UNIVERZITET U NOVOM SADU
PRIRODNO-MATEMATIČKI FAKULTET
DEPARTMAN ZA MATEMATIKU I INFORMATIKU

Marko Baba, 42/19

ASCII

(prodavnica video igara i konzola)

- seminarski rad iz predmeta Skript jezici-

Novi Sad, januar 2020.

1. Uvod 3
2. Opis programa 5
2.1 Glavni meni 5
2.2 Funkcionalnosti namenjene kupcu 9
2.3 Funkcionalnosti namenjene administratoru (prodavcu) 14
2.4 Funkcionalnosti namenjene svim korisnicima 18
3. Zaključak 20
4. Literatura 21

1. Uvod

ASCII prodavnica video igara i konzola je jednostavan program namenjen za prodavce, kao i za kupce. Program je napisan u programskom jeziku Python 3.8, a za skladištenje podataka korišćen je MYSQL Workbench.

Program je urađen bez grafičkog interfejsa, pa korisnik kroz program prolazi na jednostavan način, tako što svakoj funkcionalnosti pristupa odabirom, odnosno unosom odgovarajućeg broja za prelaz u odgovarajuću funkcionalnost.

ASCII prodavnica koristi sledeće tabele u MYSQL Workbench-u:

- Prodavac (Listing 1): koji je jedan od korisnika i ujedno i administrator programa. On je zadužen za kontrolu količine, cene, kategorija i popusta svih proizvoda u prodavnici. Jedinstveno je označen svojim id brojem (*idProdavca*), a pored toga ima svoje *ime*, *prezime*, korisničko ime (*user*) i lozinku (*passwd*) za korisnički nalog i *platu*.

```
mycursor.execute("CREATE TABLE Prodavac (idProdavca int PRIMARY KEY AUTO_INCREMENT,
ime VARCHAR(50), prezime VARCHAR(50), user VARCHAR(50), passwd VARCHAR(50), plata
int NOT NULL)")
```

Listing 1 Izrada tabele *Prodavac* u bazi podataka

- Kupac (Listing 2): koji predstavlja drugu vrstu korisnika ovog programa. On ima ograničen pristup prodavnici i njegove mogućnosti su pretraga i kupovina proizvoda. Jedinstveno je označen svojim id brojem (*idKupca*), a pored toga ima svoje ime (*imeKupca*), prezime (*prezimeKupca*), korisničko ime (*userKupca*), lozinku (*passwdKupca*) i *kupon* koji dobija nakon kupovine proizvoda koji prelaze određenu sumu novca

```
mycursor.execute("CREATE TABLE Kupac (idKupca int PRIMARY KEY AUTO_INCREMENT,
imeKupca VARCHAR(50), prezimeKupca VARCHAR(50), userKupca VARCHAR(50), passwdKupca
VARCHAR(255), kupon int NOT NULL)")
```

Listing 2 Izrada tabele *Kupac* u bazi podataka

- Proizvod (Listing 3): je artikal koji se prodaje u prodavnici. Tu spadaju razne vrste video igara i konzola. Jedinstveno je označen svojim id brojem (*idProizvoda*), a pored toga ima i svoj *naziv*, *cenu*, *popust* koji unosi prodavac i količinu prodatih komada za taj konkretan proizvod (*brojProdatih*)

```
mycursor.execute("CREATE TABLE Proizvod (idProizvoda int PRIMARY KEY
AUTO_INCREMENT, naziv VARCHAR(50), cena int NOT NULL, popust int NOT NULL,
brojProdatih int NOT NULL")
```

Listing 3 Izrada tabele *Proizvod* u bazi podataka

- Kategorija (Listing 4): je vrsta grupisanja proizvoda na celine. Dve osnovne kategorije su video igra i konzola, međutim one ne postoje zasebno, jer svaka video igra i konzola imaju podkategoriju, kao što su *PS4 Igra*, ili *Xbox One Konzola*. Kategorija je jedinstveno označena svojim id brojem (*idKategorije*), a pored toga ima i svoj naziv (*nazivKategorije*)

```
mycursor.execute("CREATE TABLE Kategorija (idKategorije int PRIMARY KEY
AUTO_INCREMENT, nazivKategorije VARCHAR(50))")
```

Listing 4 Izrada tabele *Kategorija* u bazi podataka

- ProizvodKategorija (Listing 5): je mehanizam koji se koristi da se proizvod i kategorija povežu tako da se proizvod svrsta u posebnu kategoriju i time vrši evidencija o tome koliko je u prodavnici dostupno određenog proizvoda u određenoj kategoriji. Kao svoje kolone koristi *idProizvoda* i *idKategorije*, kao i dostpunost proizvoda u odgovarajućoj kategoriji na lageru (*na_lageru*)

```
mycursor.execute("CREATE TABLE ProizvodKategorija (idProizvoda int, idKategorije
int, na_lageru int)")
```

Listing 5 Izrada tabele *ProizvodKategorija* u bazi podataka

2. Opis programa

Aplikacija se sastoji iz četiri modula: main.py, korisnici.py (sadrži funkcije namenjene svim korisnicima aplikacije), kupac.py (funkcije namenjene isključivo kupcima), prodavac.py (funkcije namenjene isključivo prodavcima, odnosno administratorima aplikacije).

2.1 Glavni meni

Aplikacija se pokreće iz modula main.py, koji je ustvari i glavni meni. Odmah po pokretanju glavnog menija, pokreće se funkcija iz modula korisnici.py, korisnici.grafikon(), koja prikazuje pet najprodavanijih proizvoda u prodavnici.

Korisnik, zatim u glavnom meniju, u beskonačnoj while petlji unosi broj 1 za pristup prodavnici kao kupac, ili broj 2 za pristup prodavnici kao administrator, odnosno radnik/prodavac (Listing 6).

```
1.  while 1:
2.      print("Unesite 1 za pristup prodavnici kao kupac...")
3.      print("Unesite 2 za administrativni pristup...")
4.      korisnik = eval(input("Unesite vas izbor: "))
```

Listing 6 Odabir načina pristupa prodavnici

Korisnik dobija upozorenje da je njegov unos bio pogrešan ukoliko ne unese neki od ova dva broja (Listing 7).

```
1.  else:
2.          print("Pogresno ste uneli broj! Pokusajte ponovo.")
3.          print("\n")
4.          continue
```

Listing 7 Upozorenje za pogrešan unos

Ukoliko se korisnik odluči za pristup prodavnici kao kupac, odmah dobija obaveštenje o aktuelnoj specijalnoj akciji u prodavnici i program ulazi u beskonačnu while petlju, odnosno u *Login System Kupca* (Listing 8). Kupac ima opciju da unese broj 1 da napravi novi korisnički nalog, nakon čega se pokreće funkcija *kupac.signupKupca()*, zatim da unese broj 2 da se uloguje u postojeći korisnički nalog, čime se pokreće funkcija *kupac.loginKupca()*, kao i da unese broj 3 da izađe iz menija *Login System Kupca* i nastavi sa kupovinom bez naloga, čime gubi pravo na specijalne kupone. Ukoliko korisnik unese pogrešan karakter, dobija obaveštenje o tome i dobija priliku da pokuša ponovo.

```
1.  if (korisnik == 1):
2.          print("Uz specijalnu akciju, za svakih potrosenih 7000 dinara dobijate popust u
    iznosu od 500 dinara na sledecu kupovinu! SAMO UKOLIKO STE ULOGOVANI U KORISNICKI
    NALOG!")
3.          print("\n")
4.          while 1:
5.              print("********* Login System Kupca *********")
6.              print("Unesite 1 da napravite nalog...")
7.              print("Unesite 2 da se ulogujete u postojeci nalog...")
8.              print("Unesite 3 da izadjete iz login sistema i nastavite sa kupovinom bez
    korisnickog naloga...")
9.              login = eval(input("Unesite vas izbor: "))
```

```
10.             print("\n")
11.             if login == 1:
12.                 kupac.signupKupca()
13.                 continue
14.             elif login == 2:
15.                 username = kupac.loginKupca()
16.             elif login == 3:
17.                 break
18.             else:
19.                 print("Pogresno ste uneli broj! Pokusajte ponovo.")
20.                 print("\n")
21.                 continue
```

Listing 8 *Login System Kupca*

Nakon uspešno napravljenog naloga, korisnik se vraća u *Login System Kupca* kako bi imao prilike da pristupi svom novom nalogu. Odmah nakon pristupa nalogu, kupac ulazi u meni *Pretraga proizvoda* (Listing 9), koji se nalazi u novoj beskonačnoj while petlji. Zatim, korisnik ima opciju da unese 1 kako bi dobio uvid u sve proizvode, kategorije i dostupnost proizvoda u određenim kategorijama (*korisnici.pretragaSvihProizvoda()*). Ukoliko unese broj 2, kupac dobija listu svih kategorija koje postoje u prodavnici, i zatim ima mogućnost da preko identifikacionog broja kategorije pretraži sve proizvode u jednoj ili više kategorija koje odabere unosom brojeva preko tastature (*kupac.pretragaProizvodaPoKategorijama()*). Ukoliko se kupac ipak odluči za broj 3, on dobija mogućnost da sam ukuca tačno ime proizvoda koji želi da pretraži, i ako takav proizvod postoji u prodavnici, biće izlistan taj proizvod sa svojom cenom, lista svih kategorija i lista kategorija u kojima je taj proizvod dostupan, kao i dostupna količina na lageru (*kupac.pretragaSpecificnogProizvoda()*).

```
1.  while 1:
2.              print("********* Pretraga proizvoda *********")
3.              print("Unesite 1 za prikaz svih proizvoda...")
4.              print("Unesite 2 za pretragu proizvoda po kategorijama...")
5.              print("Unesite 3 za pretragu specificnog proizvoda...")
6.              pretraga = eval(input("Unesite vas izbor: "))
7.              print("\n")
8.              if pretraga == 1:
9.                  korisnici.pretragaSvihProizvoda()
10.             elif pretraga == 2:
11.                 kupac.pretragaProizvodaPoKategorijama()
12.             elif pretraga == 3:
13.                 kupac.pretragaSpecificnogProizvoda()
14.             else:
15.                 print("Pogresno ste uneli broj! Pokusajte ponovo.")
16.                 print("\n")
17.                 continue
```

Listing 9 *Pretraga proizvoda*

Šta god da kupac odabere od prethodno navedenih opcija, nakon pretrage će automatski biti prebačen na meni *Kupovina* (Listing 10), gde ponovo preko beskonačne petlje ima mogućnost da bira između dve opcije. Prva je da unese broj 1 i pristupi kupovini proizvoda, čije tačno ime će morati da ukuca, a druga je da unese broj 2 i vrati se nazad na meni *Pretraga proizvoda*.

Kada se kupac odluči za kupovinu proizvoda, proizvod se preko funkcije *kupac.kupovinaProizvoda()* skladišti u korpu, ili odmah štampa na račun, u zavisnosti za šta se korisnik odlučio (više o ovome u sekciji **2.2 Funkcionalnosti namenjene kupcu**). Ukoliko je kupac pristupio svom korisničkom nalogu, sa računa mu se skida onoliko dinara koliko ima na svom kuponu, a *kupon* polje u tabeli *Kupac* u bazi podatak se anulira. Ako kupon ima vrednost 0

račun ostaje isti. Korisnik nakon toga dobija obaveštenje o iznosu računa. Na kraju, ukoliko je račun veći od 7000 dinara, kupac za svakih potrošenih 7000 dobija 500 dinara popusta na sledeću kupovinu. Ukupan iznosa računa biva podeljen sa 7000, a zatim se taj količnik množi sa 500 i dobija se ukupan iznos kupona koji se pamti na korisnikovom nalogu. Od korisnika se takođe traži da unese i adresu na koju će proizvod biti isporučen.

```
1.  while 1:
2.                  print("********* Kupovina proizvoda *********")
3.                  print("Unesite 1 za kupovinu proizvoda...")
4.                  print("Unesite 2 za povratak na pretragu proizvoda...")
5.                  kupovina = eval(input("Unesite vas izbor: "))
6.                  print("\n")
7.                  if kupovina == 1:
8.                      racun = kupac.kupovinaProizvoda()
9.                      if username is not None:
10.                         try:
11.                             Qkupon = "SELECT kupon FROM Kupac WHERE userKupca = %s"
12.                             mycursor.execute(Qkupon, (username,))
13.                             kupon = mycursor.fetchone()
14.                             racun = racun - kupon
15.                             adresa = input("Unesite adresu i broj za isporuku
    proizvoda: ")
16.                             print("Vas racun iznosi", racun, "dinara. Proizvod ce biti
    isporucen na adresu", adresa,".")
17.                             print("\n")
18.                             Qkupon2 = "UPDATE Kupac SET kupon = %s WHERE userKupca =
    %s"
19.                             mycursor.execute(Qkupon2, (0, username))
20.                             db.commit()
21.                             if racun >= 7000:
22.                                 koeficijent = int(racun / 7000)
23.                                 popust = koeficijent * 500
24.                                 Qp = "UPDATE Kupac SET kupon = %s WHERE userKupca = %s"
25.                                 mycursor.execute(Qp, (popust, username))
26.                                 db.commit()
27.                                 print("Uz vasu kupovinu ste dobili", popust, "dinara
    popusta na vasu sledecu kupovinu!")
28.                                 print("\n")
29.                         except:
30.                             continue
31.                     else:
32.                         adresa = input("Unesite adresu i broj za isporuku proizvoda: ")
33.                         print("Vas racun iznosi", racun, "dinara. Proizvod ce biti
    isporucen na adresu", adresa,".")
34.                         print("\n")
35.                 elif kupovina == 2:
36.                     break
37.                 else:
38.                     print("\n")
39.                     print("Pogresno ste uneli broj! Pokusajte ponovo.")
40.                     print("\n")
41.                     continue
```

Listing 10 *Kupovina proizvoda*

Ukoliko se korisnik odluči za administrativni pristup prodavnici, od njega se odmah traži unos administrativne šifre kako bi uopšte mogao da nastavi dalje (Listing 11). Zbog jednostavnosti administrativna šifra je *admin*, a ukoliko korisnik pogrešno unese šifru, dobiće upozorenje.

```
1.  elif (korisnik == 2):
2.          pristup = input("Unesite administrativnu sifru: ")
```

```
        print("\n")
        if (pristup == 'admin'):


else:
            print("\n")
            print("Pogresna sifra! Pokusajte ponovo.")
            print("\n")
            continue
```

Listing 11 Administrativni pristup prodavnici

Nakon što korisnik unese šifru, program ulazi u *Login System Prodavca* (Listing 12), koji se nalazi u beskonačnoj while petlji. Od korisnika se traži da unese odgovarajuć broj kako bi napravio korisnički nalog ili pristupio postojećem korisničkom nalogu. Za razliku od kupca, prodavac mora da se uloguje u nalog kako bi pristupio administrativnim funkcionalnostima. Ukoliko korisnik unese bilo koji drugi karakter sem 1 ili 2, vraća se u početni meni. Ukoliko korisnik unese broj 1, pokreće se funkcija *prodavac.signupProdavca()*, nakon čega se ponovo vraća na *Login System Prodavca*, a ukoliko unese broj 2, pokreće se funkcija *prodavac.loginProdavca()* i ukoliko se korisnik uspešno uloguje u svoj nalog, program pamti korisničko ime.

```
while 1:
                print("********* Login System Prodavca *********")
                print("Unesite 1 da napravite novi nalog...")
                print("Unesite 2 da se ulogujete u postojeci nalog...")
                print("Unesite bilo koji karakter da izadjete iz login sistema...")
                ch = input("Unesite vas izbor: ")
                print("\n")
                if ch == '1':
                    prodavac.signupProdavca()
                elif ch == '2':
                    user = prodavac.loginProdavca()


else:
                    break
```

Listing 12 *Login System Prodavca*

Pamćenje korisničkog imena se realizuje preko promenljive koju vraća funkcija *prodavac.loginProdavca()* i program preko te promenljive pristupa meniju *Administrativne funkcije* (Listing 13). U ovom meniju korisnik ima ukupno sedam opcija:

- 1: za prikaz svih proizvoda u prodavnici,
- 2: za unos novog proizvoda,
- 3: za dopunu količine odredjenog proizvoda na lageru,
- 4: za promenu cene postojećeg proizvoda,
- 5: za dodavanje popusta na postojeći proizvod,
- 6: za uklanjanje popusta sa proizvoda i
- 7: za izlaženje iz menija *Administrativne funkcije* i vraćanje na *Login System Prodavca*.

Korisnik dobija i upozorenje ako nije uneo neki od pet ponuđenih karaktera, nakon čega može da pokuša ponovo.

```
if user is not None:
                        while 1:
```

```
3.                              print("********* Administrativne funkcije *********")
4.                              print("Unesite 1 za prikaz svih proizvoda u prodavnici...")
5.                              print("Unesite 2 za unos novog proizvoda...")
6.                              print("Unesite 3 za dopunu kolicine odredjenog proizvoda u
    prodavnici...")
7.                              print("Unesite 4 za promenu cene postojeceg proizvoda...")
8.                              print("Unesite 5 za dodavanje popusta na proizvod...")
9.                              print("Unesite 6 za uklanjanje popusta sa proizvoda...")
10.                             print("Unesite 7 da izadjete iz menija administrativnih
    funkcija...")
11.                             admin = eval(input("Unesite vas izbor: "))
12.                             print("\n")
13.                             if admin == 1:
14.                                 korisnici.pretragaSvihProizvoda()
15.                             elif admin == 2:
16.                                 prodavac.unosProizvoda()
17.                             elif admin == 3:
18.                                 prodavac.dopunaKolicine()
19.                             elif admin == 4:
20.                                 prodavac.promenaCene()
21.                             elif admin == 5:
22.                                 prodavac.dodavanjePopusta()
23.                             elif admin == 6:
24.                                 prodavac.uklanjanjePopusta()
25.                             elif admin == 7:
26.                                 break
27.                             else:
28.                                 print("\n")
29.                                 print("Pogresno ste uneli broj! Pokusajte ponovo.")
30.                                 print("\n")
```

Listing 13 *Administrativne funkcije*

2.2 Funkcionalnosti namenjene kupcu

Funkcionalnosti namenjene kupcu odrađene su u modulu *kupac.py*.

Funkcionalnosti koje su pružene kupcu kao korisniku su: kreiranje naloga, prijavljivanje na postojeći nalog, pretraga proizvoda po kategorijama, pretraga specifičnog proizvoda i kupovina proizvoda. Što se tiče pretrage, odnosno prikaza svih proizvoda, ta funkcionalnost je pružena svim korisnicima i više o njoj će biti rečeno u sekciji **2.4 Funkcionalnosti namenjene svim korisnicima**.

Kreiranje naloga omogućeno je funkcijom *def signupKupca()* (Listing 14). U meniju *Formular za kreiranje naloga kupca* korisnik unosi ime, prezime, korisničko ime i lozinku, a takođe mora i da potvrdi lozinku. Ako se lozinka i potvrde lozinke poklapaju i ako korisničko ime i lozinka nemaju manje od šest a više od dvanaest karaktera, nalog se prihvata i lozinka se preko funkcije *encode()* konvertuje u bajtove tako da budu prihvatljivi za hash funkciju. Zatim se preko *hashlib.md5* funkcije uzima hash vrednost lozinke i preko *hexdigest()* vraća u heksadecimalnom formatu. Sve ove vrednosti se unose u tabelu kupca i na kraju kupac dobija poruku da je nalog uspešno napravljen. Ukoliko je korisnik pogrešno uneo neki od podataka, program ga obaveštava i daje šansu da ponovi unos. Provera postojanja identičnog korisničkog imena nije napravljena radi jednostavnosti.

```
1.  def signupKupca():
2.      print("*** Formular za kreiranje naloga kupca *** \n")
3.      ime_kupca = input("Unesite vase ime: ")
4.      prezime_kupca = input("Unesite vase prezime: ")
```

```
5.
6.      username = input("Unesite zeljeno korisnicko ime (Minimum 6 karaktera i maksimalno
    12 karaktera!): ")
7.      password = input("Unesite zeljenu lozinku (Minimum 6 karaktera i maksimalno 12
    karaktera!): ")
8.      confirm_password = input("Potvrdite lozinku: ")
9.      print("\n")
10.
11.     if (confirm_password == password and len(confirm_password) in range (6, 13) and
    len(username) in range(6, 13)):
12.         enc = confirm_password.encode()
13.         hash1 = hashlib.md5(enc).hexdigest()
14.
15.         Qk = "INSERT INTO Kupac (imeKupca, prezimeKupca, userKupca, passwdKupca, kupon)
    VALUES (%s,%s,%s,%s,%s)"
16.         mycursor.execute(Qk, (ime_kupca, prezime_kupca, username, hash1, 0,))
17.         db.commit()
18.         print("Uspesno ste napravili nalog!")
19.         print("\n")
20.
21.     else:
22.         print("\n")
23.         print("Korisnicko ime/lozinka imaju vise od 6 ili manje od 12 karaktera, ili se
    lozinke ne podudaraju! \n")
24.         print("\n")
```

Listing 14 Funkcija *def signupKupca()*

Prijavljivanje na postojeći nalog radi relativno slično, preko funkcije *def loginKupca()* (Listing 15). Korisnik, naravno, mora da unese svoje korisničko ime i lozinku, a program pokušava da konvertuje unesenu lozinku i konvertovanu vrednost pretraži u bazi podataka. Ako se korisničko ime i lozinka podudaraju sa nekom od istih kombinacija iz baze, prijavljivanje je uspešno, korisnik dobija obaveštenje o tome i funkcija vraća korisničko ime kupca. U suprotnom, korisnik dobija obaveštenje da se desila greška i da pokuša ponovo.

```
1.  def loginKupca():
2.      username = input("Unesite korisnicko ime: ")
3.      password = input("Unesite lozinku: ")
4.
5.      pwd = password.encode()
6.      pwd_hash = hashlib.md5(pwd).hexdigest()
7.      Qpwd = "SELECT * FROM Kupac WHERE userKupca = %s AND passwdKupca = %s"
8.      mycursor.execute(Qpwd, (username, pwd_hash,))
9.      nalog = mycursor.fetchone()
10.     if nalog:
11.         print("\n")
12.         print("Uspesno ste se ulogovali!")
13.         print("\n")
14.         return username
15.     else:
16.         print("\n")
17.         print("Desila se greska! Pokusajte ponovo!")
18.         print("\n")
```

Listing 15 Funkcija *def loginKupca()*

Pretraga proizvoda po kategorijama je funkcionalnost koja od kupca zahteva nekoliko unosa sa tastature. Ako je kupcu potrebno da pretražuje proizvode u samo željenim kategorijama (npr. samo PS4 igre, ili samo PS4 i PS5 igre), funkcija *def pretragaProizvodaPoKategorijama()* upravo to i omogućava (Listing 16). Ona prvo izlista nazive svih kategorija i njihove id brojeve. Onda od korisnika traži da u beskonačnoj while petlji unosi id brojeve tih kategorija sve dok to

ne želi više, u kom slučaju unosi 0. Svaki od brojeva koji unosi se skladišti u listi *kategorije*. While petlja ne sme da se prekine ako u listi *kategorije* ne postoji bar jedan element. Tada korisnik dobija poruku da mora uneti bar jednu kategoriju i program ga vraća na unos kategorije. Ako nakon while petlje, lista *kategorije* ima više od jednog elementa, od nje se pravi sekvenca tuple (kako bi se lakše radilo sa bazama) i iz baze se izvlače svi proizvodi i njihova dostupnost na lageru za kategorije koje su unete. Ako lista *kategorije* ipak sadrži samo jedan element, onda se izvlači samo taj jedan element iz liste i radi isti proces sa bazom. Na kraju se na ekran štampaju svi proizvodi u izabranim kategorijama, njihova količina na lageru i cena.

```
1.  def pretragaProizvodaPoKategorijama():
2.      print("Lista kategorija proizvoda i njihov identifikacioni brojevi: ")
3.      Qk = "SELECT * FROM Kategorija"
4.      mycursor.execute(Qk)
5.      for x in mycursor:
6.          print(x)
7.      kategorije = []
8.      broj = None
9.      while 1:
10.         broj = input("Unesite sve id brojeve kategorija za koje zelite da uradite
    pretragu proizvoda (Ukucajte 0 za kraj!): ")
11.         if (broj != '0'):
12.             try:
13.                 kategorije.append(int(broj))
14.             except:
15.                 print("\n")
16.                 print(broj + " nije broj. Pokusajte ponovo!")
17.         else:
18.             if len(kategorije) > 0:
19.                 break
20.             else:
21.                 print("\n")
22.                 print("Morate uneti bar jednu kategoriju!")
23.                 continue
24.     if len(kategorije) > 1:
25.         kategorije_tuple = tuple(kategorije)
26.         Qfetch = "SELECT p.idProizvoda, p.naziv, p.cena, k.idKategorije,
    k.nazivKategorije, pk.na_lageru FROM Proizvod p, Kategorija k, ProizvodKategorija pk
    WHERE pk.idProizvoda = p.idProizvoda AND pk.idKategorije = k.idKategorije AND
    k.idKategorije IN {}".format(kategorije_tuple)
27.         mycursor.execute(Qfetch)
28.     else:
29.         kategorije_tuple = kategorije[0]
30.         Qfetch = "SELECT p.idProizvoda, p.naziv, p.cena, k.idKategorije,
    k.nazivKategorije, pk.na_lageru FROM Proizvod p, Kategorija k, ProizvodKategorija pk
    WHERE pk.idProizvoda = p.idProizvoda AND pk.idKategorije = k.idKategorije AND
    k.idKategorije = %s"
31.         mycursor.execute(Qfetch, (kategorije_tuple,))
32.     proizvodi = mycursor.fetchall()
33.     print("\n")
34.     print("Proizvodi koji se nalaze u kategorijama koje ste uneli (ID proizvoda, naziv
    proizvoda, cena, ID kategorije, naziv kategorije i kolicina proizvoda na lageru):\n")
35.     for x in proizvodi:
36.         print(x)
37.     print("\n")
```

Listing 16 Funkcija *def pretragaProizvodaPoKategorijama()*

Pretraga specifičnog proizvoda se realizuje preko funkcije *def pretragaSpecificnogProizvoda()* (Listing 17). Korisnik preko tastature unosi tačno ime

proizvoda koji želi da pretraži. Program onda u slučaju da takav proizvod postoji u bazi, izlistava taj proizvod i njegovu cenu, a u suprotnom obaveštava da proizvod ne postoji u prodavnici. Nakon što izlista proizvod, program će izlistati i sve postojeće kategorije proizvoda. Onda će napraviti parove unesenog proizvoda i dostupnih kategorija za taj proizvod i njih izlistati preko tabele *ProizvodKategorija* iz baze. To će uraditi u formatu njihovih id brojeva. Takođe, za svaki par će izlistati i koliko je trenutno dostupno na lageru.

```
1.  def pretragaSpecificnogProizvoda():
2.      proizvod = input("Unesite tacno ime proizvoda koji zelite da pretrazite: ")
3.      print("\n")
4.      Qp = "SELECT naziv, cena FROM Proizvod WHERE naziv=%s"
5.      try:
6.          print("Proizvod koji ste uneli i njegova cena u dinarima: ")
7.          mycursor.execute(Qp, (proizvod,))
8.          pr = mycursor.fetchone()
9.          print(pr)
10.         print("\n")
11.         print("Kategorije proizvoda i njihovi identifikacioni brojevi: ")
12.         Qk = "SELECT * FROM Kategorija"
13.         mycursor.execute(Qk)
14.         for kat in mycursor:
15.             print(kat)
16.         print("\n")
17.         print("ID proizvoda koji ste uneli (prvi broj), id brojevi dostupnih kategorija
    za njega (drugi broj) i kolicina proizvoda na lageru (treci broj):")
18.         Qpk = "SELECT pk.idProizvoda, pk.idKategorije, pk.na_lageru FROM
    ProizvodKategorija pk, Proizvod p WHERE pk.idProizvoda = p.idProizvoda AND
    p.idProizvoda=%s"
19.         Qid = "SELECT idProizvoda FROM Proizvod WHERE naziv=%s"
20.         mycursor.execute(Qid, (proizvod,))
21.         id_p = mycursor.fetchone()[0]
22.         mycursor.execute(Qpk, (id_p,))
23.         for pk in mycursor:
24.             print(pk)
25.         print("\n")
26.     except:
27.         print("\n")
28.         print("Proizvod koji ste uneli ne postoji u prodavnici!")
29.         print("\n")
```

Listing 17 Funkcija *def pretragaSpecificnogProizvoda()*

Nakon svih pretraga sledi kupovina proizvoda, realizovana preko funkcije *def kupovinaProizvoda()* (Listing 18). Pre same beskonačne while petlje račun se postavlja na 0. U while petlji korisnik unosi id broj proizvoda koji želi da kupi, kategoriju u kojoj se on nalazi, kao i količinu koju želi da kupi. Program onda pokušava da iz tabele *ProizvodKategorija* u bazi podataka izvuče tačno taj proizvod, odnosno njegovu količinu na lageru. Ako ne uspe, dobija upozorenje da je pogrešno uneo podatke. Ako je kupac uneo količinu koja je bar jednaka ili manja od one koja postoji u prodavnici, program iz baze uzima cenu jednog takvog proizvoda, uzima popust na taj proizvod (ako postoji) i na kraju množi količinu koju je korisnik uneo sa sniženom cenom, a rezultat je ukupna cena kupovine. Korisnik dobija poruku ako je uneo veću količinu proizvoda nego što je dostupno na lageru.

Onda se na račun dodaje vrednost proizvoda, a zatim se u bazi menjaju vrednosti za količinu proizvoda na lageru (*na_lageru*) i količinu ukupno prodatih komada tog proizvoda (*brojProdatih*). Zatim, kupac dobija priliku da proizvod koji je odabrao zadrži u korpi i nastavi sa kupovinom, ili odmah kupi proizvode koje je odabrao i plati račun. Ako se odluči da smesti proizvode u korpu, program se samo vraća nazad na novi unos proizvoda i kupac to može da

radi koliko puta želi, sve dok ne unese sve proizvode koje želi da kupi. Korisnik ne može da skida proizvode sa korpe. Ukoliko se korisnik odluči da pređe na plaćanje, funkcija vraća iznos računa. Korisnik dobija upozorenje ukoliko je uneo pogrešan karatker. Sama završnica kupovine se obavlja u glavnom meniju, gde se obračunavaju popusti i kuponi.

```
1.  def kupovinaProizvoda():
2.      racun = 0
3.      while 1:
4.          proizvod = eval(input("Unesite ID proizvoda koji zelite da kupite: "))
5.          kategorija = eval(input("Unesite ID kategorije proizvoda: "))
6.          kolicina = eval(input("Unesite kolicinu zeljenog proizvoda: "))
7.          print("\n")
8.          try:
9.              Q1 = "SELECT pk.idProizvoda, pk.idKategorije, pk.na_lageru FROM
    ProizvodKategorija pk, Proizvod p, Kategorija k WHERE pk.idProizvoda = p.idProizvoda
    AND pk.idKategorije = k.idKategorije AND p.idProizvoda=%s AND k.idKategorije=%s"
10.             mycursor.execute(Q1, (proizvod, kategorija))
11.             kol = mycursor.fetchone()[2]
12.             if (kol - kolicina >= 0):
13.                 nova_kolicina = kol - kolicina
14.                 Q2 = "SELECT cena FROM Proizvod WHERE idProizvoda=%s"
15.                 mycursor.execute(Q2, (proizvod,))
16.                 cena_jednog = mycursor.fetchone()[0]
17.                 Q3 = "SELECT popust FROM Proizvod WHERE idProizvoda=%s"
18.                 mycursor.execute(Q3, (proizvod,))
19.                 popust = mycursor.fetchone()[0]
20.                 ukupna_cena = int(kolicina * (cena_jednog - popust))
21.
22.                 racun += ukupna_cena
23.                 Q4 = "UPDATE ProizvodKategorija SET na_lageru=%s WHERE idProizvoda=%s
    AND idKategorije=%s"
24.                 mycursor.execute(Q4, (nova_kolicina, proizvod, kategorija))
25.                 db.commit()
26.                 Q5 = "UPDATE Proizvod SET brojProdatih = brojProdatih + %s WHERE
    idProizvoda = %s"
27.                 mycursor.execute(Q5, (kolicina, proizvod))
28.                 db.commit()
29.                 while 1:
30.                     print("Unesite 1 za dodavanje proizvoda u korpu...")
31.                     print("Unesite 2 za placanje...")
32.                     placanje = eval(input("Unesite vas izbor: "))
33.                     if placanje == 1:
34.                         break
35.                     elif placanje == 2:
36.                         return racun
37.                     else:
38.                         print("\n")
39.                         print("Pogresno ste uneli broj! Pokusajte ponovo.")
40.                         continue
41.             else:
42.                 print("\n")
43.                 print("Kolicina koju ste odabrali prelazi kolicinu proizvoda na
    lageru!")
44.                 continue
45.         except:
46.             print("\n")
47.             print("Doslo je do greske! Pokusajte ponovo sa unosom proizvoda.")
48.             continue
```

Listing 18 Funkcija *def kupovinaProizvoda()*

2.3 Funkcionalnosti namenjene administratoru (prodavcu)

Funkcionalnosti koje su pružene administratoru, odnosno prodavcu su: kreiranje novog korisničkog naloga, pristup postojećem nalogu, unos novog proizvoda u prodavnicu, dopuna količine proizvoda na lageru, promena cene postojećeg proizvoda, dodavanje popusat, kao i uklanjanje popusta sa postojećeg proizvoda. Prodavac, kao i kupac, ima i mogućnost da dobije prikaz svih proizvoda u prodavnici. Više o toj funkcionalnosti u sekciji **2.4 Funkcionalnosti namenjene svim korisnicima**.

Kreiranje novog korisničkog naloga prodavca realizovano je preko funkcije *def signupProdavca()* (Listing 19). Funkcija je urađeno veoma slično kao i funkcija *def signupKupca()* i jedina razlika je što se od prodavca traži da unese i svoju platu, kako bi i ona bila uneta u bazu podataka.

```
1.  def signupProdavca():
2.      print("*** Formular za kreiranje naloga prodavca *** \n")
3.      ime_prodavca = input("Unesite vase ime: ")
4.      prezime_prodavca = input("Unesite vase prezime: ")
5.      plata_prodavca = input("Unesite vasu platu: ")
6.
7.      username = input("Unesite zeljeno korisnicko ime (Minimum 6 karaktera i maksimalno
    12 karaktera!): ")
8.      password = input("Unesite zeljenu lozinku (Minimum 6 karaktera i maksimalno 12
    karaktera!): ")
9.      confirm_password = input("Potvrdite lozinku: ")
10.     print("\n")
11.
12.     if (confirm_password == password and len(confirm_password) in range (6, 13) and
    len(username) in range(6, 13)):
13.         enc = confirm_password.encode()
14.         hash1 = hashlib.md5(enc).hexdigest()
15.
16.         Qk = "INSERT INTO Prodavac (ime, prezime, user, passwd, plata) VALUES
    (%s,%s,%s,%s,%s)"
17.         mycursor.execute(Qk, (ime_prodavca, prezime_prodavca, username, hash1,
    plata_prodavca,))
18.         db.commit()
19.         print("Uspesno ste napravili nalog!")
20.         print("\n")
21.     else:
22.         print("\n")
23.         print("Korisnicko ime/lozinka imaju vise od 6 ili manje od 12 karaktera, ili se
    lozinke ne podudaraju! \n")
24.         print("\n")
```

Listing 19 Funkcija *def signupProdavca()*

Pristup postojećem nalogu prodavca je realizovan preko funkcije *def loginProdavca()* (Listing 20) i odrađen je na identičan način kao i funkcija *def loginKupca()*.

```
1.  def loginProdavca():
2.      username = input("Unesite korisnicko ime: ")
3.      password = input("Unesite lozinku: ")
4.
5.      pwd = password.encode()
6.      pwd_hash = hashlib.md5(pwd).hexdigest()
7.      Qpwd = "SELECT * FROM Prodavac WHERE user = %s AND passwd = %s"
8.      mycursor.execute(Qpwd, (username, pwd_hash,))
9.      nalog = mycursor.fetchone()
10.     if nalog:
```

```
        print("\n")
        print("Uspesno ste se ulogovali!")
        print("\n")
        return username
    else:
        print("\n")
        print("Desila se greska! Pokusajte ponovo!")
        print("\n")
```

Listing 20 Funkcija *def loginProdavca()*

Unos proizvoda urađen je preko funkcije *def unosProizvoda()* (Listing 21). Važno je odmah na početku napomenuti da je cela funkcionalnost urađena sa namerom da prodavac u prodavnicu unosi proizvod sa istim imenom u više kategorija, ali će proizvod u svakoj od tih kategorija imati identičnu cenu (npr. video igra *Cyberpunk 2077* će imati istu cenu i za *PS5* kategoriju i za *PC* kategoriju, kao i za bilo koju drugu kategoriju u kojoj će se eventualno nalaziti). Takođe, funkcija je odrađena i sa namerom da se proizvod sa istim imenom unosi u jednakim količinama za svaku kategoriju, a prodaje u proizvoljnim količinama. Dakle, kada prodavac jednom odabere količinu unosa (npr. 10) i odabere proizvod i sve njegove kategorije, po deset komada tog proizvoda će biti ubačen u prodavnicu za svaku od kategorija (npr. za *Cyberpunk 2077* video igru, biće dodato 10 *Cyberpunk 2077* PS4 igara, 10 *Cyberpunk 2077* PC igara, itd.).

Sama funkcionalnost počinje korisnikovim unosom naziva, cene i količine proizvoda koji želi da doda u prodavnicu. Program odmah pokušava da proizvod doda u bazu, u tabelu *Proizvod*, a ukoliko to ne uspe, upozorava korisnika i daje mu šansu da pokuša ponovo. Program onda iz baze vadi sve postojeće kategorije i štampa ih na ekran. Prodavac onda preko beskonačne while petlje, treba da unosi id brojeve svih kategorija u kojima želi da se uneseni proizvod nalazi, sve dok ne unese 0. Korisnik mora uneti bar jedan id broj kategorije, kako bi se proizvod mogao svrstati u bar jednu kategoriju. Nakon petlje, program preko id broja proizvoda koji je korisnik uneo i odabranih id brojeva kategorija, u for petlji pravi parove u bazi u tabeli *ProizvodKategorija* i zapisuje unetu količinu. Na kraju korisnik dobija poruku o uspešnom unosu.

```
def unosProizvoda():
    naziv = input("Unesite naziv proizvoda koji zelite da dodate: ")
    cena = eval(input("Unesite cenu proizvoda: "))
    na_lageru = eval(input("Unesite kolicinu proizvoda koji unosite u sistem: "))
    Qp = "INSERT INTO Proizvod (naziv, cena, popust, brojProdatih) VALUES(%s,%s,%s,%s)"
    while 1:
        try:
            mycursor.execute(Qp, (naziv, cena, 0, 0))
            db.commit()
            break
        except:
            print("Desila se greska! Vodite racuna da unesete naziv, kolicinu proizvoda na lageru i cenu u validnom formatu i pokusajte ponovo!")
            print("\n")
            break
    print("-----------------------------------------------------------------\n")
    print("Lista kategorija proizvoda i njihov identifikacioni brojevi: ")
    Qk = "SELECT * FROM Kategorija"
    mycursor.execute(Qk)
    for x in mycursor:
        print(x)
    kategorije = []
    broj = None
    while 1:
        broj = input("Unesite sve id brojeve kategorija kojima vas proizvod pripada (Ukucajte 0 za kraj!): ")
```

```
25.         if (broj != '0'):
26.             try:
27.                 kategorije.append(int(broj))
28.             except:
29.                 print(broj + " nije broj. Pokusajte ponovo!")
30.         else:
31.             if len(kategorije) > 0:
32.                 break
33.             else:
34.                 print("\n")
35.                 print("Morate uneti bar jednu kategoriju!")
36.                 continue
37.     Qfetch = "SELECT idProizvoda FROM Proizvod ORDER BY idProizvoda DESC LIMIT 1"
38.     mycursor.execute(Qfetch)
39.     poslednji_id = mycursor.fetchone()
40.     Qpk = "INSERT INTO ProizvodKategorija (idProizvoda, idKategorije, na_lageru) VALUES
    (%s,%s,%s)"
41.     for value in kategorije:
42.         mycursor.execute(Qpk, (poslednji_id[0], value, na_lageru))
43.         db.commit()
44.     print("\n")
45.     print("Uspesno ste uneli proizvod!")
46.     print("\n")
```

Listing 21 Funkcija *def unosProizvoda()*

Dopuna količine proizvoda na lageru urađena je preko funkcije *def dopunaKolicine()* (Listing 22). Prodavac treba da unese id broj proizvoda koji želi da dopuni i količinu koju želi da doda. Ukoliko je njegov unos bio tačan (takav id broj postoji u prodavnici i unosi brojeva su u korektnom formatu), polje *na_lageru* u tabeli *ProizvodKategorija* u bazi se sabira sa vrednošću koje je uneo prodavac i on dobija poruku o uspešnom unosu. Ukoliko se desi greška, ispisuje se poruka.

```
1.  def dopunaKolicine():
2.      idbroj = input("Unesite id broj proizvoda u prodavnici za koji zelite uradite
    dopunu: ")
3.      kolicina = eval(input("Unesite za koliko zelite da dopunite proizvod: "))
4.      Q1 = "UPDATE ProizvodKategorija SET na_lageru = na_lageru + %s WHERE idProizvoda =
    %s"
5.      try:
6.          mycursor.execute(Q1, (kolicina, idbroj))
7.          db.commit()
8.          print("\n")
9.          print("Uspesno ste dopunili kolicinu proizvoda na lageru!")
10.         print("\n")
11.     except:
12.         print("\n")
13.         print("Desila se greska! Pokusajte ponovo sa unosom.")
14.         print("\n")
```

Listing 22 Funkcija *def dopunaKolicine()*

Promena cene proizvoda vrši se preko funkcije *def promenaCene()* (Listing 23). Kako bi korisnik promenio cenu nekog proizvoda u prodavnici, mora da unese tačno ime tog proizvoda. Ako je unos pogrešan, korisnik dobija upozorenje. Ako se unos podudara sa nekim od naziva proizvoda u bazi, korisnik prelazi u if upit i dužan je da unese novu cenu tog proizvoda. Program pokušava da podatak o ceni promeni u tabeli *Proizvod* u bazi podataka i ako u tome uspe, obaveštava korisnika. Ukoliko ne uspe, dobija upozorenje da je došlo do greške i da pripazi na format unete cene.

```
1.  def promenaCene():
2.      naziv_proizvoda = input("Unesite tacno ime proizvoda ciju cenu zelite da promenite:
    ")
3.      Qp = "SELECT naziv FROM Proizvod WHERE naziv = %s"
4.      mycursor.execute(Qp, (naziv_proizvoda,))
5.      fetch = mycursor.fetchone()
6.      if fetch != None:
7.          nova_cena = input("Unesite novu cenu proizvoda: ")
8.          Qc = "UPDATE Proizvod SET cena = %s WHERE naziv = %s"
9.          try:
10.             mycursor.execute(Qc, (int(nova_cena), naziv_proizvoda,))
11.             db.commit()
12.             print("Cena " + naziv_proizvoda + " je uspesno promenjena!")
13.             print("\n")
14.         except:
15.             print("Doslo je do greske! Obratite paznju na format unete cene i pokusajte
    ponovo!")
16.             print("\n")
17.     else:
18.         print("Doslo je do greske! Obratite paznju na format unetog naziva proizvoda i
    pokusajte ponovo!")
19.         print("\n")
```

Listing 23 Funkcija *def promenaCene()*

Dodavanje popusta vrši se preko funkcije *def dodavanjePopusta()* (Listing 24). Pošto su popusti uglavnom kratkoročni, ova funkcija, kao i polje *popust* u tabeli *Proizvod* u bazi podataka postoje kako bi se cena snizila na lakši način i isto tako vratila na stari iznos nakon isteka popusta. Ova funkcionalnost, zbog svoje sličnosti sa dodavanjem popusta ima i veoma sličnu funkciju u programu. Princip je identičan kao i kod prethodne funkcije, sa tim što se umesto promene polja *cena*, u bazi podataka menja polje *popust* u tabeli *Proizvod*.

```
1.  def dodavanjePopusta():
2.      proizvod = input("Unesite naziv proizvoda za koji zelite da unesete popust: ")
3.      Qp = "SELECT naziv FROM Proizvod WHERE naziv = %s"
4.      mycursor.execute(Qp, (proizvod,))
5.      fetch = mycursor.fetchone()
6.      if fetch != None:
7.          popust = input("Unesite popust u dinarima: ")
8.          Qpopust = "UPDATE Proizvod SET popust = %s WHERE naziv = %s"
9.          try:
10.             mycursor.execute(Qpopust, (int(popust), proizvod))
11.             db.commit()
12.             print("Popust je uspesno unet u sistem!")
13.             print("\n")
14.         except:
15.             print("Doslo je do greske! Obratite paznju na format unete cene i pokusajte
    ponovo!")
16.             print("\n")
17.     else:
18.         print("Doslo je do greske! Obratite paznju na format unetog naziva proizvoda i
    pokusajte ponovo!")
19.         print("\n")
```

Listing 24 Funkcija *def dodavanjePopusta()*

Uklanjanje popusta je još jednostavnije i vrši se preko funkcije *def uklanjanjePopusta()* (Listing 25). Korisnik takođe unosi naziv proizvoda za koji želi da ukloni popust, a program onda pokušava da polje *popust* u tabeli *Proizvod* u bazi podataka postavi na vrednost 0. Ukoliko naziv proizvoda koji je unesen postoji, korisnik dobija obaveštenje da je popust uklonjen.

Ukoliko je došlo do greške (proizvod koji je unesen ne postoji u bazi, ili korisnik nije uneo ništa), korisnik takođe dobija obaveštenje o tome.

```
def uklanjanjePopusta():
    proizvod = input("Unesite naziv proizvoda za koji zelite da uklonite popust: ")
    Qpopust = "UPDATE Proizvod SET popust = %s WHERE naziv = %s"
    if proizvod != None:
        try:
            mycursor.execute(Qpopust, (0, proizvod))
            db.commit()
            print("Popust je uspesno uklonjen iz sistema!")
            print("\n")
        except:
            print("Doslo je do greske! Obratite paznju na format unete cene i pokusajte ponovo!")
            print("\n")
    else:
        print("Doslo je do greske! Obratite paznju na format unetog naziva proizvoda i pokusajte ponovo!")
        print("\n")
```

Listing 25 Funkcija *def uklanjanjePopusta()*

2.4 Funkcionalnosti namenjene svim korisnicima

Prodavnica *ASCII* poseduje dve funkcionalnosti koje su nazavisne od tipa korisnika koji koristi program. Radi se o prikazu svih proizvoda u prodavnici, realizovanom preko funkcije *def pretragaSvihProizvoda()* i stubičastom grafikonu, koji preko funkcije *def grafikon()* u rastućem poretku pokazuje pet najprodavanijih proizvoda u prodavnici.

Pretraga svih proizvoda se pokreće preko funkcije *def pretragaSvihProizvoda()* (Listing 26). Iz baze se vade prvo svi proizvodi i štampaju. Onda se vadi lista svih kategorija i takođe štampa. Pošto proizvod može da spada u više kategorija, ali da njegova dostupnost na lageru ne bude jednaka za sve kategorije, iz baze se takođe uzimaju i vrednosti iz tabele *ProizvodKategorija* i štampa na ekran. Ovim putem, korisnik ima id brojeve svih proizvoda i kategorija i unošenjem željene kombinacije ova dva parametra može da dođe do željenog proizvoda.

```
def pretragaSvihProizvoda():
    print("Lista svih proizvoda (ID proizvoda, naziv, cena i kolicina proizvoda na lageru): ")
    Qp = "SELECT idProizvoda, naziv, cena FROM Proizvod"
    mycursor.execute(Qp)
    for x in mycursor:
        print(x)
    print("\n")
    print("Lista svih kategorija (ID kategorije i naziv): ")
    Qk = "SELECT * FROM Kategorija"
    mycursor.execute(Qk)
    for y in mycursor:
        print(y)
    print("\n")
    print("ID proizvoda (prvi broj), ID kategorija u kojima je proizvod dostupan (drugi broj) i kolicina proizvoda na lageru (treci broj):")
    Qpk = "SELECT * FROM ProizvodKategorija"
    mycursor.execute(Qpk)
    for z in mycursor:
        print(z)
```

```
19.     print("\n")
```

Listing 26 Funkcija *def pretragaSvihProizvoda()*

Funkcija *def grafikon()* uzima nazive i ukupnu količinu prodatih komada pet najprodavanijih proizvoda iz baze podataka (Listing 27). Nazive smešta u jednu, a količinu prodatih komada u drugu listu. Na osnovu ovih lista se kreiraju ose na grafikonu preko *matplotlib.pyplot* interfejsa i prikazuju svakom korisniku pri pokretanju aplikacije.

```
1.  def grafikon():
2.      Q1 = "SELECT naziv, brojProdatih FROM Proizvod ORDER BY brojProdatih ASC LIMIT 5"
3.      mycursor.execute(Q1)
4.
5.      proizvodi = []
6.      prodati = []
7.
8.      for x in mycursor:
9.          proizvodi.append(x[0])
10.         prodati.append(x[1])
11.
12.     plt.bar(proizvodi, prodati)
13.     plt.xlabel('proizvodi')
14.     plt.xticks(rotation = 90)
15.     plt.ylabel('prodato komada')
16.     plt.title("TOP 5 NAJPRODAVANIJIH PROIZVODA")
17.     plt.show()
```

Listing 27 Funkcija *def grafikon()*

3. Zaključak

Aplikacija *ASCII* poseduje najosnovnije funkcionalnosti potrebne kupcu da bez grafičkog interfejsa na brz način izvrši kupovinu željenog proizvoda. I prodavcu, odnosno administratoru je dostpuno da lako proveri stanje u kojima se proizvodi nalaze i brzo to stanje promeni. *Python* se pokazao kao izvrsno okruženje za implementaciju ovih funkcionalnosti, jer pruža sve potrebne alate, uz maksimalnu jednostavnost.

Sa dodavanjem još bezbednijeg i kompleksnijeg pristupa bazi, bolje provere unosa korisnika, nekoliko dodatnih funkcionalnosti, kompleksnije realizacije korpe i grafičkog interfejs, ili modernog web dizajna, ova aplikacija bi bila ozbiljna konkurencija modernim kupoprodajnim web servisima.

4. Literatura

1. https://stackoverflow.com/questions/33305975/how-to-set-up-an-sql-tables-structure-for-product-categories
2. https://dev.mysql.com/doc/connector-python/en/connector-python-api-mysqlcursor-executemany.html
3. https://www.geeksforgeeks.org/md5-hash-python/
4. https://www.geeksforgeeks.org/graph-plotting-in-python-set-1/