korištenja i instaliranja.

Velika prednost ovakvog programskog okruženja osim dostupnosti velikog broja jezika u jednoj aplikaciji je to što Microsoft ovaj softver pruža studentima potpuno besplatno preko Microsoft-ovog DreamSpark programa.



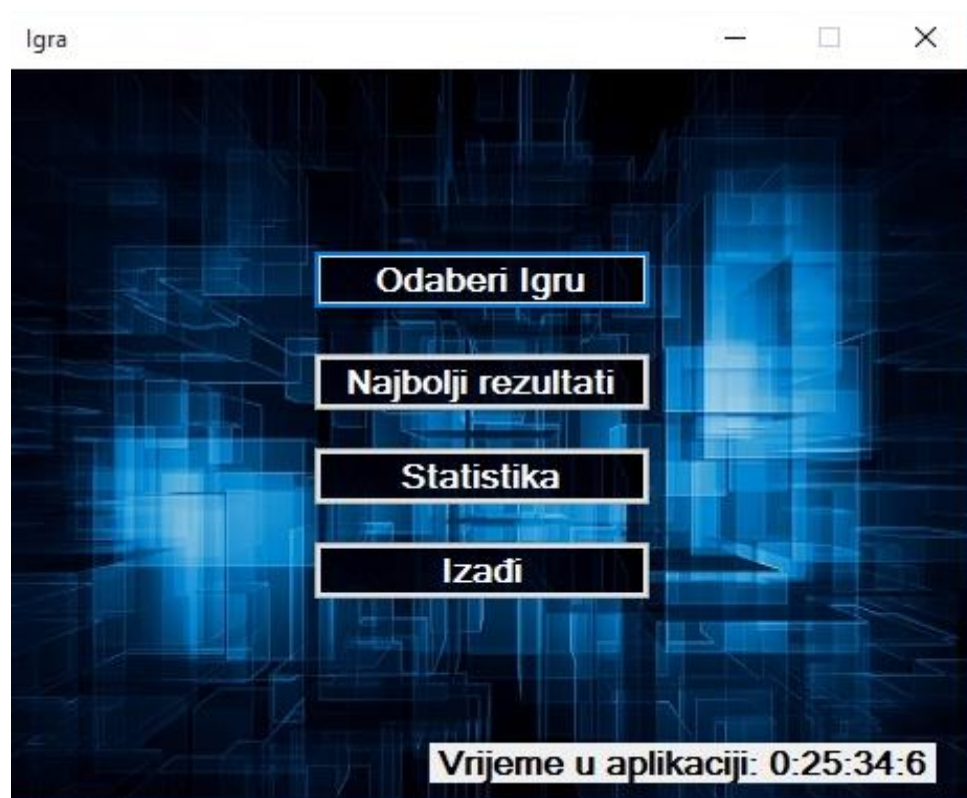
Sl. 3.1. Radno okruženje aplikacije „Microsoft Visual Studio“

4. IZRADA ZADATKA

4.1 Uvod u aplikaciju

Nakon proučavanja teorijske podloge [7] može se krenuti na realizaciju aplikacije, aplikacija je izrađena u 12 formi koje će biti detaljno obrađene u daljnjem tekstu.

Prva forma je prozor koji se prvi otvara prilikom pokretanja aplikacije, nudi mogućnost pregleda najboljih rezultata, pregleda ukupne statistike, te mogućnost otvaranja druge forme na kojoj se odabire željena igra.



Sl. 4.1. Prva Forma aplikacije

Forma jedan jednostavno je izrađena [8], sastoji se od četiri tipke te natpisa na kojem je ispisano ukupno vrijeme aktivnosti u aplikaciji, programski kod ovakve forme je trivijalan te sadrži samo funkcije za čuvanje ukupnog vremena igranja i kod za promjenu forme.

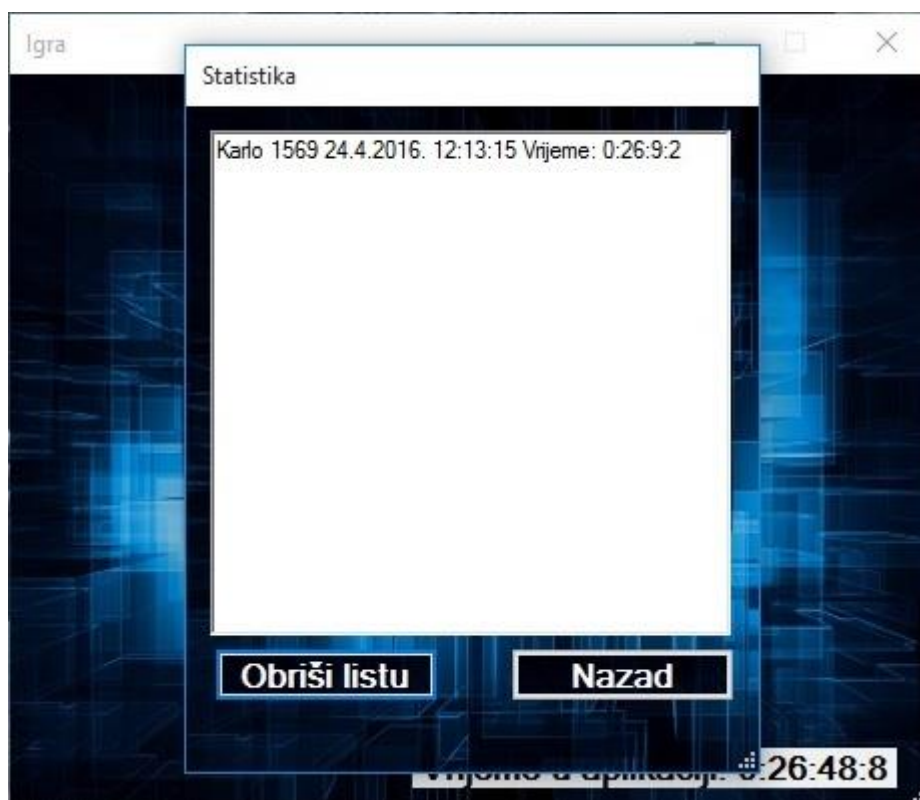
```
Form9 f2 = new Form9();  
f2.ShowDialog();
```

Programski kod 4.1. Promjena forme

Izvođenjem Programskog koda 4.1. – Promjena forme, trenutna forma zamijeniti će se Formom 9.

Forma statistika prati ukupno vrijeme aktivnosti pojedinog korisnika u aplikaciji, u *TextBox* učitava podatke iz datoteke koja se nalazi u instalacijskoj mapi aplikacije.

Tekstualna datoteka u kojoj je spremljena statistika sadrži ime korisnika računala, datum i vrijeme zadnje aktivnosti, te ukupno vrijeme aktivnosti u aplikaciji. Statistika se sprema prilikom izlaska iz aplikacije, zato se vrijeme prikazano u prozoru statistika razlikuje od onog na natpisu forme jedan.



Sl. 4.2. Izgled prozora statistika

Zbog kompliciranosti samog spremanja statistike koja sadrži vrijeme igranja korišten je kod koji se koristi za spremanje i učitavanje rezultata. Uz manje preinake takav programski kod

uspješno izvodi zadatak, iza korisničkog imena prozor sadrži vrijeme u sekundama, te se to vrijeme prilikom pokretanja aplikacije učitava te transformira u sate i minute.

Odabirom najboljih rezultata daje se mogućnost odabira konkretne igre čiji su rezultati od interesa, u *TextBox* se učitava sadržaj tekstualne datoteke koja sadrži rezultate, te se takvi rezultati sortiraju od najvećeg prema najmanjem.

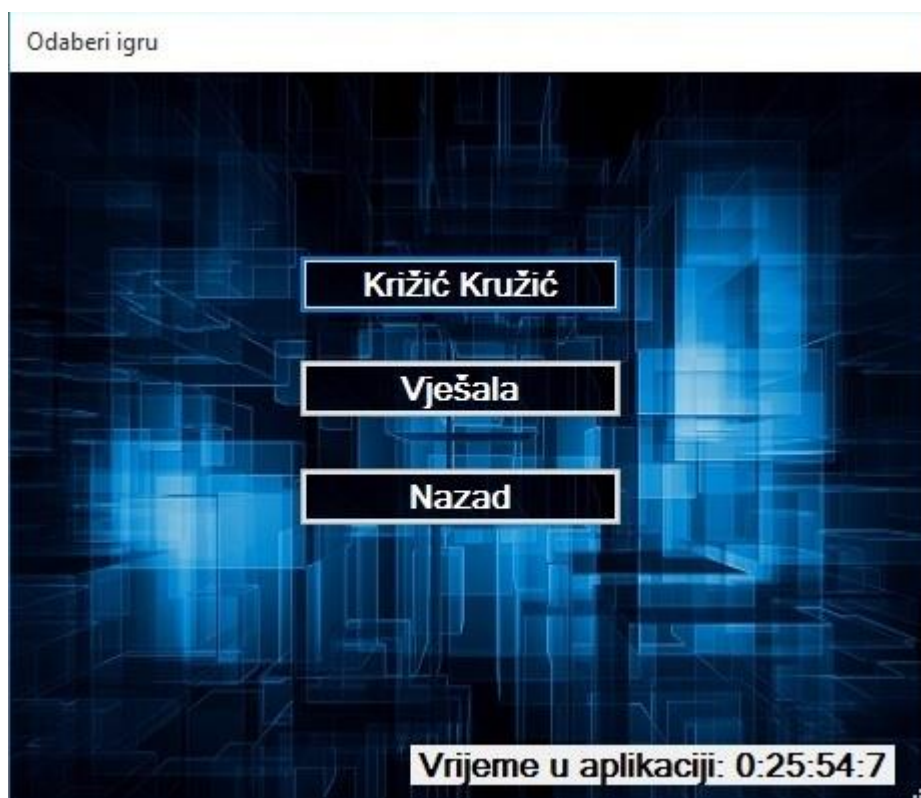


Sl. 4.3. Izgled prozora najbolji rezultati

Za razliku od prijašnjih prozora koji su sadržavali trivijalne programske kodove, ovakav prozor sadrži kompleksan kod kako bi učitao podatke iz tekstualne datoteke, te takve podatke sortirao.

Već je spomenuto kako se programski kod za spremanje rezultata koristi pri spremanju i učitavanju statistike, jedina je razlika u prikazu gdje kod statistike nemamo sortiranje

Pritiskom na prvu tipku odnosno na tipku „*Odaberi igru*“ otvara se forma dva, koja je također trivijalna kao polazni prozor aplikacije, te sadrži samo tri tipke i natpis koji prikazuje ukupno vrijeme igranja koje nasljeđuje od forme jedan. Programska pozadina ovakve forme također sadrži samo funkcije za čuvanje vremena te kodove za promjenu prozora.



Sl. 4.4. Izgled prozora za odabir igre

4.1 Križić – Kružić

Ukoliko korisnik pritisne tipku naziva „*Križić Kružić*“ daje mu se mogućnost odabira načina igranja. Ovaj prozor sadrži samo programski kod za promjene prozora, te služi kao prijelazni prozor.



Sl. 4.5. Pojava prozora biranja načina igre

Ako korisnik odluči igrati protiv računala otvoriti će mu se novi prozor u kojem će aplikacija automatski postaviti ime korisnika računala, korisnik po svojoj volji to ime može promijeniti.



Sl. 4.6. Prozor postavljanja igre protiv računala

Korisniku se također daje mogućnost odabira težine igre, te odabir željenog znaka koji će ga predstavljati u igri protiv računala. Ovakav prozor sadrži funkcije koje spremaju korisničke postavke zahtjevnosti, imena i znaka, te takve postavke prosljeđuju u formu tri koja je glavna forma igre „Križić-Kružić“.

Odabirom igre „U Dvoje“ otvara se prozor sličan prozoru igre „Protiv računala“.



Sl. 4.7. Prozor postavljanja igre u dvoje

Kao i u prozoru postavljanja igre protiv računala ovaj prozor sadrži ime prvog igrača koje se automatski postavlja te se može promijeniti, također sadrži i odabir znaka prvog igrača. Razlika je u tome što ovaj prozor pruža mogućnost upisa imena drugog igrača koje će se postaviti u „*Igrač 2*“ ukoliko korisnik ovo polje ostavi prazno.

Programski kod je gotovo identičan kao programski kod za postavljanje igre protiv računala, jedina je razlika u tome što nema funkciju spremanja zahtjevnosti, te ima funkciju spremanja imena drugog igrača.

Nakon postavljanja svojstava igre otvara se prozor koji sadrži samu igru „Križić-Kružić“.



Sl. 4.8. Prozor igre Križić-Kružić

Prozor sadrži meni traku koja korisniku omogućava prikaz najboljih rezultata, igranje nove igre, resetiranje rezultata, prikaz statistike, te izlaz iz aplikacije. Na prozoru se nalazi mnoštvo natpisa koji prikazuju imena igrača, njihov trenutni rezultat, vrijeme igre, tko drži najbolji rezultat, te trenutni rezultat.

U desnom kutu nalazi se tipka koja gasi/pali zvučne animacije aplikacije, ovisno o trenutnom stanju mijenja se slika tipke. Na sredini prozora nalazi se polje za igru koje sadrži devet tipaka koje dijele istu funkciju za praćenje koji igrač je na potezu, pomakom miša preko tipke privremeno se postavlja znak „X“ ili „O“ ovisno o znaku igrača koji je na potezu, kada se miš makne sa tipke taj se znak uklanja ukoliko korisnik nije pritisnuo tipku. Programski kod prati je li igra završena, odnosno je li popunjeno svih devet pozicija ili je jedan igrač ostvario niz od tri znaka.

```

private void button_click(object sender, EventArgs e)
{
    if (zvuk == 1) // provjerava je li zvuk omogućen
    {
        string location = @"Backs.wav";
        mediaSoundPlayer = new SoundPlayer(location);
        mediaSoundPlayer.Play();
    }
    Button b = (Button)sender;
    if (turn) {
        b.Text = ply1_zn; //postavlja tekst tipke u znak igrača koji je na
potezu
    }
    else
    {
        b.Text = ply2_zn;
        turn = !turn;
        turn_count++;
        b.Enabled = false;
        timer2.Start();
        CheckForWinner();
        if ((!turn) && against_comp) // provjerava je li drugi igrač na potezu te
je li drugi igrač računalo
        {
            computer_move();
        }
    }
}

```

Programski kod 4.2. Funkcija koja reagira na pritisak tipke s polja igre

Funkcija „*CheckForWinner()*“ provjerava sve moguće pozicije te traži pobjednički potez, odnosno tri znaka u nizu koja bi predstavljala pobjedu korisnika kojeg taj znak reprezentira.

Funkcija „*computer_move()*“ konstruirana je kao umjetna inteligencija koju računalo mora posjedovati ukoliko želi biti dostojan protivnik korisniku protiv kojeg igra, u takvoj funkciji postoje dva načina rada, način rada se odabire na prozoru postavljanja igre protiv računala, te ovisno o odabranoj težini računalo izvodi pametnije ili manje pametne poteze.

```

private void computer_move()
{
    Button move = null;
    if (kraj != true)
    {
        //pregledaj mogućnosti
        move = potrazi_mjesto(ply2_zn); //mogu li pobjediti
        if (move == null)
        {
            if (tezina == 1) { move = potrazi_mjesto(ply1_zn); } //blokiraj
            težina zahtjevno
            else move = trazi_slobodan_prostor();
            if (move == null)
            {
                if (tezina == 1 && turn_count !=0) { move = vidi_sredinu(); }
                else move = vidi_kut();
                if (move == null)
                {
                    move = trazi_slobodan_prostor();
                }
            }
        }

        move.PerformClick();
    }
}

```

Programski kod 4.3. Funkcija umjetne inteligencije zadužena za poteze računala

Izvedba Programskog koda 4.3. Funkcija umjetne inteligencije koja je zadužena za poteze računala zahtijevala je proučavanje same igre „Križić-Kružić“ te zaključivanje koji potezi donose povoljan rezultat za računalo ovisno o mogućnostima.

Funkcija „*potrazi_mjesto()*“ provjerava ima li računalo (*ply2_zn*) dva znaka u nizu, te nakon što pronađe dva znaka u nizu provjerava je li treće mjesto slobodno kako bi ostvario pobjedu postavljanjem znaka na to slobodno mjesto. Nadalje ukoliko je težina postavljena na zahtjevno sada računalo provjerava istom funkcijom „*potrazi_mjesto()*“ ima li protivnik dva znaka u nizu, te ukoliko ima popunjava treće mjesto kako protivnik ne bi ostvario pobjedu. Ukoliko je težina postavljena na lagano, tada će računalo samo popuniti slučajno slobodno mjesto na polju za igru.

Proučavanjem igre „Križić-Kružić“ zaključeno je da je najbolji potez ukoliko je računalo drugo na potezu postaviti znak u sredinu, naravno ako je ta pozicija slobodna, stoga ova funkcija upravo to i radi ako je težina postavljena na zahtjevnju, ukoliko nije, tada računalo traži prvi slobodan kut i popunjava isti. Ako računalo ima prvenstvo poteza tada popunjava slučajnu poziciju.

4.2 Vješala

Ukoliko na prozoru za odabir igre korisnik odabere „*Vješala*“ otvara mu se prozor za postavljanje igre vješala.



Sl. 4.9. Prozor za postavljanje igre vješala

Korisniku se daje mogućnost odabira područja iz kojeg želi pogađati riječi, ako korisnik ne odabere područje, automatski mu se postavlja područje „*Brendovi*“, ime mu se kao i kod igre „*Križić-Kružić*“ automatski postavlja u ime računala, koje po želji korisnik može promijeniti. Nadalje korisnik iz ovog prozora može provjeriti najbolje rezultate igre *Vješala*, te može dodati novu riječ u jednu od baza riječi. Odabirom tipke „*Dodaj riječ*“ otvara mu se prozor za dodavanje nove riječi.

Dodaj riječ u bazu

Odaberi područje: ☐ Životinje ☐ Države ☐ Gradovi ☐ Sport ☐ Automobili ☐ Brendovi

Unesi novu riječ:

Sl. 4.10. Prozor za dodavanje nove riječi u bazu riječi

Korisnik kao i na prethodnom prozoru može odabrati područje u koje želi dodati novu riječ, ukoliko ne odabere niti jedno područje automatski će se riječ dodati u bazu riječi „*Brendovi*“, te ukoliko korisnik ne unese niti jednu riječ u za to predviđeno područje neće dobiti mogućnost dodavanja riječi.

```
private void button2_Click(object sender, EventArgs e)
{
    if (radioButton1.Checked) { bazarijeci = "zivotinje.txt"; }
    else if (radioButton2.Checked) { bazarijeci = "drzave.txt"; }
    else if (radioButton3.Checked) { bazarijeci = "gradovi.txt"; }
    else if (radioButton4.Checked) { bazarijeci = "sport.txt"; }
    else if (radioButton5.Checked) { bazarijeci = "automobili.txt"; }
    else if (radioButton6.Checked) { bazarijeci = "brendovi.txt"; }
    if (nova_rijec.Text == "") { MessageBox.Show("Nisi upisao novu riječ!", ""); }
}

TextWriter tw = new StreamWriter(bazarijeci, true);
tw.WriteLine(nova_rijec.Text);
tw.Close();
this.Hide();
Form5 f2 = new Form5();
f2.ShowDialog();
}
```

Programski kod 4.4. Funkcija za unos novih riječi u bazu podatka

Kada korisnik pritisne tipku „Igraj“ na prozoru za postavljanje igre Vješala otvara mu se glavni prozor igre Vješala, prozor koji sadrži sve potrebne funkcije kako bi takva igra uspješno izvršavala svoju funkciju.



Sl. 4.11. Prozor igre „Vješala“

Ovaj prozor sadrži meni traku u kojoj je korisniku omogućen uvid u rezultate igre, korisnik po želji može promijeniti kategoriju iz koje želi pogađati riječ, te kao i u igri „Križić-Kružić“ daje mu se mogućnost uvida u statistiku, resetiranja rezultata, te početka nove igre.

Sadrži tri takozvane grupe elemenata, u prvoj grupi nalaze se natpisi koji daju uvid u trenutni rezultat, prikazuje najbolji rezultat, odabranu kategoriju, slova koja su promašena pokušaj, te duljinu tražene riječi.

Osim natpisa sadrži i tri sličice, dvije sličice reprezentiraju pomoć pri igranju (đokere), sličica „reload“ vraća pogreške na nulu, te tako korisniku daje mogućnost testiranja više slova, sličica „A“ korisniku otkriva jedno slovo iz riječi. Ovakve pomoći pri igranju mogu se koristiti po jednom u svakoj riječi i tri puta tijekom igre. Treća sličica predstavlja zvuk, odnosno pritiskom na tu sličicu zvuk se pali / gasi ovisno o prijašnjem stanju.

Druga grupa elemenata sadrži sličicu koja se mijenja ovisno o prijedlogu slova, ukoliko se prijedlog nalazi u samoj riječi tada se prikazuje zelena sličica sa palcem gore, dok se prijedlog

slova ne nalazi u riječi prikazuje se znak „X“ vidljivo na slici 4.11. Osim sličice sadrži trideset i jedno slovo te znak „razmak“, pritiskom na pojedinu tipku označava prijedlog tog slova. Tipka „Nazad“ vraća aplikaciju u prethodni prozor, dok tipka „Izlaz“ gasi aplikaciju.

Treća grupa elemenata sadrži vizualni prikaz pogrešaka, odnosno prilikom pokretanja igre pojavljuje se samo pozadina koja prikazuje drvo i liticu, svakom novom pogreškom dodaju se dijelovi tijela čovjeka. Moguće je pet puta promašiti slovo ukoliko se ne koriste pomoći pri igri, s pomoćima ovaj broj raste do deset.

Programski kod vezan za prozor igre „Vješala“

```
void prikazi_labelu()
{
    w = uzmi_rijecdb().ToLower(); // uzima slučajnu riječ iz baze riječi koja
    je odabrana
    w.Replace(" ", "");
    if (prosle_rijeci.Contains(w)) prikazi_labelu(); // ukoliko se odabrana
    riječ pojavila u trenutnoj igri pronadi drugu
    prosle_rijeci.Add(w); // dodaj riječ u riječi koje su se pojavile u
    trenutnoj igri
    char[] ch = w.ToCharArray();

    int space = 569 / ch.Length - 1; // razmak između crtica (slova)

    . . . .
```

Programski kod 4.5. Glavni dio funkcije koja prikazuje slučajnu riječ

```

string uzmi_rijecdb()
{
    string list = "";
    if (choice == 0) list = System.IO.File.ReadAllText("brendovi.txt");
    else if (choice == 1) list = System.IO.File.ReadAllText("zivotinje.txt");
    else if (choice == 2) list = System.IO.File.ReadAllText("drzave.txt");
    else if (choice == 3) list = System.IO.File.ReadAllText("gradovi.txt");
    else if (choice == 4) list = System.IO.File.ReadAllText("sportovi.txt");
    else if (choice == 5) list = System.IO.File.ReadAllText("automobili.txt");
    string[] words = list.Split('\n');
    Random rand = new Random();
    return words[rand.Next(0, words.Length - 1)];
}

```

Programski kod 4.6. Funkcija za odabir slučajne riječi iz baze riječi

Programski kod 4.6. Funkcija za odabir slučajne riječi iz baze riječi učitava u listu sve riječi iz odabrane tekstualne datoteke, te nakon toga bira jednu nasumično riječ sa kreirane liste.

```

private void provjeri_slovo()
{
    label5.Text = "Najbolji rezultat: " + highname + ": " + highscore;
    try
    {
        char letter = letter2; // pridružuje predloženo slovo varijabli "letter"

        if (tocniodgovori.Contains(letter)) // slovo se nalazi u listi tocnih odgovora
        {
            MessageBox.Show("Već si probao slovo: " + letter + " , bilo je ispravno!",
"Greška!");
            return;
        }
        else if (netocniodgovori.Contains(letter)) // slovo se nalazi u listi netocnih
odgovora
        {
            MessageBox.Show("Već si probao slovo: " + letter + " , bilo je neispravno!",
"Greška!");
            return;
        }
        if (w.Contains(letter)) // predloženo slovo se nalazi u zadanoj riječi
        {
            if (zvuk == 1) // provjerava da li je zvuk omogućen
            {
                string location = @"Backs.wav"; // zvuk "klika" na tipku
                mediaSoundPlayer = new SoundPlayer(location);
                mediaSoundPlayer.Play();
            }
            char[] LS = w.ToLower().ToCharArray();
            for (int i = 0; i < LS.Length; i++)
            {
                if (LS[i].ToString() == letter.ToString()) // provjerava na kojoj se
poziciji nalazi predloženo slovo
                {
                    labels[i].Text = letter.ToString(); // prikaži slovo u riječi
                    answer.Image = Image.FromFile("okay.jpg"); // pokaži sliku palca gore
                    tocniodgovori.Add(letter); // dodaj slovo u listu točnih odgovora
                }
            }

            foreach (Label l in labels)
            {
                if (l.Text == "__") return; // ukoliko ima "praznih" mjesta korisnik još
pogađa
            }

            MessageBox.Show("Pogodio si " + ime + "\n\n Spasio si čovjeka");
            score += (5 - missed);
            if (zvuk == 1)
            {
                string locationz = @"yay_z.wav"; // zvuk pobjede
                mediaSoundPlayer = new SoundPlayer(locationz);
                mediaSoundPlayer.Play();
            }
            newgame();
            label3.Text = "Rezultat: " + score.ToString();
            osvjezi_tipke();
        } else {
            . . . . .
        }
    }
}

```

Programski kod 4.7. Funkcija provjere predloženog slova,
dio funkcije koja se obavlja ukoliko je predloženo slovo ispravno

```

private void pictureBox3_Click(object sender, EventArgs e)
{
    if (slovo == 0) // provjerava je li joker već korišten u trenutnoj riječi
    {
        slovo_c++; // ukupan broj korištenja ovog jokera se povećava
        slovo = 1; // označava kako je ovaj joker korišten, zbog prevencije ponovnog
        korištenja
        char[] LS = w.ToLower().ToCharArray();
        for (int i = 0; i < LS.Length; i++)
        {
            if (!tocniodgovori.Contains(LS[i]) && !netocniodgovori.Contains(LS[i])) //
            ukoliko slovo u igri nije niti promašeno niti pogođeno
            {
                labels[i].Text = LS[i].ToString(); // provjerava sva moguća slova
                koje_sl = LS[i].ToString(); // sprema ispravno slovo u varijablu
                "koje_sl"
                answer.Image = Image.FromFile("okay.jpg"); // postavlja sliku u palac
                gore
                tocniodgovori.Add(LS[i]); // dodaje pronađeno slovo u točne odgovore
                break;
            }
        }
        for (int i = 0; i < LS.Length; i++)
        {
            if (LS[i].ToString() == koje_sl) { labels[i].Text = koje_sl; } //
            prikazuje pronađeno slovo u riječi
        }
        foreach (Label l in labels) // provjerava je li to posljednje slovo, te
        započinje novu igru ukoliko je
        {
            if (l.Text == "__") return;
            MessageBox.Show("Pogodio si " + ime + "\n\n Spasio si čovjeka");
            score += (5 - missed);
            newgame();
            label3.Text = "Rezultat: " + score.ToString();
            osvjezi_tipke();
        }
        else if (slovo == 1 && slovo_c < 4) MessageBox.Show("Već si iskoristio ovaj đoker
        na ovoj riječi!", "Greška!");
        else if (slovo_c == 4) MessageBox.Show("Nemaš više pravo na đoker!", "Greška!");
        // iskorišten je tri puta u cijeloj igri
    }
}

```

Programski kod 4.8. Funkcija pomoći u igri u vidu otkrivanja jednog slova u riječi

5. ZAKLJUČAK

Za realizaciju ove aplikacije bilo je potrebno proučiti veliku količinu gradiva ako se osoba po prvi puta ozbiljno susreće sa C# programskim jezikom, na internetu je dostupan veliki broj videa iz kojih se može naučiti sve što je potrebno, također postoji puno knjiga koje u dubinu objašnjavaju principe i mogućnosti C# programskog jezika. Prije početka ovakvog projekta važno je proučiti što više literature kako bi s razumijevanjem i bez većih problema savladali prepreke koje će se gotovo sigurno pojaviti prilikom ispunjavanja traženih zadataka.

Tijekom izrade aplikacije pojavljivali su se razni problemi koji su brzo i efikasno riješeni uz pomoć navedene literature, izradom ove aplikacije stečeno je veliko znanje o samom objektno orijentiranom programiranju, te rješavanju problema s kojim se programer susreo.

Prostor za napredak aplikacije postoji, od zadataka koji bi se mogli odraditi su vizualni efekti, moguće je grafički urediti aplikaciju, kao što je zamjena svih tipka slikama, takva nadogradnja aplikacije ne bi predstavljala problem. Od većih nadogradnja postoji mogućnost dodavanja više igara u samu aplikaciju, te prebacivanje aplikacije u Web aplikaciju, zamjena datoteka iz datoteka aplikacije sa MySQL bazom podataka, online igra protiv drugih korisnika. Važno je napomenuti kako će iskustvo stečeno stvaranjem ove aplikacije značajno pomoći u daljnjem radu u području programiranja.

LITERATURA

- [1] Head first C# (3rd edition), Jennifer Green and Andrew Stellman, O' Reilly® Media, Inc., USA, 2013.
- [2] Programming C# 5.0: Building Windows 8, Web, and Desktop Applications for the .NET 4.5 Framework, Ian Griffiths, O' Reilly® Media, Inc., USA, 2012.
- [3] Upute za C# programiranje, <https://msdn.microsoft.com/en-us/library/67ef8sbd.aspx>, 26.04.2016.
- [4] C# Programming, https://en.wikibooks.org/wiki/C_Sharp_Programming, 26.04.2016.
- [5] Objektno orijentirano programiranje - vježbe,
http://www.etfos.unios.hr/~lukic/oop/Auditorne_vje%C5%BEbe_5.pdf , 26.04.2016.
- [6] C# Tutorials, [https://msdn.microsoft.com/en-us/library/aa288436\(v=vs.71\).aspx](https://msdn.microsoft.com/en-us/library/aa288436(v=vs.71).aspx) ,
26.04.2016.
- [7] C# Tutorial for absolute beginners, https://www.youtube.com/watch?v=bFdP3_TF7Ks ,
26.04.2016.
- [8] Primjer izrade igre u C#, <https://www.youtube.com/watch?v=p3gYVcggQOU>, 26.04.2016.
- [9] Microsoft Visual Studio, https://en.wikipedia.org/wiki/Microsoft_Visual_Studio,
26.04.2016.
- [10] Hrvatska industrija video igara, <http://dnevnik.hr/vijesti/hrvatska/hrvatska-industrija-videoigara---419620.html>, 17.06.2016.
- [11] Povijest računalnih igara, <https://dmkbj10.wikispaces.com/Povijest+racunalnih+igara>,
17.06.2016.

SAŽETAK

Tema završnog rada je „2D računalna igra u C# programskom jeziku“. Aplikacija je izrađena pomoću Microsoft Visual Studio programskog okruženja. Cjelokupni projekt se sastoji od izrade višestrukih prozora koji obavljaju različitu svrhu u aplikaciji. Svi ti dijelovi projekta čine funkcionalnu cjelinu kako bi aplikacija radila bez problema, te kako bi se podaci uspješno izmjenjivali između različitih prozora. Zadatak je bio omogućiti igranja igre „Križić-Kružić“ u dvoje ili protiv računala, za igru protiv računala bilo je potrebno konstruirati umjetnu inteligenciju koja bi bila dostojan protivnik korisniku. U igri „Vješala“ bilo je potrebno omogućiti korisniku odabir između više kategorija riječi, dodavanje nove riječi u bazu riječi, te pomoći pri igri u vidu otkrivanja slova u riječi. Za obje igre bilo je potrebno izraditi spremanje rezultata, te ispis najboljeg rezultata tijekom igre. Aplikacija također sprema ukupnu statistiku igranja, u kojoj se nalazi ime korisnika, zadnja aktivnost, te ukupno vrijeme aktivnosti korisnika u aplikaciji. Aplikacija je na kraju grafički dotjerana, te su dodani zvučni efekti kako bi aplikacija bila atraktivnija.

KLJUČNE RIJEČI:

C#, Visual Studio, iks oks, križić,kružić,vješala, forme, statistika, umjetna inteligencija

ABSTRACT

Topic of this final paper is „2D computer game in C# programming language“. Application is created using Microsoft Visual Studio programming environment. The whole project consists of making multiple forms performing different purpose in the application. All these forms make functional unit so application can work without problems, and so application can successfully exchange data between different forms. The task is to enable playing the game „Tic-Tac-Toe“ against the computer or against other player, to play against the computer it is necessary to construct an artificial intelligence that would be a worthy opponent to the user. In the game „Hangman“ it is necessary to allow the user to select between several categories of words, and add new word to the base of words, allow user to use jokers like revealing letter in word. For both games it is necessary to save results, and show the best result while playing the game. The application also saves overall statistic, statistic is consisted of user name, last activity, and the total time user was active in the application. The application is finally graphically polished and sound effects were added to make application more attractive.

KEYWORDS:

C Sharp, Visual Studio, Tic Tac Toe, Hangman, Form, statistic, artificial intelligence

ŽIVOTOPIS

Karlo Žnidarec je rođen 18.11.1994. u Našicama. Osnovnu školu je upisao 2001. godine u školi Josipa Jurja Strossmayera u Đurđenovcu koju je završio 2009. godine. Nakon završene osnovne upisao je srednju tehničku školu, smjer tehničar za elektroniku u srednjoj školi Isidora Kršnjavog u Našicama iste godine, koju je završio 2013. godine. Nakon završene srednje škole upisao je smjer elektrotehnika na Elektrotehničkom fakultetu u Osijeku u 2013. godini.

.

KARLO ŽNIDAREC
