PROJEKTNI ZADATAK

Studenti:

1. Ermin Hamzic

Sadržaj

[1 Tema 1](#_Toc84573880)

[2 Izrada i predavanje projekta 1](#_Toc84573881)

[3 Tehnički zahtjevi 1](#_Toc84573882)

[4 Formalni zahtjevi 2](#_Toc84573883)

[5 Izvještaj o projektu 3](#_Toc84573884)

# Tema

Napisati program za upravljanje radom banke.

# Izrada i predavanje projekta

1. Projekat rade grupe po najviše 4 studenta.
2. Rok za predaju projekta je 20.12.2023. godine.
3. Završen CodeBlocks C++17 projekat i Word dokument arhivirati kao ZIP datoteku i poslati na e-mail adresu predmetnog asistenta sa subject-om “PJIP Projekat”. Obavezno obrisati izvršne/objektne datoteke (.exe i .o) iz projekta prije arhiviranja!
4. Projekat se predaje i ocjenjuje prije izlaska na završni ispit.
5. Ukoliko student odluči ne raditi projekat, može izaći na završni ispit.
6. Ukoliko student pošalje projekat nakon termina završnog ispita, mora ponovo izaći na završni ispit kako bi se ostvarili bodovi na projekat.
7. Ispunjenjem tehničkih i formalnih zahtjeva grupa studenata ostvaruje po 20 bodova.
8. Djelimičnim ispunjenjem tehničkih i formalnih zahtjeva asistent odlučuje o broju bodova koji će se dodjeliti grupi za projekat.

# Tehnički zahtjevi

1. Omogućiti operacije unosa, izmjene, pregleda i brisanja: korisnika, radnika, tekućih i štednih računa, kredita i transakcija.
2. Unos novih stavki, te izmjenu postojećih vršiti sa tastature.
3. Željenu radnju (unos, izmjena, pregled, brisanje) korisnik unosi preko tastature na osnovu ponuđenih opcija u terminalu.
4. Za novi unos definirati potrebne kolone i ograničiti unos ukoliko se ne unesu potrebni podaci.
5. Za povezivanje tabela koristit će se ključevi, koji predstavljaju jedinstveni identifikator određene stavke, kao npr:

* radnik/korisnik: tekstualni zapis JMBG od 13 cifara,
* tekući/štedni račun: tekstualni zapis od 16 cifara;

1. Napraviti dokumentaciju projekta u obliku Word dokumenta. Koristiti ovaj dokument kao šablon, te početi sa 5. poglavljem svoj izvještaj o projektu.

# Formalni zahtjevi

1. Programski kôd (*backend*) pisati u C++ programskom jeziku.
2. Koristiti što više elemenata C++17 standarda.
3. Programsko sučelje (*frontend*) bit će terminal (*Command Prompt* ili UNIX terminal).
4. Strogo se pridržavati principa objektno – orijentiranog programiranja.
5. Jasno definirati potrebne klase.
6. Koristiti naslijeđivanje gdje je potrebno.
7. Maksimalno ograničiti vidljivost i pristup članovima klase.
8. Implementirati sve konstruktore i destruktore.
9. Napraviti odgovarajuće verzije konstruktora ovisno o očekivanom unosu.
10. Koristiti STL kontejnere, a gdje to nije moguće koristiti dinamički alociranu memoriju.
11. Smanjiti ponavljanje kôda kroz implementaciju funkcija.
12. Gdje je moguće koristiti metode umjesto funkcija.
13. Deklaracije izdvojiti u .h datoteke, definicije u .cpp datoteke, a napraviti i main.cpp datoteku koja predstavlja glavni program.
14. Paziti na zauzimanje resursa radne memorije.
15. Objekte, klase, članove klase i njene metode imenovati razumljivo.
16. Nastojati na jedinstven način imenovati metode koji obavljaju iste funkcije u više klasa.

# Izvještaj o projektu

## Uvod u projektni zadatak

1. Projektni zadatak je napravljen u programsku jeziku C++. Ovaj projekat su napravila 3 studenta iz druge godine IPI akademije , smjer informacione tehnologije. Projekat se bazira na switch statementu u kojem korisnik može da se “kreće” po programu i da unosi,briše,mijenja,gleda podatke o korisniku i/ili radniku. U ovom projektnom su napravljene dvije klase (Korisnik i Radnik) i niz funkcija vezanih za te dvije klase. Main naše aplikacije ima pet opcije da se izaberu:
2. Korisnik
3. Radnik
4. Transakcija Korsnika
5. Transakcija Radnika
6. Exit

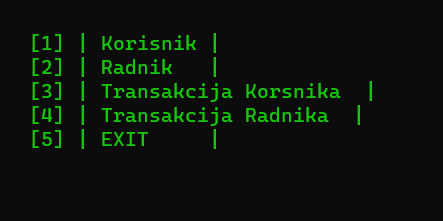
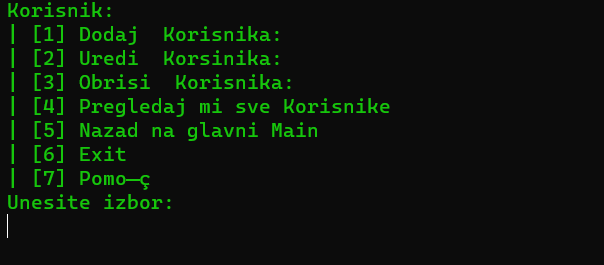


Image 1 - Upravljanje bankom

Klase

Klasa “Korisnik” i klasa “Radnik” su veoma slični. Obje klase koriste getter i setter funkcije.Getter funkcije nam služe da nam vrate vrijednost te klase i postoji svaka za svaki atribut u klase. Setteri su void zbog toga što ne vraćaju ništa , tačnije ne vraćaju nikakvu vrijednost , i oni upisuju vrijednost koju dobiju kad budu pozvani.



Tako da za klasu Book imamo funkcije:

1. Dodaj Korisnika
2. Uredi Korsnika
3. Obriši Korisnika
4. Pregledaj mi sve korsnike
5. Nazad na glavni main
6. Exit
7. Pomoc

Image 2 - Upravljanje Korisnikom

Dok za klasu Radnik imamo sljedeće funkcije:

1. Dodaj Radnika
2. Izbriši Radnika
3. Uredi Radnika
4. Pronadi sve Radnike
5. Nazad na glavni Main
6. Ext
7. Pomoć

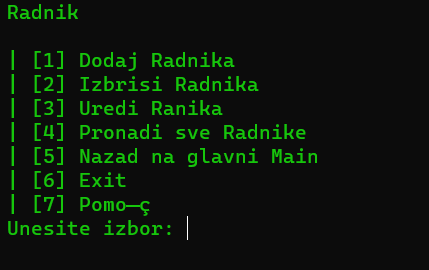


Image 3 - Radnik

Vezano za korsnika, Radnikauzima sljedeće podatke , a to su:

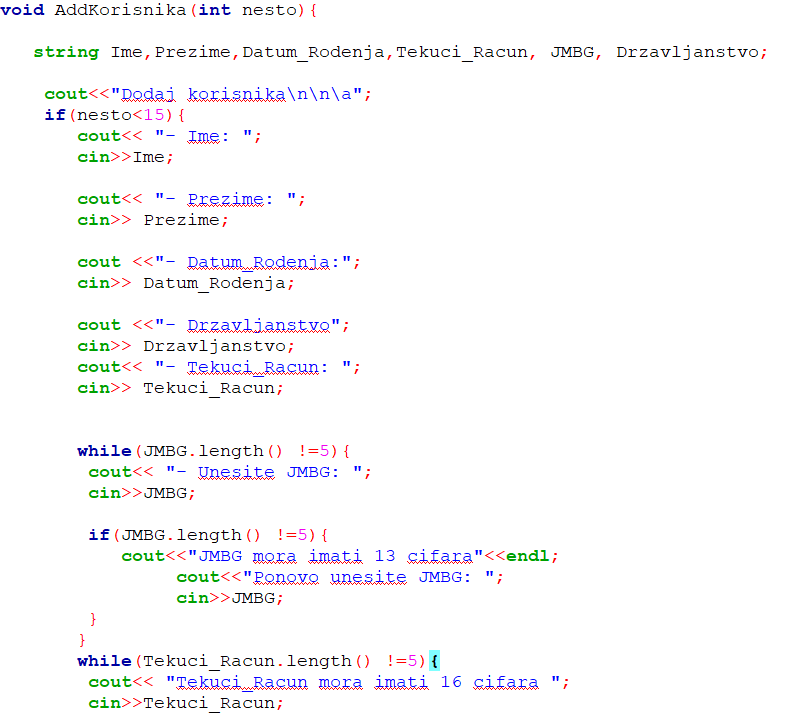
1. JMBG() - u našem programu , JMBGstriktno mora imati 13 cifara
2. Ime
3. Prezime
4. Datum\_rodenja
5. Tekuci\_Racun
6. Drzavljanstvo

Dok Radnik, prima sljedece podatke , a to su:

1. Dodaj Radnika
2. Izbrisi Radnika
3. Uredi Radnika
4. Pronadi sve Radnike
5. Nazad na glavni Main
6. Exit
7. Pomoć

## Funkcije

1. **AddKorsnik funkcija** - to je funkcija pomoću koje dodajemo Korsnika.Ona prima neke podatke , kao što su Ime,Prezime,Datum\_rodenja,Tekući\_Račun,JMBG,Drzavljanstvo  
   .Ograničeno je da se može maksimalno dodati 10 Korsnika.



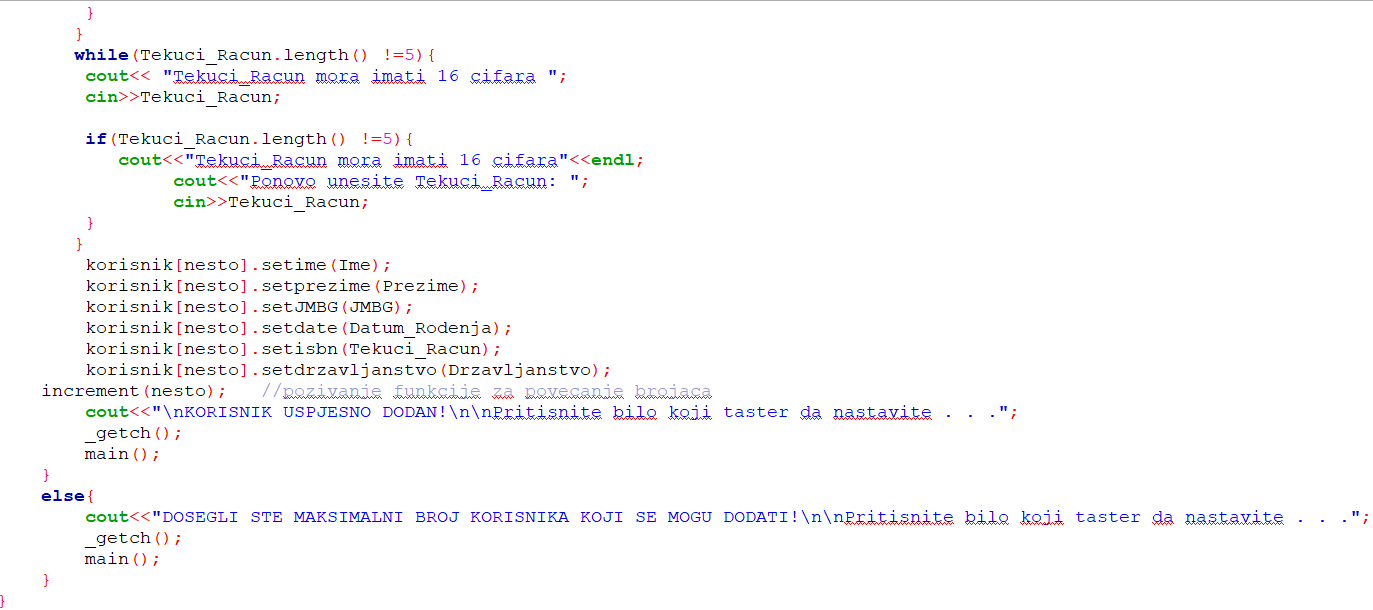


Image 4 - AddKorsnikfunkcija

1. **ObrisiKorisnika funkcija** - to je funkcija pomoću koje brišemo Korsnik. Prvo prolazi kroz for petlju, zatim se ispituje uslov , ako je brojač jednak 0 , tada će se ispisati poruka “NEMA KORSNIKAZA BRISANJE” . Ako ima neka Korsnik, tada program traži od Vas da unesete JMBG Korsnika, ako unesete ispravan JMBG tada Vas program pita da li želite izbrisati Korsnika, broj 1 briše Korsinika, broj 2 ne briše Kornsika. Ako pritisnete broj 1 izbrisat će se Korsnika koju ste željeli izbrisati.

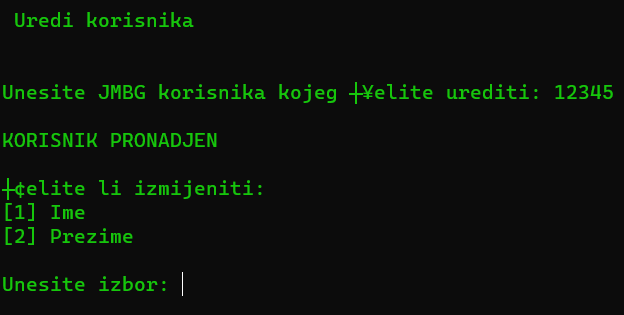


Image 5 - **Deletebook** funkcija

Sljedeca funkcija je **EditKorisnika.** To je funkcija koja mijenja neke podatke o određenoj Korsnik. Prvo prolazi kroz for petlju , i ispituje uslov , ako nema dodanih Korsnika, tada vam ispisuje poruku da nema dodanih Korsnikai vraća vas na main. Ako ima dodanih Korsnik, tada trazi od Vas da unesete JMBG, ako unesete identican JMBG od KOrisnik koja je dodana , tada Vas pita da li zelite da uredite Korsnika, ako pritisnete 1 , tada imate nove opcije za uredjivanje Korsnika, tačnije da ponovo unesete sve podatke vezano za Korsnika . Nakon toga Vam ispisuje poruku da ste uspješno uredili tu Korsnika, A te su opcije Zelite li izmjeniti 1 ime, 2 prezime.

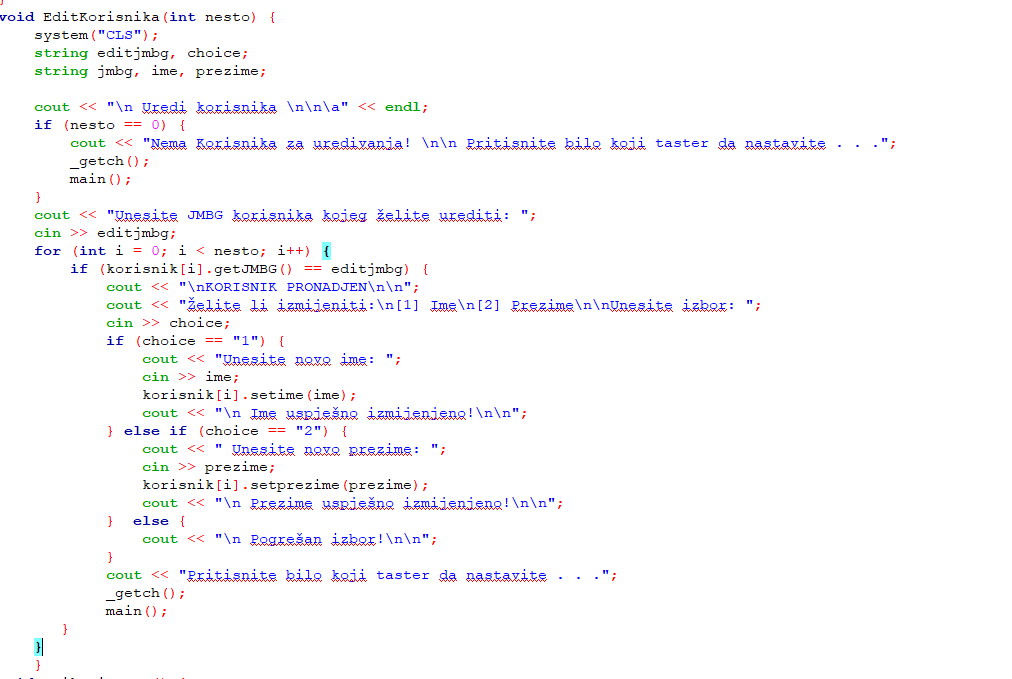
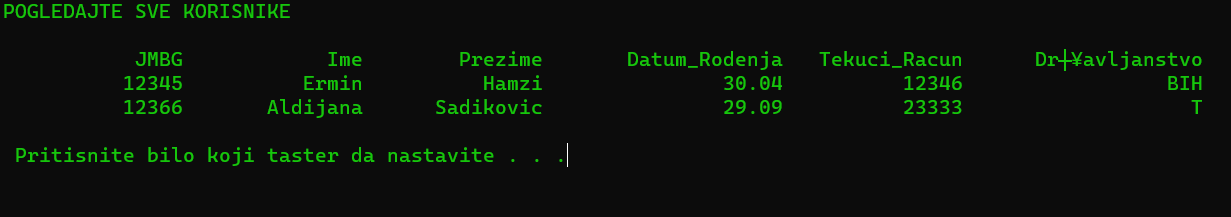


Image 6 - EditKorsnika funkcija

1. **Sljedeća funkcija je ViewPogledajKosinika**. To je funkcija gdje će Vam ispisati Korsnike koje ste prethodno unijeli.Ponovo prolazi kroz for petlju , i ima uslov(if) ako je brojač jednak nuli , ispisuje se poruka “Trenutn nema dodanih Korsnici”. A ako imaju dodane Korsnici tada se ispisuju sve Korsnike Vama na ekran sa svim podacima.



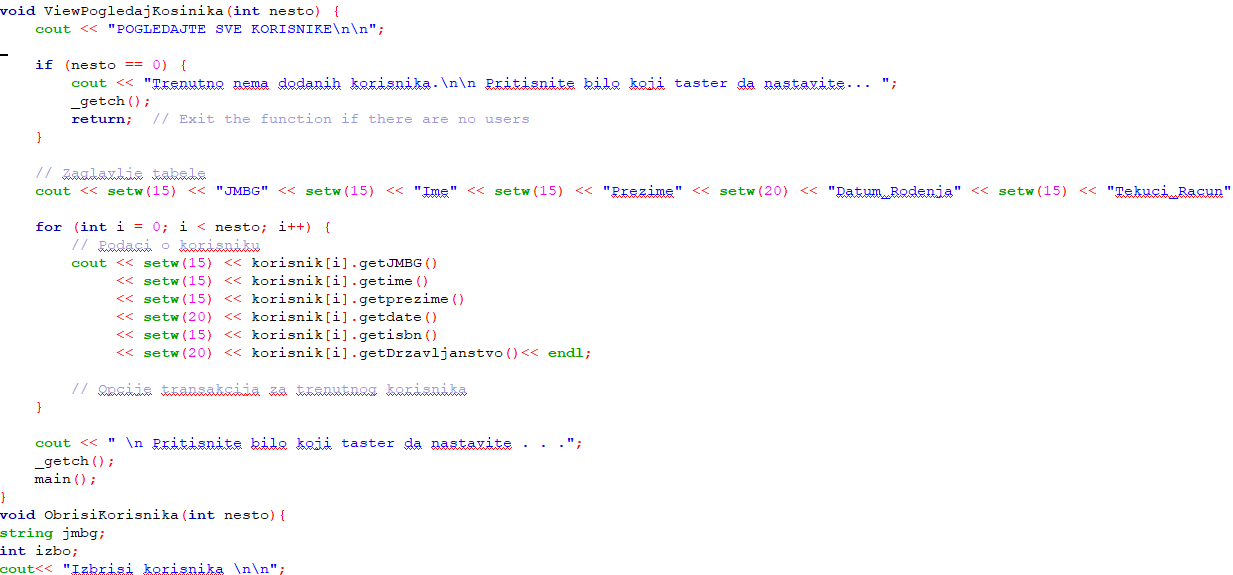
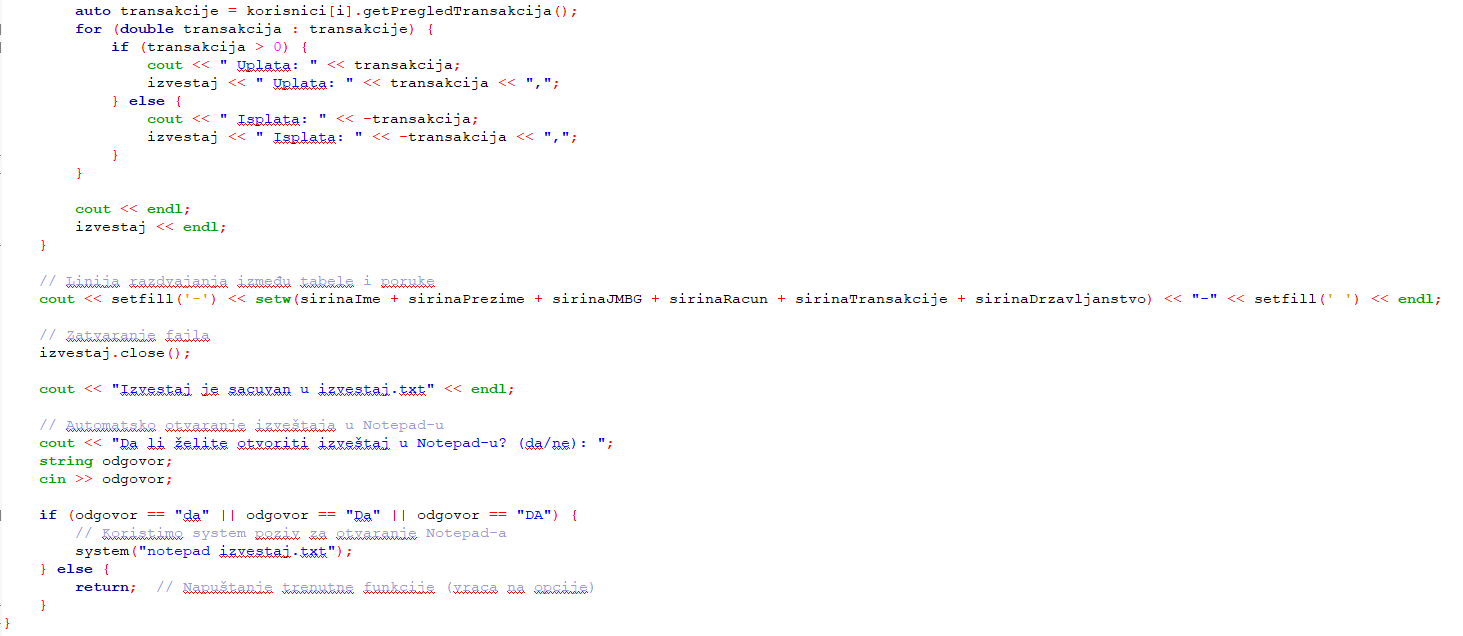


Image 7 - ViewPogledajKosinika Funkcija

Ovo su bile funkcije vezane za Korsnik.

1. **Sljedeća funkcija je pregledTabeleTransakcija** to je funkcija:  
   Otvaranje Fajla: Funkcija otvara fajl pod nazivom "izvestaj.txt" u modu pisanja koristeći ofstream.Postavljanje Širina Tabele: Postavlja širine za svaki dio tabele, od imena i prezimena do transakcija, kako bi izgled tabele bio uredan.Prikaz Zaglavlja Tabele: Ispisuje zaglavlje tabele na konzoli sa nazivima kolona. Linija Razdvajanja: Dodaje liniju razdvajanja između zaglavlja i podataka radi bolje čitljivosti. Prikaz i Upis Podataka: Prolazi kroz sve korisnike i za svakog prikazuje informacije (ime, prezime, JMBG, račun, državljanstvo) na konzoli i upisuje ih u fajl "izvestaj.txt". Također, prikazuje transakcije korisnika, a upisuje ih u fajl zajedno sa vrstom transakcije (uplata/isplata). Linija Razdvajanja između Tabele i Poruke: Dodaje još jednu liniju razdvajanja između tabele i poruke koja će biti ispisana na konzoli. Zatvaranje Fajla: Zatvara fajl "izvestaj.txt".





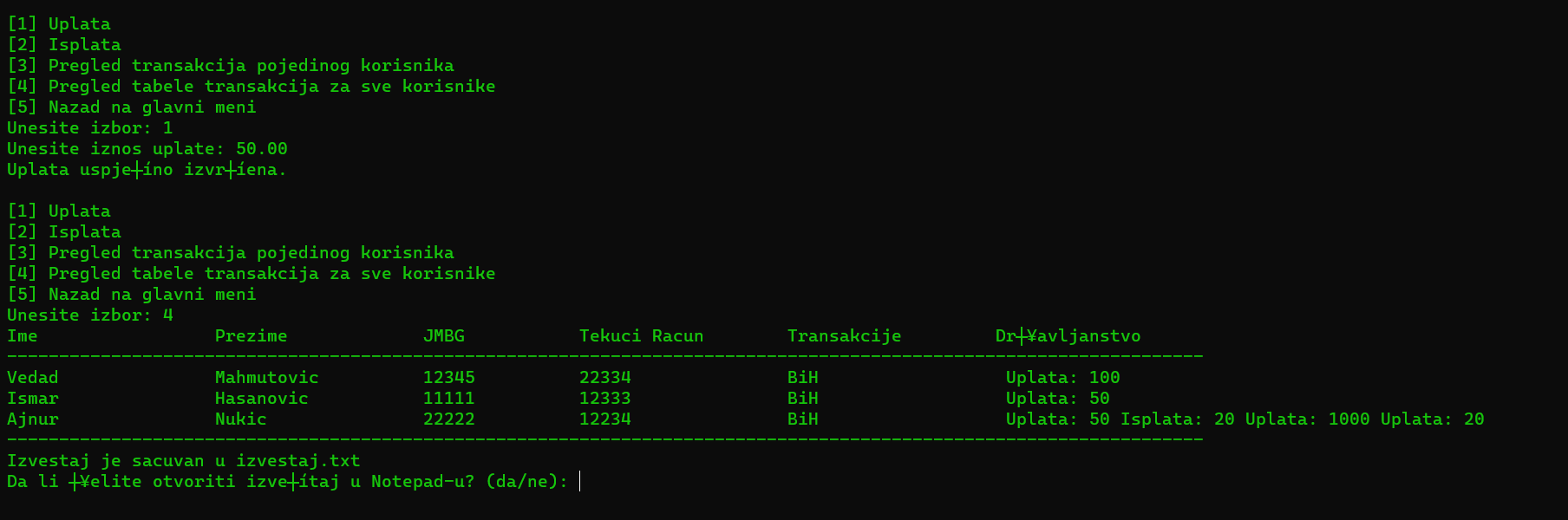
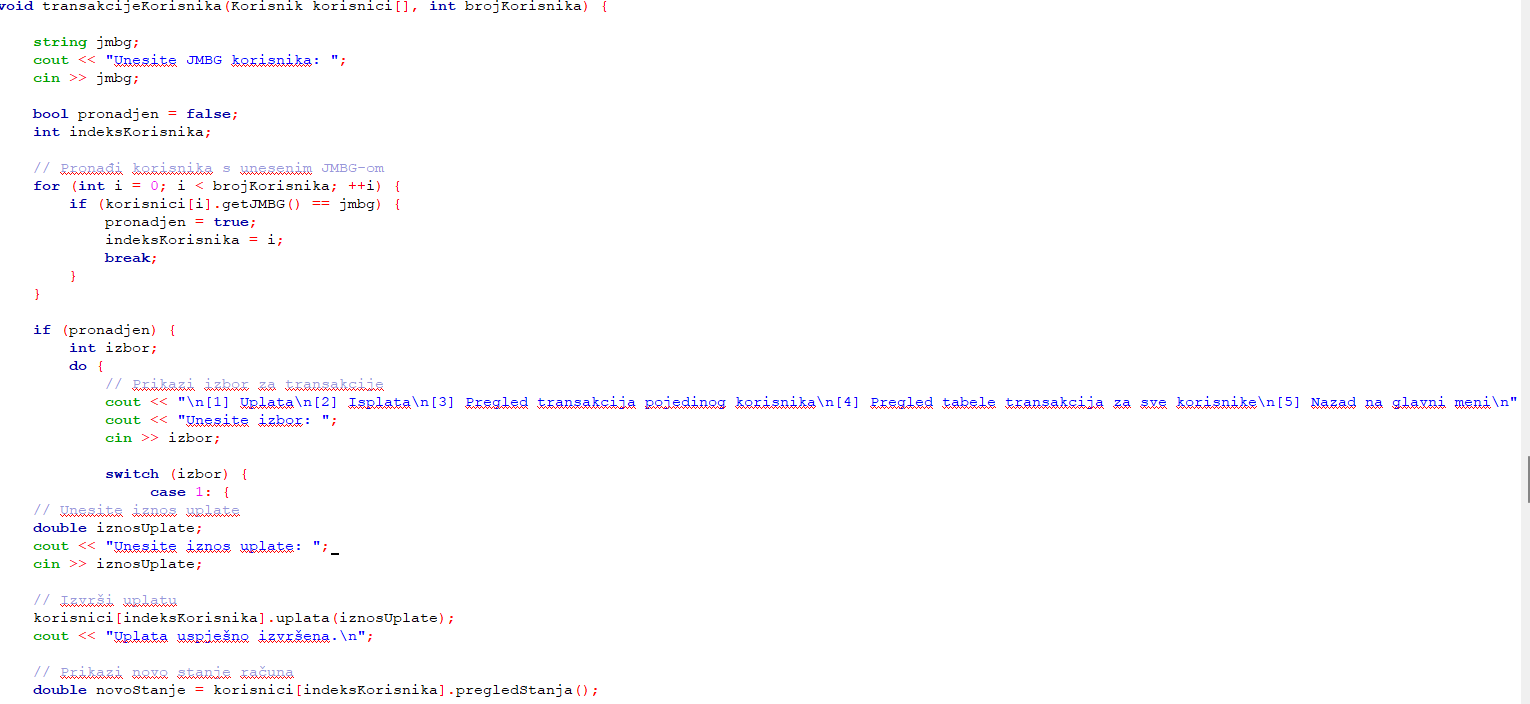


Image 8 - pregledTabeleTransakcijaFunkcija

5. **transakcijeKorisnika** u C++-u omogućava korisnicima upravljanje transakcijama za određenog korisnika. Evo kratak opis:  
Unos JMBG-a: Korisnik unosi JMBG korisnika čije transakcije želi upravljati.Pretraga Korisnika: Funkcija pretražuje niz korisnika kako bi pronašla korisnika s unesenim JMBG-om. Ako pronađe korisnika, postavlja zastavicu pronadjen na true i bilježi indeks tog korisnika.Izbor Transakcije: Ako je korisnik pronađen, funkcija ulazi u petlju koja prikazuje izbornik za transakcije. Korisnik može izabrati uplatu, isplatu, pregled transakcija za pojedinog korisnika, pregled tabele transakcija za sve korisnike ili povratak na glavni meni. Izvršavanje Transakcija: Ovisno o korisnikovom izboru, unosu se iznosi uplate/isplate, izvršavaju se odgovarajuće transakcije na korisnikovom računu, a zatim se prikazuje novo stanje računa. Prikaz Transakcija Korisnika: Ako korisnik odabere pregled transakcija pojedinog korisnika, funkcija prikazuje sve transakcije tog korisnika. Prikaz Tabele Transakcija za Sve Korisnike: Ako korisnik odabere pregled tabele transakcija za sve korisnike, funkcija poziva funkciju pregledTabeleTransakcija koja prikazuje sve transakcije svih korisnika. Povratak na Glavni Meni: Ako korisnik odabere opciju za povratak na glavni meni, funkcija poziva glavnu funkciju main.



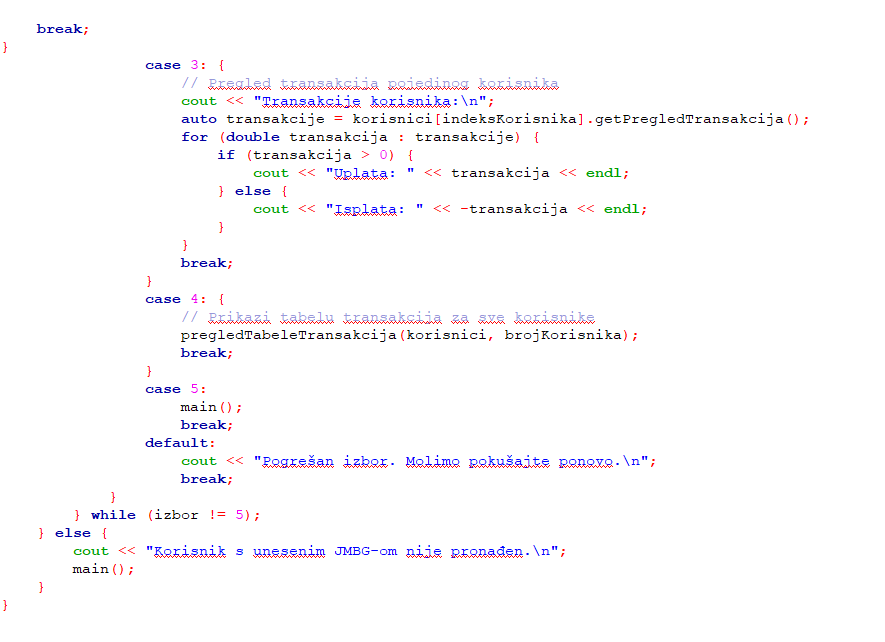


Image 8 - transakcijeKorisnika Funkcija

Ova funkcija `prikaziPomoc` u C++-u ima zadatak prikazivanja pomoći, odnosno ispisivanja informacija o različitim opcijama koje korisnik može odabrati u programu. Evo kratkog objašnjenja:  
1.system Clear Screen (`system("CLS")`): Poziva sistemsku funkciju za brisanje ekrana (čišćenje konzole), što ima za cilj da osvježi prikaz i poboljša čitljivost.  
2. Ispis Naslova Pomoći: Ispisuje naslov pomoći, označavajući da slijede informacije o opcijama.  
3. Ispis Informacija o Opcijama:

-Opcija [1]: Dodaj mi korisnika - omogućava dodavanje novog korisnika.

-Opcija [2]: Uredi mi korisnika - omogućava uređivanje podataka postojećeg korisnika.

-Opcija [3]: Obriši mi korisnika - omogućava brisanje korisnika iz baze podataka.

-Opcija [4]: Pregledaj mi sve korisnike - prikazuje sve korisnike u sustavu.

-Opcija [5]: Nazad na glavni meni - vraća se na glavni izbornik.

-Opcija [6]: Izlaz - zatvara program.

-Opcija [7]: Pretraži korisnike - omogućava pretraživanje korisnika po određenim kriterijima.

-Opcija [8]: Pomoć - prikazuje ovu pomoć.

4. Završetak Naslova Pomoći:\*\* Ispisuje odvojnicu (liniju) nakon popisa opcija.

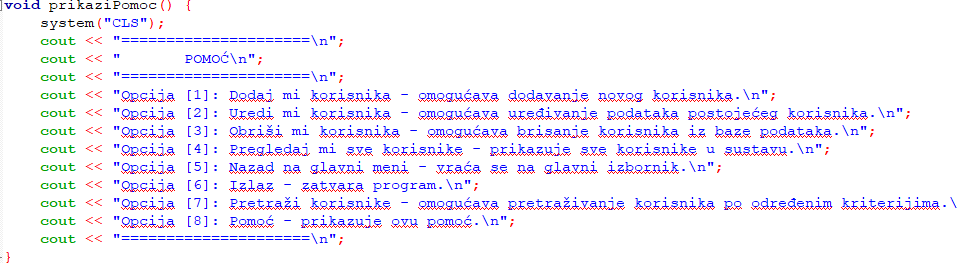
Ova funkcija ima ulogu pružanja jasnih uputa i informacija korisniku o svakoj dostupnoj opciji u programu.

Image 9 - prikaziPomoc Funkcija

Ostale funkcije su vezane za klasu “Radnika”.

Prva funkcija je **AddRadnika**. To je funkcija koja dodaje radnika i traži se da se unese ime , prezime i jmbg korisnika. JMBG je striktno ograničen na 13 cifara, ako se unese više ili manje od 13 tada ispisuje grešku “Pogrešan unos” i ponovo Vam daje mogućnost da unesete JMBG.

SLIKA

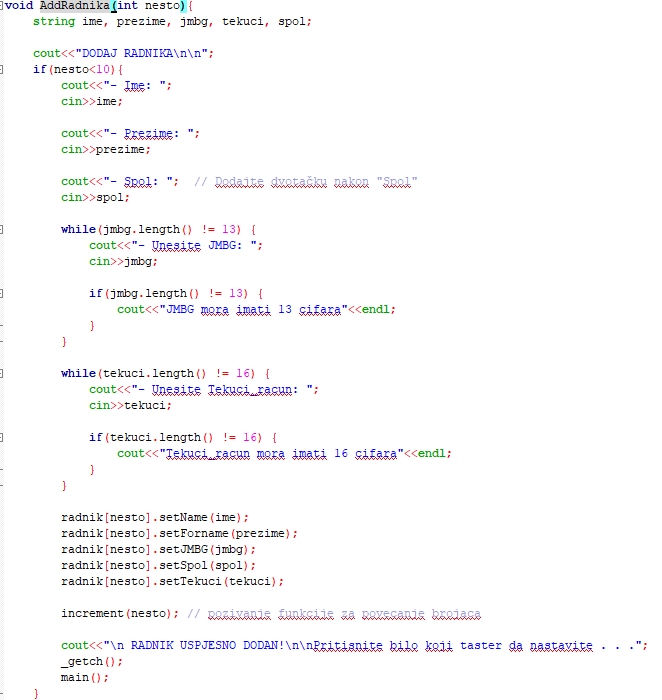


Image 10 - **AddRadnika**Funkcija

Sljedeća funkcija je **EditUser**. To je ista funkcija kao i editBook prethodno navedena. Traži se od korisnika JMBG , ako se unese tačan JMBG , tada imate mogućnost da uredite korisnika.

Druga funkcija **EditRadnika** To je funkcija koja mijenja neke podatke o određenoj Radniku. Prvo prolazi kroz for petlju , i ispituje uslov , ako nema dodanih radnika, tada vam ispisuje poruku da nema dodanih radnika vraća Vas na main. Ako ima dodanih radnik, tada trazi od Vas da unesete JMBG, ako unesete identican JMBG od Radnika koja je dodana , tada Vas pita da li zelite da uredite Korsnika, ako pritisnete 1 , tada imate nove opcije za uredjivanje adnika , tačnije da ponovo unesete sve podatke vezano za radnika. Nakon toga Vam ispisuje poruku da ste uspješno uredili tu radnika.



Image 11 - **EditRadnika** Funkcija

Treca funkcija **ViewRadnika**  To je funkcija gdje će Vam ispisati Korsnike koje ste prethodno unijeli.Ponovo prolazi kroz for petlju , i ima uslov(if) ako je brojač jednak nuli , ispisuje se poruka “Trenutn nema dodanih Korsnici”. A ako imaju dodane Korsnici tada se ispisuju sve Korsnike Vama na ekran sa svim podacima.

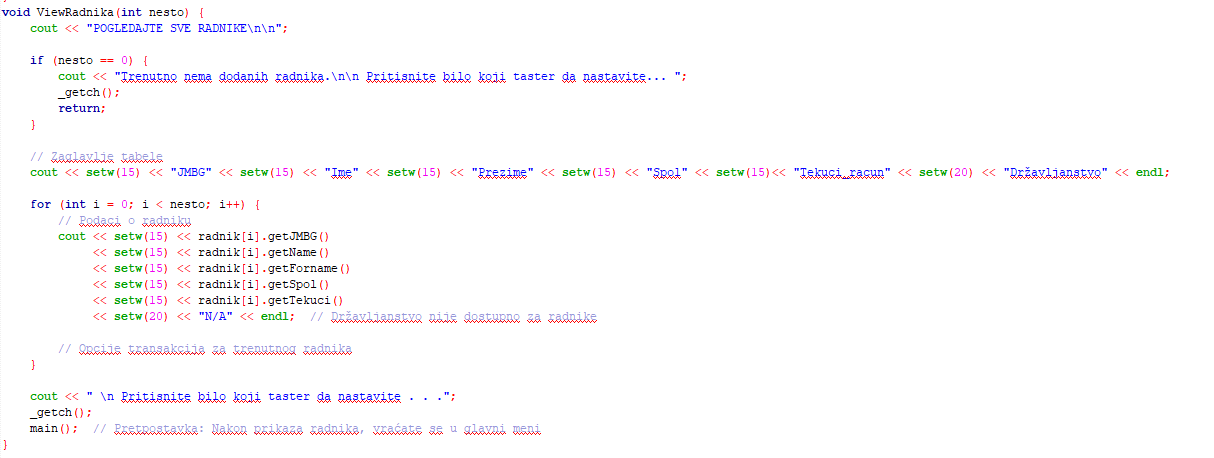


Image 11 - **ViewRadnika** Funkcija

Cetvrta funkcija **DeleteRadnika** radi na prinicipu kao **ObrisiKorisnika funkcija**.



Image 12 - **DeleteRadnika f**unkcija

Ova funkcija, nazvana "quit", ima zadatak ispisivanja poruke "Hvala Vam što ste koristili naš program" na standardni izlaz, uz dodatak zvučnog signala (\a), što može privući pažnju korisnika. Nakon toga, funkcija koristi "\_exit(1)" kako bi prekinula izvršavanje programa s kodom završetka 1. Ovo može signalizirati izlazak iz programa s određenim statusom ili nekim drugim značenjem koje je programer odredio.

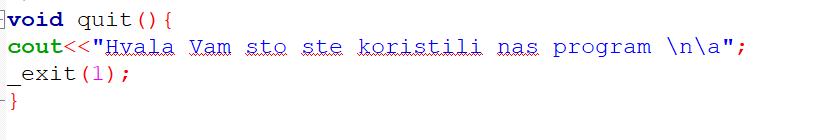


Image 13 - **Quit** funkcija

Ova funkcija, nazvana "ProvjeriLogin", ima zadaću provjeriti korisničko ime i lozinku prilikom prijave u program. Evo kratkog opisa rada funkcije:

1. Koristi statičku varijablu `loginCompleted` koja označava je li prijava već izvršena. Ova varijabla omogućuje da se prijava provjeri samo jednom tijekom izvršavanja programa.

2. Ako prijava nije već izvršena (kada je `loginCompleted` postavljena na `false`), ispisuje poruku "Dobro dosli u Banku" s pauzom između pojedinih znakova kako bi stvorila efekt postupnog pojavljivanja.

3. Zatim traži od korisnika unos korisničkog imena i lozinke.

4. Uspoređuje uneseno korisničko ime i lozinku s predefiniranim podacima (vjerovatno dostupnim u globalnom kontekstu pod nazivom `adminPodaci`). Ako su podaci ispravni, označava da je prijava uspješna (postavlja `loginCompleted` na `true`) i vraća `true`.

5. Ako podaci nisu ispravni, ispisuje poruku o pogrešnom korisničkom imenu ili lozinci te vraća `false`.

6. Ako je prijava već izvršena (kada je `loginCompleted` postavljena na `true`), odmah vraća `true` jer nema potrebe za ponovnom provjerom prijave.

Ova funkcija čini dio sustava za prijavu u bankovni program, čime se osigurava da se prijava izvršava samo jednom tijekom izvođenja programa.

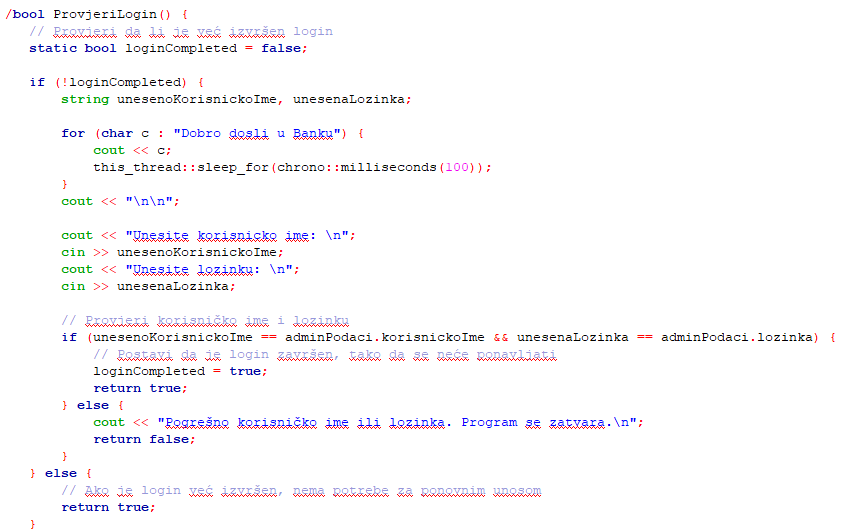


Image 13 - **ProvjeriLoginf**

**Zakljucak**

Unutar ove aplikacije vidljiva je potreba za bazom podataka. Svi podaci koji prođu kroz ovu aplikaciju bivaju izgubljeni momenta kada se proces zaustavi. Što se tiče dosadašnjih funkcionalnosti i logičke postavke koda, ima mnogo prostora za poboljšanja. Neki dijelovi postojećeg koda se mogu presložiti logički i semantički kako bi kod imao više smisla ili na efikasniji način obavljao svoju funkciju. Korisničko sučelje je ograničeno što se tiče formatiranja jer je sučelje tekstualnog oblika. Postoje načini da se C++ kod iskoristi za građenje korisničkog sučelja poput Qt framework-a, te se u budućnosti može napraviti funckionalno grafičko korisničko sučelje na bazi dosadašnjeg backend koda. Eventualne sitne korekcije su neizbježne.

Funkcionalnosti rade kako bi trebale, te za vrijeme odbrane ovog projekta postoji mogućnost isprobavanja svih funkcionalnosti i praćenja uticaja njihovog izvršavanja unutar aplikacije.