

TEHNIČKA ŠKOLA BJELOVAR

Dr. Ante Starčevića 24

ERIK BAUKOVAC

ZAVRŠNI RAD

IZRADA KVIZA U C++

Područje rada: ELEKTROTEHNIKA

Program: TEHNIČAR ZA RAČUNALSTVO

Bjelovar, svibanj 2017.

TEHNIČKA ŠKOLA BJELOVAR

Dr. Ante Starčevića 24

ERIK BAUKOVAC

ZAVRŠNI RAD

IZRADA KVIZA U C++

Područje rada: ELEKTROTEHNIKA

Program: TEHNIČAR ZA RAČUNALSTVO

Mentor: EMINA GRMIĆ, prof.

Bjelovar, svibanj 2017.

Nadnevak predaje rada: _____

(mentor je prihvatio pisani dio izradbu)

Potpis mentora: _____

Ocjena izradbe završnog rada: _____

Nadnevak obrane završnog rada: _____

Ocjena obrane završnog rada: _____

KONAČNA OCJENA: _____

(prijedlog Povjerenstva)

Povjerenstvo:

Mentor: _____

Profesor struke: _____

Profesor struke: _____

Profesor struke: _____

Profesor struke: _____

Izdvojeno mišljenje ili eventualni komentar:

Rad je izrađen u Tehničkoj školi Bjelovar (ili naziv poduzeća u kojem je rad izrađen).

Rad ima 34 stranice i 11 slika.

Sadržaj

1. UVOD	2
2. MICROSOFT VISUAL STUDIO 2015.....	3
2.1. UREĐIVANJE KODA	3
2.2. PROGRAM ZA PRONALAŽENJE POGREŠAKA.....	3
2.3. ALATI ZA TESTIRANJE	3
3. ZAGLAVLJA I FUNKCIJE U C++	4
4. KLASSE	5
4.1 OBJEKTNO ORIJENTIRANO PROGRAMIRANJE	5
4.2 KLASSE UNUTAR KODA.....	5
5. GRAĐA KVIZA.....	7
5.1. PITANJE.H	7
5.2. PITANJE.CPP.....	8
5.3. KVIZ.H	11
5.4. KVIZ.CPP	12
5.5. KVIZ E.CPP.....	16
6.RAD KVIZA.....	25
7. ZAKLJUČAK.....	28
8. POPIS LITERATURE	29

1. UVOD

Tema završnog rada programiranje je kviza u C++. Cilj ovog rada je objasniti osnove korištenog softvera i detaljno opisati korišteni kod i kako radi. U radu se pobliže objašnjavaju osnove korištenog softvera i koda. Rad je podijeljen u više cjelina.

Prvo je opisan softver korišten za programiranje, zatim funkcije programskog jezika C++, a nakon toga detaljno je opisan cjelokupni kod kviza. Ovu temu sam izabrao jer mi se svidio izazov koji mi pruža programiranje u C++ te sam želio dodatno unaprijediti svoje znanje u programiranju s C++. Programiranje u C++ nije jednostavno za savladati i zahtjeva puno prakse, logičnog razmišljanja i strpljenja. C++ je opće namjenski, objektno orijentirani programski jezik koji se koristi za programiranje desktop aplikacija, servera i softvera te je jedan od najčešće korištenih programskih jezika danas. Izrada završnog rada će se vršiti u programu Microsoft Visual Studio 2015. Ovim radom, kroz zadatak i teoriju objasniti ću princip rada kviza te ponešto o samom C++ i njegovim funkcijama. Pitanja u kvizu napisana su na temu HTML-a i CSS-a.

2. MICROSOFT VISUAL STUDIO 2015

Microsoft Visual Studio 2015 softver je namijenjen programiranju. Podržava veliki broj programskih jezika poput C, C++, Visual C++, Visual Basic .NET, Visual C#, Python i mnogih drugih. Microsoft Visual Studio 2015 ima i mnoge značajke poput editora koda, programa za pronalaženje pogrešaka, puno različitih dizajnera, ostalih alata i fleksibilnost na ekstenzije. Microsoft Visual Studio dolazi u 5 različitih verzija, a to su Community, Professional, Enterprise, Test Professional i Express.

2.1. UREĐIVANJE KODA

Microsoft Visual Studio sadrži značajku IntelliSense. IntelliSense značajka je koja ubrzava proces programiranja smanjivanjem grešaka pri upisu i uobičajenih grešaka. Pri kodiranju ispisuje se lista objekata, informacije o parametrima u funkciji, informacije o deklaraciji identifikatora i završava se ostatak varijable, naredbe ili funkcije koju smo započeli pisati. Visual Studio također pruža funkcije navigacije u kodu pomoću GoTo i Peek, pogleda na povezane funkcije pomoću Code Lens, lake promjene strukture koda pomoću integriranog razvojnog okruženja i brzog ispravljanja pogrešaka odmah tijekom programiranja ili pogledom u Error List.

2.2. PROGRAM ZA PRONALAŽENJE POGREŠAKA

Microsoft Visual Studio pruža program za pronalaženje pogrešaka neovisno o programskom jeziku koji se koristi i platformi na kojoj se radi. Pomoću njega može se odrediti breakpoint to jest mjesto na kojem se može provjeriti što se događa s varijablom na tom mjestu. Pauziranjem programa moguće je provjeriti vrijednost varijable na razne načine. Nedostatci u kodu i neočekivane situacije se manifestiraju kao izuzetci čije upozorenje slijedi odmah. Sve povezane niti mogu se vidjeti u jednom prozoru i upravljati se s njima.

2.3. ALATI ZA TESTIRANJE

Microsoft Visual Studio ima i veliki broj alata za testiranje koda. Unit Test trga funkcionalnost programa na male jedinice. IntelliTest provjerava .NET kod i za svaku liniju u kodu generira se unos koji izvršava liniju. UI Test provjerava ispravnost aplikacije i njenog sučelja. Postoje još mnogi testovi za provjeru performansi na Internetu i cjelokupnog koda.

3. ZAGLAVLJA I FUNKCIJE U C++

U radu je korišteno šest C++ zaglavlja, a to su `<string>`, `<vector>`, `<iostream>`, `<fstream>`, `<time.h>` i `<cstdlib>`.

Zaglavlje `<string>` koristi se za uvođenje stringa u kod.

Zaglavlje `<vector>` koristi se za uvođenje vektora u kod.

Zaglavlje `<iostream>` standardno je C++ zaglavlje koje služi za definiranje upisa i ispisa objekata.

Zaglavlje `<fstream>` koristi se za interakciju s datotekom.

Zaglavlje `<time.h>` koristi se za dobivanje i upravljanje podacima o vremenu.

Zaglavlje `<cstdlib>` koristi se za definiranje nekoliko funkcija opće namijene.

Funkcije koje su korištene u programu su: *find* - pronalazi zadanu varijablu, *length* - daje dužinu varijable, *erase* - briše dio varijable, *stoi* - pretvara string u integer tip podatka, *cout* - služi za ispis podataka, *rand()* - generira nasumičan broj, *getline* - izvlači članove iz pohrane, *push_back* - stavlja varijablu na kraj vektora, *close* - zatvara korištenu datoteku, *srand* - pseudo nasumično generira vrijeme na temelju zadane vrijednosti, *time* - daje trenutno vrijeme, *cin* - služi za unos podataka, *ignore* - ignorira vrijednost koja mu je dodijeljena, *open* - otvara datoteku, *out* - otvara datoteku za pisanje, *trunc* - briše sav kontekst koji je postojao u datoteci, *clear* - briše greške u cin funkciji

Osim korištenih zaglavlja postoje i druga, a neka od njih su `<cassert>`, `<cctype>`, `<cerrno>`, `<cfenv>`, `<cfloat>`, `< cinttypes>`, `<ciso646>`, `<climits>`, `<locale>`, `<cmath>`, `<list>`, `<random>`, `<cstdarg>`, `<regex>`, `<set>`, `<ratio>`, `<new>`, `<locale>`.

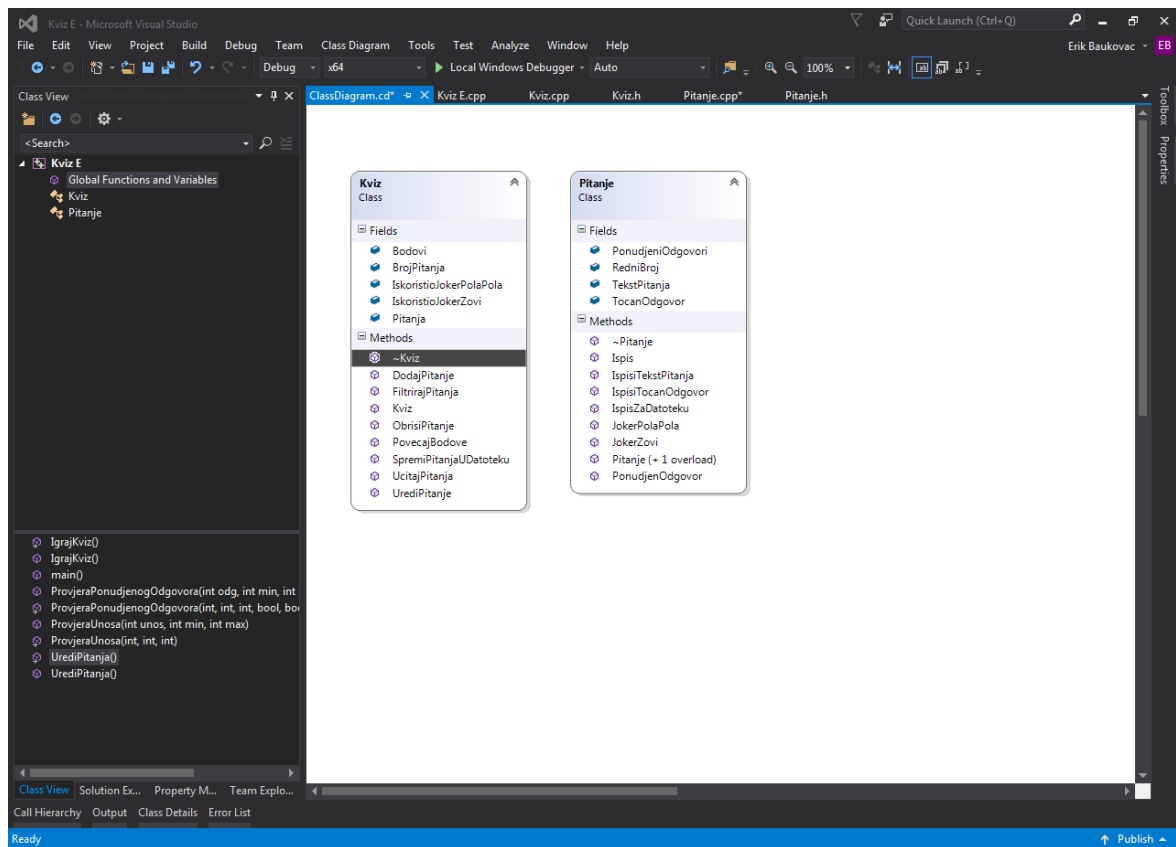
4. KLASSE

4.1 OBJEKTNO ORIJENTIRANO PROGRAMIRANJE

Objektno orijentirano programiranje metoda je programiranja koja se temelji na definiranju klasa kao samostalne programske cjeline. Objekt je naziv za skup svojstava koja se mogu objediniti u smislenu cjelinu. Konstruktor je funkcijski član koji se automatski poziva prilikom stvaranja objekta. On se deklarira kao funkcijski član koji nema određeni povratni tip i identičnog je imena kao i klasa. Destruktor je funkcijski član zadužen za oslobađanje svih memorijskih resursa dodijeljenih objektu i postoji samo jedan destruktor po klasi. On se deklarira kao funkcijski član koji nema određeni povratni tip i identičnog je imena kao i klasa ispred kojeg stoji znak tilda (~). Funkcije koje su povezane s objektom nazivaju se metode. Prijatelj klase je klasa, funkcija ili funkcijski član koji imaju pravo pristupa privatnim i zaštićenim članovima neke druge klase. S istim nazivom može se definirati više metoda s različitim tipovima podataka, to se naziva polimorfizam. I najvažnije svojstvo klasa nasljeđivanje je gdje se jedna klasa definira unutar druge klase to jest jedna klasa nasljeđuje drugu klasu. Pomoću nasljeđivanja se brže i efikasnije stvara nova klasa iz postojećih.

4.2 KLASSE UNUTAR KODA

Program je podijeljen na dvije klase (slika 1.). Klasa *Pitanje* sadrži četiri varijable i devet metoda. Varijable klase *Pitanje* su *PonudjeniOdgovori*, *RedniBroj*, *TekstPitanja* i *TocanOdgovor*, a metode korištene u klasi *Kviz* su *~Pitanje*, *Ispis*, *IspisiTeksPitanja*, *IspisiTocanOdgovor*, *IspisZaDatoteku*, *JokerPolaPola*, *JokerZovi*, *Pitanje* i *PonudjenOdgovor*. Klasa *Kviz* sadrži pet varijabli i devet metoda. Varijable klase *Kviz* su *Bodovi*, *BrojPitanja*, *IskoristioJokerPolaPola*, *IskoristioJokerZovi* i *Pitanja*, a korištene metode su *~Kviz*, *DodajPitanje*, *FiltrirajPitanja*, *Kviz*, *ObrisiPitanje*, *PovecajBodove*, *SpremiPitanjaUDatoteku*, *UcitajPitanja* i *UrediPitanje*. Globalne metode programa su *IgrajKviz()*, *main()*, *ProvjeraPonudjenogOdgovora*, *ProvjeraUnosa* i *UrediPitanja*.



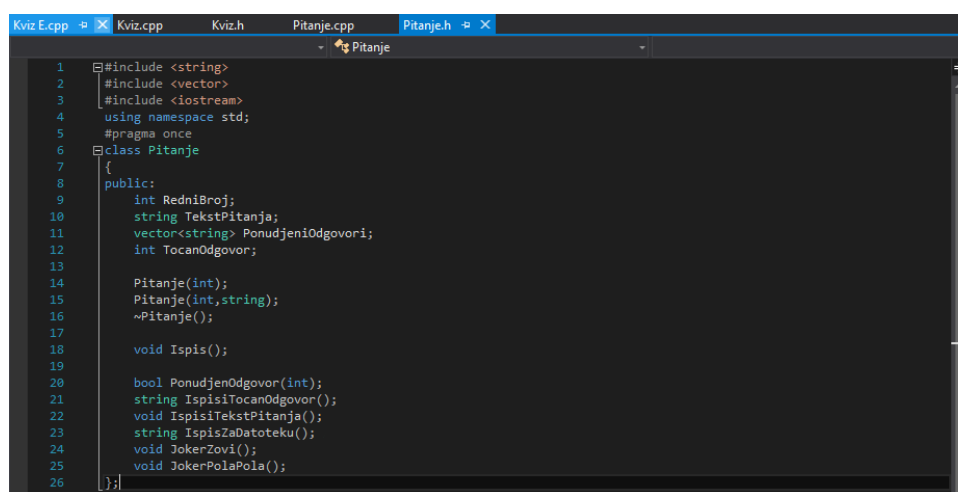
Slika 1. Klase

5. GRAĐA KVIZA

Kod za kviz sastoji se od petsto četrdeset linija koda i dvije klase podijeljenih u pet različitih datoteka. Kviz radi s datotekom Pitanja.txt u kojoj su spremljena pitanja i odgovori na temu HTML i CSS. Kod je podijeljen u pet dijelova radi lakšeg snalaženja i preglednosti.

5.1. PITANJE.H

Pitanje.h temelj je programa. U njemu su uključena zaglavlja `<string>`, `<vector>` i `<iostream>`. *Using namespace std* skraćuje kod i pomaže kod raspoznavanja identifikatora. *#pragma once* uključuje izvorni file samo jednom po kompajliranju, a to znači da program ima manje koda, brže se kompajlira i da su izbjegnuti sudari imena. Prvo je otvorena klasa *Pitanje* u kojoj su javni članovi *RedniBroj*, *TekstPitanja*, *PonudjeniOdgovori* i *TocanOdgovor*. Zatim je pozvan konstruktor za redni broj pitanja, konstruktor za redni broj i tekst pitanja te destruktor. Funkcije u klasi su funkcija *Ispis* koja se koristi za ispis pitanja i odgovora, funkcija *PonudjeniOdgovor* koja se koristi za provjeru točnosti odgovora, funkcija *IspisiTocanOdgovor* koja se koristi kako bi program vratio točan odgovor, funkcija *IspisiTekstPitanja* koja se koristi da bi program ispisao tekst pitanja, funkcija *IspisZaDatoteku* koja se koristi za upisivanje pitanja u datoteku te funkcije *JokerZovi* i *JokerPolaPola* koje služe za definiciju jokera zovi i jokera pola pola.



```
1 #include <string>
2 #include <vector>
3 #include <iostream>
4 using namespace std;
5 #pragma once
6 class Pitanje
7 {
8 public:
9     int RedniBroj;
10    string TekstPitanja;
11    vector<string> PonudjeniOdgovori;
12    int TocanOdgovor;
13
14    Pitanje(int);
15    Pitanje(int,string);
16    ~Pitanje();
17
18    void Ispis();
19
20    bool PonudjeniOdgovor(int);
21    string IspisiTocanOdgovor();
22    void IspisiTekstPitanja();
23    string IspisZaDatoteku();
24    void JokerZovi();
25    void JokerPolaPola();
26 }
```

Slika 2. Pitanje.h

5.2. PITANJE.CPP

Ovdje su definirane funkcije koje su navedene u zaglavlju *Pitanje.h*. Funkcije su uključene pomoću *#include* funkcije. Prvo je pozvan konstruktor za redni broj i dodijeljena mu je varijabla *Rb*.

```
Pitanje::Pitanje(int Rb)
{
    RedniBroj = Rb;
}
```

Zatim je pozvan konstruktor za redni broj i tekst pitanja.

```
Pitanje::Pitanje(int Rb, string Linija)
```

Dodijeljena je varijabla rednom broju i oznaka za delimiter.

```
RedniBroj = Rb;
string delimiter = ",";
```

Pomoću funkcije *substr* program dohvaća dio pitanja koji je označen s delimiterom i upisuje ga pod varijablom *TekstPitanja* te briše taj dio pitanja zajedno s delimiterom i to ponavlja četiri puta kako bi izvadilo i odgovore na isti način.

```
TekstPitanja = Linija.substr(0, Linija.find(delimiter));
Linija.erase(0, Linija.find(delimiter) +
delimiter.length());
```

```
PonudjeniOdgovori.push_back(Linija.substr(0,
Linija.find(delimiter)));
Linija.erase(0, Linija.find(delimiter) +
delimiter.length());
```

```
PonudjeniOdgovori.push_back(Linija.substr(0,
Linija.find(delimiter)));
Linija.erase(0, Linija.find(delimiter) +
delimiter.length());
```

```
PonudjeniOdgovori.push_back(Linija.substr(0,
Linija.find(delimiter)));
```

```
Linija.erase(0, Linija.find(delimiter) +  
delimiter.length());
```

```
PonudjeniOdgovori.push_back(Linija.substr(0,  
Linija.find(delimiter)));  
Linija.erase(0, Linija.find(delimiter) +  
delimiter.length());
```

Pomoću funkcije *stoi* string *Linija* se pretvara u integer tip podatka.

```
TocanOdgovor = stoi(Linija);
```

Nakon toga je pozvan destruktor u kod.

```
Pitanje::~Pitanje()  
{  
}
```

Zatim je definirana funkcija za ispis gdje je u prvom redu definiran ispis rednog broja i teksta pitanja, a zatim redni broj i tekst odgovora.

```
void Pitanje::Ispis()  
{  
    cout << RedniBroj << ". " << TekstPitanja << endl;  
    for (unsigned int i = 1; i <= PonudjeniOdgovori.size();  
i++)  
        cout << i << ") " << PonudjeniOdgovori[i - 1] << endl;  
    cout << endl;  
}
```

Zatim je definirana funkcija za provjeru točnosti odgovora.

```
bool Pitanje::PonudjenOdgovor(int Odgovor)  
{  
    if (Odgovor == TocanOdgovor)  
        return true;  
    return false;  
}
```

Zatim je definirana funkcija za ispis točnog odgovora.

```
string Pitanje::IspisiTocanOdgovor()  
{  
    return PonudjeniOdgovori[TocanOdgovor - 1];  
}
```

Zatim je definirana funkcija za ispis teksta pitanja.

```
void Pitanje::IspisiTekstPitanja() {  
    cout << TekstPitanja << endl;}
```

Sljedeća funkcija je funkcija za upis u datoteku

```
string Pitanje::IspisZaDatoteku  
{  
    string r = TekstPitanja + "," + PonudjeniOdgovori[0] + ","  
+ PonudjeniOdgovori[1] + "," + PonudjeniOdgovori[2] + "," +  
PonudjeniOdgovori[3] + "," + to_string(TocanOdgovor) + "\n";  
    return r;  
}
```

Nakon nje dolazi funkcija za joker zovi.

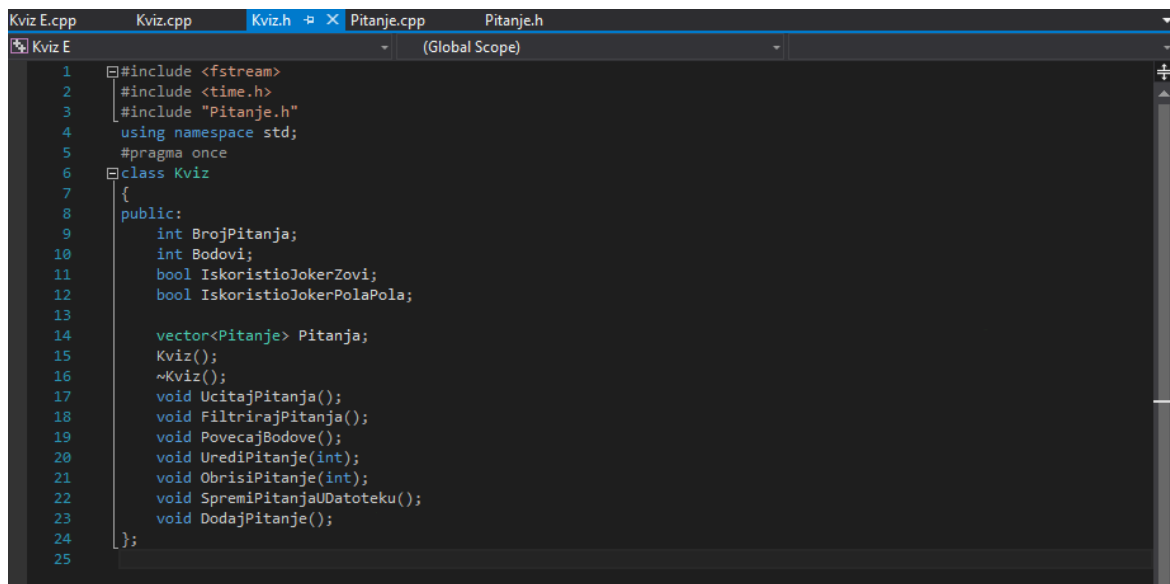
```
void Pitanje::JokerZovi()  
{  
    cout << "\nPrijatelj: Tocan odgovor je " <<  
PonudjeniOdgovori[TocanOdgovor - 1] << endl;  
}
```

I na kraju se definira funkciju za joker pola-pola.

```
void Pitanje::JokerPolaPola()  
{  
    while (PonudjeniOdgovori.size() > 2)  
    {  
        int randNum = rand() % PonudjeniOdgovori.size();  
  
        if (randNum != TocanOdgovor - 1)  
        {  
            if (randNum < TocanOdgovor - 1)  
            {  
                TocanOdgovor--;  
            }  
  
            PonudjeniOdgovori.erase(PonudjeniOdgovori.begin() +  
randNum);  
        }  
    }  
}
```

5.3. KVIZ.H

Ovo je drugo zaglavlje programa. U njemu se uključuje `<fstream>`, `<time.h>` i "Pitanje.h" zaglavlje. Ovdje se također koristi `#pragma once` i ima istu funkciju u ovom dijelu koda kao i kod Pitanje.h zaglavlja. Otvara se klasa *Kviz* u kojoj su javni članovi *BrojPitanja*, *Bodovi*, *IskoristioJokerZovi* i *IskoristioJokerPolaPola*. Prvo se stvara vektor klase *Pitanje* zvan *Pitanja*, zatim konstruktor i destruktor klase. Funkcije u ovoj klasi su funkcija *UcitajPitanja* pomoću koje se učitavaju pitanja u datoteku, funkcija *FiltrirajPitanja* koja služi za nasumičan odabir pitanja iz liste pitanja, funkcija *PovecajBodove* koja se koristi za povećanje broja bodova, funkcija *UrediPitanje* koja služi za uređivanje pitanja koja se već nalaze u datoteci, funkcija *ObrisiPitanje* koja služi za brisanje pitanja s liste pitanja, funkcija *SpremiPitanjaUDatoteku* koja služi za spremanje pitanja u datoteku i funkcija *DodajPitanje* koja služi za dodavanje novih pitanja na kraj liste.



```
1 #include <fstream>
2 #include <time.h>
3 #include "Pitanje.h"
4 using namespace std;
5 #pragma once
6 class Kviz
7 {
8 public:
9     int BrojPitanja;
10    int Bodovi;
11    bool IskoristioJokerZovi;
12    bool IskoristioJokerPolaPola;
13
14    vector<Pitanje> Pitanja;
15    Kviz();
16    ~Kviz();
17    void UcitajPitanja();
18    void FiltrirajPitanja();
19    void PovecajBodove();
20    void UrediPitanje(int);
21    void ObrisiPitanje(int);
22    void SpremiPitanjaUDatoteku();
23    void DodajPitanje();
24 };
25
```

Slika 3. Kviz.h

5.4. KVIZ.CPP

Ovdje su definirane funkcije koje su navedene u zaglavlju Kviz.h. Funkcije su uključene pomoću *#include* funkcije. Prvo se definira konstruktor klase u kojemu se namješta broj bodova i pitanja na 0 te da joker zovi i joker pola-pola nisu iskorišteni stavljajući njihove vrijednosti na false.

```
Kviz::Kviz()
{
    Bodovi = 0;
    BrojPitanja = 0;
    IskoristioJokerPolaPola = false;
    IskoristioJokerZovi = false;
}
```

Nakon toga se poziva destruktor u kod.

```
Kviz::~Kviz()
{
}
```

Zatim se definira funkciju za učitavanje pitanja u program. U while petlji definirano je da program učitava red po red i sprema učitano u string varijablu. Funkcija *push_back* stavlja novi element na kraj vektora.

```
void Kviz::UcitajPitanja()
{
    ifstream file("Pitanja.txt");
    string str;
    int i = 1;
    while (getline(file, str))
    {
        Pitanja.push_back(Pitanje(i, str));
        i++;
        BrojPitanja++;
    }
    file.close();
}
```


Zatim se definira funkcija koja nasumično odabire polovicu pitanja s liste koristeći funkcije *srand* i *time*. Kombinacijom funkcije *srand* koja pseudo nasumično generira brojeve po zadanoj vrijednosti i funkcije *time* koja daje trenutnu vrijednost vremena napravljen je generator nasumičnih brojeva koji se ne ponavljaju i uvijek su drugačiji.

```
void Kviz::FiltrirajPitanja()
{
    vector<Pitanje> temp;

    srand(time(NULL));
    int brPitanja = Pitanja.size() / 2;
    for (int i = 0; i < brPitanja; i++)
    {
        int randNum = rand() % (Pitanja.size() - 1);
        temp.push_back(Pitanja[randNum]);
        Pitanja.erase(Pitanja.begin() + randNum);
    }
    Pitanja = temp;

    for (int i = 0; i < Pitanja.size(); i++)
    {
        Pitanja[i].RedniBroj = i + 1;
    }

    BrojPitanja = Pitanja.size();
}
```

Zatim se definira funkcija za povećavanje broja bodova.

```
void Kviz::PovecajBodove()
{
    Bodovi++;
}
```

Zatim se definira funkcija za uređivanje pitanja.

```
void Kviz::UrediPitanje(int Index)
```

Pomoću `system("cls")` briše se sav sadržaj iz konzolne aplikacije radi bolje preglednosti.

```
system("cls");  
string Tekst;  
int Tocan;  
cout << "Unesite pitanje:" << endl;
```

`cin.ignore()` stavljena je kako bi program ignorirao novi red nakon pritiska tipke Enter kod unosa pitanja.

```
cin.ignore();
```

`getline(cin, Tekst)` služi za upis teksta u zadanu varijablu, a ovdje je to `Tekst`.

```
getline(cin, Tekst);  
Pitanja[Index].TekstPitanja = Tekst;  
cout << endl << "Unesite prvi odgovor:" << endl;  
getline(cin, Tekst);  
Pitanja[Index].PonudjeniOdgovori[0] = Tekst;  
cout << endl << "Unesite drugi odgovor:" << endl;  
getline(cin, Tekst);  
Pitanja[Index].PonudjeniOdgovori[1] = Tekst;  
cout << endl << "Unesite treci odgovor:" << endl;  
getline(cin, Tekst);  
Pitanja[Index].PonudjeniOdgovori[2] = Tekst;  
cout << endl << "Unesite cetvrti odgovor:" << endl;  
getline(cin, Tekst);  
Pitanja[Index].PonudjeniOdgovori[3] = Tekst;  
cout << endl << "Unesite broj tocnog odgovora (1-4):" <<  
endl;
```

Nakon što se unesu pitanja i odgovori, unosi se i broj točnog odgovora.

```
cin >> Tocan;  
Pitanja[Index].TocanOdgovor = Tocan;
```

Zatim se definira funkcija za brisanje pitanja koja briše pitanje s liste i smanjuje broj pitanja na listi.

```
void Kviz::ObrisiPitanje(int Index)
{
    Pitanja.erase(Pitanja.begin() + Index);
    BrojPitanja--;
}
```

Zatim se definira funkcija za spremanje pitanja u datoteku.

```
void Kviz::SpremiPitanjaUDatoteku()
    ofstream file;
```

Otvora se datoteka i briše se sav sadržaj tijekom pokretanja programa.

```
file.open("Pitanja.txt", ofstream::out | ofstream::trunc);

for each(Pitanje p in Pitanja)
{
    file << p.IspisZaDatoteku();
}
file.close();
```

I na kraju se definira funkcija za unos novih pitanja na listu.

```
void Kviz::DodajPitanje()
```

Stvara se objekt za redni broj.

```
Pitanje tempPitanje(BrojPitanja + 1);
string Tekst;
int Tocan;
cout << "Unesite pitanje:" << endl;
cin.ignore();
getline(cin, Tekst);
tempPitanje.TekstPitanja = Tekst;
cout << endl << "Unesite prvi odgovor:" << endl;
getline(cin, Tekst);
tempPitanje.PonudjeniOdgovori.push_back(Tekst);
cout << endl << "Unesite drugi odgovor:" << endl;
getline(cin, Tekst);
tempPitanje.PonudjeniOdgovori.push_back(Tekst);
```

```

cout << endl << "Unesite treci odgovor:" << endl;
getline(cin, Tekst);
tempPitanje.PonudjeniOdgovori.push_back(Tekst);
cout << endl << "Unesite cetvrti odgovor:" << endl;
getline(cin, Tekst);
tempPitanje.PonudjeniOdgovori.push_back(Tekst);
cout << endl << "Unesite broj tocnog odgovora (1-4):" <<
endl;
cin >> Tocan;
tempPitanje.TocanOdgovor = Tocan;

```

Dodaju se pitanja u vektor pomoću *push_back* funkcije.

```

Pitanja.push_back(tempPitanje);
BrojPitanja++;

```

5.5. KVIZ E.CPP

Ovdje su definirane funkcije navedene u zaglavlju Kviz.h koje se uključuju pomoću *#include* funkcije. Prvo se kreiraju četiri funkcije *IgrajKviz*, *UrediPitanja*, *ProvjeraPonudjenogOdgovora* i *ProvjeraUnosa*. Zatim se otvara main program i definira se while petlja za odabir radnji pri otvaranju programa u kojem je definirano da je 3. odabir izlaz iz programa.

```

int Odabir = 0;
while (Odabir != 3)
{
    system("cls");
    cout << "Dobro dosli.\nOdaberite redni broj ispred
akcije:"<<endl<<endl;
    cout << "1. Igraj kviz" << endl;
    cout << "2. Uredi pitanja" << endl;
    cout << "3. Izlaz" << endl;

    cin >> Odabir;
}

```

Provjerava se unos. Ako korisnik odabere 1., tada je odabrao igranje kviza. Pokreće se funkcija *IgrajKviz* nakon čijeg se izvršenja *Odabir* vraća na 0 kako bi ponovno biranje bilo moguće.

```

        if (ProvjeraUnosa(Odabir, 1, 3))
        {
            if (Odabir == 1)
            {
                IgrajKviz();

                Odabir = 0;
            }

```

Ako je odabir 2., tada je korisnik odabrao uređivanje pitanja, pokreće se funkcija *UrediPitanja* nakon čijeg se izvršenja *Odabir* vraća na 0 kako bi ponovno biranje bilo moguće.

```

        if (Odabir == 2)
        {
            UrediPitanja();

            Odabir = 0;
        }
    }

```

Ako je odabir drugačiji od tri ponuđena odgovora, tada program ignorira unos zbog funkcije *cin.ignore* kojoj je postavljena granica s *numeric_limits<streamsize>* i vraća odabir na 0.

```

    else
    {
        cin.clear();
        cin.ignore(numeric_limits<streamsize>::max(), '\n');
        Odabir = 0;
    }

    system("cls");

```

Nakon toga definira se funkcija *IgrajKviz* gdje se prvo inicijalizira objekt klase *Kviz*, poziva se funkcija *UcitajPitanja* i *FiltrirajPitanja* i odgovor se postavlja na 0.

```

Kviz Kviz;
Kviz.UcitajPitanja();
Kviz.FiltrirajPitanja();

```

```
int Odgovor = 0;
```

Definira se predložak funkcije *for_each* koji primjenjuje definiranu funkciju na odabrani raspon. Odabrano je da se funkcija primjenjuje na svako pitanje.

```
for each(Pitanje p in Kviz.Pitanja)
```

Nakon toga definira se *while* petlja to jest funkcija koja će primjenjivati *for_each*.

```
while (Odgovor < 1 || Odgovor > 6)
```

Poziva se funkcija *Ispis* i definira se poziv na funkciju *JokerZovi*.

```
p.Ispis();
```

```
if (Odgovor == -5)
{
    p.JokerZovi();
}
```

Provjerava se jesu li jokeri iskorišteni. Ako nisu, pokazuju se korisniku kako bi ih mogao izabrati te se definira unos i ispis.

```
if(!Kviz.IskoristioJokerZovi)
```

```
    cout << "\n5) Joker zovi" << endl;
```

```
if (!Kviz.IskoristioJokerPolaPola)
```

```
    cout << "\n6) Joker pola-pola" << endl;
```

```
cout << endl;
```

```
cin >> Odgovor;
```

Definira se *if* naredba grananja unutar koje se stavlja *switch case* i provjerava je li uneseni odgovor točan.

```
if (ProvjeraPonudjenogOdgovora(Odgovor, 1, 6,
    Kviz.IskoristioJokerPolaPola, Kviz.IskoristioJokerZovi))
    switch (Odgovor)
    {
        case 1:
        case 2:
```

```

        case 3:
        case 4:
            if (p.PonudjenOdgovor(Odgovor))

```

Ako je odgovor točan, broj bodova se inkrementira te se ispisuje poruka.

```

Kviz.PovecajBodove();
cout << endl << "Bravo, tocan odgovor!" << endl << endl;

```

Ako odgovor nije točan, ispisuje se poruka da odgovor nije točan i poruka koji je odgovor točan.

```

else {
    cout << endl << "Vas odgovor nije tocan!\nTocan odgovor je pod
    rednim brojem " << p.TocanOdgovor << ". " <<
    p.IspisiTocanOdgovor() << endl << endl;
}

```

```

system("Pause");

```

```

break;

```

Zatim se definira slučaj 5 i 6 to jest slučajevi za jokere se pozivaju i postavlja se da su iskorišteni.

```

        case 5:
            Odgovor = -5;
            Kviz.IskoristioJokerZovi = true;
            break;
        case 6:
            p.JokerPolaPola();
            Kviz.IskoristioJokerPolaPola = true;
            Odgovor = 0;
            break;

```

Zatim ako je unos za if naredbu ispred switch case netočan, ignorira se unos i vraća se vrijednost odgovora na 0.

```

else
{
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
    Odgovor = 0;}

```

Zatvara se while petlja i postavlja se odgovor na 0. Na kraju se ispisuje broj bodova i postotak.

```
Odgovor = 0;
    }
    system("cls");
    cout << "Vas rezultat: " << Kviz.Bodovi << "/" <<
Kviz.BrojPitanja <<endl<<endl;
    cout << "Vas rezultat: " << Kviz.Bodovi * 100 /
Kviz.BrojPitanja <<"%"<< endl << endl;

    system("Pause");
```

Nakon toga definira se funkcija *UrediPitanja* u kojoj se prvo inicijalizira objekt klase *Kviz*, poziva funkcija *UcitajPitanja* i postavlja pitanje i odabir na 0.

```
Kviz Kviz;
Kviz.UcitajPitanja();
int OdabranoPitanje = 0;
int OdabranaAkcija = 0;
```

Zatim se prvo otvara while petlja

```
while (OdabranoPitanje != Kviz.BrojPitanja + 2)
```

unutar koje se inicijalizira varijabla i definira for_each koji ispisuje broj pitanja i pitanje.

```
int j = 1;
system("cls");
for each(Pitanje p in Kviz.Pitanja)
{
    cout << j << ". ";
    p.IspisiTekstPitanja();
    j++;
}
```

Zatim se ispisuju opcije za dodavanje pitanja i povratak te se dodaje unos.

```
cout << endl;
cout << Kviz.BrojPitanja + 1 << ". Dodaj pitanje" << endl;
```



```
cout << Kviz.BrojPitanja + 2 << ". Povratak" << endl << endl;
cin >> OdabranoPitanje;
```

Otvara se prva if naredba.

```
if (ProvjeraUnosa(OdabranoPitanje, 1, Kviz.BrojPitanja + 2))
```

Otvara se druga if naredba.

```
if (OdabranoPitanje > 0 && OdabranoPitanje <=
Kviz.BrojPitanja)
```

Otvara se druga while petlja koja ispisuje izbornik i definira unos

```
while (OdabranaAkcija != 3){
    system("cls");
    Kviz.Pitanja[OdabranoPitanje - 1].Ispis();
    cout << endl;
    cout << "Odaberite redni broj ispred akcije:" << endl << endl;
    cout << "1. Uredi pitanje!" << endl;
    cout << "2. Obrisi pitanje!" << endl;
    cout << "3. Izlaz!" << endl << endl;

    cin >> OdabranaAkcija;
```

Otvara se treća if naredba s kojom se bira hoće li se urediti pitanje ili će ga se obrisati. Ako je unos nevažeći, program ignorira unos i vraća odabir na 0. Zatvara se treća if i while petlja.

```
if (ProvjeraUnosa(OdabranaAkcija, 1, 3))
{
    if (OdabranaAkcija == 1)
        Kviz.UrediPitanje(OdabranoPitanje - 1);
    if (OdabranaAkcija == 2)
    {
        Kviz.ObrisiPitanje(OdabranoPitanje - 1);
        break;
    }
}
else{
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
```

```

                                OdabranaAkcija = 0;
                                }
                                }

```

Vraća se unos na 0 i zatvara se druga if naredba grananja .

```
OdabranaAkcija = 0;
```

Postavlja se funkcija za dodavanje pitanja, zatvara se prva if i while funkcija te se postavlja funkciju za ignoriranje netočnog unosa.

```

if (OdabranoPitanje == Kviz.BrojPitanja + 1)
{
    system("cls");
    Kviz.DodajPitanje();
}
else
{
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
    OdabranoPitanje = 0;
}
}

```

Postavlja se odabrano pitanje i akcija na 0 te se otvara while petlja za izlaz iz UrediPitanja.

```

OdabranoPitanje = 0;

system("cls");

OdabranaAkcija = 0;

while (OdabranaAkcija != 1 && OdabranaAkcija != 2)
{

```

Ispisuje se tekst za mogućnost spremanja promjena. Definira se unos i s if naredbom provjerava se hoće li se pitanja spremiti li ne. Ako hoće, poziva se funkcija za spremanje pitanja.

```

cout << "Zelite li spremiti vase promjene?" << endl << endl;
    cout << "1. Da" << endl;
    cout << "2. Ne" << endl << endl;
    cin >> OdabranaAkcija;

```

```

    if (ProvjeraUnosa(OdabranaAkcija, 1, 2))
    {
        if (OdabranaAkcija == 1)
        {
            Kviz.SpremiPitanjaUDatoteku();
        }
    }

```

Stavlja se zaštita protiv krivog unosa i zatvaraju sve petlje.

```

else
{
    cin.clear();
    cin.ignore(numeric_limits<streamsize>::max(), '\n');
    OdabranaAkcija = 0;
}

```

Nakon toga definira se funkcija za provjeru ponuđenog odgovora. Bool tip znači da može biti samo istinit ili lažan.

```

bool ProvjeraPonudjenogOdgovora(int odg, int min, int max,
bool jokerpp, bool jokerz)
{
    bool dobarOdg = false;
    for (int i = min; i <= max; i++)
    {
        if (odg == i)
        {
            if (i >= 1 && i <= 4)
                dobarOdg = true;
            if (i == 5 && !jokerz)
                dobarOdg = true;
            if (i == 6 && !jokerpp)
                dobarOdg = true;

            break;
        }
    }
    return dobarOdg;
}

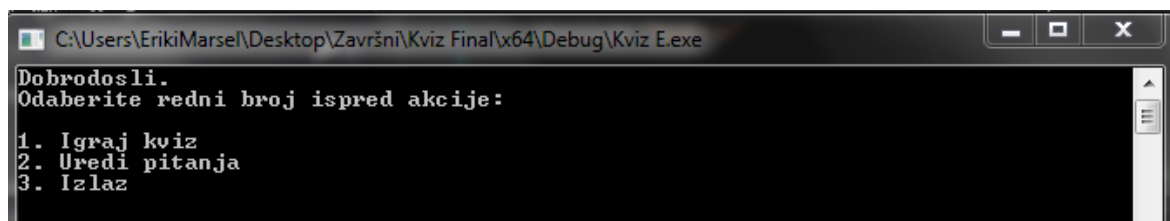
```

Nakon toga definira se funkcija za provjeru unosa i ovdje moj program završava.

```
bool ProvjeraUnosa(int unos, int min, int max)
{
    bool dobarUnos = false;
    for (int i = min; i <= max; i++)
    {
        if (unos == i)
        {
            dobarUnos = true;
            break;
        }
    }
    return dobarUnos;
}
```

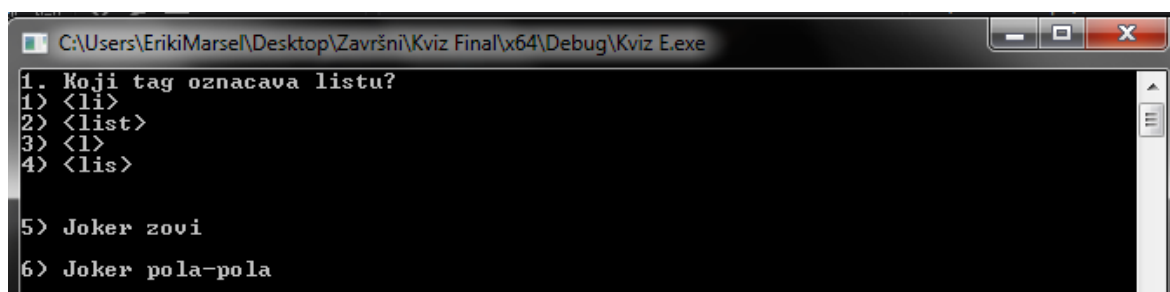
6.RAD KVIZA

Pri pokretanju programa izbacuje se konzolna aplikacija.



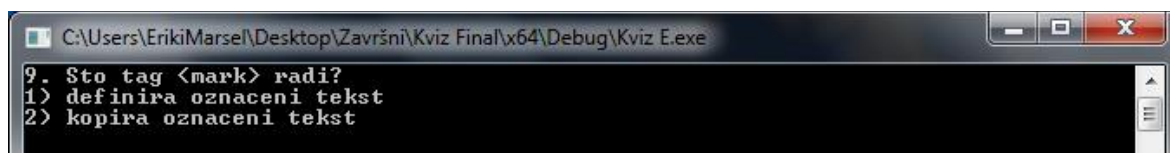
Slika 4. Start izbornik

Kada se pokrene kviz, pitanja imaju četiri ponuđena odgovora i mogućnosti korištenja jokera zovi i jokera pola-pola.

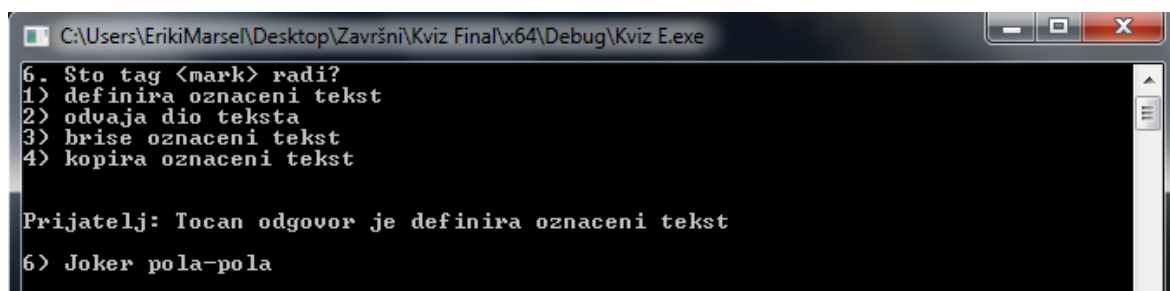


Slika 5. Prvo pitanje

Ovdje je primjer kada se iskoristi joker pola-pola pa zatim primjer za joker zovi.

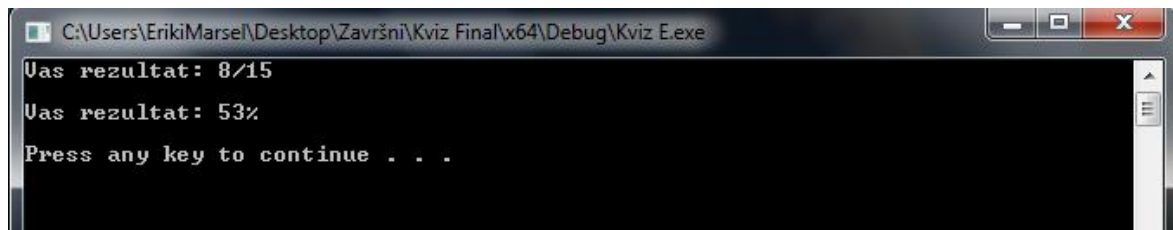


Slika 6. Joker pola-pola



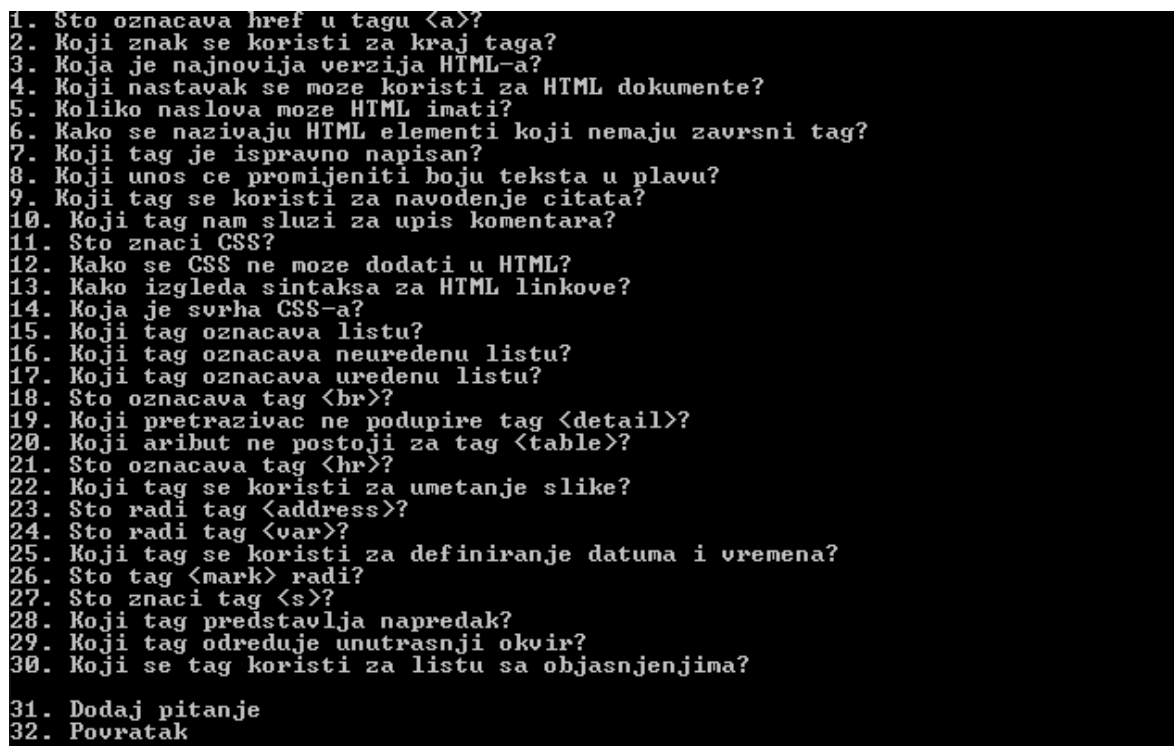
Slika 7. Joker zovi

Na kraju se ispisuje rezultat u bodovima i postotcima.



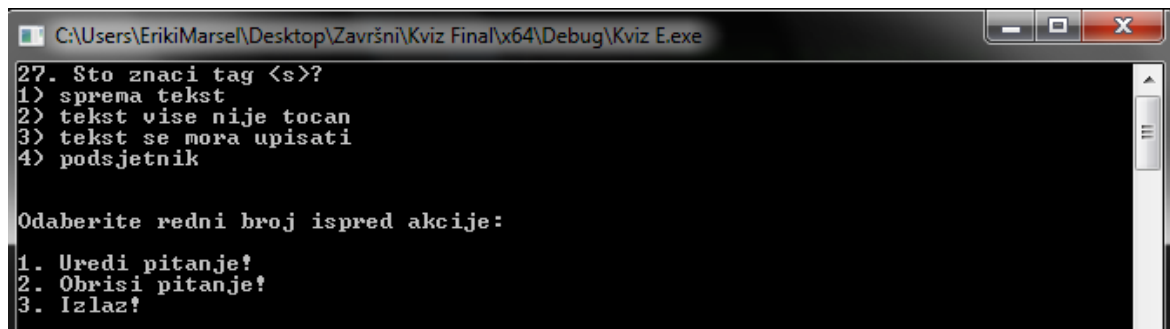
Slika 8. Rezultat

Kada se na izborniku izabere uređivanje pitanja, pojavljuje se lista svih pitanja i opcije za dodavanje pitanja ili povratak.



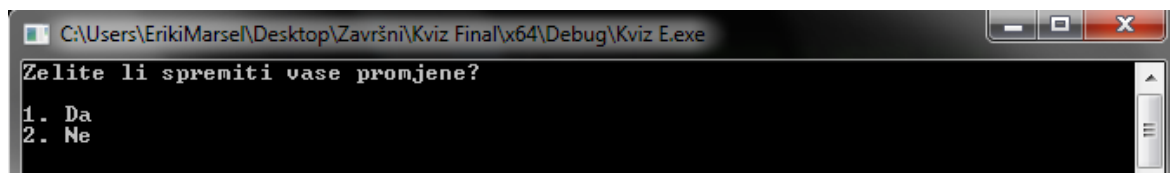
Slika 9. Uređivanje pitanja

Kada je jedno pitanje odabrano otvara se pitanje i daje se opcije za uređivanje pitanja, brisanje pitanja ili povratak na izbornik pitanja.



Slika 10. Uređivano pitanje

Kada se završi s uređivanjem i odabere se povratak, postoji opcija za spremanje unesenih promjena.



Slika 11. Spremanje promjena

7. ZAKLJUČAK

C++ jedan je od kompliciranijih programskih jezika i zato je ovaj kviz bilo kompliciranije napraviti nego što bi možda bilo u drugim jezicima. Kviz sam po sebi zahtjeva mnogo razmišljanja i logičkog zaključivanja. Za nekoga tko se planira baviti programiranjem preporučio bih da krene od nekog jednostavnijeg jezika te da se zatim prebaci na C++ ili da ostane na lakšem jeziku ako je moguće.

8. POPIS LITERATURE

1. Tatjana, Stranjak i Vesna, Tomić. C jezik. Zagreb, Školska knjiga, 2007.
2. Julijan, Šribar i Boris, Motik. Demistificirani C++. Zagreb, Element, 2010.
3. Stack Overflow, <https://stackoverflow.com> (20. veljača 2017.)
4. cplusplus.com – The C++ Resources Network, <http://www.cplusplus.com> (12. siječanj 2017.)
5. C++ Tutorial, <https://www.tutorialspoint.com/cplusplus/> (12. siječanj 2017.)
6. C++ - Wikipedia, [wikihttps://en.wikipedia.org/wiki/C%2B%2B](https://en.wikipedia.org/wiki/C%2B%2B) (12. siječanj 201.)
7. renaissanceincroatia ,<https://www.youtube.com/user/renaissanceincroatia> (12. siječanj 201.)
8. Nauči programirati,
<https://www.youtube.com/channel/UCfHfy20CJt6WZtQRtT47Zzw> (12. siječanj 201.)
9. Pluralsight | Unlimited Online Developer, It and Creative Training,
<https://www.pluralsight.com> (31. siječanj 2017)

KONZULTACIJSKI LIST

Ime i prezime učenika: _____, razred: _____

Red. br.	Nadnevak konzultacija	Bilješke o napredovanju	Potpis mentora