

## PROGRAMIRANJE 1



**Milena Vujošević Janičić, Jovana Kovačević,  
Danijela Simić, Anđelka Zečević**

**PROGRAMIRANJE 1**  
**Zbirka zadataka sa rešenjima**

**Beograd  
2016.**

Autori:

*dr Milena Vujošević Jančić*, docent na Matematičkom fakultetu u Beogradu

*dr Jovana Kovačević*, docent na Matematičkom fakultetu u Beogradu

*Danijela Simić*, asistent na Matematičkom fakultetu u Beogradu

*Anđelka Zečević*, asistent na Matematičkom fakultetu u Beogradu

PROGRAMIRANJE 1

Zbirka zadataka sa rešenjima

# Sadržaj

<b>1</b>	<b>Uvodni zadaci</b>	<b>1</b>
1.1	Rešenja	10
<b>2</b>	<b>Kontrola toka</b>	<b>29</b>
2.1	Naredbe grananja	29
2.2	Rešenja	39
2.3	Petlje	76
2.4	Rešenja	95
2.5	Funkcije	134
2.6	Rešenja	145
<b>3</b>	<b>Predstavljanje podataka</b>	<b>181</b>
3.1	Nizovi	181
3.2	Rešenja	193
3.3	Pokazivači	241
3.4	Rešenja	247
3.5	Niske	268
3.6	Rešenja	279
3.7	Višedimenzioni nizovi	302
3.8	Rešenja	309
3.9	Strukture	310
3.10	Rešenja	316
<b>4</b>	<b>Ulaz i izlaz programa</b>	<b>347</b>
4.1	Standardni tokovi	347
4.2	Argumenti komandne linije	347
4.3	Datoteke	347
4.4	Rešenja	361

<b>5</b>	<b>Razni zadaci</b>	<b>389</b>
5.1	Rešenja . . . . .	389

# Predgovor

U okviru kursa *Programiranje 1* na Matematičkom fakultetu vežbaju se zadaci koji imaju za cilj da studente nauče ...

*Autori*





# 1

## Uvodni zadaci

**Zadatak 1.1** Napisati program koji na standardni izlaz ispisuje tekst **Zdravo svete!**. Resenje se ne uklapa sa tekstem zadatka. Ono sto su altrenativni ispisi u resenju moze da ostane pod komentarima, ali da se postavi kao dva razlicita resenja, ali uvek resenje mora da se poklapa sa test primerom. U okviru prvog resenja potrebno je i kratko uputstvo kako da se prevede i pokrene program. Takođe, komentar uz include i uz int main i uz return - to posle naravno necemo komentarisati.

[Rešenje 1.1]

**Zadatak 1.2** Napisati program koji poziva korisnika da unese dva cela broja sa standardnog ulaza, a zatim ispisuje:

- (a) unete vrednosti
- (b) njihov zbir Ovo bi mogao da bude zadatak sa prodavnicom i kupovinom dva artikla, tako da bih takav neki zadatak dodala.
- (c) njihovu razliku Ovo je u sustini isti zadatak kao zadatak sa prodavnicom i kusurom, samo je taj malko komplikovaniji. Ostavila bih oba, ali bih ovaj drugi priblizila po redosledu ovom zadatku.
- (d) njihov proizvod Ovo je u sustini isti zadatak kao zadatak sa jabukama, ali bih ovaj drugi priblizila po redosledu ovom zadatku.

Ovde je problem sto oni jos nisu ucili if, a deljenje je zbog toga nezgodno. Ja bih izbacila poslednje dve stavke i uradila samo a,b,c i d  
U resenju prokomentarisati deklaracije int x i int y: Deklariše se promenljiva

celobrojnog tipa u kojoj će biti čuvana vrednost prvog broja koji se unosi sa ulaza ili tako nekako, pa slično za y i rezultat. Na ovom mestu bih stavila punu recenicu koja pominje reci i deklaracija i celobrojni tip i slično, a u ostalim resenjima samo komentare sta promenljiva predstavlja, ukoliko joj ime nije dovoljno deskriptivno. Na mestu gde je prvi put deklaracija float ili double promenljive, tu bih isto pomenula tip, da pomenemo jednostruku i dvostruku tacnost.

[Rešenje 1.2]

**Zadatak 1.3** Napisati program koji sa standardnog ulaza učitava realnu vrednost izraženu u inčima, konvertuje tu vrednost u centimetre i ispisuje je na standardni izlaz zaokruženu na dve decimale. UPUTSTVO: *Jedan inč ima 2.54 centimetra.*

[Rešenje 1.3]

**Zadatak 1.4** Napisati program koji sa standardnog ulaza učitava dužinu poluprečnika kruga i na standardni izlaz ispisuje njegov obim i površinu.

[Rešenje 1.4]

**Zadatak 1.5** Napisati program koji za uneti pozitivan broj na standardni izlaz ispisuje njegove cifre jedinica, desetica i stotina. *Preformulisala sam da broj nije trocifren jer nemamo if da proverimo da li je ulaz ispravan. Sa ovakvom formualcijom program moze da radi ispravno i ako broj nije trocifren. Ispraviti u resenju tipove, tj da brojevi budu unsigned kako bi uvek radio ispravno tj kako ne bi dobijao negativne cifre. Ispraviti komentar u printf-u koji pominje cele brojeve.*

[Rešenje 1.5]

**Zadatak 1.6** Napisati program koji učitava trocifreni broj sa standardnog ulaza i ispisuje broj dobijen obrtanjem njegovih cifara.

[Rešenje 1.6] *Ovaj zadatak mozda prebaciti u if? Kako proveriti da li je uneti broj stvarno trocifren? Ili nekako formulisati zadatak tako da nije neophodno da broj bude trocifren. Npr, ispisuje broj koji se dobije kada cifra jedinica i cifra stotina zamene mesta. Sve zadatke koji se bave radom sa ciframa broja grupisati na jedno mesto.*

**Zadatak 1.7** Ovaj zadatak je radjen na praktikumu (jednakostranici tro-ugao), dole je naveden. Da li da ostane dole ili da se prebaci ovde? *Mislim da redosled treba da bude po smislu, a ne po vezbe/praktikum podeli. Zato, ako mu je mesto ovde, onda ga tu treba i prebaciti. Zadatke poredjati na pocetku i*

po tipovima, a posle mogu i da se mesaju: dakle da prvo ide par zadataka koji rade sa celobrojnim vrednostima, a onda zadaci koji rade sa realnim vrednostima. Redosled zadataka je vrlo osetljiva stvar i to bih ostavila za kasniju fazu: za sada samo okvirno kako nam se ucini da treba, a onda posle ce biti jos tumbanja. Zato nije vazno da se brojevi zadataka i resenja poklapaju u ovom trenutku, kada napravimo finalno rasporedjivanje onda cemo da menjamo te brojeve. Za rasporedjivanje zadataka vazna su nam i resenja.

[Rešenje 1.7]

**Zadatak 1.8** Napisati program koji za unetu cenu proizvoda ispisuje najmanji broj novčanica koje je potrebno izdvojiti prilikom plaćanja proizvoda. Na raspolaganju su novčanice od 1000, 100, 50, 10 i 1 dinar.

[Rešenje 1.8]

**Zadatak 1.9** Napisati program koji za tri cela broja koja se unose sa standardnog ulaza ispisuje njihovu aritmetičku sredinu na standardni izlaz.

[Rešenje 1.9]

**Zadatak 1.10** Ovaj zadatak je radjen na praktikumu (razmena vrednosti), dole je naveden. Da li da ostane dole ili da se prebaci ovde?

[Rešenje 1.10]

**Zadatak 1.11** Napisati program za uneti ceo broj ispisuje njegov kvadrat i kub.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite ceo broj: 4  
|| Kvadrat:16  
|| Kub: 64
```

[Rešenje ??]

**Zadatak 1.12** Napisati program koji za unete dužine stranica pravougaonika ispisuje njegov obim i površinu.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite duzine stranica pravougaonika: 2 8  
|| Obim: 20  
|| Povrsina: 16
```

[Rešenje 1.12]

**Zadatak 1.13** Sve geometrijske zadatke bih grupisala da budu susedni. Malo me zbunjuje sto su resenja za kvadrat i pravougaonik sa int, a za ove druge sa float, kao da stranica pravougaonika ne moze da bude realan broj? Mozda bi svi ti goemetrijski trebalo da rade sa realnim brojevima? Tako bi bili konzistentni... Napisati program koji za unetu dužinu stranice jednakostraničnog trougla ispisuje njegov obim i površinu.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:  
Unesite duzine stranica trougla: 3 4 5  
Obim: 12.00  
Povrsina: 6.00
```

[Rešenje 1.13]

**Zadatak 1.14** Napisati program koji pomaže moleru da izračuna površinu zidova prostorije koju treba da okreči. Za unete dimenzije sobe u metrima (dužinu, širinu i visinu), program treba da ispiše površinu zidova za krečenje pod pretpostavkom da na vrata i prozore otpada oko 20%. Omogućiti i da na osnovu unosene cene usluge po kvadratnom metru program izračuna ukupnu cenu krečenja.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenzije sobe: 4 4 3  
Unesite cenu po kvadratnom metru: 500  
Moler treba da okreci 51.2 kvadratna metra  
Cena krecenja je 25600
```

[Rešenje 1.14]

**Zadatak 1.15** Napisati program koji za unetu količinu jabuka u kilogramima i unetu cenu po kilogramu ispisuje ukupnu vrednost date količine jabuka.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:  
Unesite kolicinu jabuka (u kg): 6  
Unesite cenu (u dinarima): 82  
Molimo platite 492 dinara.
```

[Rešenje 1.15]

**Zadatak 1.16** Napisati program koji pomaže kasirki da obračuna kusur koji treba da vrati kupcu. Za unetu cenu artikla, količinu artikla i iznos koji je kupac dao, program treba da ispiše vrednost kusura.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom cenu, kolicinu i iznos: 132 2 500
|| Kusur je 236 dinara.
```

[Rešenje 1.16]

**Zadatak 1.17** Napisati program koji za uneti prirodni četvorocifreni broj:

- (a) izračunava proizvod cifara
- (b) izračunava razliku sume krajnjih i srednjih cifara
- (c) izračunava sumu kvadrata cifara
- (d) izračunava broj koji se dobija ispisom cifara u obrnutom poretku
- (e) izračunava broj koji se dobija zamenom cifre jedinice i cifre stotine

Ponovo imamo problem sa time sto je broj cetvorocifren a nemamo if proveru?

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: 2371
|| Proizvod cifara: 42
|| Razlika sume krajnjih i srednjih: -7
|| Suma kvadrata cifara: 63
|| Broj u obrnutom poretku: 1732
|| Broj sa zamenjenom cifrom jedinica i stotina: 2173
```

[Rešenje 1.17]

**Zadatak 1.18** Napisati program koji ispisuje broj koji se dobija izbacivanjem cifre desetica u unetom prirodnom broju.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1349
|| Rezultat je: 139
```

[Rešenje 1.18]

## 1 Uvodni zadaci

---

**Zadatak 1.19** Napisati program koji za uneti prirodan broj  $x$  i unete vrednosti  $c$  i  $p$  ispisuje broj koji se dobija ubacivanjem cifre  $c$  u broj  $x$  na poziciji  $p$ . Podrazumeva se da je broj  $p$  manji od ukupnog broja cifara broja  $x$  i da numeracija cifara počinje od 1. UPUTSTVO: *Koristiti funkciju `pow` iz `math.h` biblioteke.*

Izmenila bih da numeracija cifara pocinje od 0, jer se to uklapa sa tezijskim faktorom i nekako je logicnije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom x, c i p: 140 2 2  
|| Rezultat je: 1420
```

[Rešenje 1.19]

**Zadatak 1.20** Razbiti ovaj zadatak na tri zadatka i staviti da idu zajedno uz zadatak sa incima. Napisati program koji:

- unetu dužinu u miljama konvertuje u kilometre ( $1 \text{ mi} = 1.609344 \text{ km}$ )
- unetu težinu u funtama konvertuje u kilograme ( $1 \text{ lb} = 0.45359237 \text{ kg}$ )
- unetu temperaturu u celzijusima konvertuje u farenhajte ( $F = \frac{9 \cdot C}{5} + 32$ )

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite duzinu u miljama: 1.8  
|| Vrednost duzine u kilometrima je: 2.896819  
|| Unesite tezinu u funtama: 10  
|| Vrednost tezine u kilogramima je: 4.535923  
|| Unesite temperaturu u celzjusima: 37.2  
|| Vrednost temperature u farenhajtima je: 98.960007
```

[Rešenje 1.20]

**Zadatak 1.21** Napisati program koji za uneta vremena poletanja i sletanja aviona ispisuje dužinu trajanja leta. Možemo pretpostaviti da su poletanje i sletanje u istom danu.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite vreme poletanja: 8 5 0  
|| Unesite vreme sletanja: 12 41 30  
|| Duzina trajanja leta: 4 h 36 min 30 sec
```

[Rešenje 1.21]

**Zadatak 1.22** Sa standardnog ulaza se učitavaju vrednosti dve realne promenljive. Napisati program koji razmenjuje njihove vrednosti i ispisuje ih na standardni izlaz.

[Rešenje 1.22]

**Zadatak 1.23** Napisati program koji za unete koordinate suprotnih temena pravougaonika (gornje levo i donje desno teme) ispisuje njegov obim i površinu. Pretpostaviti da su stranice pravougaonika paralelene koordinatnim osama.

[Rešenje 1.23]

**Zadatak 1.24** Date su dve celobrojene promenljive  $a$  i  $b$ . Napisati program koji promenljivoj  $a$  dodeljuje njihovu sumu, a promenljivoj  $b$  njihovu razliku. NAPOMENA: *Ne koristiti pomoćne promenljive.*

[Rešenje 1.24]

**Zadatak 1.25** Meni je ovaj zadatak potpuno nejasan, a kako nema rešenje ne umem da ga formulisem kako treba ?? Napisati program koji na standardni izlaz ispisuje sledeći tekst:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Karakter : % { * + = a
Brojevi: 43, -56, 455
```

[Rešenje 1.25]

**Zadatak 1.26** Sa standardnog unosa se unosi prirodan broj  $n$  i cifre  $c_1$  i  $c_2$ . Napisati program ispisuje broj dobijen umetanjem cifara  $c_1$  i  $c_2$  na mesta stotina i hiljada broja  $n$ . NAPOMENA: *Za neke ulazne podatke može se dobiti neočekivan rezultat zbog prekoračenja, što ilustruje test primer broj xx. Meni ovo deluje kao tekst zadatka cije je rešenje dato kod 1.33, možda gresim.*

[Rešenje 1.26]

**Zadatak 1.27** Sve zadatke sa operatorom  $?$  grupisati na kraju ovog poglavlja. Napisati program koji za uneta dva cela broja ispisuje njihov maksimum.

[Rešenje 1.27]

**Zadatak 1.28** Data su tri cela broja  $a, b, c$ . Napisati program koji dodeljuje promenljivoj *rez* vrednost 1 ako važi jedan od sledećih uslova: **Deluje mi da ce resenje sa ? biti ruzno i da je ovo vise zadatak za if. Mozda od ovoga napraviti tri zadatka?**

- a)  $a, b, c$  su različiti brojevi
- b)  $a, b, c$  su parni brojevi
- c)  $a, b, c$  su pozitivni brojevi, ne veći od 100

U suprotnom, promenljivoj *rezultat* dodeliti vrednost 0. Ispisati vrednost promenljive *rezultat* na standardni izlaz.

[Rešenje 1.28]

**Zadatak 1.29** Ne bih stampala da ne jer to bez if-a ne moze lepo da se uradi, tj printf u okviru operatora ? je mnogo ruzan stil programiranja. Zato bih i ovo formulisala kao štampanje vrednosti odgovarajuće promenljive koja ima vrednost 0 ili 1. Isto i u sledećem zadatku. Ili prebaciti zadatke u if. Napisati program koji ispituje da li se tačke  $A(x_1, y_1)$  i  $B(x_2, y_2)$  nalaze u istom kvadrantu i ispisuje odgovor DA ili NE.

[Rešenje 1.29]

**Zadatak 1.30** Ne bih stampala da ne jer to bez if-a ne moze lepo da se uradi, tj printf u okviru operatora ? je mnogo ruzan stil programiranja. Zato bih i ovo formulisala kao štampanje vrednosti odgovarajuće promenljive koja ima vrednost 0 ili 1. Ili prebaciti zadatke u if. Napisati program koji ispituje da li se tačke  $A(x_1, y_1)$ ,  $B(x_2, y_2)$  i  $C(x_3, y_3)$  nalaze na istoj pravoj i ispisuje odgovor DA ili NE.

[Rešenje 1.30]

**Zadatak 1.31** Ovo bih mozda ostavila za if — potrebno je da proverim da li su koordinate ispravno unesene Polje šahovske table se definiše parom prirodnih brojeva ne većih od 8: prvi se odnosi na red, drugi na kolonu. Ako su dati takvi parovi, napisati program koji proverava:

- a) da li su polja  $(k, m)$  i  $(l, n)$  iste boje
- b) da li kraljica sa  $(k, l)$  ugrozava polje  $(m, n)$



c) da li konj sa (k, l) ugrozava polje (m, n)

[Rešenje 1.31]

**Zadatak 1.32** Ovo mi je ok da bude reseno sa ?. Napisati program koji za unete vrednosti promenljivih x i y ispisuje vrednost sledećeg izraza:

$$rez = \frac{\min(x, y) + 0.5}{1 + \max^2(x, y)}$$

.

[Rešenje 1.32]

**Zadatak 1.33** Ovaj zadatak je uredu ukoliko mu se definise odgovarajući tekst, a kao jedan test primer postavi ilustracija prekoracenja. ) Ovo je ilustrativni zadatak kakav Milena ne voli da dolazi na vezbe tako da sumnjam da ce se odrzati ovde :)

[Rešenje 1.33]

**Zadatak 1.34** Napisati program koji za unete realne vrednsoti  $a_{11}$ ,  $a_{12}$ ,  $a_{21}$ ,  $a_{22}$  ispisuje vrednost determinante matrice:

```
a11 a12
a21 a22
```

Umesto verbatim staviti odgovarajući format za prikaz matrice. Pri ispisu vrednost zaokružiti na 4 decimale.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 1 2 3 4
|| -2.0000
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: -1 0 0 1
|| -1.0000
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 1.5 -2 3 4.5
|| 12.7500
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 0.01 0.01 0.5 7
|| 0.0650
```

[Rešenje 1.34]

### 1.1 Rešenja

#### Rešenje 1.1

```
1  /*
   Napisati program koji na standardni izlaz ispisuje tekst "Zdravo
   svete!"
3  */
   #include<stdio.h>
5
   int main()
7  {
   /* printf - funkcija pomocu koje se vrši ispis */
   /* oznaka \n : prelazak u novi red */
9  printf("Zdravo svete!\n");
11
   /* naredne dve naredbe ispisace reci Zdravo i svete u istom redu*/
13  printf("Zdravo ");
   printf("svete \n");
15
   /* naredne dve naredbe ispisace reci Zdravo i svete u posebnim
   redovima */
17  /* jer se u prvoj printf naredbi na kraju oznakom \n prelazi u novi
   red */
19  printf("Zdravo \n");
   printf("svete \n");
21
   return 0;
23 }
```

#### Rešenje 1.2

```
1  /*
   Napisati program koji poziva korisnika da unese dva cela broja sa
   standardnog ulaza,
3  a zatim ispisuje:
   1) unete vrednosti
5  2) njihov zbir
   3) njihovu razliku
7  4) njihov proizvod
   5) ceo deo pri deljenju jednog broja drugim brojem
9  6) ostatak pri deljenju jednog broja drugim brojem
11 */
13 #include<stdio.h>
15 int main()
```

```

{
17  int x;
    int y;
19  int rezultat;

21  printf("Unesi vrednost celobrojne promenljive x:");
    scanf("%d", &x); /* "%d" - specifikator tipa koji treba uneti (%d
        za int)
23                      &x    - adresa promenljive x
                        */

25  printf("Unesi vrednost celobrojne promenljive y:");
27  scanf("%d", &y);

29  /* 1) ispis unetih vrednosti */
    printf("x=%d, y=%d\n", x,y); /* umesto prvog %d bice ispisana
        vrednost promenljive x */
31                      /* umesto drugog %d bice ispisana
        vrednost promenljive y */

33  /* 2) ispis zbira */
    rezultat = x+y; /* dodelimo vrednost promenljivoj rezultat */
35  printf("Zbir je %d\n", rezultat);

37  /* 3) ispis razlike */
    printf("Razlika je %d\n",x-y); /* mozemo ispisivati direktno
        vrednost izraza x-y i bez */
39                      /* njegovog dodeljivanja posebno
        promenljivoj */

41  /* 4) ispis proizvoda */
43  printf("%d*%d=%d\n",x,y,x*y);

45  /* 5) ispis kolicnika */
    rezultat = x/y;
47  printf("celobrojno deljenje: %d/%d=%d\n",x,y,rezultat); /*
        promenljiva rezultat je celobrojna (int) */
                                                /* ona ne
        moze sadrzati realan broj */
49                      /* ukoliko
        je x=7, a y=2, tada ce nakon naredbe */
                                                /*
        rezultat=x/y; promenljiva rezultat imati vrednost 2 */
51                      /* a ne
        2.5 */

53  printf("ostatak pri celobrojnem deljenju: %d %% %d=%d\n",x,y,x%y);
                                                /*
        operator % izracunava ostatak pri celobrojnem deljenju */
55                      /* 7%2 ima
        vrednost 1 (jer je 7=3*2+1) */

```

## 1 Uvodni zadaci

---

```
57     % u naredbi printf pisemo %% */
    return 0;
}
```

### Rešenje 1.3

```
2  /*
   Napisati program koji sa standardnog ulaza ucitava realnu vrednost
   izrazenu
   u incima, konvertuje tu vrednost u centimetre i ispisuje je na
   standardni izlaz
   zaokruzenu na dve decimale.
   */
6  #include <stdio.h>

8  int main()
{
10     float in;
    float cm;

12     printf("Unesi broj inca: ");
14     scanf("%f", &in);          /* "%f" specifikator za unos
    /ispis float promenljivih */

16     cm = in*2.54; /* 1 inch = 2.54 cm */

18     printf("%.2f in = %.2f cm\n", in, cm); /* "%.4f" - ispis realne
    promenljive na 4 decimale */

20     return 0;
}
```

### Rešenje 1.4

```
1  /*
   Napisati program koji sa standardnog ulaza ucitava duzinu
   poluprecnika kruga
   i na standardni izlaz ispisuje njegov obim i povrstinu
   */
5
7  #include <stdio.h>
   #include <math.h> /* biblioteka matematickih funkcija; za prevodjenje
   je neophodno ukljuciti opciju -lm
   npr. gcc primer.c -lm */
9  int main()
{
11     int r;
    float O;
    float A;
```

```
13 float P;
14 printf("Unesi poluprecnik kruga:");
15 scanf("%d", &r);

17 O=2*r*M_PI; /* M_PI - konstanta pi koja se nalazi u biblioteci math
    .h */
18 P=r*r*M_PI;

19 printf("Obim: %f, površina: %f\n",O,P);

21 return 0;
23 }
```

### Rešenje 1.5

```
1  /*
2  Napisati program koji učitava trocifreni broj koji se
3  unosi sa standardnog ulaza i ispisuje njegove cifre na
4  standardni izlaz.
5  */
6  #include <stdio.h>
7  int main()
8  {
9      int x;
10     int cifra_jedinice;
11     int cifra_desetice;
12     int cifra_stotine;

13     printf("Unesi trocifreni broj:");
14     scanf("%d", &x);

15     cifra_jedinice = x%10;
16     cifra_desetice = (x/10)%10;
17     cifra_stotine = x/100;

18     printf("Cifre unetog broja su %d,%d,%d\n", cifra_jedinice,
19         cifra_desetice, cifra_stotine);

20     /*
21      2. nacin, bez uvođenja dodatnih promenljivih cifra_jedinice,
22      cifra_desetice i cifra_stotine:
23     */

24     printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10, x/100)
25     ;
26     /*
27     return 0;
28     */
29 }
```

### Rešenje 1.6

## 1 Uvodni zadaci

---

```
1  /*
2  Napisati program koji ucitava trocifreni broj koji se
3  unosi sa standardnog ulaza i ispisuje broj dobijen obrtanjem
4  njegovih cifara.
5  */
6  #include <stdio.h>
7  int main()
8  {
9      int x;
10     int obrnuto_x;
11
12     int cifra_jedinice;
13     int cifra_desetice;
14     int cifra_stotine;
15
16     printf("Unesi trocifreni broj:");
17     scanf("%d", &x);
18
19     cifra_jedinice = x%10;
20     cifra_desetice = (x/10)%10;
21     cifra_stotine = x/100;
22
23     obrnuto_x = cifra_jedinice*100 + cifra_desetice*10 + cifra_stotine
24     ;
25
26     printf("Obrnuto x: %d\n", obrnuto_x);
27
28     return 0;
29 }
```

### Rešenje 1.7

```
1  /*
2  Napisati program koji za unetu duzinu stranice jednakostranicnog
3  trougla
4  ispisuje njegovu povrstinu.
5  */
6  #include <stdio.h>
7  #include <math.h>
8  int main()
9  {
10     unsigned int a;
11     float P;
12
13     printf("Unesi duzinu stranice jednakostranicnog trougla:");
14     scanf("%d",&a);
15
16     P = (a*a*sqrt(3))/4;
```

```

18     printf("Povrsina jednakostraniceg trougla stranice %d je %f\n",a
    ,P);
    return 0;
20 }

```

### Rešenje 1.8

```

/*
2   Napisati program koji za unetu cenu proizvoda ispisuje najmanji
   broj
   novcanica koje je potrebno izdvojiti da bi se proizvod platio. Na
4   raspolaganju su novcanice od 1000,100,50,10 i 1 dinar. Na primer,
   za unetu cenu 5178, program na standardni izlaz treba da ispise:
6   5178=5*1000+ 1*100 +1*50 +2*10 +8*1
*/
8
#include <stdio.h>
10
int main()
12 {
    int x;
14     printf("Unesi cenu:");
    scanf("%d", &x);
16
    printf("%d=%d*1000+ ", x,x/1000);
18     x=x%1000;
    printf("%d*100 +", x/100);
20     x=x%100;
    printf("%d*50 +",x/50);
22     x=x%50;
    printf("%d*10 +", x/10);
24     x=x%10;
    printf("%d*1\n", x);
26     return 0;
}

```

### Rešenje 1.9

```

1  /*
   Napisati program koji za tri cela broja koja se unose sa standardnog
   ulaza
3  ispisuje njihovu aritmeticku sredinu na standardni izlaz.
   */
5
#include<stdio.h>
7
int main()
9 {
    int a, b, c;

```

## 1 Uvodni zadaci

---

```
11  float as;

13  printf("Unesi tri cela broja:");
    scanf("%d%d%d",&a,&b,&c);

15  as=(a+b+c)/3.0; /* da bismo dobili kolicnik, jedan argument mora da
        bude realan broj */

17  /*
19  moguće je i:
    as=1.0*(a+b+c)/3;
21  ili
    as=((float)(a+b+c))/3;
23  */

25  printf("Aritmeticka sredina unetih brojeva je %f\n", as);
    return 0;
27 }
```

### Rešenje 1.10

```
1  /*
    Napisati program koji poziva korisnika da unese dve celobrojne
    vrednosti,
3   smesta ih u promenljive x i y, zamenjuje vrednosti tih
    promenljivih i
    stampa ih na standardni izlaz.
5  */
#include<stdio.h>
7  int main()
{
9   int x,y;
    int t;
11  printf("Unesi dve celobrojne vrednosti:");
    scanf("%d%d",&x,&y);
13  printf("x=%d, y=%d\n",x,y);
    t=x; /* promenljiva t dobija vrednost promenljive x */
15  x=y; /* promenljiva x dobija vrednost promenljive y */
    y=t; /* promenljiva y dobija vrednost promenljive t */
17  printf("nakon zamene, x=%d, y=%d\n",x,y);
    return 0;
19 }
```

### Rešenje 1.11

```
/* Napisati program koji omogucava korisniku da unese ceo broj, a
   zatim
2  ispisuje njegov kvadrat i kub */
```



```
4 #include <stdio.h>

6 int main(){
    int n;

8     /* Ucitavamo broj */
10    printf("Unesite ceo broj: ");
    scanf("%d", &n);

12    /* Ispisujemo trazene vrednosti */
14    printf("Kvadrat: %d\n", n*n);
    printf("Kub: %d\n", n*n*n);

16    /* Završavamo sa programom */
18    return 0;
}
```

### Rešenje 1.12

```
/* Napisati program koji za unete stranice pravougaonika ispisuje
   njegov obim i
   površinu */
2

4 #include <stdio.h>

6 int main(){
    int a, b;
    int obim, površina;

8     /* Ucitavamo potrebne podatke */
10    printf("Unesite duzine stranica pravougaonika: ");
12    scanf("%d %d", &a, &b);

14    /* Obim */
    obim=2*(a+b);

16    /* Površina */
18    površina=a*b;

20    /* Ispisujemo trazene vrednosti */
22    printf("Obim: %d\n", obim);
    printf("Površina: %d\n", površina);

24    /* Završavamo sa programom */
26    return 0;
}
```

### Rešenje 1.13

## 1 Uvodni zadaci

---

```
/* Napisati program koji za unete duzine stranica trougla, ispisuje
   njegov obim
2 i povrstinu */

4 /* Napomena: prilikom prevodjenja programa treba navesti opciju -lm
   * na primer, gcc trougao.c -lm
6 */

8 #include <stdio.h>
   #include <math.h>

10

12 int main(){
   float a, b, c;
   float obim, s, površina;

14

   /* Ucitavamo potrebne podatke */
16   printf("Unesite duzine stranica trougla: ");
   scanf("%f %f %f", &a, &b, &c);

18

   /* Obim */
20   obim=a+b+c;

22   /* Površina - koristimo Heronov obrazac*/
   s=obim/2;
24   površina=sqrt(s*(s-a)*(s-b)*(s-c));

26   /* Ispisujemo trazene vrednosti */
   printf("Obim: %.2f\n", obim);
28   printf("Površina: %.2f\n", površina);

30   return 0;
}
```

### Rešenje 1.14

```
1 /* Napisati program koji za unete dimenzije sobe u metrima (duzinu,
   sirinu i
   visinu) ispisuje koju povrstinu treba da okreći moler. Uracunati da na
   vrata i
3 prozore otpada oko 20%. Omogućiti i unos cene usluge po kvadratnom
   metru i
   izracunati zaradu koju ostvaruje moler.
5 */

7 #include <stdio.h>

9 int main(){
   int dužina, sirina, visina;
11   int cena;
   float površina_za_krećenje;
```

```

13  float ukupna_cena;

15  /* Ucitavamo duzinu, sirinu i visinu sobe */
printf("Unesite dimenzije sobe: ");
17  scanf("%d %d %d", &duzina, &sirina, &visina);

19  /* Ucitavamo cenu krecenja */
printf("Unesite cenu po kvadratnom metru: ");
21  scanf("%d", &cena);

23  /* Povrsina za krecenje odgovara povrshini kvadra - bez poda jer se
    on ne
kreći */
25  površina_za_krecenje=0.8*(duzina*sirina+2*duzina*visina+2*sirina*
    visina);
    ukupna_cena=povrsina_za_krecenje*cena;

27  /* Ispisujemo trazene podatke */
29  printf("Moler treba da okreći %.2f kvadratna metra\n",
    površina_za_krecenje);
    printf("Cena krecenja je %.2f\n", ukupna_cena);

31  /* Završavamo sa programom */
33  return 0;
}

```

### Rešenje 1.15

```

/* Napisati program koji za unetu kolicinu jabuka u kilogramima i
    unetu cenu
2 po kilogramu ispisuje ukupan iznos koji treba platiti */

4 #include <stdio.h>

6 int main(){
    int kolicina;
8    int cena;

10    /* Ucitavamo potrebne podatke */
printf("Unesite kolicinu jabuka (u kg): ");
12    scanf("%d", &kolicina);

14    printf("Unesite cenu (u dinarima): ");
    scanf("%d", &cena);

16    /* Ispisujemo trazeni iznos */
18    printf("Molimo platite %d dinara.\n", kolicina*cena);

20    /* Završavamo sa programom */
    return 0;
22

```

## 1 Uvodni zadaci

---

```
}
```

### Rešenje 1.16

```
1  /* Napisati program koji pomaze kasirki da obracuna kusur tako sto od
   nje trazi
   da unese cenu artikla, kolicinu artikla i iznos koji je dobila od
   kupca. */
3
4  #include <stdio.h>
5
6  int main(){
7      int cena;
8      int kolicina;
9      int iznos;
10     int kusur;
11
12     /* Ucitavamo potrebne podatke */
13     printf("Unesite redom cenu, kolicinu i iznos: ");
14     scanf("%d %d %d", &cena, &kolicina, &iznos);
15
16     /* Izracunavamo kusur */
17     kusur=iznos - kolicina*cena;
18
19     /* I ispisujemo trazenu vrednost */
20     printf("Kusur je %d dinara.\n", kusur);
21
22     return 0;
23 }
```

### Rešenje 1.17

```
1  /* Napisati program koji prirodnom cetvorocifrenom broju koji se
   unosi sa
   standardnog ulaza:
3
4  a) izracunava proizvod cifara
5  b) izracunava razliku sume krajnjih i srednjih cifara
6  c) izracunava sumu kvadrata cifara
7  d) odredjuje broj koji se dobija ispisom cifara u obrnutom poretku
8  e) odredjuje broj koji se dobija zamenom cifre jedinice i cifre
   stotine
9  */
10
11  #include <stdio.h>
12
13  int main(){
14
15      int n;
```

```
17  int j, d, s, h;
18  int proizvod_cifara, razlika_cifara, suma_kvadrata, broj_obrnutu,
    broj_zamena;

19  /* Ucitavamo vrednost sa ulaza */
20  printf("Unesite cetvorocifreni broj: ");
21  scanf("%d", &n);

22  /* Izdvajamo cifre broja i to redom: j -jedinice, d - desetice, s -
    stotine i
23  h - hiljade */
24  j=n%10;
25  d=(n/10)%10;
26  s=(n/100)%10;
27  h=n/1000;

28  /* Izracunavamo proizvod cifara */
29  proizvod_cifara=j*d*s*h;
30  printf("Proizvod cifara: %d\n", proizvod_cifara);

31  /* Izracunavamo razliku sume krajnjih i srednjih cifara */
32  razlika_cifara=(h+j)-(s+d);
33  printf("Razlika sume krajnjih i srednjih: %d\n", razlika_cifara);

34  /* Izracunavamo sumu kvadrata cifara */
35  suma_kvadrata=j*j+d*d+s*s+h*h;
36  printf("Suma kvadrata cifara: %d\n", suma_kvadrata);

37  /* Odredjujemo broj zapisan istim ciframa ali u obrnutom redosledu
    */
38  broj_obrnutu= j*1000+d*100+s*10+h;
39  printf("Broj u obrnutom poretku: %d\n", broj_obrnutu);

40  /* Odredjujemo broj u kojem su cifra jedinica i cifra stotina
    zamenile mesta
41  */
42  broj_zamena=h*1000+j*100+d*10+s;
43  printf("Broj sa zamenjenom cifrom jedinica i stotina: %d\n",
    broj_zamena);

44  return 0;
45  }
```

Rešenje 1.18

Rešenje 1.19

## 1 Uvodni zadaci

```
1  /* Napisati program koji u datom prirodnom broju x ubacuje cifru c na
   poziciju p
   i rezultat ispisuje na standardni izlaz. Brojevi x, c i p se unose sa
3 standardnog ulaza. Podrazumeva se da je broj p manji od ukupnog broja
   cifara broja i da numeracija cifara pocinje od 1.
5
   Uputstvo: koristiti funkciju pow iz math.h biblioteke za racunanje
   stepena
7 broja. Na primer, pow(10, 2)=100.000.
   S obzirom da funkcija pow vraca double vrednosti, pre upotrebe je
   potrebno
9 konvertovati (kastovati) ove vrednosti u int tip.
11
   Prilikom prevodjenja programa koristiti -lm opciju.
   */
13
15 #include <stdio.h>
   #include <math.h>
17
19 int main(){
   int x, c, p;
   int levo, desno;
   int novo_x;
21
   /* Ucitavamo potrebne vrednosti */
23 printf("Unesite redom x, c i p: ");
   scanf("%d %d %d", &x, &c, &p);
25
   /* Odredjujemo deo broja koji se nalazi desno od pozicije p */
27 desno=x%(int)pow(10, p-1);
29
   /* Odredjujemo deo broja koji se nalazi levo od pozicije p */
   levo=x/(int)pow(10, p-1);
31
   /* Odredjujemo novi broj */
33 novo_x=levo*(int)pow(10, p) +c*(int)pow(10, p-1) + desno;
35
   /* Ispisujemo dobijenu vrednost */
   printf("Rezultat je: %d\n", novo_x);
37
   /* Završavamo sa programom */
39 return 0;
41 }
```

### Rešenje 1.20

```
1  /* Napisati program koji:
   1) unetu duzinu u miljama konvertuje u kilometre (1 mi = 1.609344 km)
3  2) unetu tezinu u funtama konvertuje u kilograme ( 1 lb = 0.45359237
   kg)
```

```

3) unetu temperaturu u celzijusima konvertuje u farenhajte ( $F = 9 * C$ 
   /5 + 32)
5 */
7 #include <stdio.h>
9
11 int main(){
12     float milje;
13     float kilometri;
14
15     printf("Unesite duzinu u miljama: ");
16     scanf("%f", &milje);
17     kilometri=milje*1.609344;
18     printf("Vrednost duzine u kilometrima je: %f\n", kilometri);
19
20     float funte;
21     float kilogrami;
22
23     printf("Unesite tezinu u funtama: ");
24     scanf("%f", &funte);
25     kilogrami=funte*0.45359237;
26     printf("Vrednost tezine u kilogramima je: %f\n", kilogrami);
27
28     float celzjusi;
29     float farenhajti;
30
31     printf("Unesite temperaturu u celzijusima: ");
32     scanf("%f", &celzjusi);
33     farenhajti=9*celzjusi/5+32;
34     printf("Vrednost temperature u farenhajtima je: %f\n", farenhajti);
35
36     return 0;
37 }

```

### Rešenje 1.21

```

/* Napisati program koji ucitava sa standardnog ulaza vreme poletanja
   i vreme
2 sletanja aviona, a potom ispisuje duzinu trajanja leta. Mozemo
   pretpostaviti da
   su poletanje i sletanje u istom danu. */
4
6 #include <stdio.h>
8
10 int main(){
11     int poletanje, poletanje_sat, poletanje_minut, poletanje_sekund;
12     int sletanje, sletanje_sat, sletanje_minut, sletanje_sekund;
13     int duzina, duzina_sat, duzina_minut, duzina_sekund;

```

```
12  printf("Unesite vreme poletanja: ");
14  scanf("%d %d %d", &poletanje_sat, &poletanje_minut, &
    poletanje_sekund);

16
18  printf("Unesite vreme sletanja: ");
    scanf("%d %d %d", &sletanje_sat, &sletanje_minut, &sletanje_sekund)
    ;

20  /* Pretvoricemo i vreme poletanja i vreme sletanja u sekunde */
    poletanje=poletanje_sat*3600+poletanje_minut*60+poletanje_sekund;
22    sletanje=sletanje_sat*3600 + sletanje_minut*60 +sletanje_sekund;

24    /* I izracunati razliku u sekundama */
    duzina=sletanje-poletanje;

26
28    /* Izdvajamo broj sati, broj minuta i broj sekundi */
    duzina_sat=duzina/3600;
    duzina_minut=(duzina%3600)/60;
30    duzina_sekund=(duzina%3600)%60;

32
34    /* I ispisujemo rezultat */
    printf("Duzina trajanja leta je: %d h %d min %d sec\n", duzina_sat,
        duzina_minut, duzina_sekund);

36
38    return 0;
}
```

Rešenje [1.22](#)

Rešenje [1.23](#)

Rešenje [1.24](#)

Rešenje [1.25](#)

Rešenje [1.26](#)

Rešenje [1.27](#)

Rešenje [1.28](#)



## Rešenje 1.29

## Rešenje 1.30

## Rešenje 1.31

## Rešenje 1.32

## Rešenje 1.33

```
1  #include <stdio.h>
   #include <math.h>
3  #include <limits.h>

5  /* u zaglavlju limits.h
   su definisane maksimalne i minimalne
7  vrednosti za svaki tip podataka
   npr. INT_MAX konstanta je najveći ceo
9  broj koji može da se stavi
   u promenljivu tipa int
11 zbog toga za poslednji test primer
   ne dobijamo željeni broj
13 jer je došlo do prekoracenja
   novibroj je veći od INT_MAX
15 */

17 /* test primeri:
   broj: 140
19 c1: 2
   c2: 3

21 novibroj: 13240
23 -----
   broj: 526
25 c1: 7
   c2: 4

27 novibroj: 54726
29 -----
   broj: 25
31 c1: 9
   c2: 5

33 novibroj: 5925
35 -----
   test primer koji dovodi do prekoracenja, pa zbog toga
37 ne dobijamo željeni rezultat:
```

## 1 Uvodni zadaci

---

```
39 broj: 100000000
41 c1: 5
41 c2: 1

43 novibroj: neocekivan rezultat ---> PREKORACENJE

45 */

47 int main(){
47 int broj,c1,c2,z1,z2;
49 int novibroj;
49 int dostatak1, dostatak2 ;
51 printf("unesi broj: ");
51 scanf("%d", &broj);
53 printf("unesi c1: ");
53 scanf("%d", &c1);
55 printf("unesi c2: ");
55 scanf("%d", &c2);

57 /* najbolje odmah da se kastuje z1 jer se kasnije cesto
59 koristi u racunu pa da ne ponavljamo (int) */
59 // za stotine pozicija je 3 ---> z1 = (int)pow(10,3-1);
61 z1 = (int)pow(10,2);

63 dostatak1 = broj % z1;

65 /*
65 levi ostatak je u stvari ovaj deo --> broj / z1 * z1 * 10
67 inace taj deo moze da se racuna i kao --> (broj - broj % z1) * 10
67 */
69 novibroj = broj / z1 * z1 * 10 + z1 * c1 + dostatak1 ;

71 //sada u novibroj insertujemo cifru c2 na poziciju 4 - za hiljade

73 z2 = (int)pow(10,3);

75 dostatak2 = novibroj % z2;

77 /*
77 levi ostatak je u stvari ovaj deo --> broj / z2 * z2 * 10
79 inace taj deo moze da se racuna i kao --> (broj - broj % z2) * 10
79 */
81 novibroj = novibroj / z2 * z2 * 10 + z2 * c2 + dostatak2 ;

83
85 printf("Novi broj je: %d\n", novibroj);
85 printf("Maksimalna vrednost za int je: %d\n", INT_MAX);

87 return 0;
87 }
```

Rešenje [1.34](#)



## 2

# Kontrola toka

## 2.1 Naredbe grananja

TODO Iz svih resenja pobrisati formulaciju zadatka.

TODO U resenjima gde imena promenljivih nisu deskriptivna treba dodati komentare prilikom deklaracija cemu sluze odgovarajuca imena promenljivih.

TODO Da li pominjati stadndardni ulaz/izlaz? Negde se pominju, negde ne, deluje mi da to opterecuje zadatke, ali bi u svakom slucaju to rebalo da je konzistentno.

**Zadatak 2.1** Napisati program koji za uneto vreme (broj sati iz intervala  $[0, 24)$  i broj minuta iz intervala  $[0, 60)$ ) ispisuje koliko je sati i minuta ostalo do ponoći. **TODO** Dodati u rešenje proveru ispravnosti unetog vremena, tj ako neko unese neispravno vreme.

[Rešenje 2.1]

**Zadatak 2.2** Napisati program koji za uneti ceo broj ispisuje njegovu recipročnu vrednost. **NAPOMENA:** *Voditi računa da program radi ispravno za sve unete vrednosti.* **TODO** U resenje dodati komentar na temu implicitne konverzije kod deljenja

[Rešenje 2.2]

**Zadatak 2.3** Napisati program koji za uneti ceo broj  $x$  ispisuje njegov znak, tj da li je broj jednak nuli, manji od nule ili veći od nule.

[Rešenje 2.3]

**Zadatak 2.4** Napisati program koji za godinu koja se unosi sa standardnog ulaza na standardni izlaz ispisuje da li je prestupna.

[Rešenje 2.4]

**Zadatak 2.5** Napisati program koji za dva cela broja uneta sa standardnog ulaza ispisuje njihov minimum na standardni izlaz.

[Rešenje 2.5]

**Zadatak 2.6** Sa standardnog ulaza se unose cene tri artikla. Ukoliko se najjeftiniji artikal dobija za 1 dinar, napisati program koji izračunava ukupnu cenu, kao i koliko dinara se uštedi zahvaljujući popustu.

[Rešenje 2.6]

**Zadatak 2.7** Napisati program koji za koeficijente kvadratne jednačine, koji se unose sa standardnog ulaza, ispisuje na standardni izlaz koliko realnih rešenja jednačina ima i ako ih ima, ispisuje rešenja jednačine zaokružena na dve decimalne.

[Rešenje 2.7]

**Zadatak 2.8** Napisati program koji za karakter koji učitava jedan karakter i :

- u slučaju da je uneta cifra, ispisuje nju i njen ASCII kod
- u slučaju da je uneto malo slovo, ispisuje njega, njegov ASCII kod, odgovarajuće veliko slovo i njegov ASCII kod
- u slučaju da je uneto veliko slovo, ispisuje njega, njegov ASCII kod, odgovarajuće malo slovo i njegov ASCII kod
- u ostalim slučajevima, ispisuje uneti karakter i njegov ASCII kod

[Rešenje 2.8]

**Zadatak 2.9** Napisati program koji učitava tri cela broja i ispisuje zbir onih unetih brojeva koji su pozitivni.

[Rešenje 2.9]

**Zadatak 2.10** Napisati program koji za realan broj unet sa standardnog ulaza ispisuje njegovu apsolutnu vrednost.

[Rešenje 2.10]

**Zadatak 2.11** Napisati program koji za karakter unet sa standardnog ulaza ispisuje da li je samoglasnik.

[Rešenje 2.11]

**Zadatak 2.12** Napisati program koji za uneti dan i mesec ispisuje godišnje doba kojem pripadaju. NAPOMENA: *Podrazumevati da je unos korektan.*

[Rešenje 2.12]

**Zadatak 2.13** Napisati program koji za uneti četvorocifreni broj proverava da li su njegove cifre uređene rastuće, opadajuće ili nisu uređene i štampa odgovarajuću poruku na standardni izlaz. Voditi računa o nekorektnim unosima.

[Rešenje 2.13]

**Zadatak 2.14** Ovaj zadatak je mnogo tezak.

Sa standardnog ulaza unose se jedan karakter i 8 realnih brojeva koji predstavljaju koordinate četiri tačke:  $A(x_1, y_1)$ ,  $B(x_2, y_2)$ ,  $C(x_3, y_3)$ ,  $D(x_4, y_4)$ . Na osnovu unetog karaktera ispisuje se odgovarajuća poruka na standardni izlaz:

- $k$  - proverava da li su date tačke temena pravougaonika čije su stranice paralelne koordinatnim osama i u slučaju da jesu, ispisuje vrednost obima datog pravougaonika. Možemo podrazumevati da će korisnik koordinate tačaka unositi redom  $A, B, C, D$ , pri čemu  $ABCD$  opisuje pravougaonik čije su stranice  $AB, BC, CD, DA$ , a dijagonale  $AC$  i  $BD$ . Na primer, tačke  $(1, 1), (2, 1), (2, 2), (1, 2)$  čine pravougaonik čije su stranice paralelne koordinatnim osama i čiji je obim 4 a tačke  $(1, 1), (2, 2), (3, 3), (4, 4)$  ne čine pravougaonik.
- $h$  - proverava da li su unete tačke kolinearne i ukoliko jesu, ispisuje jednačinu prave kojoj pripadaju. Na primer, tačke  $(1, 2), (2, 3), (3, 4), (4, 5)$  su kolinearne i pripadaju pravoj  $y = x + 1$ , tačke  $(1, 1), (1, 2), (1, 3), (1, 4)$  su kolinearne i pripadaju pravoj  $x = 1$ , a tačke  $(1, 1), (2, 1), (2, 2), (1, 2)$  nisu kolinearne.
- $j$  - Kramerovim pravilom proverava da li je sistem jednačina  $x_1 * p + x_2 * q = x_4 - x_3$ ,  $y_1 * p + y_2 * q = y_4 - y_3$  određen, neodređen ili nema rešenja, i u slučaju da je određen ispisuje rešenja.

[Rešenje 2.14]

## 2 Kontrola toka

---

**Zadatak 2.15** Napisati program koji za uneti četvorocifreni ceo broj ispisuje njegovu najveću cifru.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 6835  
|| Najveća cifra je: 8
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 238  
|| Greška: Niste uneli četvorocifren broj!
```

[Rešenje 2.15]

**Zadatak 2.16** Broj je Armstrongov ako je jednak zbiru kubova svojih cifara. Napisati program koji za dati trocifren broj proverava da li je Armstrongov.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 153  
|| Broj je Armstrongov.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 111  
|| Broj nije Armstrongov.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 84  
|| Greška: Niste uneli trocifren broj!
```

[Rešenje 2.16]

**Zadatak 2.17** U nizu 12345678910111213....9899 ispisani su redom brojevi od 1 do 99. Napisati program koji za uneti ceo broj  $k$  ( $1 \leq k \leq 189$ ) ispisuje cifru koja se nalazi na  $k$ -toj poziciji datog niza.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite k: 13  
|| Na 13-toj poziciji je broj 1.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite k: 105  
|| Na 105-toj poziciji je broj 7.
```

[Rešenje 2.17]

**Zadatak 2.18** Sa standardnog ulaza se unosi četvorocifreni pozitivan broj. Napisati program koji ispisuje proizvod parnih cifara datog broja. Ukoliko uneti broj nije pozitivna četvorocifrena vrednost ispisati poruku *Greška!*.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 8123  
|| Proizvod parnih cifara: 16
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 3579  
|| Proizvod parnih cifara: 0
```



### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 288  
|| Greska!
```

[Rešenje 2.18]

**Zadatak 2.19** Sa standardnog ulaza se unosi 5 karaktera. Napisati program koji u slučaju da je prvi karakter veliko ili malo slovo *a* ispisuje unete karaktere obrnutim redosledom, a u suprotnom ništa ne ispisuje.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: A u E f h  
|| h f E u A
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: k L M 9 o
```

[Rešenje 2.19]

**Zadatak 2.20** Sa standardnog ulaza se unosi karakter *c*. Napisati program koji:

- a) ako je *c* malo slovo, zamenjuje ga odgovarajućim velikim i ispisuje na standardni izlaz
- b) ako je *c* veliko slovo, zamenjuje ga odgovarajućim malim i ispisuje na standardni izlaz
- c) ako je *c* cifra, ispisuje poruku *cifra*
- d) u ostalim slučajevima, ispisuje karakter *c* između dve zvezdice.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: K  
|| k
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: 8  
|| cifra
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: >  
|| **>
```

[Rešenje 2.20]

## 2 Kontrola toka

---

**Zadatak 2.21** Napisati program koji za unetih 5 karaktera ispisuje koliko je među njima malih slova.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite karaktere: A u E f h  
| Broj malih slova: 3
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite karaktere: k L M 9 o  
| Broj malih slova: 2
```

[Rešenje 2.21]

**Zadatak 2.22** Sa standardnog ulaza se unosi četvorocifren ceo broj. Napisati program koji ispisuje broj koji se dobija kada se unetom broju razmene najmanja i najveća cifra. Ako uneti broj nije četvorocifren ispisati poruku *Greska!*.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 2863  
| Novi broj: 8263
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 247  
| Greska!
```

[Rešenje 2.22]

**Zadatak 2.23** Sa standardnog ulaza se unose tri neoznačena trocifrena broja. Napisati program koji ispisuje broj dobijen spajanjem dva najveća od njih u šestocifren broj. Spajanje izvršiti tako da najveći od trocifrenih brojeva bude na početku šestocifrenog broja. Dobijeni šestocifreni broj ispisati na izlazu. Ako neki od unetih brojeva nije trocifren, ispisati poruku *Greska!*.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite brojeve: 185 247 311  
| Trazeni broj je: 311247
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite brojeve: 865 11 298  
| Greska!
```

[Rešenje 2.23]

**Zadatak 2.24** Sa standardnog ulaza se učitavaju realni koeficijenti  $A$  i  $B$  linearne jednačine  $Ax + B = 0$ . Napisati program koji ispisuje rešenja ove jednačine. Ukoliko jednačina nema rešenja ili ukoliko ima više od jednog rešenja ispisati odgovarajuće poruke.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koeficijente A i B: 2 -5
|| x=2.5
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koeficijente A i B: 0 18.5
|| Jednacina nema resenja.
```

[Rešenje 2.24]

**Zadatak 2.25** Napisati program koji za dva data intervala realne prave (a1, b1) i (a2, b2) određuje:

- dužinu zajedničkog dela ta dva intervala
- najveći interval sadržan u datim intervalima (presek), a ako on ne postoji dati odgovarajuću poruku. (?! zar ovo nije isto što i a?)
- dužinu realne prave koju pokrivaju ta dva intervala
- najmanji interval koji sadrži date intervale

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom a1, b1, a2 i b2: 2 9 4 11
|| Duzina zajednickog dela: 5
|| Presek intervala: [4,9]
|| Zajednicka duzina intervala: 9
|| Najmanji interval: [2, 11]
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom a1, b1, a2 i b2: 1 2 10 13
|| Duzina zajednickog dela: 0
|| Presek intervala: prazan
|| Zajednicka duzina intervala: 4
|| Najmanji interval: [1, 13]
```

[Rešenje 2.25]

**Zadatak 2.26** Data je funkcija  $f(x) = 2 \cdot \cos(x) - x^3$ . Sa standardnog ulaza se unosi realan broj  $x$  i broj  $k$  koje može biti 1, 2 ili 3. Napisati program koji izračunava  $F(k, x) = f(f(f(\dots f(x))))$  gde je funkcija  $f$  primenjena  $k$ -puta.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom x i k: 2.31 2
|| F(2.31, 2)=2557.516602
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom x i k: 12 1
|| F(12, 1)=-1726.312256
```

[Rešenje 2.26]

**Zadatak 2.27** Napisati program koji za uneti redni broj dana u nedelji ispisuje ime dana. U slučaju pogrešnog unosa ispisati odgovarajuću poruku.

## 2 Kontrola toka

---

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 4  
|| U pitanju je: cetvrtak
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 7  
|| U pitanju je: nedelja
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 8  
|| Greska: nedozvoljni unos!
```

[Rešenje 2.27]

**Zadatak 2.28** Sa standardnog ulaza se učitavaju dva cela broja i jedan od karaktera +, -, \*, / ili % koji predstavlja računsku operaciju. Napisati program koji ispisuje vrednost izraza dobijenog primenom ove operacije na date argumente. Koristiti naredbu *switch*. U slučaju pogrešnog unosa ispisati odgovarajuću poruku.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite operator i dva cela broja: - 8 11  
|| Rezultat je: -3
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite operator i dva cela broja: / 14 0  
|| Greska: deljenje nulom nije dozvoljeno!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite operator i dva cela broja: ? 5 7  
|| Greska: nepoznat operator!
```

[Rešenje 2.28]

**Zadatak 2.29** Napisati program koji za uneti datum u formatu *dan.me-sec.godina*. proverava da li je korektan.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 25.11.1983.  
|| Datum je korektan!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 1.17.2004.  
|| Datum nije korektan!
```

[Rešenje 2.29]

**Zadatak 2.30** Napisati program koji za korektno unet datum u formatu *dan.mesec.godina*. ispisuje datum prethodnog dana.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 30.4.2008.  
|| Prethodni datum: 29.4.2008.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 1.12.2005.  
|| Prethodni datum: 30.11.2005.
```

[Rešenje 2.30]

**Zadatak 2.31** Napisati program koji za korektno unet datum u formatu *dan.mesec.godina*. ispisuje datum narednog dana.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 30.4.2008.  
|| Naredni datum: 1.5.2008.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 1.12.2005.  
|| Naredni datum: 2.12.2005.
```

[Rešenje 2.31]

**Zadatak 2.32** Sa standardnog ulaza unosi se jedan karakter. Ako je karakter malo slovo zameniti ga velikim slovom, ako je veliko slovo zameniti malim slovom, ako je cifra ispisati u pitanju je cifra. Ako je bilo koji drugi karakter onda ga ispisati na standardni izlaz.

[Rešenje 2.135]

**Zadatak 2.33** Ovaj zadatak je isti kao 2.22! Kako se brise?! Ima vise test primera nego 2.22. Sa standardnog ulaza se unosi četvorocifren ceo broj. Napisati program koji datom broju razmenjuje najmanju i najveću cifru. Dobijeni broj ispisati na izlaz. Ako broj nije četvorocifren ispisati -1.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 3842  
|| 3248
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -4239  
|| -4932
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 123  
|| -1
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -45678  
|| -1
```

[Rešenje 2.83]

**Zadatak 2.34** Sa standardnog ulaza se unosi 5 karaktera. Napisati program koji ispisuje koliko se puta pojavilo veliko ili malo slovo a.

## 2 Kontrola toka

---

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: aBcAe  
|| 2
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: aa4A_  
|| 3
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: aAaAa  
|| 5
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: B6(vV  
|| 0
```

[Rešenje 2.84]

**Zadatak 2.35** Sa standardnog ulaza se unose 5 karaktera. Napisati program koji ispisuje koliko puta su se pojavile cifre.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: A1cA3  
|| 2
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: 2a45_  
|| 2
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: 43986  
|| 5
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: B6(vV  
|| 0
```

[Rešenje 2.115]

**Zadatak 2.36** Isti kao 2.23 ali ima bolje test primere! Sa standardnog ulaza se unose tri neoznačena trocifrena broja. Spojiti dva najveća u šestocifren broj. Spajanje izvršiti tako da najveći od trocifrenih brojeva bude na početku šestocifrenog broja. Dobijeni šestocifreni broj ispisati na izlazu. Ako neki od unetih brojeva nije trocifren, ispisati -1.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 384 123 245  
|| 384245
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 123 345 5  
|| -1
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 1242 234 324  
|| -1
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 374 23 898  
|| -1
```

[Rešenje 2.36]

**Zadatak 2.37** Korisnik unosi 3 cela broja: (p), (q) i (r). Nakon toga unosi i dva karaktera, koji imaju sledeci smisao:

'k' -logička konjukcija

'd' -logička disjunkcija

'm' -relacija manje

'v' -relacija veće

Nakon toga se računa vrednost izraza (p) op1 (q) op2 (r) i ispisuje rezultat.

[[Rešenje 2.37](#)]

**Zadatak 2.38** Tekst

[[Rešenje 2.38](#)]

**Zadatak 2.39** Tekst

[[Rešenje 2.39](#)]

**Zadatak 2.40** Tekst

[[Rešenje 2.40](#)]

**Zadatak 2.41** Tekst

[[Rešenje 2.41](#)]

## 2.2 Rešenja

**Rešenje 2.1**

```
1  /*
   Napisati program koji za uneto vreme ispisuje koliko je sati i
   minuta ostalo
3  do ponoci.
   */
5  #include<stdio.h>
```

## 2 Kontrola toka

```
7 int main()
{
9     int sati;
    int minuti;
11    int preostali_sati;
    int preostali_minuti;

13    printf("Unesi vreme (broj sati u intervalu [0,24), broj minuta u
        intervalu [0,60]):");
15    scanf("%d%d",&sati,&minuti);

17    preostali_sati = 24-sati-1;
    preostali_minuti = 60-minuti;
19    if (preostali_minuti==60)
    {
21        preostali_sati++;
        preostali_minuti=0;
23    }

25    printf("Do ponoci je ostalo %d sati i %d minuta\n", 24-sati-1, 60-
        minuti);
    return 0;
27 }
```

### Rešenje 2.2

```
1 /*
    Napisati program koji za uneti ceo broj ispisuje njegovu reciprocnu
    vrednost.
3  Ukoliko je uneti broj jednak nuli, ispisati poruku "Nedozvoljeno
    deljenje nulom".
    */

5  #include <stdio.h>

7  int main()
9  {
    int x;
11    float rx;

13    printf("Unesi jedan ceo broj:");
    scanf("%d",&x);

15

17    /*
        obratiti paznju:
        x==0 - relacija jednakosti (da li je vrednost promenljive x
        jednaka nuli)
19    x=0 - naredba dodele (promenljiva x dobija vrednost nula)
    */

21    if (x==0)
```



```

23     printf("Nedozvoljeno deljenje nulom\n");
    else
25     {
        rx = 1.0/x;
27     printf("Recipročna vrednost unetog broja:%f\n",rx);
    }
29     return 0;
31 }

```

### Rešenje 2.3

```

1  #include <stdio.h>
   /*
3  Napisati program koji za uneti ceo broj x ispisuje da li je jednak
   nuli,
   manji od nule ili veci od nule.
   */
5  int main()
7  {
    int x;
9    printf("Unesi ceo broj:");
    scanf("%d",&x);
11
    /*
13    obratiti paznju:
    x==0 - relacija jednakosti (da li je vrednost promenljive x
    jednaka nuli)
15    x=0 - naredba dodele (promenljiva x dobija vrednost nula)
    */
17    if (x==0)
        printf("Broj je jednak nuli\n");
19    else if (x<0)
        printf("Broj je manji od nule\n");
21    else
        printf("Broj je veci od nule\n");
23
    return 0;
25 }

```

### Rešenje 2.4

```

1  /*
   Napisati program koji za godinu koja se unosi sa standardnog ulaza
   na standardni izlaz
3  ispisuje da li je prestupna.
   */
5
   #include <stdio.h>

```

## 2 Kontrola toka

---

```
7  int main()
9  {
    int x;
    printf("Unesi godinu:");
    scanf("%d",&x);

    if ((x%4==0 && x%100!=0) || x%400==0)
        printf("Godina je prestupna\n");
    else
        printf("Godina nije prestupna\n");
    return 0;
19 }
```

### Rešenje 2.5

```
1  /*
   * Napisati program koji za 2 cela broja uneta sa standardnog ulaza
   * ispisuje njihov minimum na standardni izlaz.
   */
5
   #include <stdio.h>
7  int main()
   {
9      int a,b;
      int min1;
      int min2;
      int min3;

      scanf("%d%d",&a,&b);

15

      /* 1. nacin */
      if (a<b)
          min1=a;
      else
          min1=b;

23      printf("Minimum unetih brojeva (1.nacin) je %d\n",min1);

      /* 2. nacin */
      min2 = (a<b) ? a : b;
27      printf("Minimum unetih brojeva (2.nacin) je %d\n",min2);

      /* 3. nacin */
      min3=a;
      if (b<a)
          min3 = b;
33      printf("Minimum unetih brojeva (3.nacin) je %d\n",min3);

35      return 0;
```

}

## Rešenje 2.6

```

2  /*
3  a) Napisati program koji za 3 cela broja uneta sa standardnog ulaza
4  ispisuje njihov minimum na standardni izlaz.
5  b) Neka uneti brojevi predstavljaju cene artikla. Ukoliko se
6  najjeftiniji
7  artikal dobija za 1 dinar, napisati kolika je ukupna cena, kao i
8  koliko
9  dinara se uštedi zahvaljujuci popustu.
10 */
11
12 #include <stdio.h>
13 int main()
14 {
15     int a,b,c;
16     int min;
17     int min1;
18     int min2;
19     int cena_bez_popusta, cena_sa_popustom;
20
21     scanf("%d%d%d",&a,&b,&c);
22
23     if (a<b)
24         if (a<c) /* poredak: a<b,a<c => a,b,c ili a,c,b */
25             min=a;
26         else /* poredak: a<b, a>=c => a<b, c<=a => c,a,b */
27             min=c;
28     else /* b<=a */
29         if (b<c) /* poredak: b<=a,b<c => b,a,c ili b,c,a */
30             min=b;
31         else /* poredak: b<=a, c<=b => c,b,a */
32             min=c;
33
34     printf("Minimum unetih brojeva (1.nacin) je %d\n",min);
35
36     /* 2. nacin */
37     /* najpre odredimo minimum brojeva a,b */
38     if (a<b)
39         min1=a;
40     else
41         min1=b;
42
43     if (c<min1)
44         min1=c;
45     printf("Minimum unetih brojeva (2.nacin) je %d\n",min1);
46
47     /* 3. nacin */

```

## 2 Kontrola toka

```
46     min2=a;
47     if(min2>b)
48         min2=b;
49     if(min2>c)
50         min2=c;

52     printf("Minimum unetih brojeva (3.nacin) je %d\n",min2);

54     cena_bez_popusta=a+b+c;
55     cena_sa_popustom = cena_bez_popusta - min2 + 1;

56     printf("Cena sa popustom: %.2f\n Cena bez popusta: %d\n Usteda:
57     %.2f\n", cena_sa_popustom, cena_bez_popusta, cena_bez_popusta-
58     cena_sa_popustom);

59     return 0;
60 }
```

### Rešenje 2.7

```
/*
2  Napisati program koji za koeficijente kvadratne jednacine
3  koji se unose sa standardnog ulaza na standardni izlaz
4  ispisuje koliko realnih resenja jednacina ima i ako ih ima, ispisuje
5  resenja jednacine
6  zaokruzena na dve decimale.
7  */
8  #include <stdio.h>
9  #include <math.h>
10 int main()
11 {
12     float a,b,c;
13     float D;
14     float x1,x2;
15     printf("Unesi koeficijente kvadratne jednacine:");
16     scanf("%f%f%f",&a,&b,&c);

17     /* proveravamo da li je kvadratna jednacina korektno zadata */
18     if (a==0)
19         if (b==0)
20             if(c==0) /* slucaj a==0 && b==0 && c==0 */
21                 printf("Jednacina ima beskonacno mnogo resenja\n");
22             else /* slucaj a==0 && b==0 && c!=0 */
23                 printf("Jednacina nema resenja\n");
24         else /* slucaj a==0 && b!=0 */
25         {
26             x1=-c/b;
27             printf("Jednacina ima jedinstveno realno resenje %.2f\n",x1);
28         }
29     else /* slucaj a!=0 */
```

```

30 {
32     D=b*b-4*a*c; /* funkcija sqrt nalazi se u biblioteci math.h (
    prevodjenje sa -lm opcijom) */
34     if (D<0)
        printf("Jednacina nema realnih resenja\n");
36     else if (D>0)
        {
38         x1 = (-b+sqrt(D))/(2*a);
            x2 = (-b-sqrt(D))/(2*a);
            printf("Jednacina ima dva razlicita realna resenja %.2f i %.2
f\n",x1,x2);
40        }
        else
42        {
            x1 = (-b)/(2*a);
44            printf("Jednacina ima jedinstveno realno resenje %.2f\n",x1);
        }
46    }
48    return 0;
}

```

## Rešenje 2.8

```

1  /*
3  Napisati program koji za karakter koji ucitava jedan karakter i :
    - u slucaju da je uneta cifra, ispisuje nju i njen ascii kod
5  - u slucaju da je uneto malo slovo, ispisuje njega, njegov ascii kod,
    odgovarajuce veliko slovo i njegov ascii kod
    - u slucaju da je uneto veliko slovo, ispisuje njega, njegov ascii
    kod, odgovarajuce malo slovo i njegov ascii kod
7  - u ostalim slucajevima, ispisuje uneti karakter i njegov ascii kod
    */
9  #include <stdio.h>
    int main()
11 {
        char c;
13     printf("Unesi jedan karakter:");
        scanf("%c", &c);
15
        if (c>='0' && c<='9')
17         printf("cifra:%c ascii:%d\n",c,c);
        else if (c>='A' && c<='Z')
19         printf("veliko slovo:%c ascii:%d odgovarajuce malo:%c, ascii:%d
\n",c,c,c-'A'+ 'a',c-'A'+ 'a'); /* Razlika izmedju ascii koda
        svakog malog i odgovarajuceg velikog slova
                                     je konstanta koja se moze
        sracunati izrazom 'a'-'A' (i iznosi 32) */

```

## 2 Kontrola toka

```
21     else if (c>='a' && c<='z')
        printf("malo slovo:%c ascii:%d odgovarajuće veliko:%c, ascii:%d
23         \n",c,c,c-'a'+'A',c-'a'+'A');
    else
        printf("karakter:%c ascii:%d\n",c,c);
25     return 0;
27 }
```

### Rešenje 2.9

```
1  /*
3  Napisati program koji učitava tri cela broja i ispisuje zbir onih
   unetih brojeva
   koji su pozitivni.
5
6  */
7  #include<stdio.h>
8  int main()
9  {
10     int a,b,c;
11     int s;
12     printf("Unesi prvi ceo broj:");
13     scanf("%d",&a);
14     printf("Unesi drugi ceo broj:");
15     scanf("%d",&b);
16     printf("Unesi treci ceo broj:");
17     scanf("%d",&c);
18
19     s=0; /* inicijalizujemo promenljivu s na nulu */
20
21     if (a>0)
22         s=s+a; /* naredba dodele: vrednost izraza a desne strane znaka
   jednakosti
23         dodeljujemo promenljivoj sa leve strane znaka
   jednakosti.
   Staru vrednost promenljive s saberemo sa vrednoscu
   promenljive a
25         i dobijenu vrednost upisemo u promenljivu s */
26
27     if (b>0)
28         s+=b; /* operator +=
29         s+=b je skraceni zapis za s=s+b
30         */
31
32     if (c>0)
33         s+=c;
34
35     printf("Suma unetih pozitivnih brojeva: %d\n",s);
36     return 0;
```

37 | }

## Rešenje 2.10

```
1  /*
3  Napisati program koji za realan broj unet sa standardnog ulaza
   ispisuje njegovu apsolutnu vrednost.
5
   */
7
   #include<stdio.h>
9   #include<math.h>
   #include<stdlib.h>
11  int main()
   {
13     float x;
       float y;
15
       printf("Unesi jedan realan broj:");
17     scanf("%f",&x);
19
       /* 1. nacin */
       if (x>0)
21         y=x;
       else
23         y=-x;
       printf("Apsolutna vrednost broja %f je %f\n",x,y);
25
       /* 2. nacin */
       y=x;
27       if (y<0)
29         y=-y;
31
       printf("Apsolutna vrednost broja %f je %f\n",x,y);
33
       /* 3. nacin - pogresan!*/
       y=abs(x); /* funkcija abs vraca ceo broj! za racunanje apsolutne
                  vrednosti realnog broja treba koristiti funkciju fabs */
35         /* funkcija abs se nalazi u zaglavlju stdlib.h */
       printf("Apsolutna vrednost broja %f je %f\n",x,y);
37
       /* 4. nacin */
39       y=fabs(x); /* funkcija fabs se nalazi u zaglavlju math.h */
       printf("Apsolutna vrednost broja %f je %f\n",x,y);
41       return 0;
   }
```

## Rešenje 2.11

## 2 Kontrola toka

---

```
2  /*
   Napisati program koji poziva korisnika da unese jedan karakter i
       ispisuje
4  da li je uneti karakter samoglasnik.
   */
6
   #include <stdio.h>
8
   int main()
10  {
       char c;
12  printf("Unesi jedan karakter:");
       scanf("%c", &c);
14  switch(c)
       {
16      case 'A' :
17      case 'E' :
18      case 'I' :
19      case 'O' :
20      case 'U' :
21      case 'a' :
22      case 'e' :
23      case 'i' :
24      case 'o' :
25      case 'u' : printf("Uneli ste samoglasnik\n");
26                  break;
27      default : printf("Niste uneli samoglasnik\n");
28                  break;
29  }
30
       return 0;
32 }
```

### Rešenje 2.12

```
1  /*
   Napisati program koji za uneti dan i mesec ispisuje godisnje doba kom
3  pripadaju. Mozemo podrazumevati da je unos korektan.
   */
5
   #include <stdio.h>
7
   int main()
9  {
       int d,m;
11  printf("Unesi dan i mesec");
       scanf("%d%d",&d,&m);
13 }
```



```
switch(m) /* argument u naredbi switch mora biti celobrojna
promenljiva */
{
    case 1: /* argument u naredbi case mora biti celobrojna
konstanta */
    case 2: /* ispitujemo da li je m==2 */
        printf("zima\n");
        break;
    case 3:
        if (d<21)
            printf("zima\n");
        else
            printf("prolece\n");
        break;
    case 4:
    case 5:
        printf("prolece\n");
        break;
    case 6:
        if (d<21)
            printf("prolece");
        else
            printf("leto");
        break;
    case 7:
    case 8:
        printf("leto");
        break;
    case 9:
        if (d<23)
            printf("leto\n");
        else
            printf("jesen\n");
        break;
    case 10:
    case 11:
        printf("jesen\n");
        break;
    case 12:
        if (d<22)
            printf("jesen\n");
        else
            printf("zima\n");
    }
    return 0;
}
```

### Rešenje 2.13

```
1 /*
   Napisati program koji od korisnika zahteva da unese
```

## 2 Kontrola toka

---

```
3  cetvorocifreni broj. Program za taj broj proverava
   da li su cifre uredjene rastuce, opadajuce ili nisu
5  uredjene i stampa odgovarajucu poruku na standardni
   izlaz. Voditi racuna o nekorektnim unosima. Na primer,
7  pokretanje programa moze da izgleda ovako:

9  Unesi jedan cetvorocifreni broj: -1357
   Cifre su mu uredjene neopadajuce.

11 ili ovako

13 Unesi jedan cetvorocifreni broj: 9952
   Cifre su mu uredjene nerastuce.

15 ili ovako

17 Unesi jedan cetvorocifreni broj: 9572
   Cifre su mu nisu uredjene.

21 Unesi jedan cetvorocifreni broj: 123
   Uneti broj nije cetvorocifren.

23
25 */

27 #include <stdio.h>
29 #include <stdlib.h>

31 int main()
32 {
33     int x;
34     char c1;    /* cifre su brojevi {0,1,2,3,4,5,6,7,8,9} */
35     char c10;
36     char c100;
37     char c1000;

39     printf("Unesi jedan cetvorocifreni broj:");
40     scanf("%d", &x);

41     x=abs(x); /* u slucaju da je broj negativan, uzimamo njegovu
42                apsolutnu vrednost
43                kako ne bismo za cifre dobili negativne brojeve */
44                /* funkcija abs nalazi se u zaglavlju stdlib.h */

45     if (x<1000 || x>9999)
46         printf("Uneti broj nije cetvorocifren\n");
47     else
48     {
49         c1 = x%10;
50         c10 = (x/10)%10;
51         c100 = (x/100)%10;
52         c1000 = (x/1000)%10;
```

```

55     printf("Cifre broja: %d,%d,%d,%d\n",c1000,c100,c10,c1);
57
58     if (c1000<=c100 && c100<=c10 && c10<=c1)
59         printf("Cifre su uredjene neopadajuće \n");
60     else if (c1000>=c100 && c100>=c10 && c10>=c1)
61         printf("Cifre su uredjene nerastuće \n");
62     else
63         printf("Cifre nisu uredjene\n");
64 }
65 return 0;
66 }

```

### Rešenje 2.14

```

1  /*
2  Sa standardnog ulaza unose se jedan karakter i 8 realnih brojeva koji
   predstavljaju
3  koordinate cetiri tacke: A(x1, y1), B(x2, y2), C(x3, y3), D(x4, y4).
   Na osnovu unetog karaktera
   ispisuje se odgovarajuca poruka na standardni izlaz:
4  k - proverava da li su date tacke temena pravougaonika cije su
   stranice paralelne koordinatnim osama i
   u slucaju da jesu, ispisuje obim datog pravougaonika; mozemo
   podrazumevati da ce korisnik koordinate tacaka
5  unositi redom A,B,C,D, pri cemu ABCD opisuje pravougaonik cije su
   stranice AB,BC,CD i DA, a dijagonale AC i BD
   na primer, tacke (1,1),(2,1),(2,2),(1,2) cine pravougaonik cije
   su stranice paralelne koordinatnim osama i ciji je obim 4
6  a tacke (1,1),(2,2),(3,3),(4,4) ne cine pravougaonik
7  h - proverava da li su unete tacke kolinearne i ukoliko jesu,
   ispisati jednacinu prave kojoj pripadaju
   na primer, tacke (1,2),(2,3),(3,4),(4,5) su kolinearne i
8  pripadaju pravoj y=x+1
   tacke (1,1),(1,2),(1,3),(1,4) su kolinearne i pripadaju pravoj x
   =1
9  a tacke (1,1),(2,1),(2,2),(1,2) nisu kolinearne
10 j - Kramerovim pravilom proverava da li je dati sistem jednacina
11 x1 * p + x2 * q = x4 - x3
   y1 * p + y2 * q = y4 - y3
12 odredjen, neodredjen ili nema resenja, i u slucaju da je odredjen
   ispisati resenja.
   na primer, za unete koordinate (1,1),(1,1),(1,0),(2,2) sistem
   nema resenja
13 za unete koordinate (1,1),(1,1),(1,1),(1,1) sistem je
   neodredjen ili nema resenja
   za unete koordinate (6,1),(8,3),(10,-4),(9,1) sistem
   ima jedinstveno resenje 4.30, 3.10
14 */
15
16
17
18
19
20
21
22
23

```

```
#include<stdio.h>
25 #include<math.h>
int main()
27 {
    char c;
    float x1,y1,x2,y2,x3,y3,x4,y4;
    float kab,kbc,kad;
    float dab,dad;
    float delta, deltap, deltaq;
    float 0;
    float k,n;

    printf("Unesi jedan karakter:");
    scanf("%c",&c);

    printf("Unesi realne koordinate 4 tacke:");
    scanf("%f%f%f%f%f%f%f", &x1,&y1,&x2,&y2,&x3,&y3,&x4,&y4);

    switch (c)
    {
        case 'k':
            if (y1==y2 && y3==y4 && x1==x4 && x2==x3)
            {
                dab = sqrt(pow(x1-x2,2)+pow(y1-y2,2)); // funkcija pow(x
                ,y) racuna vrednost stepene funkcije x^y
                dad = sqrt(pow(x1-x4,2)+pow(y1-y4,2)); // x i y su
                realne vrednosti
                0 = 2*dab + 2*dad;
                printf("Obim pravougaonika je %f\n",0);
            }
            else
            printf("Tacke ne cine pravougaonik sa stranicama koje su
            paralelne koordinatnim osama\n");
            break;
            case 'h':
                if ((x1-x2)!=0) // ukoliko se tacke A(x1,y1) i B(x2,y2) ne
                nalaze na pravoj koja je paralelna x osi
                {
                    k = (y1-y2)/(x1-x2); //izracunamo k,n za pravu odredjenu
                    tackama A(x1,y1) i B(x2,y2)
                    n = y1-k*x1;

                    if (y3==x3*k+n && y4==x4*k+n) // proverimo da li tacke
                    C(x3,y3) i D(x4,y4) nalaze na toj pravoj
                    printf("Tacke su kolinearne, pripadaju pravoj y=%f*x
                    +%f\n",k,n);
                    else
                    printf("Tacke nisu kolinearne\n");
                }
                else // ukoliko se A i B nalaze na pravoj koja je paralelna
                x osi
                if (x3==x1 && x4==x1) // proverimo da li tacke C(x3,y3)
```

```

69     i D(x4,y4) nalaze na toj pravoj
        printf ("Tacke su kolinearne, pripadaju pravoj x=%f\n",x1);
71     else
        printf("Tacke nisu kolinearne\n");
        break;
73     case 'j':
        delta = x1*y2-x2*y1;
        deltap = x2*(y4-y3)-y2*(x4-x3);
75        deltaq = x1*(y4-y3)-y1*(x4-x3);
        if (delta!=0)
77            printf("Sistem ima jedinstveno resenje %.2f, %.2f\n",
deltap/delta, deltaq/delta);
        else if (deltap==0 && deltaq==0)
79            printf("Sistem je neodredjen ili nema resenja.\n");
        else
81            printf("Sistem nema resenja\n");
        break;
83        default:
        printf("Nekorektan unos\n");
85    }
    return 0;
87 }

```

### Rešenje 2.15

```

1  /* Sa standardnog ulaza se unosi ceo cetvorocifren broj. Napisati
   program koji
   ispisuje njegovu najveću cifru na standardni izlaz. */
3
   #include <stdio.h>
5
   int main(){
7       int n, j, d, s, h, max;

9       /* Ucitavamo broj */
       printf("Unesite broj: ");
11      scanf("%d", &n);

13      /* Proveravamo da li se radi o cetvorocifrenom broju */
       if(n<1000 || n>9999){
15          /* Ako broj nije cetvorocifren, prijavljujemo gresku */
          printf("Greska: Niste uneli cetvorocifren broj!\n");
17      }
       else{
19
           /* Ako je broj cetvorocifren, izdvajamo cifre broja:
           j -jedinice, d - desetice, s - stotine i h - hiljade
           */
21          j=n%10;
           d=(n/10)%10;
23

```

## 2 Kontrola toka

---

```
25     s=(n/100)%10;
26     h=n/1000;
27
28     /* Odredjujemo maksimalnu cifru */
29     max=j;
30     if(d>max)
31         max=d;
32     if(s>max)
33         max=s;
34     if(h>max)
35         max=h;
36
37     /* II nacin:
38     * if(j>d && j>s && j>h)
39     *     max=j;
40     * if(d>j && d>s && d>h)
41     *     max=d;
42     * if(s>j && s>d && s>h)
43     *     max=s;
44     * if(h>j && h>d && h>s)
45     *     max=h;
46     */
47
48     /* Ispisujemo rezultat */
49     printf("Najveca cifra je: %d\n", max);
50
51 }
52
53 return 0;
54 }
```

### Rešenje 2.16

```
1  /* Napisati program koji za dati trocifren broj proverava da li je
2     Armstrongov. Broj je Armstrongov ako je jednak zbiru kubova svojih
3     cifara.
4     */
5
6  #include <stdio.h>
7
8  int main(){
9     int n, j, d, s;
10
11     /* Ucitavamo broj */
12     printf("Unesite broj: ");
13     scanf("%d", &n);
14
15     /* Proveravamo da li je broj trocifren */
16     if(n<100 || n>999){
17         /* Ako broj nije trocifren, prijavljujemo gresku */
18         printf("Greska: Niste uneli trocifren broj!\n");
19     }
```

```

19     }
    else{

21         /* Ako je broj trocifre, izdvajamo cifre broja:
           j -jedinice, d - desetice, s - stotine
23         */
        j=n%10;
25         d=(n/10)%10;
        s=n/100;

27         /* Proveravamo da li je broj Armstrongov */
29         if(n==j*j*j+d*d*d+s*s*s){
            printf("Broj je Armstrongov.\n");
31         }
        else{
33             printf("Broj nije Armstrongov.\n");
        }
35     }

37     return 0;
39 }

```

### Rešenje 2.17

```

1  /* Za ceo broj k između 1 i 189 koji se unosi sa standardnog ulaza,
   odrediti
   cifru koja se nalazi na k-toj poziciji
3  niza 12345678910111213....9899 u kom su redom ispisani brojevi od 1
   do 99. */

5

7  #include <stdio.h>

9  int main(){
   int k, n, broj;

11

13     printf("Unesite k: ");
     scanf("%d", &k);

15     if(k<10){
        /* Traži se jednocifren broj */
17         printf("Na %d-toj poziciji je broj %d.\n", k, k);
     }
     else
19         /* Traži se dvocifreni broj */
21         if(k>=10 && k<=189){

23             /* Odredjujemo broj dvocifrenih brojeva koji se mogu zapisati
               pomocu

```

```

    k cifara */
25
    if(k%2!=0){
27        /* Ako je k neparan broj, zapisan je ceo broj dvocifrenih
        brojeva */
        /* 9 oduzimamo jer je 9 broj cifara potrebnih za zapis
        jednocifrenih
29        * brojeva */
        n=(k-9)/2;
31
        /* Broj o kojem se radi je */
33        broj=9+n;

        /* Ujedno, za neparno k se trazi cifra jedinica izdvojenog
        broja */
35        printf("Na %d-toj poziciji je broj %d.\n", k, broj%10);
37    }
    else{
39        /* Ako je k paran broj, zapisan je ceo broj dvocifrenih
        brojeva i
        zapoceto je sa zapisom sledeceg */
41        /* 9 oduzimamo jer je 9 broj cifara potrebnih za zapis
        jednocifrenih
43        * brojeva */
        n=(k-9)/2 +1;
45
        /* Broj o kojem se radi je */
47        broj= 9 + n;

        /* Ujedno, za parno k se trazi cifra desetica izdvojenog
        broja */
49        printf("Na %d-toj poziciji je broj %d.\n", k, broj/10);
51    }
53 }
    else{
55     printf("Greska: Nedozvoljena vrednost broja k!\n");
    }
57
59 return 0;
}
```

### Rešenje 2.18

```

1 /* Sa standardnog ulaza se unosi cetvorocifreni pozitivan broj.
   Napisati program
   koji racuna i ispisuje proizvod parnih cifara datog broja. Ukoliko
   uneti broj
3 nije pozitivna cetvorocifrena vrednost ispisati poruku Greska!. */
```



```
5 #include <stdio.h>

7 int main(){
    int n, j, d, s, h;
9     int broj_parnih, proizvod_parnih;

11     /* Ucitavamo broj */
    printf("Unesite broj: ");
13     scanf("%d", &n);

15     /* Proveravamo da li je unet cetvorocifreni broj */
    if(n<1000 || n>9999){
17         /* Ako nije, prijavljujemo gresku */
        printf("Greska!\n");
19     }
    else{

21         /* Ako jeste: */

23         /* Izdvajamo cifre broja:
25          j -jedinice, d - desetice, s - stotine i h - hiljade
          */
27         j=n%10;
        d=(n/10)%10;
29         s=(n/100)%10;
        h=n/1000;

31         /* Inicijalizujemo broj parnih cifara na 0 */
33         broj_parnih=0;
        /* Postavljamo proizvod parnih cifara na 1 (neutral za mnozenje)
          */
35         proizvod_parnih=1;

37         /* Proveravamo da li je cifra jedinica parna */
        if(j%2==0){
39             proizvod_parnih=proizvod_parnih*j;
            broj_parnih++;
41         }

43         /* Proveravamo da li je cifra desetica parna */
        if(d%2==0){
45             proizvod_parnih=proizvod_parnih*d;
            broj_parnih++;
47         }

49         /* Proveravamo da li je cifra stotina parna */
        if(s%2==0){
51             proizvod_parnih=proizvod_parnih*s;
            broj_parnih++;
53         }

55         /* Proveravamo da li je cifra hiljada parna */
```

```
57     if(h%2==0){
        proizvod_parnih=proizvod_parnih*h;
        broj_parnih++;
59     }

61     /* Proveravamo da li u zapisu broja ima parnih cifara i
        ispisujemo
        rezultat */
63     if(broj_parnih==0){
        printf("Proizvod parnih cifara: 0\n");
65     }
        else{
67         printf("Proizvod parnih cifara: %d\n", proizvod_parnih);
        }
69     }

71     return 0;
73 }
}
```

### Rešenje 2.19

```
1  /* Sa standardnog ulaza unosi se 5 karaktera. Proveriti da li je prvi
    karakter
    veliko ili malo slovo a. Ako jeste, ispisati karaktere obrnutim
3  redosledom, a ako nije, nista ne ispisivati. */

5  #include <stdio.h>

7  int main(){

9      char c1, c2, c3, c4, c5;

11     /* Citamo karaktere */
    /* Obratiti paznju na format učitavanja */
13     printf("Unesite karaktere: ");
    scanf("%c %c %c %c %c", &c1, &c2, &c3, &c4, &c5);

15     /* Proveravamo da li je prvi karakter malo ili veliko slova a */
17     if(c1=='a' || c1=='A'){
        /* I ako jeste, ispusujemo karaktere u obrnutom redosledu */
19         printf("%c %c %c %c %c\n", c5, c4, c3, c2, c1);
        }

21     return 0;
23 }
```

### Rešenje 2.20

```

1  /* Sa standardnog ulaza unosi se jedan karakter. Ako je u pitanju malo
   slovo,
   zameniti ga odgovarajućim velikim slovom i ispisati na standardni
   izlaz. Ako je
3  u pitanju veliko slovo, zameniti ga odgovarajućim malim slovom
   i ispisati ga na standardni izlaz. Ako je u pitanju cifra ispisati
   poruku cifra.
5  Ako je u pitanju bilo koji drugi karakter, onda ga ispisati na
   standardni izlaz
   između dveju zvezdica. */
7
   #include <stdio.h>
9
   int main(){
11
       char c;
13
       /* Citamo karakter */
15       printf("Unesite karakter: ");
       scanf("%c", &c);
17
       /* Proveravamo da li je karakter malo slovo */
19       if(c>='a' && c<='z'){
           /* I ako jeste, ispusujemo odgovarajuće veliko slovo */
21           printf("%c\n", c-'a'+'A');
       }
23       else{
           /* Proveravamo da li je karakter veliko slovo */
25           if(c>='A' && c<='Z'){
               /* I ako jeste, ispusujemo odgovarajuće malo slovo */
27               printf("%c\n", c-'A'+'a');
           }
29           else{
               /* Proveravamo da li je karakter cifra */
31               if(c>='0' && c<='9'){
                   /* I ako jeste, ispusujemo odgovarajuću poruku */
33                   printf("cifra\n");
               }
35               else{
                   /* Inace ispisujemo karakter između dveju zvezdica */
37                   printf("**%c*\n",c);
               }
39           }
       }
41       return 0;
43   }

```

## Rešenje 2.21

```
1  /* Sa standardnog ulaza se unosi 5 karaktera. Ispisati na izlazu broj
   unetih
   malih slova. */
3
   #include <stdio.h>
5
   int main(){
7
       char c1, c2, c3, c4, c5;
9       int broj_malih_slova=0;
11
       /* Citamo karaktere */
13       printf("Unesite karaktere: ");
       scanf("%c %c %c %c %c", &c1, &c2, &c3, &c4, &c5);
15
       /* Proveravamo da li je prvi karakter malo slovo */
17       if(c1>='a' && c1<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
19           broj_malih_slova++;
       }
21
       /* Proveravamo da li je drugi karakter malo slovo */
23       if(c2>='a' && c2<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
25           broj_malih_slova++;
       }
27
       /* Proveravamo da li je treci karakter malo slovo */
29       if(c3>='a' && c3<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
31           broj_malih_slova++;
       }
33
       /* Proveravamo da li je cetvrti karakter malo slovo */
35       if(c4>='a' && c4<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
37           broj_malih_slova++;
       }
39
       /* Proveravamo da li je peti karakter malo slovo */
41       if(c5>='a' && c5<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
43           broj_malih_slova++;
       }
45
       /* Ispisujemo rezultat */
47       printf("Broj malih slova: %d\n", broj_malih_slova);
49
       return 0;
```

51 | }

## Rešenje 2.22

## Rešenje 2.23

## Rešenje 2.24

## Rešenje 2.25

```

/* Napisati program koji za dva data intervala realne prave (a1, b1)
   i (a2, b2)
2 odreduje:
a) duzinu zajednickog dela ta dva intervala
4 b) najveći interval sadržan u datim intervalima (presek), a ako on ne
   postoji
   dati odgovarajuću poruku.
6 c) duzinu realne prave koju pokrivaju ta dva intervala
d) najmanji interval koji sadrži date intervale
8 */

10 #include <stdio.h>
#include <stdlib.h>
12
14 int main() {
16     int a1, b1, a2, b2;
17     int a3, b3;
18     int duzina_zajednickog_dela, zajednicka_duzina;
19     int x, y; // krajevi najmanjeg intervala koji pokriva oba intervala

20     printf("Unesite redom a1, b1, a2 i b2: ");
21     scanf("%d%d%d%d", &a1, &b1, &a2, &b2);

22     /* Presek intervala [a1, b1] i [a2, b2]
23     * racuna se kao:
24     * [a3, b3] = [max{a1,a2}, min{b1, b2}] */

26     a3 = a1 > a2 ? a1 : a2;
27     b3 = b1 < b2 ? b1 : b2;

28     /* U ovom slucaju, presek je prazan */
29     if(a3 >= b3) {
31         duzina_zajednickog_dela = 0;
32         zajednicka_duzina = abs(b1-a1) + abs(b2-a2);
33     }
34     else {
36

```

```
38     duzina_zajednickog_dela = abs(b3-a3);
    zajednicka_duzina = abs(b2-a1);
40 }

42 /* Racunanje "pokrivaca" */
x = a1 < a2 ? a1 : a2;
44 y = b1 > b2 ? b1 : b2;

46 printf("Duzina zajednickog dela: %d\n", duzina_zajednickog_dela);

48 if(a3 >= b3)
    printf("Presek intervala: prazan\n");
50 else
    printf("Presek intervala: [%d, %d]\n", a3, b3);
52
    printf("Zajednicka duzina intervala: %d\n", zajednicka_duzina);
54 printf("Najmanji interval: [%d, %d]\n", x, y);

56 return 0;
}
```

### Rešenje 2.26

```
1 /* Data je funkcija  $f(x) = 2 * \cos(x) - x*x*x$  . Sa standardnog ulaza
   se unosi
   realan broj x i broj k koje moze biti 1, 2 ili 3. Napisati program
   koji
3   izracunava  $F(k, x) = f(f(f(...f(x))))$  gde je funkcija f
   primenjena k-puta.
   */
5
   #include <stdio.h>
7   #include <math.h>
   int main(){
9       float x;
       int k;
11      float F;

13      printf("Unesite redom x i k: ");
       scanf("%f %d", &x, &k);
15

       /* Proveravamo vrednost za k */
17      if(k<1 || k>3){
           printf("Greska: nedozvoljena vrednost za k!\n");
19          return 0;
       }
21      printf("F(%f,%d)=", x, k);

23      /* Analiziramo moguće slučajeve */
       if(k==1){
```

```
25     F=2*cos(x)-x*x*x;
    }
27     else{
        if(k==2){
29             x=2*cos(x)-x*x*x;
            F=2*cos(x)-x*x*x;
31         }
        else{
33             x=2*cos(x)-x*x*x;
            x=2*cos(x)-x*x*x;
35             F=2*cos(x)-x*x*x;
        }
37     }
    }
39
    /* Ispisujemo rezultat */
41    printf("%f\n", F);
43
    return 0;
}
```

### Rešenje 2.27

```
1  /* Napisati program koji za uneti broj n (1 n 7) koji predstavlja
   redni broj
   dana u nedelji ispisuje ime dana. U slucaju pogresnog unosa ispisati
   odgovaraju
3  poruku. */

5  #include <stdio.h>

7  int main(){

9      int dan;

11     printf("Unesite broj: ");
    scanf("%d", &dan);

13     switch(dan){
15         case 1:
            printf("ponedeljak\n");
            break;
17         case 2:
            printf("utorak\n");
            break;
19         case 3:
            printf("sreda\n");
            break;
21         case 4:
            printf("cetvrtak\n");
            break;
23
25     }
```

## 2 Kontrola toka

---

```
27     case 5:
28         printf("petak\n");
29         break;
30     case 6:
31         printf("subota\n");
32         break;
33     case 7:
34         printf("nedelja\n");
35         break;
36     default:
37         printf("Greska: nedozvoljeni unos!\n");
38     }
39
40     return 0;
41 }
```

### Rešenje 2.28

```
1  /* Sa standardnog ulaza se ucitavaju dva cela broja i jedan od
   * karakterima +, -,
   *, / ili % koji predstavlja operaciju koju treba izvesti nad unetim
   brojevima.
3  Napisati program koji koriscenjem switch naredbe analizira o kom
   karakteru je
   rec i na standardni izlaz ispisuje rezultat. U slucaju pogresnog
   unosa ispisati
5  odgovarajuću poruku. */
6
7  #include <stdio.h>
8
9  int main(){
10
11     char op;
12     int x, y;
13
14     printf("Unesite operator i dva cela broja: ");
15     scanf("%c %d %d", &op, &x, &y);
16
17     switch(op){
18         case '+':
19             printf("Rezultat je: %d\n", x+y);
20             break;
21         case '-':
22             printf("Rezultat je: %d\n", x-y);
23             break;
24         case '*':
25             printf("Rezultat je: %d\n", x*y);
26             break;
27         case '/':
28             if(y==0)
29                 printf("Greska: deljenje nulom nije dozvoljeno!\n");
```



```

    else
31         printf("Rezultat je: %f\n", x*1.0/y);
        break;
33     case '%':
        printf("Rezultat je: %d\n", x%y);
35         break;
        default:
37         printf("Greska: nepoznat operator!\n");
    }
39
    return 0;
41 }

```

### Rešenje 2.29

```

1  /* Napisati program koji za uneti datum u formatu dan.mesec.godina.
   proverava da
   li je korektan. */
3
   #include <stdio.h>
5
   int main(){
7       int dan, mesec, godina, dozvoljen_broj_dana;

9       /* Citamo datum */
       printf("Unesite datum: ");
11      scanf("%d.%d.%d", &dan, &mesec, &godina);

13      /* Proveravamo godinu */
       if(godina<0){
15          printf("Datum nije korektan (neispravna godina)!\n");
          return 0;
17      }

19      /* Proveravamo mesec */
       if(mesec<1 || mesec>12){
21          printf("Datum nije korektan (neispravan mesec)!\n");
          return 0;
23      }

25      /* Ako je mesec korektan, proveravamo broj dana */
       switch(mesec){
27         case 1:
           case 3:
29         case 5:
           case 7:
31         case 8:
           case 10:
33         case 12:
           /* Dozvoljeni broj dana za januar, mart, maj, jul, avgust,
35            * oktobar i decembar je 31 */

```

## 2 Kontrola toka

```
        dovoljen_broj_dana=31;
37     break;
    case 2:
39     /* Proveravamo da li je godina prestupna */
        if(godina%4==0 && godina%100!=0 || godina%400==0)
41         /* Ako jeste, dovoljeni broj dana za februar je 29 */
            dovoljen_broj_dana=29;
43     else
        /* Ako nije, dovoljeni broj dana za februar je 28 */
45         dovoljen_broj_dana=28;
        break;
47     case 4:
49     case 6:
49     case 9:
49     case 11:
51     /* Dozvoljeni broj dana za april, jun, septembar i novembar
je 30 */
        dovoljen_broj_dana=30;
53     break;
    }
55     /* Proveravamo dan */
    if(dan<0 || dan>dovoljen_broj_dana){
57         printf("Datum nije korektan (neispravan dan)!\n");
        return 0;
59     }

61     /* Sve provere su ispunjene pa zakljucujemo da je datum korektan
*/
    printf("Ispravan datum!\n");

63

65     return 0;
}
```

### Rešenje 2.30

```
1  /* Napisati program koji za korektno unet datum u formatu dan.mesec.
   godina.
   ispisuje datum prethodnog dana. */
3
   #include <stdio.h>
5
   int main(){
7       int dan, mesec, godina;
       int prethodni_dan, prethodni_mesec, prethodni_godina;
9
       /* Citamo datum */
11      printf("Unesite datum: ");
       scanf("%d.%d.%d", &dan, &mesec, &godina);
13
       /* Racunamo dan, mesec i godinu prethodnog dana */
}
```

```
15     prethodni_dan=dan-1;
16     prethodni_mesec=meseć;
17     prethodni_godina=godina;

19     /* I po potrebi vrsimo korekcije */

21     /* Ako je u pitanju prvi u mesecu */
22     if(prethodni_dan==0){
23         /* Treba korigovati mesec */
24         prethodni_mesec=meseć-1;
25         /* Ako je u pitanju januar */
26         if(prethodni_mesec==0){
27             /* Treba korigovati i godinu */
28             prethodni_mesec=12;
29             prethodni_godina=godina-1;
30         }

31         /* Analiziramo redni broj meseca kako bi odredili tacan dan*/
32         switch(prethodni_mesec){
33             case 1:
34             case 3:
35             case 5:
36             case 7:
37             case 8:
38             case 10:
39             case 12:
40                 prethodni_dan=31;
41                 break;
42             case 2:
43                 if((prethodni_godina%4==0 && prethodni_godina%100!=0) ||
44                    prethodni_godina%400==0)
45                     prethodni_dan=29;
46                 else
47                     prethodni_dan=28;
48                 break;
49             case 4:
50             case 6:
51             case 9:
52             case 11:
53                 prethodni_dan=30;
54         }
55     }

57     /* Ispisujemo datum koji smo izracunali */
58     printf("Prethodni datum: %d.%d.%d\\n",
59           prethodni_dan, prethodni_mesec,prethodni_godina);

61     return 0;
63 }
```

## Rešenje 2.31

### Rešenje 2.135

### Rešenje 2.83

```
1 #include <stdio.h>
3 int main()
4 {
5     int broj;
6     scanf("%d", &broj);
7
8     // Da bismo lakse odredili da li je cetvorocifren
9     int absBroj = broj < 0 ? -broj : broj;
10    if ( absBroj <= 999 || absBroj >= 10000)
11    {
12        printf("-1");
13        return -1;
14    }
15
16    int a = absBroj % 10;
17    int b = (absBroj / 10) % 10;
18    int c = (absBroj / 100) % 10;
19    int d = absBroj / 1000;
20
21    int max = a, min = a;
22    // cuvamo i stepen da bismo lakse zamenili cifre
23    /* Ideja:
24       4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1,
25       ^ ^  odnosno oduzemo 100 i dodamo 900 */
26    int stepenMax = 1, stepenMin = 1;
27
28    if (b > max)
29    {
30        max = b;
31        stepenMax = 10;
32    }
33    if (b < min)
34    {
35        min = b;
36        stepenMin = 10;
37    }
38
39    if (c > max)
40    {
41        max = c;
42        stepenMax = 100;
43    }
44    if (c < min)
45    {
46        min = c;
47        stepenMin = 100;
```

```

    }
49
    if (d > max)
51    {
        max = d;
53        stepenMax = 1000;
    }
55    if (d < min)
    {
57        min = d;
        stepenMin = 1000;
59    }

61
    int rez;
63    /* Ideja:
        4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1,
65        ^ ^  odnosno oduzemo 100 i dodamo 900 */

    if (broj > 0)
67        rez = broj - max*stepenMax + min*stepenMax
69                - min*stepenMin + max*stepenMin;
    else
71        rez = broj + max*stepenMax - min*stepenMax
                + min*stepenMin - max*stepenMin;
73
    printf("%d\n", rez);
75    return 0;
77 }

```

### Rešenje 2.84

```

1  #include <stdio.h>
   #include <ctype.h>
3
   int main()
5   {
       int br_a = 0;
7       if (tolower(getchar()) == 'a')
           br_a++;
9       if (tolower(getchar()) == 'a')
           br_a++;
11      if (tolower(getchar()) == 'a')
           br_a++;
13      if (tolower(getchar()) == 'a')
           br_a++;
15      if (tolower(getchar()) == 'a')
           br_a++;
17
       printf("%d\n", br_a);

```

## 2 Kontrola toka

---

```
19     return 0;
21 }
```

### Rešenje 2.115

```
1  #include <stdio.h>
2  #include <ctype.h>
3
4  int main()
5  {
6      int br_cif = 0;
7      if (isdigit(getchar()))
8          br_cif++;
9      if (isdigit(getchar()))
10         br_cif++;
11     if (isdigit(getchar()))
12         br_cif++;
13     if (isdigit(getchar()))
14         br_cif++;
15     if (isdigit(getchar()))
16         br_cif++;
17
18     printf("%d\n", br_cif);
19
20     return 0;
21 }
```

### Rešenje 2.36

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int br1, br2, br3;
6      scanf("%d%d%d", &br1, &br2, &br3);
7
8      if (br1 > 999 || br1 < 100 || br2 > 999 || br2 < 100
9          || br3 > 999 || br3 < 100)
10     {
11         printf("-1");
12         return -1;
13     }
14
15     int max1 = br1;
16     if (br2 > max1)
17         max1 = br2;
18     if (br3 > max1)
19         max1 = br3;
```

```

21  /* Ako je br1 vec najveci, onda pretragu
    za sledecim najvecim krecemo od br2 */
23  int max2 = br1 != max1 ? br1 : br2;
    if (br1 > max2 && br1 != max1)
25      max2 = br1;
    if (br2 > max2 && br2 != max1)
27      max2 = br2;
    if (br3 > max2 && br3 != max1)
29      max2 = br3;

31  int rez = max1*1000 + max2;
    printf("%d\n",rez);
33
35  return 0;
}

```

## Rešenje 2.37

## Rešenje 2.38

```

#include <stdio.h>
2  #include <ctype.h> // !!!

4  // Upotreba funkcija isalpha, isdigit, toupper, tolower

6  // isalpha( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
    je karakter slovo (malo ili veliko), inace 0
    // isdigit( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
    je karakter cifra, inace 0
8  // isupper( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
    je karakter veliko slovo, inace 0
    // islower( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
    je karakter malo slovo, inace 0
10 // toupper( karakter ) - ukoliko je karakter malo slovo, funkcija
    vraca odgovarajuce veliko slovo,
    //                               inace vraca isti karakter
12 // tolower( karakter ) - ukoliko je karakter veliko slovo, funkcija
    vraca odgovarajuce malo slovo,
    //                               inace vraca isti karakter

14 int main()
16 {
    char c;
    char veliko_slovo;
    char malo_slovo;

20
    printf("Unesite karakter: ");
22    scanf("%c",&c);

```

```
24     if(isalpha(c))
25     {
26         printf("Karakter %c je slovo\n",c);
27
28     if(isupper(c))
29         printf("Veliko slovo\n");
30     else
31         printf("Malo slovo\n");
32
33         veliko_slovo = toupper(c); // malo -> veliko slovo
34         malo_slovo = tolower(c);   // veliko -> malo slovo
35
36         printf("Veliko slovo: %c, malo slovo: %c\n", veliko_slovo,
37             malo_slovo);
38     }
39     else if(isdigit(c))
40         printf("Karakter %c je cifra\n",c);
41     else
42         printf("Karakter %c je znak\n",c);
43
44
45     printf("=====Bez koriscenja funkcija=====\\n");
46
47     // Isti rezultat bez koriscenja ugradjenih funkcija
48
49     if((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z'))
50     {
51         printf("Karakter %c je slovo\n",c);
52
53     if(c >= 'A' && c <= 'Z')
54         printf("Veliko slovo\n");
55     else
56         printf("Malo slovo\n");
57
58         if(c >= 'a' && c <= 'z')
59         {
60             veliko_slovo = c - ('a' - 'A');
61             malo_slovo = c;
62         }
63         else if(c >= 'A' && c <= 'Z')
64         {
65             malo_slovo = c + ('a' - 'A');
66             veliko_slovo = c;
67         }
68     }
69
70     printf("Veliko slovo: %c, malo slovo: %c\\n", veliko_slovo,
71         malo_slovo);
72 }
```



```

74     else if(c >= '0' && c <= '9')
        printf("Karakter %c je cifra\n",c);
76     else
        printf("Karakter %c je znak\n",c);
78
        return 0;
80 }

```

### Rešenje 2.39

```

#include <stdio.h>
2
// Za uneti redni broj dana u nedelji ispisati njegov naziv
4
int main()
6 {
    int broj_dana;
8
    printf("Unesite broj dana: ");
10    scanf("%d",&broj_dana);
12
    switch(broj_dana)
    {
14        case 1: printf("Dan je ponedeljak\n");
                break; // Obavezan izlazak iz case-a!
16        case 2: printf("Dan je utorak\n");
                break; // Obavezan izlazak iz case-a!
18        case 3: printf("Dan je sreda\n");
                break; // Obavezan izlazak iz case-a!
20        case 4: printf("Dan je cetvrtak\n");
                break; // Obavezan izlazak iz case-a!
22        case 5: printf("Dan je petak\n");
                break; // Obavezan izlazak iz case-a!
24        case 6: printf("Dan je subota\n");
                break; // Obavezan izlazak iz case-a!
26        case 7: printf("Dan je nedelja\n");
                break; // Obavezan izlazak iz case-a!
28        default: printf("Lose unet broj!\n"); // Ako ni jedna provera
                ne prolazi
    }
30
    return 0;
32 }

```

### Rešenje 2.40

```

#include <stdio.h>
2 #include <stdlib.h> // Potrebno za exit funkciju

```

```
4 // Za unetu godinu i mesec ispisuje se naziv meseca i koliko dana ima
   u tom mesecu te godine

6 int main()
{
8     int godina;
    int mesec;

10
    int prestupna;

12

14     printf("Unesite godinu: ");
    scanf("%d",&godina);

16

    if(godina < 0)
18     {
        printf("Lose uneta godina!\n");
20         exit(EXIT_FAILURE);
    }

22

    if((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
        // Provera da li je godina prestupna,
24         prestupna = 1;
    // bitno za mesec februar
    else
26         prestupna = 0;

28     printf("Unesite redni broj meseca: ");
    scanf("%d",&mesec);

30

    switch(mesec)
32    {
        case 1: printf("Januar, 31 dan\n");
34                break;
        case 2:
36                if(prestupna)
                    printf("Februar, 29 dana\n");
38                else
                    printf("Februar, 28 dana\n");
40                break;

42        case 3: printf("Mart, 31 dan\n");
                break;
44        case 4: printf("April, 30 dana\n");
                break;
46        case 5: printf("Maj, 31 dan\n");
                break;
48        case 6: printf("Jun, 30 dana\n");
                break;
50        case 7: printf("Jul, 31 dan\n");
                break;
```

```

52     case 8: printf("Avgust, 31 dan\n");
           break;
54     case 9: printf("Septembar, 30 dana\n");
           break;
56     case 10: printf("Oktobar, 31 dan\n");
           break;
58     case 11: printf("Novembar, 30 dana\n");
           break;
60     case 12: printf("Decembar, 31 dan\n");
           break;
62     default: printf("Lose unet redni broj meseca!\n");
}
64
66 return 0;
}

```

### Rešenje 2.41

```

#include <stdio.h>
2
// Za uneti datum određuje ispisuje se naziv godisnjeg doba kome
   datum pripada
4
int main()
6 {
   int godina;
   int mesec;
   int dan;
8
   printf("Unesite datum (DD MM GGGG): ");
   scanf("%d%d%d", &dan, &mesec, &godina);
10
   if(dan < 0 || godina < 0)
       printf("Lose unet datum!\n");
12
   switch(mesec) // Dodati provere za redni broj dana!
   {
       case 1: printf("Zima\n");
               break;
14
       case 2: printf("Zima\n");
               break;
16
       case 3:
               if(dan < 21)
                   printf("Zima\n");
               else
                   printf("Prolece\n");
               break;
18
       case 4: printf("Prolece\n");
               break;
20
       case 5: printf("Prolece\n");
               break;
22
       case 6: printf("Prolece\n");
               break;
24
       case 7: printf("Prolece\n");
               break;
26
       case 8: printf("Prolece\n");
               break;
28
       case 9: printf("Prolece\n");
               break;
30
       case 10: printf("Prolece\n");
               break;
32
       case 11: printf("Prolece\n");
               break;
       case 12: printf("Prolece\n");
               break;
   }
}

```

```

        break;
34
    case 5: printf("Prolece\n");
36        break;

    case 6:
38        if(dan < 21)
40            printf("Prolece\n");
        else
42            printf("Leto\n");
        break;
44

    case 7: printf("Leto\n");
46        break;

    case 8: printf("Leto\n");
48        break;

    case 9:
50        if(dan < 23)
52            printf("Leto\n");
        else
54            printf("Jesen\n");
56        break;

    case 10: printf("Jesen\n");
58        break;

    case 11: printf("Jesen\n");
60        break;

    case 12:
62        if(dan < 22)
64            printf("Jesen\n");
        else
66            printf("Zima\n");
68        break;

    default: printf("Lose unet redni broj meseca!\n");
70
72 }

74 return 0;
}
```

## 2.3 Petlje

### Zadatak 2.42 Tekst

[Rešenje [2.42](#)]

**Zadatak 2.43** Tekst

[Rešenje [2.43](#)]

**Zadatak 2.44** Tekst

[Rešenje [2.44](#)]

**Zadatak 2.45** Tekst

[Rešenje [2.45](#)]

**Zadatak 2.46** Tekst

[Rešenje [2.46](#)]

**Zadatak 2.47** Tekst

[Rešenje [2.47](#)]

**Zadatak 2.48** Tekst

[Rešenje [2.48](#)]

**Zadatak 2.49** Tekst

[Rešenje [2.49](#)]

**Zadatak 2.50** Tekst

[Rešenje [2.50](#)]

**Zadatak 2.51** Tekst

[Rešenje [2.51](#)]

**Zadatak 2.52** Tekst

[Rešenje [2.52](#)]

**Zadatak 2.53** Tekst

[Rešenje [2.53](#)]

**Zadatak 2.54** Tekst

[Rešenje [2.54](#)]

### Zadatak 2.55 Tekst

[Rešenje 2.55]

**Zadatak 2.56** Sa standardnog ulaza unosi se ceo pozitivan broj  $n$ , a potom i  $n$  celih brojeva. Izračunati i ispisati zbir onih brojeva koji su neparni i negativni.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 5  
|| Unesite n brojeva: 1 -5 -6 3 -11  
|| -16
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 4  
|| Unesite n brojeva: -1 1 0 3  
|| -1
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 4  
|| Unesite n brojeva: 5 8 13 17  
|| 0
```

[Rešenje 2.56]

**Zadatak 2.57** Sa standardnog ulaza unosi se realan broj  $m$ , ceo pozitivan broj  $n$  i  $n$  realnih brojeva. Izračunati i ispisati koliko je brojeva među unetima manje od zadatog broja  $m$ .

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj m: 12.37  
|| Unesite broj n: 5  
|| Unesite n brojeva: 11 54.13 -6 13 8  
|| 3
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj m: 2  
|| Unesite broj n: 4  
|| Unesite n brojeva: -1 11 4.32 3  
|| 1
```

[Rešenje 2.57]

**Zadatak 2.58** Sa standardnog ulaza unosi se ceo pozitivan broj  $n$ , a potom i  $n$  karaktera. Za svaki od samoglasnika ispisati koliko puta se pojavio među unetim karakterima. Prilikom implementacije koristiti *switch* naredbu. Ne praviti razliku između malih i velikih slova.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| Unesite n karaktera: u A b a o
|| Samoglasnik a: 2
|| Samoglasnik e: 0
|| Samoglasnik i: 0
|| Samoglasnik o: 1
|| Samoglasnik u: 0

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| Unesite n karaktera: j k + E E a e
|| Samoglasnik a: 1
|| Samoglasnik e: 3
|| Samoglasnik i: 0
|| Samoglasnik o: 0
|| Samoglasnik u: 0

```

[Rešenje 2.58]

**Zadatak 2.59** Sa standardnog ulaza unosi se ceo neoznačen broj. Napisati program koji proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu ili ne.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1857
|| Cifra 5 se nalazi u zapisu!

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 84
|| Cifra 5 se ne nalazi u zapisu!

```

[Rešenje 2.59]

**Zadatak 2.60** Napisati program koji unetom broju uklanja nule sa desne strane. Novodobijeni broj ispisati na standardni izlaz.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 12000
|| 12

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 856
|| 856

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 140
|| 14

```

[Rešenje 2.60]

**Zadatak 2.61** Napisati program koji uneti neoznačeni ceo broj transformiše tako što svaku parnu cifru u zapisu broja uveća za 1.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 2417
|| 3517

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 138
|| 139

```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 59  
|| 59
```

[Rešenje 2.61]

**Zadatak 2.62** Sa standardnog ulaza unosi se neoznačen ceo broj. Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre polaznog broja. Cifre se posmatraju sa desna na levo.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 21854  
|| 284
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 18  
|| 8
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 1  
|| 1
```

[Rešenje 2.62]

**Zadatak 2.63** Sa standardnog ulaza unose se realan broj  $x$  i ceo neoznačen broj  $n$ . Napisati program koji izračunava  $x^n$ .

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom brojeve x i n: 4 3  
|| 64.00000
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom brojeve x i n: 5.8 5  
|| 6563.56768
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom brojeve x i n: 11.43 0  
|| 1.00000
```

[Rešenje 2.63]

**Zadatak 2.64** Sa standardnog ulaza unose se realan broj  $x$  i ceo broj  $n$ . Napisati program koji izračunava  $x^n$ .



*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 2 -3
|| 0.125

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: -3 2
|| 9.000

```

[Rešenje ??]

**Zadatak 2.65** Sa standardnog ulaza unose se realan broj  $x$  i ceo neoznačen broj  $n$ . Napisati program koji izračunava sumu  $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \dots + n \cdot x^n$ .

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 2 3
|| S=34.000000

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 1.5 5
|| S=74.343750

```

[Rešenje 2.65]

**Zadatak 2.66** Sa standardnog ulaza unose se realan broj  $x$  i ceo neoznačen broj  $n$ . Napisati program koji izračunava sumu  $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \dots + \frac{1}{x^n}$ .

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 2 4
|| S=1.937500

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 1.8 6
|| S=2.213249

```

[Rešenje 2.66]

**Zadatak 2.67** Napisati program koji sa zadatom tačnošću izračunava sumu  $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x: 2
|| Unesite tacnost eps: 0.001
|| S=7.388713

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x: 3
|| Unesite tacnost eps: 0.01
|| S=20.079666

```

[Rešenje 2.67]

**Zadatak 2.68** Napisati program koji sa zadatom tačnošću izračunava sumu  $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} \dots$

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x: 3
|| Unesite tacnost eps: 0.001
|| S=0.049997
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x: 3.14
|| Unesite tacnost eps: 0.01
|| S=0.049072
```

[Rešenje 2.68]

**Zadatak 2.69** Sa standardnog ulaza unosi se neoznačen ceo broj. Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem cifara koje su jednake zbiru svojih suseda. Cifre se posmatraju sa desna na levo.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 28631
|| 2631
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 440
|| 40
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 242
|| 22
```

[Rešenje 2.69]

**Zadatak 2.70** Napisati program koji proverava da li je dati prirodan broj palindrom. Broj je palindrom ako se isto čita i sa leve i sa desne strane.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 25452
|| Broj je palindrom!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 895
|| Broj nije palindrom!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 5
|| Broj je palindrom!
```

[Rešenje 2.70]

**Zadatak 2.71** Sa standardnog ulaza se unosi ceo pozitivan broj  $n$ , a zatim i  $n$  celih brojeva. Napisati program koji ispisuje broj sa najvećom cifrom desetica. Ukoliko ima više takvih, ispisati prvi.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| Unesite n brojeva: 18 365 25 1 78
|| 78

```

[Rešenje 2.71]

**Zadatak 2.72** Sa standardnog ulaza se unosi ceo pozitivan broj  $n$ , a zatim i  $n$  celih brojeva. Napisati program koji ispisuje broj sa najvećim brojem cifara. Ukoliko ima više takvih, ispisati prvi.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| Unesite n brojeva: 18 365 25 1 78
|| 365

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| Unesite n brojeva: 3 892 18 21 639 742 85
|| 892

```

[Rešenje 2.72]

**Zadatak 2.73** Sa standardnog ulaza se unosi ceo pozitivan broj  $n$ , a zatim i  $n$  celih brojeva. Napisati program koji ispisuje broj sa najvećom vodećom cifrom. Vodeća cifra je prva cifra iz zapisa broja. Ukoliko ima više takvih, ispisati prvi.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| Unesite n brojeva: 8 964 32 511 27
|| 964

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| Unesite n brojeva: 41 669 8
|| 8

```

[Rešenje 2.73]

**Zadatak 2.74** Sa standardnog ulaza se unose celi pozitivni brojevi  $n$  ( $n > 1$ ) i  $d$ , a zatim i  $n$  celih brojeva. Napisati program koji izračunava koliko ima parova uzastopnih brojeva među unetim brojevima koji se nalaze na rastojanju  $d$ . Rastojanje između brojeva je definisano sa  $d(x, y) = |y - x|$ . Rezultat ispisati na standardni izlaz.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve n i d: 5 2
|| Unesite n brojeva: 2 3 5 1 -1
|| Broj parova: 2

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve n i d: 10 5
|| Unesite n brojeva: -3 6 11 -20 -25 -8 42 37 1 6
|| Broj parova: 4

```

[Rešenje 2.74]

**Zadatak 2.75** Sa standardnog ulaza se unosi ceo broj  $n$ , a zatim i  $n$  karaktera. Napisati program koji proverava da li se od unetih karaktera može napisati reč *Zima*.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 4  
|| Unestite 1. karakter: +  
|| Unestite 2. karakter: o  
|| Unestite 3. karakter: Z  
|| Unestite 4. karakter: j  
|| Ne moze se napisati rec Zima.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 10  
|| Unestite 1. karakter: i  
|| Unestite 2. karakter: 9  
|| Unestite 3. karakter: 0  
|| Unestite 4. karakter: p  
|| Unestite 5. karakter: a  
|| Unestite 6. karakter: Z  
|| Unestite 7. karakter: o  
|| Unestite 8. karakter: m  
|| Unestite 9. karakter: M  
|| Unestite 10. karakter: -  
|| Moze se napisati rec Zima.
```

[Rešenje 2.75]

**Zadatak 2.76** Sa standardnog ulaza se unose celi brojevi sve do unosa broja 0. Napisati program koji izračunava i ispisuje razliku najvećeg i najmanjeg unetog broja.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 8 6 5 2 11 7 0  
|| Razlika: 9
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 8 -1 8 6 0  
|| Razlika: 9
```

[Rešenje 2.76]

**Zadatak 2.77** Sa standardnog ulaza se unose realni brojevi sve do unosa broja 0. Napisati program koji izračunava i ispisuje aritmetičku sredinu unetih brojeva.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 8 5.2 6.11 3 0  
|| Aritmeticka sredina: 5.5775
```

[Rešenje 2.77]

**Zadatak 2.78** Napisati program koji za uneti ceo broj  $n$  iscrtava rub kvadrata dimenzije  $n$ .

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
*****
*   *
*   *
*   *
*   *
*****
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 2
**
**
```

[Rešenje 2.78]

**Zadatak 2.79** Napisati program koji za uneti ceo broj  $n$  i karakter  $c$  iscrtava rub jednakokrako pravouglog trougla čije su katete dužine  $n$ .

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Unesite karakter c: *
*
**
* *
****
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite karakter c: +
+
++
+ +
+ +
++++
```

[Rešenje 2.79]

**Zadatak 2.80** Napisati program koji za uneti ceo broj  $n$  iscrtava *krstiće* dimenzije  $n$ .

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
*  *
*  *
*
*  *
*  *
*  *
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
* *
*
* *
```

[Rešenje 2.80]

**Zadatak 2.81** Napisati program koji za uneti ceo broj  $n$  iscrtava strelice dimenzije  $n$ .

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
*
*
***
*
*
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
*
*
*
*
*****
*
*
*
*
*
```

[Rešenje 2.81]

**Zadatak 2.82** Napisati program koji određuje N-ti član Fibonačijevog niza.

[Rešenje 2.135]

**Zadatak 2.83** Napisati program koji broj N transformiše tako što mu uklanja nule sa desne strane. Napr. 12000 se transformiše u 12.

[Rešenje 2.83]

**Zadatak 2.84** Napisati program koji proverava da li je dati prirodan broj N palindrom.

[Rešenje 2.84]

**Zadatak 2.85** Sa standarnog ulaza unosi se ceo broj n, a potom n realnih brojeva. Odrediti koliko puta je prilikom unosa došlo do promene znaka.

[Rešenje 2.115]

**Zadatak 2.86** Napisati program koji za realno x i prirodan broj n izračunava:

a)  $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \dots + n \cdot x^n$

b)  $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \dots + \frac{1}{x^n}$

c)  $S = (1 + \cos(x)) \cdot (1 + \cos(x^2)) \cdot \dots \cdot (1 + \cos(x^n))$  (čuvanje meurezultata)

d)  $S = 1 - 2 + 3 - 4 + 5 - \dots (-1)^n \cdot n$

[Rešenje 2.115]

**Zadatak 2.87** Za učitani broj  $n$  napisati program koji računa:

$$1 + \frac{1}{2 + \frac{1}{3 + \frac{1}{4 + \dots + \frac{1}{(n-1) + \frac{1}{n}}}}}$$

[Rešenje 2.115]

**Zadatak 2.88** Napisati program koji za zadato  $N$  ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
***
***
***
```

[Rešenje 2.115]

**Zadatak 2.89**

[Rešenje 2.115]

**Zadatak 2.90** Napisati program koji za zadato  $N$  ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
***
**
*
```

[Rešenje 2.115]

**Zadatak 2.91** Napisati program koji za zadato  $N$  ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
*
**
***
```

[Rešenje 2.115]

**Zadatak 2.92** Napisati program koji za zadato N ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
*
**
***
**
*
```

[Rešenje 2.115]

**Zadatak 2.93** Napisati program koji za zadato N ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
*
***
*****
```

[Rešenje 2.115]

**Zadatak 2.94** Napisati program koji za zadato N ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
*
* *
* * *
```

[Rešenje 2.115]

**Zadatak 2.95** Napisati program koji za zadato N ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
*
***
*****
***
*
```



[Rešenje 2.115]

**Zadatak 2.96** Napisati program koji za zadato N ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
  *
 * *
* * *
 * *
  *
```

[Rešenje 2.115]

**Zadatak 2.97** Napisati program koji za zadato N ispisuje:

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 7
 *
 * *
 ***
* * *
*****
* * * *
*****
```

[Rešenje 2.115]

**Zadatak 2.98** Sa standardnog ulaza unosi se neoznačen broj N. Napisati program koji za uneto N iscrtava kvadrat dimenzije N koji na glavnoj dijagonali ima zvezdice.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
*****
**  *
* * *
* **
*****
```

[Rešenje 2.115]

**Zadatak 2.99** Sa standardnog ulaza unose se neoznačeni celi brojevi M i N. Napisati program koji za učitane brojeve M i N ispisuje jedan do drugog N kvadrata čija je svaka strana sastavljena od M zvezdica. Zvezdice su međusobno razdvoje

## 2 Kontrola toka

---

prazninom.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5 3
*****
*       *       *       *
*       *       *       *
*       *       *       *
*       *       *       *
*****
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4 4
*****
*       *       *       *
*       *       *       *
*       *       *       *
*****
```

[Rešenje 2.115]

**Zadatak 2.100** Sa standardnog ulaza unosi se ceo pozitivan broj  $n$ . Pretpostavlja se da je unos ispravan. Napisati program koji stampa zvezdice i minuseve sledeceg oblika za:

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 6
*****
****--****
****--****
****--****
****--****
**-----**
**-----**
**-----**
**-----**
**-----**
**-----**
**-----**
**-----**
**-----**
**-----**
*****
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 2
****
***
****
```

[Rešenje 2.115]

**Zadatak 2.101** Unosi se broj  $N$  ( $N \geq 2$ ). Napisati program koji na standardni izlaz ispisuje sledeću sliku:

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
  *
 * *
*  * *
* * * *
*   *
*   *
* * * *
* * * *
```

[Rešenje 2.115]

**Zadatak 2.102** Napisati program koji ispisuje vrednost funkcije  $\cos(x)$  u 10 ravnomerno razmaknutih tačaka intervala  $[a, b]$  ( $a$  i  $b$  su vrednosti tipa `double`, za koje važi  $a < b$  i učitavaju se sa tastature). Pri ispisu vrednosti se zaokružuju na 4 decimale. Za neispravan unos, program ispisuje broj -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 1 10
|| 0.5403 -0.4161 -0.9900 -0.6536 0.2837 0.9602
|| 0.7539 -0.1455 -0.9111 -0.8391
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 0 28.274
|| 1.0000 -1.0000 1.0000 -1.0000 1.0000 -1.0000
|| 1.0000 -1.0000 1.0000 -1.0000
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 1 -3
|| -1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 0 1
|| 1.0000 0.9938 0.9754 0.9450 0.9028 0.8496 0.7859
|| 0.7125 0.6303 0.5403
```

[Rešenje 2.115]

**Zadatak 2.103** Sa standardnog ulaza unosi se broj  $n$ . Napisati program koji ispisuje brojeve od 1 do  $n$ , zatim od 2 do  $n-1$ , 3 do  $n-2$ , itd. Za neispravan unos, program ispisuje broj -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| 1 2 3 4 5 2 3 4 3
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -4
|| -1
```

*Primer 7*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| 1 2 3 4 5 6 7 2 3 4 5 6 3 4 5 4
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| 1 2 3 2
```

[Rešenje 2.115]

**Zadatak 2.104** Napisati program koji za uneto  $n$  ispisuje „trougao” sačinjen od „koordinata” svojih tačaka.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 1
|| (1, 1)
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 2
|| (1, 2) (2, 2)
|| (1, 1)
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| (1,3) (2,3) (3,3)
|| (1, 2) (2, 2)
|| (1, 1)
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 4
|| (1,4) (2,4) (3,4) (4,4)
|| (1,3) (2,3) (3,3)
|| (1,2) (2,2)
|| (1, 1)
```

[Rešenje 2.115]

**Zadatak 2.105** Napisati C program koji na standardni izlaz ispisuje odgovor da li je uneti prirodan broj deljiv sumom svojih cifara.

[Rešenje 2.115]

**Zadatak 2.106** Napisati C program koji učitava sa standardnog ulaza cele brojeve dok ih je manje od 100 ili dok ne naiđe na nulu. Program treba da ispiše na standardni izlaz broj sa minimalnom poslednjom cifrom. Ako ih ima više neka ispiše bilo koji (ne koristiti nizove).

[Rešenje 2.115]

**Zadatak 2.107** Napisati program koji sledeću sumu računa sa minimalnim brojem operacija

$$1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \dots + (-1)^n \frac{x^{2n}}{(2n)!}$$

Celobrojne vrednosti brojeva  $n$  i  $x$  se učitavaju sa standardnog ulaza.

[Rešenje 2.115]

**Zadatak 2.108** Sa standardnog ulaza unosi se broj. Izbaciti sve one cifre iz broja koje su jednake zbiru svojih suseda.

[Rešenje 2.115]

**Zadatak 2.109**  $A_0$  papir ima površinu  $1m^2$  i odnos stranica  $1 : \sqrt{2}$ .  $A_1$  papir dobija se podelom papira  $A_0$  po dužoj ivici.  $A_2$  papir dobija se podelom  $A_1$  papira po dužoj ivici itd. Napisati program koji za uneto  $k$  ispisuje dimenzije papira  $A_k$  u milimetrima.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 4
|| 297 210
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| 297 420
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| 74 105
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 9
|| 37 52
```

[Rešenje 2.115]

**Zadatak 2.110** Sa standardnog ulaza unosi se ceo pozitivan broj  $n$  veći od 0. Napisati program koji računa sledeću vrednost:

$$S = (1 + \frac{1}{2!})(1 + \frac{1}{3!}) \dots (1 + \frac{1}{n!})$$

U slučaju greške ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| 1.838108
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| 1.841026
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 0
|| -1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 10
|| 1.841077
```

[Rešenje 2.115]

**Zadatak 2.111** Napisati program koji uneti neoznačeni ceo broj transformiše tako što svaku parnu cifru u zapisu broja uveća za 1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 22
|| 33
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 1230
|| 1331
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 0
|| 1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 23456
|| 33557
```

[Rešenje 2.115]

**Zadatak 2.112** Sa standardnog ulaza unosi se ceo pozitivan broj  $N$ , a potom  $N$  celih brojeva. Naći sumu brojeva koji su deljivi sa 5, a nisu deljivi sa 7. U slučaju greške ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5 2 35 5 -175 -20
|| -15
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -3
|| -1
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 10 -5 6 175 -20 -25 -8 42 245 1
|| 6
|| -50
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 6 2205 -1904 2 7 -540 5
|| -535
```

[Rešenje 2.115]

**Zadatak 2.113** Sa standardnog ulaza unosi se ceo pozitivan neparan broj  $n$ . Napisati program koji za uneto  $n$  izračunava:

$$S = 1 \cdot 3 \cdot 5 - 1 \cdot 3 \cdot 5 \cdot 7 + 1 \cdot 3 \cdot 5 \cdot 7 \cdot 9 - 1 \cdot 3 \cdot 5 \cdot 7 \cdot 9 \cdot 11 + \dots (-1)^{\frac{n-1}{2}+1} \cdot 1 \cdot 3 \cdot \dots \cdot n$$

U slučaju greške ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 9
|| 855
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 11
|| -9540
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 20
|| -1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -3
|| -1
```

[Rešenje 2.115]

**Zadatak 2.114** Sa standardnog ulaza unose se realni brojevi  $x$  i  $a$  i ceo pozitivan broj  $n$  veći od 0. Napisati program koji za učitane vrednosti  $x$ ,  $a$  i  $n$  izračunava:

$$((\dots \underbrace{(((x+a)^2 + a)^2 + \dots a)^2}_{n}))$$

U slučaju greške ispisati -1.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3.2 0.2 5
|| 367940960.000000
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 2 1 3
|| 101.000000
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 2.6 0.3 3
|| 76.164085
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5.4 7 -2
|| -1
```

[Rešenje 2.115]

**Zadatak 2.115** Napisati program koji za argument komandne linije  $n$  ispisuje sve brojeve od 1 do  $n$ , zatim svaki drugi broj od 1 do  $n$ , zatim svaki treći broj od 1 do  $n$  itd., završavajući sa svakim  $n$ -tim (tj. samo sa 1). U slučaju greške ispisati -1.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| 1 2 3
|| 1 3
|| 1
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 1
|| 1
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| 1 2 3 4 5 6 7
|| 1 3 5 7
|| 1 4 7
|| 1 5
|| 1 6
|| 1 7
|| 1
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -23
```

[Rešenje 2.115]

## 2.4 Rešenja

### Rešenje 2.42

```
1 /*
   Napisati program koji 10 puta ispisuje tekst "We love C programming".
```

## 2 Kontrola toka

---

```
3  */
5  #include<stdio.h>
7  int main()
8  {
9      int i=0;          /* promenljiva i kontrolise koliko puta ce se petlja
10         izvrsiti */
11     while (i<10) /* pre ulaska u telo petlje proverava se da li je */
12     {
13         /* ispunjen uslov petlje */
14         printf("We love C programming\n");
15         i++; /* operator ++ uvecava i promenljivu za 1
16             ili i+=1;
17             ukoliko ne bismo menjali vrednost promenljive i doslo
18             bi
19             do beskonacne petlje!
20         */
21     }
22     /*
23     brojanje u while petlji smo mogli realizovati i preko uslova:
24
25     i=1;
26     while(i<=10)
27     {
28         ...
29     }
30
31     ili
32
33     i=2;
34     while(i<=11)
35     {
36         ...
37     }
38
39     ili
40
41     i=3;
42     while(i<13)
43     {
44         ...
45     }
46
47     Brojanje pocet od 0 uz koriscenje stroge nejednakosti
48     je u duhu programskog jezika C i zato cemo ovaj nacin
49     brojanja najcesce koristiti
50     */
51     return 0;
52 }
```



## Rešenje 2.43

```
2  /*
4  Napisati program koji poziva korisnika da unese pozitivan ceo broj n
   a potom ispisuje brojeve od 0 do n-1.
6  */
8  #include<stdio.h>
10 int main()
12 {
   int x;
   int n;

   printf("Unesi pozitivan ceo broj:");
14   scanf("%d", &n);
   x=0;
16   while (x<n)
   {
18       printf("%d\n", x);
       x++;
20   }
   return 0;
22 }
```

## Rešenje 2.44

```
2  /*
4  Napisati program koji za uneti pozitivan ceo broj
   izracunava njegov faktorijel. Testirati program
   za razlicite vrednosti promenljive x. Obratiti paznju
   da pocev od 23! dolazi do prekoracenja.
6  */
8  #include<stdio.h>
10 int main()
12 {
   int x;
   unsigned long f;
14   int i;
   int original;
16
   printf("Unesi x>=0:");
18   scanf("%d",&x);

20   original=x;
   f=1;
22   if (x<0)
       printf("Nekorektan unos\n");
```

## 2 Kontrola toka

```
24  else
25  {
26      while (x>1)
27      {
28          f=f*x; /* vrednost izraza sa desne strane naredbe dodele
                        dodeljujemo promenljivoj sa leve strane naredbe
30          dodele
                        */
31          x--; /* operator -- umanjuje vrednost promenljive x za 1
32              naredba x--; ima isti efekat kao x-=1;
33              ili x=x-1;
34              */
35      }
36      printf("%d! = %lu\n",x,f);          /* nekorektno: vrednost
promenljive x je unistena */
37      printf("%d! = %lu\n",original,f); /* korektno: promenljiva
original sadrzi vrednost promenljive x pre ulaska u petlju */
38
39  }
40
41
42  return 0;
43  }
```

### Rešenje 2.45

```
1  /*
2      Napisati program koji poziva korisnika da unese pozitivan ceo broj
3      n,
4      a zatim za unetih n celih brojeva ispisuje sumu pozitivnih i sumu
5      negativnih brojeva.
6
7  */
8
9  #include<stdio.h>
10
11  int main()
12  {
13      int n;
14      int x;
15      int suma_poz;
16      int suma_neg;
17      int i;
18
19      printf("Unesi pozitivan ceo broj:");
20      scanf("%d",&n);
21
22      suma_poz=0; /* promenljivim koje ce sadržati sumu se pre ulaska u
petlju */
23      suma_neg=0; /* dodeljuje se 0 (neutral za sabiranje) */
24      i=0;
```

```

25     while(i<n)
26     {
27         printf("Unesi ceo broj:");
28         scanf("%d", &x);
29
30         if (x<0)
31             suma_neg+=x;
32         else
33             suma_poz+=x;
34
35         i++;
36     }
37
38     printf(" Suma pozitivnih: %d\n Suma negativnih: %d\n", suma_poz,
39           suma_neg);
40     return 0;
41 }

```

### Rešenje 2.46

```

/*
2   Napisati program koji omogućava korisniku da unosi cele brojeve dok
   ne unese nulu. Nakon toga ispisati proizvod onih unetih brojeva
   koji
4   su pozitivni.
*/
6
#include <stdio.h>
8
int main()
9 {
10     int x;
11     int p;
12
13     p=1;
14     while (1) /* izraz 1 je konstantan; razlicit je od nule sto znaci
   da ga tumacimo kao tacnog */
15     {
16         printf("Unesi jedan ceo broj:");
17         scanf("%d", &x);
18         if (x==0) /* ukoliko je uneta nula */
19             break; /* break prekidamo petlju; izvršavanje se nastavlja
   od prve naredbe nakon petlje */
20
21         if (x<0) /* ukoliko je unet negativan broj, tu vrednost ne
   zelimo da pomnozimo sa ukupnim proizvodom p; zato moramo
   nastaviti dalje */
22             continue; /* sa izvršavanjem petlje; continue prekida
   trenutnu iteraciju petlje tako sto preskace sve naredbe
   koje nakon njega slede; izvršavanje se
   nastavlja od provere uslova petlje */

```

## 2 Kontrola toka

---

```
24     p=p*x;
    }
26     printf("Proizvod unetih brojeva je %d\n",p);
28     return 0;
30 }
```

### Rešenje 2.47

```
/*
2   Napisati program koji za uneti ceo broj ispisuje njegove cifre
   u obrnutom poretku.
4 */
6 #include<stdio.h>
7 #include<stdlib.h>
8 int main()
9 {
10     int x;
11     char cifra;
12     printf("Unesi ceo broj:");
13     scanf("%d", &x);
14
15     x = abs(x); /* pretvaranje u apsolutnu vrednost se vrsi za slucaj
   kada je unet
16                 negativan broj kako bismo osigurali da ce nam
   izdvojene cifre
17                 biti pozitivne
18                 */
19
20     while(x>0)
21     {
22         cifra=x%10; /* izdvajamo poslednju cifru broja x */
23         printf("%d\n", cifra);
24         x/=10; /* ako je npr x=1582, x%10 ce biti 2,
   a x/10 ce biti 158;
25                 npr x=5, x%10 ce biti 5
   a x/10 ce biti 0 */
26     }
27
28     return 0;
29 }
30
```

### Rešenje 2.48

```
1 /*
   Napisati program koji omogucava korisniku da unosi karaktere dok ne
   zada tacku i ukoliko je karakter malo slovo,
```

```

3  ispisuje odgovarajuće veliko, ukoliko je karakter veliko slovo
   ispisuje odgovarajuće malo, a u suprotnom ispisuje
   isti karakter kao i uneti.
5  */
7  #include <stdio.h>
9  int main()
10 {
11     int c;
13     /* funkcija getchar ucitava jedan karakter.
       naredbom dodele (c=getchar()) promenljivoj c bice dodeljena
       vrednost
       ascii koda unetog karaktera
       obratiti paznju na zagrade!
15     */
17     while((c=getchar())!='.')
19     {
21         if (c>='A' && c<='Z')
           putchar(c+'a'-'A'); /* Razlika izmedju ascii koda svakog malog
                                i odgovarajuceg velikog slova
                                je konstanta koja se moze sracunati
                                izrazom 'a'-'A' (i iznosi 32) */
23         else if (c>='a' && c<='z')
           putchar(c-'a'+'A');
25         else
           putchar(c);
27     }
29     return 0;
30 }

```

### Rešenje 2.49

```

/*
2  Napisati program koji omogucava korisniku da unosi karaktere dok
   ne zada EOF a potom ispisuje broj velikih slova, broj malih slova
   ,
4  broj cifara, broj belina i zbir cifara.
   */
6
8  #include <stdio.h>
10
12 int main()
13 {
14     /* promenljivoj c dodelicemo povratnu vrednost funkcije getchar()
       funkcija getchar() ucitava jedan karakter sa standardnog ulaza
       i vraca njegov ascii kod; povratna vrednost funkcije getchar je
       int, pa i promenljiva c mora biti tipa int
       */

```

## 2 Kontrola toka

```
16  int c;

18  /* brojacji moraju biti inicijalizovani na 0 */
int br_v=0;
20  int br_m=0;
int br_c=0;
22  int br_b=0;
int br_k=0;
24  int suma=0;

26  while((c=getchar())!=EOF)          /* petlja se završava kada
korisnik ne unese karakter, vec zada konstantu EOF */
{
    kombinacijom tastera CTRL+D. U tom slucaju, getchar() vraca -1*/
28  if (c>='A' && c<='Z')
    br_v++; /* <=> br_v = br_v+1; */
30  else if (c>='a' && c<='z')
    br_m++;
32  else if (c>='0' && c<='9')
    {
34      br_c++;
      suma=suma+c-'0';          /* funkcija getchar() vraca ascii
kod unetog karaktera; ascii kodovi cifara 0,1,...,9
36      su redom 48,49,...,57; Na primer,
za unetu 1
                                promenljiva c ce imati vrednost
49. Zbog toga bi bilo pogresno racunati
38      zbir kao zbir=zbir+c. Promenljivu zbir zato
racunamo kao zbir=zbir+(c-'0')
                                jer c-'0' ce za unetu 0 proizvesti 48-'0' sto je
0,
                                za unetu 1 49-'0' sto je 1, za unetu 2 50-'0' sto
40      je 2, ...*/
    }
42  else if (c=='\t' || c=='\n' || c==' ')
    br_b++;
44
    br_k++;
46 }

48  printf("velika: %d, mala: %d, cifre: %d, beline: %d, svi: %d\n",
    br_v, br_m, br_c, br_b, br_k);
printf("suma cifara: %d\n", suma);
50
    return 0;
52 }
```

### Rešenje 2.50

```
/* Niz prirodnih brojeva formira se na sledeci nacin:
2 an+1 = an/2 ako je an parno
```

```

an+1 = (3*an+1)/2 ako je an neparno
4 Napisati program koji za uneti pocetni clan niza a0 stampa niz
   brojeva sve do prvog clana jednakog
1.
6 */
#include<stdio.h>
8 int main()
{
10     int a0;
    int an,an1;
12
    printf("Unesi pocetni clan niza brojeva:");
14     scanf("%d",&a0);
16
    if (a0>0)
    {
18         printf("%d\n", a0);
20
        an=a0;
        while(an!=1)
22         {
            if (an%2) /* ukoliko je vrednost izraza an%2 razlicita od nule,
24             /* izraz se tumaci kao tacan i izvrsavaju se naredbe
            iz if grane */
                an1=(3*an+1)/2;
26         }
            else /* u suprotnom, ukoliko je vrednost izraza an%2 jednaka
            nuli, izraz */
28         { /* se tumaci kao netacan i izvrsavaju se naredbe iz else
            grane */
                an1=an/2;
30         }
            printf("%d\n",an1);
32         an=an1;
        }
34     }
    else
36         printf("Nekorektan unos\n");
38     return 0;
}

```

### Rešenje 2.51

```

1 /*
   Napisati program koji za uneti ceo broja n ispisuje n puta tekst
3   "We love C programming" koriscenjem while, for i do while petlje.
   Obratiti paznju
   na rezultat kada je n<=0.
5 */

```

```
7  #include <stdio.h>

9  int main()
10 {
11
12     int n,m;
13     int i;

14
15     printf("Unesi ceo broj:");
16     scanf("%d",&n);

17
18     /* 1. nacin - while petlja */
19     printf("while: ");

20
21     i=0;
22     while (i<n)          /* uslov petlje se proverava pre ulaska u
23         telo petlje */
24     {
25         printf("We love C programming\n");
26         i++;
27     }

28     printf("\n");

29
30     /* 2. nacin - for petlja */
31     printf("for: ");

32         /* naredba i=0 se izvsava jednom, pre prve
33         iteracije */
34     for(i=0;i<n;i++)      /* uslov petlje i<=m se proverava pre svake
35         iteracije */
36         printf("We love C programming\n"); /* naredba i++ se izvsava
37         nakon svake iteracije */

38     printf("\n");

39
40     /* 3. nacin - do while petlja */
41     printf("do while: "); /* uslov petlje se proverava na kraju svake
42         iteracije */

43         /* zbog toga se do while petlja izvsava
44         bar jednom, cak i u slucaju */
45         /* da uslov petlje nikada nije ispunjen */

46     i=0;
47     do                    /* petlja se zapocinje bez
48         provere uslova */
49     {
50         printf("We love C programming\n"); /* stampa se dati tekst */
51         i++;                               /* uvecava se vrednost
52         promenljive i */
53     }
54     while(i<n);          /* proverava se uslov i
55         ukoliko je ispunjen, nastavlja se sa sledecom iteracijom */
```



```

49                                     /* u suprotnom, petlja se
    završava i program se nastavlja od prve naredbe koja sledi za
    petljom */
    printf("\n");

51    return 0;

53 }

```

### Rešenje 2.52

```

/*
2   Napisati program koji za uneta dva cela broja n i m ispisuje sve
    cele brojeve
    iz intervala [n,m] koriscenjem while, for i do while petlje.
    Obratiti paznju
4   na rezultat kada je n>m.
*/

6   #include <stdio.h>

8   int main()
10  {

12     int n,m;
    int i;

14     printf("Unesi dva cela broja:");
16     scanf("%d%d",&n,&m);

18     /* 1. nacin - while petlja */
    printf("while: ");

20     i=n;
22     while (i<=m)          /* uslov petlje se proverava pre ulaska u
        telo petlje */
    {
24         printf("%d ", i);
        i++;
26     }

28     printf("\n");

30     /* 2. nacin - for petlja */
    printf("for: ");

32                                     /* naredba i=n se izvsava jednom, pre prve
        iteracije */
    for(i=n;i<=m;i++)          /* uslov petlje i<=m se proverava pre svake
        iteracije */
34         printf("%d ", i);      /* naredba i++ se izvsava nakon svake
        iteracije */

```

```
36     printf("\n");
38     /* 3. nacin - do while petlja */
    printf("do while: "); /* uslov petlje se proverava na kraju svake
        iteracije */
40                                /* zbog toga se do while petlja izvršava
        bar jednom, čak i u slučaju */
                                /* da uslov petlje nikada nije ispunjen */
42     i=n;
    do                                /* petlja se zapocinje bez provere uslova */
44     {
        printf("%d ",i); /* stampa se vrednost promenljive i */
46        i++;            /* uvecava se vrednost promenljive i */
    }
48     while(i<=m);                /* proverava se uslov i ukoliko je ispunjen,
        nastavlja se sa sledecom iteracijom */
                                /* u suprotnom, petlja se završava i program
        se nastavlja od prve naredbe koja sledi za petljom */
50     printf("\n");
    return 0;
52 }
```

### Rešenje 2.53

```
/*
2  Program izracunava minimum n unetih brojeva.
  Npr. za n=4 i brojeve 3 8 2 9 program ispisuje 2
  */
#include <stdio.h>
6  int main()
  {
8      int n, i;
        float x, min;
10
12     printf("Unesi n>0:");
        scanf("%d", &n);
14     if (n<=0)                                /* ako je unos neispravan */
        {
16         printf("Neispravan unos\n");
            return -1;                        /* prekidamo izvršavanje
        programa pomocu naredbe return */
18     }                                /* u slučaju greske kao što je
        neispravan unos vraćamo vrednost -1 */
        printf("Unesi realan broj:");
20     scanf("%f", &x);                        /* prvi broj je unet izvan petlje */
        min=x;                                /* kako bi bio njegova vrednost bila
        dodeljena promenljivoj min */
22                                /* neophodno je da promenljiva min
        bude inicijalizovana pre ulaska u petlju */
    }
```

```

/* da bi uslov x<min mogao da bude
ispitan u prvoj iteraciji */
24 i=0;
while(i<(n-1))
26 {
    printf("Unesi realan broj:");
    scanf("%f", &x);
    if(x<min)
30     min=x;
    i++;
32 }
printf("Minimum je: %f\n", min);
34 return 0;
}

```

## Rešenje 2.54

```

/*
2   a) Napisati program za uneti pozitivan ceo broj n ispisuje zbir
   s = 1+2^3+3^3+...+n^3. Na primer, za n=4, izlaz iz programa
4   treba da bude:
   Suma kubova od 1 do 4 je 100
6   b) Modifikovati program tako da ispisuje zbir s = 1+2^3+3^3+...+k
   ^3
   za svako i od 1 do n. Na primer, za n=4, izlaz iz programa
   treba da
8   bude:
i=1, n=1
10 i=2, n=9
i=3, n=36
12 i=4, n=100

14 */

16 #include <stdio.h>

18 int main()
{
20     int n;
    int i;
22     int s;

24     printf("Unesite jedan pozitivan ceo broj:");
26     scanf("%d", &n);

28     if (n<0)
        return -1;

30     i=1;
32     s=0; /* inicijalizacija promenljive u kojoj se cuva suma kubova */

```

```
34 for(i=1;i<=n;i++)
35 {
36     s+=i*i*i;
37     /* b) */
38     printf("i=%d, s=%d\n", i, s);
39 }
40 /* a) */
41 printf("Suma kubova od 1 do %d: %d\n", n, s);
42 return 0;
43 }
```

### Rešenje 2.55

```
1  /*
2  Napisati program koji ispisuje sve prave delioce unetog pozitivnog
3  celog broja.
4
5  */
6
7  #include<stdio.h>
8  #include<math.h>
9  int main()
10 {
11     int x;
12     int i;
13
14     printf("Unesi x>0:");
15     scanf("%d", &x);
16
17     if (x<=0)
18     {
19         printf("Neispravan unos\n");
20         return -1;
21     }
22
23     /* 1. nacin */
24     printf("----- 1. nacin ----- \n");
25     for(i=2;i<x;i++)
26     {
27         printf("proveravam za %d...\n",i);
28         if (x%i==0)
29             printf("\t delilac:%d \n",i);
30     }
31     /* 2. nacin (brzi) */
32     printf("----- 2. nacin ----- \n");
33     for(i=2;i<=sqrt(x);i++)
34     {
35         printf("proveravam za %d...\n",i);
36         if (x%i==0)
```

```

        if (i==x/i) /* u slucaju kada je delilac koren broja, npr 4
za 16, ispisujemo ga jednom */
37     printf("\t delilac:%d \n",i);
        else /* u suprotnom, npr 2 za 16, ispisujemo i 2 i 8
*/
39     printf("\t delioci:%d %d \n",i,x/i);
    }
41     return 0;
}

```

### Rešenje 2.56

```

/* Sa standardnog ulaza unosi se ceo pozitivan broj n, a potom i n
celih
2 brojeva. Izracunati i ispisati zbir onih brojeva koji su neparni i
negativni. */

4 #include <stdio.h>

6 int main(){
    int n, i, x;
8     int zbir=0;

10     printf("Unesite broj n: ");
    scanf("%d", &n);

12     printf("Unesite n brojeva: ");

14     /* Inicijalizujemo brojac kojim kontrolisemo broj ucitavanja -
* treba da ih bude tacno n
*/
16     i=0;
    while(i<n){
20         /* Ucitavamo broj */
        scanf("%d", &x);

22         /* Proveravamo da li broj negativan i neparan */
        if(x<0 && x%2!=0){
24             /* Ako jeste, dodajemo ga na zbir */
            zbir=zbir+x;
26         }

28         /* Uvecavamo brojac iteracija */
        i++;
30     }

32     /* Ispisujemo rezultat */
    printf("%d\n", zbir);
34     return 0;
36 }

```

### Rešenje 2.57

```
1  /* Sa standardnog ulaza unosi se realan broj m, ceo pozitivan broj n
   i n realnih
   brojeva. Izracunati i ispisati koliko je brojeva medju unetima manje
   od zadatog
3  broja m. */

5  #include <stdio.h>

7  int main(){

9      float m, x;
      int n, i;
11     int broj_brojeva=0;

13     printf("Unesite broj m: ");
     scanf("%f", &m);

15     printf("Unesite broj n: ");
     scanf("%d", &n);

19     printf("Unesite n brojeva: ");
     /* Inicijalizujemo brojac kojim kontrolisemo broj učitavanja -
21     * treba da ih bude tacno n
     */
23     i=0;
     while(i<n){
25         /* Ucitavamo broj */
         scanf("%f", &x);

27         /* Proveravamo da li je broj manji od zadatog broja m */
29         if(x<m){
             /* Ako jeste, uvecavamo brojac brojeva za 1 */
31             broj_brojeva++;
         }

33         /* Uvecavamo brojac iteracija */
35         i++;
     }

37     /* Ispisujemo rezultat */
39     printf("%d\n", broj_brojeva);

41     return 0;
}
```

## Rešenje 2.58

## Rešenje 2.59

```
1  /* Sa standardnog ulaza unosi se ceo neoznaceni broj. Napisati program
   koji
   proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu ili ne
   . */
3
4  #include <stdio.h>
5
6  int main(){
7      int n, cifra;
8      int indikator=0;
9
10     /* Ucitavamo broj */
11     printf("Unesite broj: ");
12     scanf("%d", &n);
13
14     /* Sve dok imamo cifara u zapisu broja */
15     while(n>0){
16
17         /* Izdvajamo poslednju cifru broja */
18         cifra=n%10;
19
20         /* Proveravamo da li je bas ona jednaka broju 5 */
21         if(cifra==5){
22             /* Ako jeste postavljamo indikator na vrednost 1 tako da
23              znamo da smo
24              * pronasli peticu i prekidamo sa izvršavanjem petlje */
25             indikator=1;
26             break;
27         }
28         /* Ako izdvojena cifra nije jednaka broju 5, broj delimo sa 10
29         kako bi
30         mogli da izdvojimo i preostale cifre broja na isti nacin */
31         n=n/10;
32     }
33
34     /* Ispisujemo rezultat */
35     if(indikator==0){
36         printf("Cifra 5 se ne nalazi u zapisu!\n");
37     }
38     else{
39         printf("Cifra 5 se nalazi u zapisu!\n");
40     }
41     return 0;
42 }
```

### Rešenje 2.60

```
1  /* Napisati program koji unetom broju uklanja nule sa desne strane.
   Novodobijeni
   broj ispisati na standardni izlaz. */
3
4  #include <stdio.h>
5
6  int main(){
7      int n;
8
9      /* Ucitavamo broj */
10     printf("Unesite broj: ");
11     scanf("%d", &n);
12
13     if(n==0){
14         printf("0\n");
15     }
16     else{
17         /* Sve dok je poslednja cifra u zapisu broja n nula */
18         while(n%10==0){
19             /* Broj delimo sa 10 tj. uklanjamo mu nulu sa kraja */
20             n=n/10;
21         }
22
23         /* Ispisujemo rezultat */
24         printf("%d\n", n);
25     }
26
27     return 0;
28 }
29
```

### Rešenje 2.61

```
1  #include <stdio.h>
2
3  int main() {
4
5      unsigned int x;
6      int pozicija;    // da li se radi o cifri jedinici, desetici,
7                      // stotini itd...
8      int cifra;      // trenutna izdvojena cifra iz broja x
9      unsigned int y;  // broj dobijen nakon transformacije
10
11     printf("Unesite broj: ");
12     scanf("%d", &x);
13
14     if(x > 0) {
15
16     }
```



```

15      /* Posto pocinjemo sa izdvajanjem cifara od cifre jedinica,
17      postavljamo tezinu (stepen) pozicije na 1 */
      pozicija = 1;
19      y = 0;

21      /* Sve dok imamo cifara u zapisu broja */
      while(x > 0) {

23          /* Izdvajamo poslednju cifru iz zapisa */
25          cifra = x % 10;

27          /* Proveravamo da li je cifra parna */
          if(cifra % 2 == 0){
29              /* I ako jeste, uvecavamo je */
              cifra++;

31          }

33          /* Novi broj formiramo tako sto izdvojenu cifru pomnozimo
          odgovarajucom
35              tezinom (stepenom) pozicije */
          y += cifra*pozicija;

37          /* Pripremamo broj za izdvajanje naredne cifre */
39          x /= 10;

41          /* I uvecavamo tezinu (stepen) pozicije */
          pozicija *= 10;
43      }

45      /* Ispisujemo izracunatu vrednost */
      printf("%d\n", y);
47  }
      else
49      printf("Nekorektan unos.\n");

51  return 0;
}

```

### Rešenje 2.62

```

1  /* Sa standardnog ulaza unosi se neoznaceni ceo broj. Napisati program
   koji
   formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre
   polaznog
3  broja. Cifre se posmatraju sa desna na levo.
   */
5
   #include <stdio.h>
7  #include <math.h>

```

```
9 int main() {
11     unsigned int x;
    int stepen_deset; // da li se radi o cifri jedinici, desetici,
        stotini itd...
13     int cifra;      // trenutna izdvojena cifra iz broja x
    int rbr; // redni broj cifre koju trenutno obradjujemo, gledano s
        desna na levo
15     unsigned int y; // broj dobijen nakon transformacije

17     /* Ucitavamo broj */
    printf("Unesite broj: ");
19     scanf("%d", &x);

21     if(x > 0) {
        /* Postavljamo vrednost stepena na 0 - to znaci da cemo prvo
        mnoziti sa
23         * 10^0=1 */
        stepen_deset = 0;

25         /* Postavljamo vrednost broja koji se formira na 0 */
27         y = 0;
        /* Postavljamo redni broj pozicije na 0 */
29         rbr = 0;

31         /* Sve dok imamo cifara u zapisu broja */
        while(x > 0) {
33             /* Izdvajamo cifru */
35             cifra = x%10;

37             /* Proveravamo da li je pozicija izdvojene cifre parna -
            * cifre na parnim pozicijama zadržavamo
            */
39             if(rbr % 2 == 0) {
41                 /* I ako jeste */

43                 /* Dodajemo izdvojenu cifru novom broju */
                /* Neophodno je izvršiti "kastovanje" tipova, jer je double
                povratni tip
45                 * funkcije pow */
                y += cifra * ((int) pow(10, stepen_deset));

47                 /* Uvecavamo stepen zbog naredne cifre */
49                 stepen_deset++;
            }

51             /* Azuriramo redni broj cifre */
53             rbr++;
            /* I pripremamo broj za naredno izdvajanje */
55             x /= 10;
```

```
    }  
57  
    /* Ispisujemo rezultat */  
59    printf("%d\n", y);  
    }  
61    else  
        printf("Nekorektan unos.\n");  
63  
    return 0;  
65 }
```

### Rešenje 2.63

```
1  /* Sa standradnog ulaza unose se realan broj x i ceo neoznaceni broj n  
   . Napisati  
   program koji izracunava x^n */  
3  
   #include <stdio.h>  
5  
   int main(){  
7  
       int n;  
9       float x;  
       float vrednost;  
11      unsigned exp;  
  
13      /* Ucitavaju se brojevi x i n */  
       printf("Unesite redom brojeve x i n: ");  
15       scanf("%f %d", &x, &n );  
  
17       /* Pocetna vrednost stepena koji se racuna */  
       vrednost=1;  
19  
       for(exp=1; exp<=n; exp++)  
21           vrednost=vrednost*x;  
  
23       /* Stampamo rezultat */  
       printf("%f\n",vrednost);  
25  
       return 0;  
27 }
```

### Rešenje 2.64

```
1  /* Sa standradnog ulaza unose se realan broj x i ceo broj n. Napisati  
   program koji izracunava x^n */  
3  
   #include <stdio.h>  
5
```

```
int main(void){
7
    int n, n_abs;
9    float x;
    float vrednost;
11    unsigned exp;

13    /* Ucitavaju se brojevi x i n */
    printf("Unesite redom brojeve x i n: ");
15    scanf("%f %d", &x, &n);

17    /* Pocetna vrednost stepena koji se racuna */
    vrednost=1;

19    /* Stepenovanje */
    n_abs=abs(n);
21    for(exp=1; exp<=n_abs; exp++)
        vrednost=vrednost*x;

23    /* Stampamo rezultat */
    if(n<0){
25        printf("%.3f\n",1/vrednost);
    }
27    else{
        printf("%.3f\n",vrednost);
29    }
31    return 0;
33 }
}
```

### Rešenje 2.65

```
1 /* Sa standardnog ulaza unose se realan broj x i ceo neoznaceni broj n
   . Napisati
   program koji izracunava sumu  $S = x + 2x^2 + 3x^3 + \dots + nx^n$  */
3
4 #include <stdio.h>
5
6 int main(){
7     unsigned n, i;
8     float x, S, stepen;
9
10    printf("Unesite redom brojeve x i n: ");
11    scanf("%f %u", &x, &n);
12
13    /* Inicijalizujemo sumu koju racunamo */
    S=0;
15
16    /* Stepen promenljiva ce sadrzati vrednosti stepena  $x^n$  -
       * pocetna vrednost joj je 1 */
17    stepen=1;
```

```

19     for(i=1; i<=n; i++){
21         stepen=stepen*x;
           S=S+i*stepen;
23     }

25     printf("S=%f\n", S);

27     return 0;
}

```

### Rešenje 2.66

```

1  /* Sa standardnog ulaza unose se realan broj x i ceo neoznaceni broj n
   *
   * Napisati program koji izracunava sumu  $S=1+1/x+1/x^2+1/x^3+\dots+1/x^n$ 
   */
3
4  #include <stdio.h>
5  int main(){
6      unsigned n, i;
7      float x, S, stepen;

9      printf("Unesite redom brojeve x i n: ");
      scanf("%f %u", &x, &n);

11
      S=1;
      stepen=1;
13     for(i=1; i<=n; i++){
15         stepen=stepen*x;
           S=S+1/stepen;
17     }

19     printf("S=%f\n", S);

21     return 0;
}

```

### Rešenje 2.67

```

1  /* Napisati program koji sa zadatom tacnoscu izracunava sumu
2   $S=1+x+x^2/2!+x^3/3!+\dots+x^n/n! + \dots$  */
   /* Napomena: ovo je razvoj funkcije  $e^x$  */
4
5  #include <stdio.h>
6  #include <math.h>
7  int main(){
8      int n, i, faktorijel;
           float S;

```

## 2 Kontrola toka

```
10     float x, eps, stepen;

12     printf("Unesite x: ");
    scanf("%f", &x);

14     printf("Unesite tacnost eps: ");
    scanf("%f", &eps);

18     /* Tacnost izracunavanja je zadovoljena ako je apsolutna vrednost
20     * razlika suma
22     * u dvema uzastopnim iteracijama manja od zadate tacnosti;
24     * Oдавde se izvodi da apsolutna vrednost opsteg clana sume
26     * mora da bude manja od zadate tacnosti da bi uslov bio ispunjen
28     */

26     S=1;
    faktorijel=1;
    stepen=x;
    i=2;
30     while(fabs(stepen/faktorijel)>eps){
        S=S+stepen/faktorijel;
32        stepen=stepen*x;
        faktorijel=faktorijel*i;
34        i++;
    }

36     printf("S=%f\n", S);

38     return 0;
40 }
```

### Rešenje 2.68

```
/* Napisati program koji sa zadatom tacnosu izracunava sumu
2 S=1-x+x^2/2!-x^3/3!+... */
/* razvoj funkcije sin(x) */

4
6 #include <stdio.h>
7 #include <math.h>
8 int main(){
9     int n, i, faktorijel, znak;
10    float S;
11    float x, eps, stepen;

12    printf("Unesite x: ");
    scanf("%f", &x);

14    printf("Unesite tacnost eps: ");
    scanf("%f", &eps);

16 }
```

```

18  /* Tacnost izracunavanja je zadovoljena ako je apsolutna vrednost
    * razlika suma
20  * u dvema uzastopnim iteracijama manja od zadate tacnosti;
    * Oдавде se izvodi da apsolutna vrednost opsteg clana sume
22  * mora da bude manja od zadate tacnosti da bi uslov bio ispunjen
    */

24
    S=1;
26  faktorijel=1;
    stepen=x;
28  i=2;
    znak=-1;
30  while(fabs(stepen/faktorijel)>eps){
        S=S+znak*stepen/faktorijel;
32        stepen=stepen*x;
        faktorijel=faktorijel*i;
34        znak=-znak;
        i++;
36    }

38    printf("S=%f\n", S);

40    return 0;
}

```

### Rešenje 2.69

```

1  /* Sa standardnog ulaza unosi se neoznaceni ceo broj. Napisati program
    koji formira i ispisuje broj koji se dobija
    izbacivanjem cifara koje su jednake zbiru svojih suseda. Cifre se
    posmatraju sa desna na levo. */
3
#include <stdio.h>
5
int main(){
7    unsigned n, novo_n;
    int stepen;
9    int cifra_levo, cifra_sredina, cifra_desno;

11    /* Ucitavamo broj sa ulaza */
    printf("Unesite broj: ");
13    scanf("%u", &n);

15
    /* Stepen broja 10 sa kojim cemo mnoziti cifre izdvojenog broja */
17    stepen=1;

19    /* Nova vrednost broja */
    novo_n=0;

21
    /* Sve dok u zapisu broja imamo barem tri cifre */

```

## 2 Kontrola toka

```
23 while(n>99){
    /* Izdvajamo srednju cifru, cifru desno od nje i cifru levo od
    nje:
25 npr. za trojku 583 8 je srednja cifra, 3 je cifra desno, a 5
    cifra levo */
    cifra_desno=n%10;
27 cifra_sredina=(n/10)%10;
    cifra_levo=(n/100)%10;

29
    /* U novi broj smestamo desnu cifru */
31 novo_n+=cifra_desno*stepen;

33
    /* Azuriramo vrednost stepena */
    stepen=stepen*10;

35
    /* Ako je srednja cifra jednaka zbiru leve i desne cifre */
37 if(cifra_levo+cifra_desno==cifra_sredina){

39
        /* Treba izbaciti srednju cifru, pa broj n azuriramo tako sto
        ga podelimo sa 100 */
        n=n/100;
41     }
    else{
43
        /* Inace, zadržavamo srednju cifru i odbacujemo samo poslednju
        */
45         n=n/10;
    }
47 }

49 /* Na novi broj dodajemo preostali dvocifreni ili jednocifreni broj
    */
    novo_n=n*stepen+novo_n;

51
    /* I ispisujemo rezultat */
53 printf("%d\n", novo_n);

55
    return 0;
57 }
```

### Rešenje 2.70

```
1 /* Napisati program koji proverava da li je dati prirodan broj
    palindrom. Broj
    je palindrom ako se isto cita i sa leve i sa desne strane. */
3
    #include <stdio.h>
5    #include <math.h>
```



```
7 int main() {
9     int x;
10    int broj_cifara;
11    int min_stepen, max_stepen;
12    int pom;
13    int leva_cifra, desna_cifra;
14    int indikator;
15
16    printf("Unesite broj: ");
17    scanf("%d", &x);
18
19    /* Ako je korisnik uneo negativan broj, analiziramo njegovu
20       apsolutnu
21       * vrednost
22       */
23    if(x < 0)
24        x=-x;
25
26    /* Odredjujemo broj cifara u zapisu broja x
27       kako bismo mogli da izdvajamo istovremeno cifre i sa leve i sa
28       desne
29       strane
30       */
31    broj_cifara = 0;
32    pom = x;
33    while(pom > 0) {
34        pom /= 10;
35        broj_cifara++;
36    }
37
38    /* Odredjujemo stepen koji stoji uz krajnju levu cifru broja */
39    max_stepen = (int) pow (10, broj_cifara-1);
40
41    /* Indikator je promenljiva koja ce nam ukazivati da li je broj
42       * palindrom ili ne
43       */
44    indikator=1;
45    while(x!=0 && indikator==1){
46        /* Izdvajamo levu cifru */
47        leva_cifra=x/max_stepen;
48        /* Izdvajamo desnu cifru */
49        desna_cifra=x%10;
50        /* Ako su cifre razlicite, odmah mozemo da zakljucimo da
51           * broj nije palindrom i da prekinemo izvorsavanje petlje */
52        if(leva_cifra!=desna_cifra){
53            indikator=0;
54            break;
55        }
56        /* Formiramo novu vrednost broja x tako sto odbacujemo
57           * krajnju levu i krajnju desnu cifru */
58    }
```

```
57     x=(x%max_stepen-x%10)/10;
59     /* I korigujemo maksimalan stepen tako dobijenog broja -
        * delimo sa 100 jer smo odbacili 2 cifre */
        max_stepen=max_stepen/100;
61 }

63 /* Ispisujemo rezultat */
64 if(indikator==1)
65     printf("Broj je palindrom!\n");
66 else
67     printf("Broj nije palindrom!\n");
69
70 return 0;
71 }
```

### Rešenje 2.71

```
/* Sa standardnog ulaza se unosi ceo pozitivan broj n, a zatim i n
   celih brojeva. Napisati program koji ispisuje
2 broj sa najvećom cifrom desetica. Ukoliko ima više takvih, ispisati
   prvi. */

4 #include <stdio.h>
5 #include <math.h>
6
7 int main(){
8
9     int n;
10    int x, x_desetica;
11    int max_desetica, broj;
12    int i;

13
14    /* Citamo vrednost sa ulaza */
15    printf("Unesite broj n: ");
16    scanf("%d", &n);

17
18    /* Postavljamo maksimalnu cifru desetice na 0 - 0 je svakako
        najmanja cifra pa je početna vrednost neutralna tj.
        ne može da utiče na maksimum koji izračunavamo. Nije uvek zgodno
        pretpostaviti da je maksimalna vrednost 0. Na primer,
20 ako tražimo maksimum celih brojeva, a korisnik unese -32 -7 i -22,
        maksimalni je broj -7 */
        max_desetica=0;

21
22    /* Učitavamo broj po broj */
23    printf("Unesite n brojeva: ");
24    for(i=0; i<n; i++){
25        scanf("%d", &x);

26
27        /* Izdvajamo cifru desetica procitanog broja */
```

```

    x_desetica=(abs(x)/10)%10;
30
    /* Ako je ona veca od maksimalne cifre desetice */
32    if(x_desetica>max_desetica){
        /* Cuvamo je */
34        max_desetica=x_desetica;
        /* Ali zbog ispisa, cuvamo i broj u kojem se ona pojavljuje */
36        broj=x;
    }
38 }

40 /* Ispisujemo rezultat */
    printf("%d\n", broj);
42
    return 0;
44 }

```

### Rešenje 2.72

```

/* Sa standardnog ulaza se unosi ceo pozitivan broj n, a zatim i n
   celih brojeva. Napisati program koji ispisuje
2 broj sa najvećim brojem cifara. Ukoliko ima više takvih, ispisati
   prvi. */

4 #include <stdio.h>
   #include <math.h>

6
   int main(){
8
    int n;
10    int x, x_kopija, br_cifara;
    int max_br_cifara, broj;
12    int i;

14    /* Citamo vrednost sa ulaza */
    printf("Unesite broj n: ");
16    scanf("%d", &n);

18    /* Postavljamo maksimalan broj cifara na 0 - svaki broj ima više
       od 0 cifara pa je ova vrednost neutralna */
    max_br_cifara=0;

20
    /* Ucitavamo broj po broj */
22    printf("Unesite n brojeva: ");
    for(i=0; i<n; i++){
24        scanf("%d", &x);

26        /* Odredjujemo broj cifara unetog broja x */
        x_kopija=abs(x);
        br_cifara=0;
28

```

## 2 Kontrola toka

```
30     while(x_kopija!=0){
31         x_kopija=x_kopija/10;
32         br_cifara++;
33     }
34     /* Ako je broj cifara unetog broja veci od maksimalnog */
35     if(br_cifara>max_br_cifara){
36         /* Cuvamo ga */
37         max_br_cifara=br_cifara;
38         /* I zbog ispisa rezultata, cuvamo i originalni broj */
39         /* Zbog ovoga smo morali i da racunamo broj cifara nad kopijom
40         broja x kako ne bismo promenili njegovu vrednost */
41         broj=x;
42     }
43 }
44
45 /* Ispisujemo rezultat */
46 printf("%d\n", broj);
47
48 return 0;
49 }
```

### Rešenje 2.73

```
/* Sa standardnog ulaza se unosi ceo pozitivan broj n, a zatim i n
celih brojeva. Napisati program koji ispisuje
2 broj sa najvecom vodecom cifrom. Vodeca cifra je prva cifra iz zapisa
broja. Ukoliko ima vise takvih, ispisati
prvi. */
4
6 #include <stdio.h>
7 #include <math.h>
8
9 int main(){
10
11     int n;
12     int x, x_kopija;
13     int broj;
14     int vodeca_cifra, max_vodeca_cifra;
15     int i;
16
17     /* Citamo vrednost sa ulaza */
18     printf("Unesite broj n: ");
19     scanf("%d", &n);
20
21     /* Postavljamo maksimalnu vodecu cifru na 0 - cifre broja su vece
ili jednake od 0 pa je ova vrednost neutralna */
22     max_vodeca_cifra=0;
23
24     /* Ucitavamo broj po broj */
25     printf("Unesite n brojeva: ");
```

```

26     for(i=0; i<n; i++){
        scanf("%d", &x);

28     /* Odredjujemo vodecu cifru broja */
        x_kopija=abs(x);
30     while(x_kopija>10){
        x_kopija=x_kopija/10;
32     }
        vodeca_cifra=x_kopija;

34     /* Ako je izdvojena cifra veca od maksimalne vodece cifre */
36     if(vodeca_cifra>max_vodeca_cifra){
        /* Cuvamo je */
38         max_vodeca_cifra=vodeca_cifra;
        /* I zbog ispisa, cuvamo i broj u kojem se ona pojavljuje */
40         /* Zbog ovoga smo morali i da racunamo vodecu cifru nad kopijom
        broja x kako ne bismo promenili njegovu vrednost */
        broj=x;
42     }
    }

44     /* Ispisujemo rezultat */
46     printf("%d\n", broj);

48     return 0;
50 }

```

### Rešenje 2.74

```

/* Sa standardnog ulaza se unose celi pozitivni brojevi n (n > 1) i d
, a zatim i n celih brojeva. Napisati program
2 koji izracunava koliko ima parova uzastopnih brojeva medju unetim
   brojevima koji se nalaze na rastojanju d.
   Rastojanje izmedu brojeva je definisano sa  $d(x, y) = |y - x|$ . Rezultat
   ispisati na standardni izlaz. */

4
#include <stdio.h>
6 #include <math.h>

8 int main(){

10     int n;
    int d;
12     int x, y;
    int broj_parova;
14     int i;

16     /* Ucitavamo vrednosti sa ulaza */
    printf("Unesite brojeve n i d: ");
18     scanf("%d %d", &n, &d);

```

```
20  /* Inicijalizujemo broj parova */
    broj_parova=0;
22
    printf("Unesite n brojeva: ");
24
    /* Ucitavamo prvi broj */
26    scanf("%d", &x);
28
    for(i=1; i<n; i++){
        /* Ucitavamo naredni broj */
30        scanf("%d", &y);
32
        /* Ako su brojevi na rastojanju d */
        if(abs(y-x)==d)
34            /* Treba uvecati broj parova */
            broj_parova++;
36
        /* Cuvamo broj iz tekuce iteracije kako bismo mogli da ga
        upotrebimo u narednoj iteraciji */
38        x=y;
    }
40
    /* Ispisujemo rezultat */
42    printf("Broj parova: %d\n", broj_parova);
44
    return 0;
46 }
```

### Rešenje 2.75

```
1  /* Sa standardnog ulaza se unosi ceo broj n, a zatim i n karaktera.
    Napisati program koji proverava da li se od
    unetih karaktera moze napisati rec Zima. */
3
    #include <stdio.h>
5    #include <math.h>
7
    int main(){
9
        int n;
        int broj_Z, broj_i, broj_m, broj_a;
11       char novi_red, c;
        int i;
13
        broj_Z=0;
15       broj_i=0;
        broj_m=0;
17       broj_a=0;
```

```

19  /* Ucitavamo broj karaktera */
21  printf("Unesite broj: ");
   scanf("%d", &n);
23
24  /* Ucitavamo karakter po karakter */
25  for(i=0; i<n; i++){
   printf("Unestite %d. karakter: ", i+1);
27  /*
   Prvo citamo znak za novi red koji je ostao neprocitan nakon
   pritiska Enter tastera
29   posle prethodnog unosa, pa tek onda citamo karakter koji treba
   obradivati
   */
31   scanf("%c%c", &novi_red, &c);
33
   /* Analiziramo karakter */
   switch(c){
35     case 'Z':
       broj_Z++;
37     break;
       case 'i':
39     broj_i++;
       break;
41     case 'm':
       broj_m++;
43     break;
       case 'a':
45     broj_a++;
       break;
47   }
   }
49
   /* Ako imamo barem jedno veliko slovo z i barem po jedno malo slovo
   i, m i a */
51   if(broj_Z && broj_i && broj_m && broj_a){
       /* Zakljucujemo da se rec moze napisati */
53   printf("Moze se napisati rec Zima.\n");
   }
55   else{
       /* Inace, obavestavamo korisnika da je to nemoguće */
57   printf("Ne moze se napisati rec Zima.\n");
   }
59
   return 0;
61 }

```

### Rešenje 2.76

```

1  /* Sa standardnog ulaza se unose celi brojevi sve do unosa broja 0.
   Napisati program koji izracunava i ispisuje

```

```
razliku najveceg i najmanjeg unetog broja. */
3
#include <stdio.h>
5 #include <math.h>

7 int main(){

9     int x;
    int min, max;

11     printf("Unesite brojeve: ");

13     /* Prvi broj učitavamo izvan petlje zbog inicijalizacije maksimuma
        i minimuma */
15     scanf("%d", &x);
    max=x;
17     min=x;

19     /* U petlji smo sve dok ne procitamo broj 0 */
    while(x!=0){

21         /* Proveravamo da li je procitani broj veci od aktuelnog
            maksimuma */
23         if(x>max)
            max=x;
25         /* Proveravamo da li je procitani broj manji od aktuelnog
            minimuma */
27         if(x<min)
            min=x;

29         /* Učitavamo naredni broj */
        scanf("%d", &x);
31     }

33     /* Ispisujemo razliku najveceg i najmanjeg broja */
    printf("Razlika: %d\n", max-min);
35
    return 0;
37 }
```

### Rešenje 2.77

```
1 /* Sa standardnog ulaza se unose realni brojevi sve do unosa broja 0.
    Napisati program koji izracunava i ispisuje
    aritmeticku sredinu unetih brojeva. */

3
#include <stdio.h>
5 #include <math.h>

7 int main(){
```



```
9   float x;
10  int broj_brojeva;
11  float suma;

12  /* Inicijalizujemo vrednosti */
13  broj_brojeva=0;
14  suma=0;

15

16  printf("Unesite brojeve: ");

17

18  /* U petlji */
19  while(1){
20      /* Ucitavamo broj sa ulaza */
21      scanf("%f", &x);

22      /* Ako je korisnik uneo 0, prekidamo sa petljom */
23      if(x==0)
24          break;

25      /* Inace .. */

26      /* Procitani broj dodajemo na sumu */
27      suma+=x;
28      /* I uvecavamo broj procitanih brojeva */
29      broj_brojeva++;
30  }

31  /* Ispisujemo trazeni rezultat */
32  printf("Aritmeticka sredina: %.4f\n", suma/broj_brojeva);

33

34  return 0;
35 }
```

### Rešenje 2.78

```
#include <stdio.h>

2  int main(){
3
4      int n;
5      int k, v;

6      /* Unosi se dimenzija kvadrata */
7      printf("Unesite broj n: ");
8      scanf("%d", &n);

9      /* Kvadrat iscrtavamo vrstu po vrstu */
10     for(v=1; v<=n; v++){
11
12         /* Ukoliko je u pitanju prva ili poslednja vrsta */
13     }
```

## 2 Kontrola toka

---

```
16     if(v==1 || v==n){
18         /* Stampamo n zvezdica */
19         for(k=1; k<=n; k++)
20             putchar('*');
21
22     }
23     else
24     {
25         /* Ukoliko vrsta nije ni prva ni poslednja */
26
27         /* Stampamo prvu zvezdu */
28         putchar('*');
29
30         /* Zatim stampamo n-2 blanko znaka */
31         for(k=1; k<=n-2; k++)
32             putchar(' ');
33
34         /* I stampamo poslednju zvezdu */
35         putchar('*');
36     }
37
38     /* Posto smo odstampali sadrzaj vrste, stampamo i znak za novi
39     red */
40     putchar('\n');
41 }
42
43 return 0;
44 }
```

### Rešenje 2.79

```
#include <stdio.h>
2
3 int main(){
4
5     int n;
6     char novi_red, c;
7     int i, j;
8
9     printf("Unesite broj n: ");
10    scanf("%d", &n);
11
12    scanf("%c", &novi_red);
13
14    printf("Unesite karakter c: ");
15    scanf("%c", &c);
16
17    /* Iscrtavamo trougao red po red */
18    for(i=1; i<=n; i++){
```

```

20  /* Ukoliko je u pitanju prvi ili poslednji red */
    if(i==1 || i==n){
22      /* Stampamo i puta karakter c */
        for(j=1; j<=i; j++)
24          putchar(c);
    }
26  else{
        /* Inace... */

28      /* Stampamo prvi karakter */
30      putchar(c);

32      /* Stampamo i-2 blanko znaka */
        for(j=1; j<=i-2; j++)
34          putchar(' ');

36      /* Stampamo poslednji karakter */
        putchar(c);
38  }

40  /* Stampamo znak za novi red */
    printf("\n");
42  }

44  return 0;
}

```

### Rešenje 2.80

```

1  #include <stdio.h>

3  int main(){

5      int n, i, j;

7      printf("Unesite broj n: ");
      scanf("%d", &n);

9      /* Krstice koje iscrtavamo mozemo posmatrati kao dijagonale
         kvadrata dimenzije n */

11     /* Prolazimo kroz sve vrste kvadrata */
13     for(i=1; i<=n; i++){

15         /* Prolazimo kroz sve kolone kvadrata */
         for(j=1; j<=n; j++){

17             /* Ako se nalazimo na glavnoj ili sporednoj dijagonali */
19             if(i==j || i+j==n+1)
                 /* Stampamo zvezdu */

```

```
21     putchar('*');
    else
23     /* Inace, stampamo blanko znak */
        putchar(' ');
25 }

27 /* Nakon uspesno iscrtane vrste, stampamo znak za novi red */
    putchar('\n');
29 }

31 return 0;
}
```

Rešenje [2.81](#)

Rešenje [2.135](#)

Rešenje [2.83](#)

Rešenje [2.84](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

Rešenje [2.115](#)

## 2.5 Funkcije

**Zadatak 2.116** Napisati funkcije `int kvadrat(int x)` i `int kub(int x)` koje računaju, redom, kvadrat i kub datog broja. Napisati program koji testira rad ovih funkcija.

[Rešenje [2.116](#)]

**Zadatak 2.117** Napisati funkciju `float stepen(float x, int n)` koja računa  $x^n$ . Napisati program koji testira rad ove funkcije.

[Rešenje [2.117](#)]

**Zadatak 2.118** Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najveći zajednički delilac primenom Euklidovog algoritma. Napisati program koji testira rad ove funkcije.

[Rešenje [2.118](#)]

**Zadatak 2.119** Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato `n` vraća zbir recipročnih vrednosti brojeva od 1 do `n`. Napisati program koji testira rad ove funkcije. Rezultat zaokružiti na dve decimale.

[Rešenje [2.119](#)]

**Zadatak 2.120** Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji testira rad napisane funkcije. Rezultat ispisivati na tri decimale.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 461
|| 3.667
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1001
|| 0.500
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -84723
|| 4.800
```

[Rešenje 2.120]

**Zadatak 2.121** Napisati funkciju `void ispis(float x, float y, unsigned n)` koja za dva realna broja  $x$  i  $y$  i jedan neoznačeni ceo broj  $n$  ispisuje vrednosti funkcije  $\sin$  u  $n$  ravnomerno raspoređenih tačaka intervala  $[x,y]$ . Napisati program koji testira rad ove funkcije.

[Rešenje 2.121]

**Zadatak 2.122** Napisati funkciju `int broj_ncifara(int x)` koja broji neparne cifre u zapisu datog celog broja. Testirati rad ove funkcije u programu koji učitava cele brojeve dok se ne unese nula i ispisuje broj neparnih cifara svakog unetog broja.

[Rešenje 2.122]

**Zadatak 2.123** Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji sa standardnog ulaza učitava tri cela broja i ispisuje rezultat poziva funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 19 8 14
|| Minimum je: 8
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: -6 11 -12
|| Minimum je: -12
```

[Rešenje 2.123]

**Zadatak 2.124** Napisati funkciju `unsigned int apsolutna_vrednost(int x)` koja izračunava apsolutnu vrednost broja  $x$ . Napisati program koji sa standardnog ulaza učitava jedan ceo broj i ispisuje rezultat poziva funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: -34  
| Apsolutna vrednost: 34
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 5  
| Apsolutna vrednost: 5
```

[Rešenje 2.124]

**Zadatak 2.125** Napisati funkciju *float razlomljeni\_deo(float x)* koja izračunava razlomljeni deo broja *x*. Napisati program koji sa standardnog ulaza učitava jedan realan broj i ispisuje rezultat poziva funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 8.235  
| Razlomljeni deo: 0.235000
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: -5.11  
| Razlomljeni deo: 0.110000
```

[Rešenje 2.125]

**Zadatak 2.126** Napisati funkciju *void romb(int n)* koja iscrtava romb čija je stranica dužine *n*. Napisati program koji učitava ceo pozitivan broj i ispisuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj n: 5  
| *****  
| *****  
| *****  
| *****  
| *****
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj n: 2  
| **  
| **
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj n: -5  
| Greska: pogresna dimenzija!
```

[Rešenje 2.126]

**Zadatak 2.127** Napisati funkciju *void grafikon\_h(int a, int b, int c, int d)* koja iscrtava horizontalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i prikazuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.



### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 4 1 7 5
****
*
*****
****
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 8 -2 5 4
Greska: pogresan unos!
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 5 2 2 10
*****
**
**
*****
```

[Rešenje [2.127](#)]

**Zadatak 2.128** Napisati funkciju `void grafikon_v(int a, int b, int c, int d)` koja iscrtava vertikalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i ispisuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 4 1 7 5
*
*
**
* **
* **
* **
****
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 8 -2 5 4
Greska: pogresan unos!
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 5 2 2 4
*
* *
* *
****
****
```

[Rešenje [2.128](#)]

**Zadatak 2.129** Napisati funkciju `int prestupna(int godina)` koja za zadanu godinu proverava da li je prestupna. Funkcija treba da vrati 1 ako je godina prestupna ili 0 ako nije. Napisati program koji učitava dva cela broja  $g1$  i  $g2$  i

## 2 Kontrola toka

---

ispisuje sve godine iz intervala  $[g1, g2]$  koje su prestupne.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2001 2010  
|| Prestupne godine su: 2004 2008
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2005 2015  
|| Prestupne godine su: 2008 2012
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2010 2001  
|| Greska: pogresan unos!
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2001 2002  
|| Nema prestupnih godina u ovom intervalu!
```

[Rešenje 2.129]

**Zadatak 2.130** Napisati funkciju `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu. Napisati program koji testira ovu funkciju. U slučaju nekorektnog unosa ispisati odgovarajuću poruku o grešci.

[Rešenje 2.135]

**Zadatak 2.131** Napisati funkciju `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan. Napisati program koji testira ovu funkciju.

[Rešenje 2.135]

**Zadatak 2.132** Napisati funkciju `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum određuje datum sledećeg dana. Napisati program koji testira ovu funkciju.

[Rešenje 2.135]

**Zadatak 2.133** Napisati funkciju `int broj_dana1(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do do datog datuma. Napisati glavni program koji testira napisanu funkciju.

[Rešenje 2.135]

**Zadatak 2.134** Napisati funkciju `int broj_dana2(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati glavni program koji testira napisanu funkciju.

[Rešenje 2.135]

**Zadatak 2.135** Napisati funkciju `int broj_dana3(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati glavni program koji testira napisanu funkciju.

[Rešenje 2.135]

**Zadatak 2.136** Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja  $n$ . Napisati program koji sa standardnog ulaza učitava ceo broj  $k$  i ispisuje zbir delilaca svakog broja od 1 do  $k$ .

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj k: 6  
|| 1 3 4 7 6 12
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj k: -2  
|| Greska: pogresan unos!
```

[Rešenje 2.136]

**Zadatak 2.137** Napisati funkciju `int ukloni_stotine(int n)` koja modifikuje zadati broj tako što iz njegovog zapisa uklanja cifru stotina (ako postoji). Napisati program koji za brojeve koji se unose sa standardnog ulaza sve do pojave broja 0 ispisuje rezultat primene funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 1210  
|| 110  
|| Unesite broj: 18  
|| 18  
|| Unesite broj: 3856  
|| 356  
|| Unesite broj: 0
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -9632  
|| -932  
|| Unesite broj: 246  
|| 46  
|| Unesite broj: -52  
|| -52  
|| Unesite broj: 0
```

[Rešenje 2.137]

**Zadatak 2.138** Napisati funkciju *int rotacija(int n)* koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sa standardnog ulaza sve do pojave broja 0 ispisuje rezultat primene funkcije.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 146
461
Unesite broj: 18
81
Unesite broj: 3856
8563
Unesite broj: 7
7
Unesite broj: 0
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 89
98
Unesite broj: -369
-693
Unesite broj: -55281
-52815
Unesite broj: 0
```

[Rešenje 2.138]

**Zadatak 2.139** Napisati funkciju *int prost (int x)* koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati 1 ako je broj prost i 0 u suprotnom. Testirati rad funkcije u programu koji za uneti ceo broj *n* ispisuje prvih *n* prostih brojeva.

[Rešenje 2.139]

**Zadatak 2.140** Napisati funkciju *int sadrzi(int x, int c)* koja ispituje da li se cifra *c* nalazi u zapisu celog broja *x*. Napisati program koji testira rad ove funkcije.

[Rešenje 2.140]

**Zadatak 2.141** Napisati program za ispitivanje svojstava cifara datog celog broja.

- Napisati funkciju *sve\_parne\_cifre* koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.
- Napisati funkciju *sve\_cifre\_jednake* koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.
- Testirati napisane funkcije na unetom celom broju i ispisati odgovarajuću poruku.

[Rešenje 2.141]

**Zadatak 2.142** Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva uneta neoznačena broja  $x$  i  $n$  utvrđuje da li je  $x$  neki stepen broja  $n$ . Ukoliko jeste, funkcija vraća izložilac stepena, a u suprotnom vraća -1. Napisati program koji testira rad ove funkcije.

[Rešenje 2.142]

**Zadatak 2.143** Napisati funkciju `double e_na_x(double x, double eps)` koja računa vrednost  $e^x$  kao parcijalnu sumu reda  $\sum_{n=0}^{\infty} \frac{x^n}{n!}$ , pri čemu se sumiranje vrši dok je razlika sabiraka u redu po apsolutnoj vrednosti manja od  $\varepsilon$ . Napisati program koji testira rad ove funkcije.

[Rešenje 2.143]

**Zadatak 2.144** Za dati broj može se formirati niz tako da je svaki sledeći član niza dobijen kao suma cifara prethodnog člana niza. Broj je srećan ako se dati niz završava jedinicom. Napisati funkciju `int srećan(int x)` koja vraća 1 ako je broj srećan, a 0 u suprotnom. Napisati program koji za uneti prirodan broj  $n$  ispisuje sve srećne brojeve od 1 do  $n$ .

[Rešenje 2.144]

**Zadatak 2.145** Napisati funkciju `int konverzija(int c)` koja prebacuje veliko slovo u ekvivalentno malo i obrnuto. Napisati program koji testira ovu funkciju na karakterima koji se unose sa standardnog ulaza do pojave EOF.

[Rešenje 2.145]

**Zadatak 2.146** Napisati funkciju `int zapis(int x, int y)` koja proverava da li se brojevi  $x$  i  $y$  zapisuju pomoću istih cifara. Funkcija treba da vrati vrednost 1 ako je uslov ispunjen, odnosno 0 ako nije. Napisati i program koji učitava dva cela broja i ispisuje rezultat primene funkcije.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: 251 125
|| Uslov je ispunjen!
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: 8898 9988
|| Uslov nije ispunjen!
```

## 2 Kontrola toka

---

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: -7391 1397  
|| Uslov je ispunjen!
```

[Rešenje 2.146]

**Zadatak 2.147** Napisati funkciju *int faktorijel(int n)* koja računa faktoriyel broja  $n$ . Napisati i program koji učitava dva cela broja  $x$  i  $y$  ( $0 \leq x, y \leq 12$ ) i ispisuje vrednost zbira  $x! + y!$ .

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 4 5  
|| 144
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 18 -5  
|| Greska: pogresan unos!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 6 0  
|| 721
```

[Rešenje 2.147]

**Zadatak 2.148** Napisati funkciju *int rastuce(int n)* koja ispituje da li su cifre datog celog broja u rastućem poretku. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i ispisuje rezultat primene funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2689  
|| Cifre su u rastucem poretku!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 559  
|| Cifre su u rastucem poretku!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 628  
|| Cifre nisu u rastucem poretku!
```

[Rešenje 2.148]

**Zadatak 2.149** Broj je Armstrongov ako je jednak sumi nekog stepena svojih cifara.

- a) Napisati funkciju `int stepen(int x, int n)` koja izračunava  $n$ -ti stepen broja  $x$ .
- b) Napisati funkciju `int armstrong(int x)` koja vraća 1 ako je broj Armstrongov, odnosno 0 ako nije.
- c) Napisati program koji za ceo broj koji se unosi sa standardnog ulaza proverava da li je Armstrongov (koristeci funkciju `armstrong`).

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 153
|| Broj je Armstrongov!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1634
|| Broj je Armstrongov!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 118
|| Broj nije Armstrongov!
```

[Rešenje 2.149]

**Zadatak 2.150** Napisati funkciju `int par_nepar(int n)` koja ispituje da li su cifre datog celog broja naizmenično parne i neparne. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i testira rad funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 2749
|| Broj ispunjava uslov!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -963
|| Broj ispunjava uslov!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 27449
|| Broj ne ispunjava uslov!
```

[Rešenje 2.150]

**Zadatak 2.151** Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj  $x$  pojavljuje u nizu brojeva koji se unose sa standardnog ulaza sve do pojave nule. Napisati program koji učitava vrednost broja  $x$  i testira

## 2 Kontrola toka

---

rad napisane funkcije.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj x: 2.84
Unesite brojeve: 8.13 2.84 5 21.6 2.84 11.5 0
Broj pojavljivanja broja 2.84 je: 2
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj x: -1.17
Unesite brojeve: -128.35 8.965 8.968 89.36 0
Broj pojavljivanja broja -1.17 je: 0
```

[Rešenje 2.151]

**Zadatak 2.152** Napisati funkciju *long int fibonaci(int n)* koja računa  $n$ -ti element Fibonačijevog niza. Fibonačijev niz je niz za koji važi:  $F_0 = 1$ ,  $F_1 = 1$ ,  $F_{n+2} = F_{n+1} + F_n$  za  $n \geq 0$ . Napisati i program koji učitava ceo broj  $n$  ( $0 \leq n \leq 50$ ) i ispisuje traženi Fibonačijev broj.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 7
21
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 65
Greska: nedozvoljena vrednost!
```

[Rešenje 2.152]

**Zadatak 2.153** Napisati funkciju *char sifra(char c, int k)* koja za dati karakter  $c$  određuje šifru na sledeći način: ukoliko je  $c$  slovo, šifra je karakter koji se nalazi  $k$  pozicija iza njega u abecedi. U suprotnom karakter ostaje nepromenjen. Šifrovanje treba da bude kružno, što znači da je, na primer, šifra za  $c='b'$  i  $k=2$  karakter  $'z'$ . Napisati program koji učitava karakter po karakter do kraja ulaza (do pojave EOF koji se generiše kombinacijom CTRL+D) i ispisuje šifrovani tekst.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj k: 2
Unesite tekst (CTRL+D za prekid):
c
a
8
+
+
Z
X
```

[Rešenje 2.153]



## 2.6 Rešenja

### Rešenje 2.116

```
1  #include <stdio.h>
2
3  int kvadrat(int x)
4  {
5      /* promenljive u listi argumenata funkcije, kao i one
6       deklarisanе u samoj funkciji, lokalne su za tu funkciju
7       sto znaci da se promenljive x i y nece "videti" nigde izvan
8       funkcije kvadrat (ni u funkciji main ni u funkciji kub)
9       */
10
11     int y;
12     y = x*x;
13     return y;
14 }
15
16 int kub(int a)
17 {
18     /*
19      u listi argumenata funkcije mozemo, a ne moramo, imati
20     promenljivu
21     istog naziva kao promenljiva koja je deklarisanа u main
22     funkciji
23     (u ovom slucaju promenljiva a); ova promenljiva se razlikuje
24     od promenljive a deklarisanе u main funkciji i vidljiva je
25     samo unutar funkcije kub
26     */
27     return a*a*a;
28 }
29
30 int main()
31 {
32     int a, kv, kb;
33     printf("Unesi ceo broj:");
34     scanf("%d", &a);
35
36     kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
37     funkcije kvadrat */
38     kb = kub(a);      /* promenljivoj kb dodeljujemo povratnu vrednost
39     funkcije kub */
40
41     printf("Kvadrat broja %d je %d, a njegov kub je %d\n", a, kv, kb);
42     return 0;
43 }
```

### Rešenje 2.117

## 2 Kontrola toka

---

```
1  /*
2  Napisati program koji za uneti realan broj x i ceo broj n ispisuje
3  vrednost stepena x^n. Unosenje promenljivih, racunanje stepena i
4  ispis promenljivih realizovati u posebnim funkcijama.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 float stepen(float a, int b)
11 {
12     float s=1;
13     int i;
14
15     for(i=0;i<abs(b);i++)
16         s=s*a;
17
18     return b>0 ? s : 1/s; /* ukoliko je izlozilac b negativan,
19                            izracunamo a^|b| i vracamo reciprocnu vrednost
20                            izracunatog stepena */
21 }
22
23 int main()
24 {
25     int n;
26     float x;
27     float s;
28
29     printf("Unesi jedan realan i jedan ceo broj:");
30     scanf("%f%d",&x,&n);
31
32     s = stepen(x,n);
33
34     printf("%f^%d=%f\n",x,n,s);
35
36     return 0;
37 }
38
39 }
```

### Rešenje 2.118

```
1  /*
2  Napisati funkciju koja za dva data cela broja odredjuje
3  najveći zajednicki delilac. Napisati potom glavni program
4  koji testira ovu funkciju.
5  */
6
7  #include <stdio.h>
```

```

9  int euklid(int x, int y)
10 {
11     int r;
12     /* Euklidov algoritam */
13     while(y) /* algoritam se zaustavlja kada vrednost */
14     {        /* promenljive y postane nula */
15         r=x%y;
16         x=y;
17         y=r;
18     }
19
20     return x; /* nzd je sacuvan u promenljivoj x */
21 }
22
23 int main()
24 {
25     int a,b;
26     int nzd;
27
28     printf("unesi dva cela broja:");
29     scanf("%d%d", &a,&b);
30
31     nzd = euklid(a,b); /* promenljivoj nzd dodeljujemo povratnu
32                        vrednost funkcije euklid */
33
34     printf("najveci zajednicki delilac za %d i %d je %d\n", a,b,nzd);
35
36     return 0;
37 }

```

### Rešenje 2.119

```

/*
2  Napisati funkciju koja za dato n vraca zbir reciprocnih vrednosti
   brojeva od 1 do n.
   Napisati program koji omogucava korisniku da unese prirodan broj n, a
   potom ispisuje zbir reciprocnih
4  vrednosti brojeva od 1 do n koristeći funkciju float zbir_reciprocnih
   (int n). Rezultat zaokruziti
   na dve decimalne.
6  */
7
8  #include <stdio.h>
9
10 float zbir_reciprocnih(int n)
11 {
12     float z=0;
13     int i;
14     for(i=1;i<=n;i++)

```

## 2 Kontrola toka

```
        z+=1.0/i; /* da bismo dobili reciprocnu vrednost broja, vazno je
                da izbegnemo celobrojno deljenje dva cela broja */
16     return z; /* tako sto ce npr deljenik biti 1.0 umesto 1 */
    }
18
    int main()
20 {
        int n;
22     printf("Unesi jedan pozitivan ceo broj:\n");
        scanf("%d", &n);
24     printf("Zbir reciprocnih vrednosti brojeva od 1 do %d je %.2f\n", n
            , zbir_reciprocnih(n));
        /* povratna vrednost funkcije zbir_reciprocnih je float; funkciju
            mozemo pozvati u okviru
26         naredbe printf i umesto specifikatora %.2f bice ispisana
            povratna vrednost funkcije
            zbir_reciprocnih zaokruzena na dve decimale */
28     return 0;
    }
```

### Rešenje 2.120

```
1  /*
    Napisati funkciju koja racuna aritmeticku sredinu cifara datog celog
    broja.
3  Napisati potom glavni program koji omogucava korisniku da unese ceo
    broj
    i racuna aritmeticku sredinu njegovih cifara primenom napisane
    funkcije. Ispisati
5  izracunatu vrednost zaokruzenu na dve decimale.
    */
7
    #include<stdio.h>
9  #include<stdlib.h>

11 float aritmeticka_sredina(int x)
    {
13         int zbir_cifara=0;
            int broj_cifara=0;
15         char cifra;

17         if (x==0) /* u slucaju da je uneta 0 */
            return 0; /* aritmeticka sredina cifara iznosi 0 i tu vrednost
                vracamo */
19

21         x=abs(x); /* uzimamo apsolutnu vrednost broja za slucaj da je
            negativan */

23         while(x)
            {
```

```

25     cifra=x%10;

27     broj_cifara++;
    zbir_cifara+=cifra;

29     x/=10;

31 }

33     return (0.0+zbir_cifara)/broj_cifara; /* posto su zbir_cifara i
        broj_cifara celobrojne vrednosti,
        jednu od njih konvertujemo u realnu      neophodno je da bar
        celobrojno deljenje */                  kako bismo izbegli
35 }

37 int main()
39 {
    int x;
    printf("Unesi jedan ceo broj:");
    scanf("%d",&x);
    printf("Aritmeticka sredina cifara broja %d iznosi %.2f\n", x,
    aritmeticka_sredina(x));
    return 0;
45 }

```

### Rešenje 2.121

```

1  /*
    Napisati funkciju koja za dva realna broja x i y i jedan neoznaceni
    ceo broj n
3  ispisuje vrednosti funkcije sin u n ravnomerno rasporedjenih tacaka
    intervala [x,y].
    Napisati potom glavni program koji omogucava korisniku da unese
    potrebne vrednosti
5  i poziva napisanu funkciju.
    */

7
9  #include <stdio.h>
    #include <math.h>

11 void ispis(float x, float y, int n) /* funkcija nema povratnu
    vrednost; zbog toga je povratni tip void */
{
13     float i;
    float korak=(y-x)/(n-1);

15     for(i=x;i<=y;i+=korak)
17         printf("sin(%.4f)=%.4f\n", i,sin(i));

19 }

```

```
21 int main()
22 {
23     float a,b;
24     int n;
25     float t;
26     printf("Unesi dva realna broja:");
27     scanf("%f%f",&a,&b);
28     printf("Unesi jedan ceo broj > 1:");
29     scanf("%u",&n);
30
31     if (n<=1 || a==b)
32     {
33         printf("Nekorektan unos\n");
34         return -1;
35     }
36     if (b<a) /* u slucaju da je desni kraj intervala manji od levog */
37     {       /* zamenimo im mesta */
38         t=a;
39         a=b;
40         b=t;
41     }
42
43
44
45     ispis(a,b,n);
46
47     return 0;
48 }
```

### Rešenje 2.122

```
/*
2  Napisati funkciju koja broji neparne cifre u zapisu datog celog broja
   . Napisati
   potom glavni program koji unosi cele brojeve dok se ne unese nula, i
   ispisuje
4  broj neparnih cifara svakog unetog broja koriscenjem napisane
   funkcije.
*/
6
8  #include<stdio.h>
9  #include<stdlib.h>
10
11 int broj_ncifara(int x)
12 {
13     int s=0;
14     char cifra;
15     x = abs(x);
16
17     while(x)
```

```

18     {
        cifra = x%10;
        s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
neparna,
20                                a 0 kada je cifra parna */
        x/=10;
22     }

24     return s;
}

26
27
28 int main()
29 {
    int x;
30     do
    {
32         scanf("%d",&x);
        printf("Broj neparnih cifara u zapisu broja %d: %d\n", x,
        broj_ncifara(x));
34     } while(x!=0);

36     return 0;
}

```

### Rešenje 2.123

```

1  #include <stdio.h>

3  /*
    Funkcija koja racuna minimum tri cela broja
5  */
int min(int x, int y, int z){
7     int min;

9     min=x;

11    if(min>y)
        min=y;

13    if(min>z)
        min=z;

15    return min;
17 }

19
20
21 int main(){
    int x,y,z;

23    /* Ucitavamo brojeve */
    printf("Unesite brojeve: ");
25    scanf("%d%d%d", &x, &y, &z);

```

## 2 Kontrola toka

---

```
27  /* Pozivamo funkciju i ispisujemo rezultat */
    printf("Minimum je: %d\n", min(x,y,z));
29
    return 0;
31 }
```

### Rešenje 2.124

```
1  #include <stdio.h>

3  /* Funkcija koja racuna apsolutnu vrednost */
   unsigned int apsolutna_vrednost(int x){
5     /* Kako funkcija vraca unsigned, a x je tipa int, vrsimo kastovanje
       rezultata u tip unsigned */
       return (unsigned)(x<0?-x:x);
7  }

9  int main(){
    int n;
11
    /* Ucitavamo broj */
13    printf("Unesite broj: ");
    scanf("%d", &n);
15
    /* Ispisujemo njegovu apsolutnu vrednost */
17    printf("Apsolutna vrednost: %u\n", apsolutna_vrednost(n));
19
    return 0;
}
```

### Rešenje 2.125

```
1  #include<stdio.h>
   #include<math.h>
3
   /* Funkcija koja vraca razlomljeni deo prosledjenog broja */
5  float razlomljeni_deo(float x){

7     /* Funkcija fabs vraca apsolutnu vrednost realnog broja
       * NAPOMENA: funkcija fabs se nalazi u zaglavlju math.h
       * NAPOMENA2: funkcija abs se nalazi u zaglavlju stdlib.h, ali se
       koristi samo za cele brojeve!
       */
11    x = fabs(x);

13    /* Razlomljeni deo broja dobijamo tako sto od samog broja oduzmemo
       njegov ceo deo*/
       return x - (int)x;
}
```



```

15 }
17 int main(){
18     float n;
19
20     /* Ucitavamo broj */
21     printf("Unesite broj:");
22     scanf("%f", &n);
23
24     /* Ispisujemo rezultat */
25     printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));
26
27     return 0;
28 }

```

### Rešenje 2.126

```

1  #include<stdio.h>
2
3  /* Funkcija koja iscrtava romb */
4  void romb(int n){
5      int i, j;
6
7      /* U svakoj liniji */
8      for(i=0; i<n; i++){
9
10         /* Prvo ispisujemo n-i-1 razmaka */
11         for(j=0; j<n-i-1; j++)
12             printf(" ");
13
14         /* Zatim ispisujemo n zvezdica */
15         for(j=0; j<n; j++)
16             printf("*");
17
18         /* Na kraju svake linije stoji oznaka za novi red */
19         printf("\n");
20     }
21 }
22
23 int main(){
24     int n;
25
26     /* Ucitavamo broj n */
27     printf("Unesite broj n: ");
28     scanf("%d", &n);
29
30     /* Proveravamo korektnost ulaza i ispisujemo rezultat */
31     if(n<=0)
32         printf("Greska: pogresna dimenzija!\n");
33     else

```

```
35 |   romb(n);
37 |   return 0;
   | }
```

### Rešenje 2.127

```
#include<stdio.h>

2
/* Funkcija koja stampa n zvezdica za kojima sledi znak za novi red
   */
4 void stampaj_zvezdice(int n){
   int i;
6   for(i=0; i<n; i++)
       printf("*");
8
   printf("\n");
10 }

12 /* Funkcija koja crta grafikon */
void grafikon_h(int a, int b, int c, int d)
14 {
   int i;
16
   /* Prvo ispisujemo a zvezdica */
18   stampaj_zvezdice(a);

20   /* Zatim u sledecem redu b zvezdica */
   stampaj_zvezdice(b);

22   /* Zatim u sledecem redu c zvezdica */
24   stampaj_zvezdice(c);

26   /* Zatim u poslednjem redu d zvezdica */
   stampaj_zvezdice(d);
28 }

30
32 int main(){
   int a,b,c,d;

34   /* Ucitavamo vrednosti a,b,c,d */
   printf("Unesite vrednosti: ");
36   scanf("%d%d%d%d", &a, &b, &c, &d);

38   /* Proveravamo korektnost ulaza i ispisujemo rezultat */
   if(a <0 || b<0 || c<0 || d<0){
40     printf("Greska: pogresan unos!\n");
   }else{
42     grafikon_h(a,b,c,d);
   }
}
```

```
44     return 0;
46 }
```

## Rešenje 2.128

```
1  #include<stdio.h>

3  int maksimum(int a, int b, int c, int d){
    int max;

5      max=a;
7      if(b>max)
        max=b;
9      if(c>max)
        max=c;
11     if(d>max)
        max=d;

13     return max;
15 }

17 /* Funkcija koja iscrtava vertikalni grafikon */
void grafikon_v(int a, int b, int c, int d){
19     int i, max;

21     /* Na pocetku je potrebno pronaci najveću od ove četiri vrednosti
        */
    max=maksimum(a, b, c, d);

23     /* Grafikon ukupno ima max horizontalnih linija */
25     for(i=0; i<max; i++){

27         /* U svakoj od horizontalnih linija se nalazi po 4 polja:
            polje za a,b,c i d uspravnu liniju.
29         U svako od polja treba da se upise ili * ili belina,
            u zavisnosti od vrednosti a i toga u kojoj liniji se trenutno
            nalazimo
31         */

33         /* Proveravamo uslov za polje a */
        if(i<max-a)
35             printf(" ");
        else
37             printf("*");

39         /* Proveravamo uslov za polje b */
        if(i<max-b)
41             printf(" ");
        else
43             printf("*");
```

```
45  /* Proveravamo uslov za polje c */
46  if(i<max-c)
47      printf(" ");
48  else
49      printf("*");
50
51  /* Proveravamo uslov za polje d */
52  if(i<max-d)
53      printf(" ");
54  else
55      printf("*");
56
57  /* Na kraju svake horizontalne linije stampamo novi red */
58  printf("\n");
59  }
60 }
61
62 int main(){
63     int a,b,c,d;
64
65     /* Ucitavamo vrednosti a,b,c,d */
66     printf("Unesite vrednosti: ");
67     scanf("%d%d%d%d", &a, &b, &c, &d);
68
69     /* Proveravamo korektnost ulaza i stampamo grafikon */
70     if(a < 0 || b < 0 || c < 0 || d < 0)
71         printf("Greska: pogresan unos!\n");
72     else
73         grafikon_v(a,b,c,d);
74
75     return 0;
76 }
```

### Rešenje 2.129

```
1  #include<stdio.h>
2
3  /* Funkcija koja proverava da li je godina prestupna */
4  int prestupna(int godina){
5      if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
6          return 1;
7      else
8          return 0;
9  }
10
11 /* Funkcija koja proverava da li postoji prestupna godina u datom
    intervalu */
12 int postoji_prestupna(int g1, int g2){
13     for(; g1<=g2; g1++){
14         if(prestupna(g1))
15             return 1;
16     }
17     return 0;
18 }
```

```

15     return 1;
16 }
17 return 0;
18 }
19
20 int main(){
21
22     int g1, g2;
23
24     /* Ucitavamo godine */
25     printf("Unesite dve godine: ");
26     scanf("%d%d", &g1, &g2);
27
28     /* Proveravamo korektnost ulaza */
29     if(g1 < 0 || g2 < 0 || g1>g2){
30         printf("Greska: pogresan unos!\n");
31     }
32     else{
33
34         /* Proveravamo da li uopste postoji prestupna godina u datom
35            intervalu */
36         if(postoji_prestupna(g1,g2)){
37             /* Ako postoje, ispisujemo ih */
38             printf("Prestupne godine su: ");
39             for(; g1<=g2; g1++){
40                 if(prestupna(g1))
41                     printf("%d ", g1);
42             }
43             printf("\n");
44         }else{
45             /* U suprotnom, stampamo odgovarajucu poruku */
46             printf("Nema prestupnih godina u ovom intervalu!\n");
47         }
48         return 0;
49     }

```

### Rešenje 2.135

```

1  #include <stdio.h>
2
3  /* Funkcija koja racuna zbir delilaca broja x */
4  int zbir_delilaca(int x){
5      int i=0;
6
7      /* Na pocetku zbir inicijalizujemo na 0 */
8      int zbir = 0;
9
10     /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
11     for(i=1; i<=x; i++){
12         if(x % i == 0)

```

```
13     zbir += i;
14 }
15
16 /* Vracamo dobijeni zbir */
17 return zbir;
18 }
19
20 int main(){
21
22     int k, i;
23
24     /* Ucitavamo broj k */
25     printf("Unesite broj k:");
26     scanf("%d", &k);
27
28     /* Proveravamo korektnost ulaza */
29     if(k <= 0)
30         printf("Greska: pogresan unos!\n");
31     else{
32
33         /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
34         for(i=1; i<=k; i++)
35             printf("%d ", zbir_delilaca(i));
36
37         printf("\n");
38     }
39
40
41     return 0;
42 }
```

Rešenje [2.135](#)

Rešenje [2.135](#)

Rešenje [2.135](#)

Rešenje [2.135](#)

Rešenje [2.135](#)

Rešenje [2.136](#)

```
1 #include <stdio.h>
2
3 /* Funkcija koja racuna zbir delilaca broja x */
```

```

5  int zbir_delilaca(int x){
7      int i=0;

7      /* Na pocetku zbir inicijalizujemo na 0 */
      int zbir = 0;

9      /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
11     for(i=1; i<=x; i++){
13         if(x % i == 0)
            zbir += i;
        }

15     /* Vracamo dobijeni zbir */
17     return zbir;
    }

19 int main(){
21
23     int k, i;

25     /* Ucitavamo broj k */
    printf("Unesite broj k:");
    scanf("%d", &k);

27     /* Proveravamo korektnost ulaza */
29     if(k <= 0)
        printf("Greska: pogresan unos!\n");
    else{

33         /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
        for(i=1; i<=k; i++)
35             printf("%d ", zbir_delilaca(i));

37         printf("\n");
    }

39

41     return 0;
}

```

### Rešenje 2.137

```

2  #include <stdio.h>

4  /* Funkcija koja uklanja broj stotina iz broja n */
    int ukloni_stotine(int n){

6      /* Ako je broj izmedju -100 i 100 nema cifru desetica pa onda
        vracamo isti taj broj */
        if(n>-100 && n<100)
8            return n;
    }

```

```

10  else
11  {
12      /* U suprotnom vracamo broj sa uklonjenom cifrom stotina */
13
14      /* Odredjujemo znak broja */
15      int znak=(n<0)? -1 : 1;
16
17      /* I nadalje radimo sa apsolutnom vrednoscu broja */
18      n=abs(n);
19
20      return znak*((n/1000)*100 + n%100);
21  }
22
23  /* Funkcija koja vraca znak broja */
24  int znak(int broj){
25      return broj<0?-1:1;
26  }
27
28  int main(){
29
30      int broj;
31
32      while(1){
33
34          /* Ucitavamo broj sa standardnog ulaza */
35          printf("Unesite broj: ");
36          scanf("%d", &broj);
37
38          /* Broj 0 oznacava kraj rada */
39          if(broj == 0)
40              break;
41
42          /* Ispisujemo rezultat, vodeci racuna da program treba da radi
43             ispravno i za negativne brojeve */
44          printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
45      }
46
47      return 0;
48  }
```

### Rešenje 2.138

```

1  #include<stdio.h>
2  #include<math.h>
3
4  int rotacija(int n){
5
6      /* U promenljivoj broj pamtimo originalnu vrednost n */
7      int broj, br = 0, znak;
```



```
8
/* Odredjujemo znak broja */
10 znak=(n<0) ? -1: 1;

12 /* I nadalje radimo sa apsolutnom vrednoscu broja */
n=abs(n);

14 /* U promenljivoj broj cuvamo kopiju broja n */
16 broj=n;

18 /* Ako je broj jednocifren, nema potrebe da ga rotiramo. */
if(n>-10 && n < 10)
20     return n;

22 /* Petljom izdvajamo cifru po cifru, kako bismo dosli do krajnje
   leve cifre broja
   (one koja treba da postane krajnje desna), npr za n = 1234, treba
   da dobijemo 1,
24   zatim da "pomerimo" 234 u levo i da na kraj nalepimo 1 = 2341 */

26 /* Na kraju ove petlje, u n se nalazi najlevlja cifra broja (koja
   treba da postane krajnje desna),
   dok se u br nalazi broj cifara unetog broja */
28 while(n >=10){
    n/=10;
    br++;
30 }

32 /*
34   Levi deo (234) dobijamo kao n%(10^broj_cifara)
   Zatim levi deo pomnozimo sa 10, da bi dobili 2340
36   Zatim na levi deo dodamo desni deo (1) koja se nalazi u
   promenljivoj n
   */
38
39 return znak* ((broj%(int)pow(10, br))*10 + n);
40 }

42 int main(){

43     int n;
44     while(1){

45         /* Ucitavamo broj */
46         printf("Unesite broj: ");
47         scanf("%d", &n);

48         /* Ako je uneta 0, izlazimo iz petlje */
49         if(n == 0)
50             break;

51         /* Stampamo broj rotiran za jedno mesto u levo */
```

## 2 Kontrola toka

---

```
56     printf("%d\n", rotacija(n));
57     }
58
60     return 0;
61 }
```

### Rešenje 2.139

```
1  /*
2  Napisati funkciju koja ispituje da li je dati ceo broj prost.
3  Funkcija treba
4  da vrati 1 ako je broj prost i 0 u suprotnom. Napisati potom glavni
5  program
6  koji za uneti ceo broj n ispisuje prvih n prostih brojeva.
7  */
8
9  #include <stdio.h>
10 #include <math.h>
11
12 int prost (int x) /* 1-broj je prost, 0-broj nije prost */
13 {
14     int i;
15
16     if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
17         return 1;
18
19     if (x%2==0) /* parni brojevi nisu prosti */
20         return 0;
21
22     for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
23         if (x%i==0) /* ako je pronadjen, to znaci da broj nije prost */
24             return 0; /* završavamo funkciju */
25
26     /* ukoliko izvršavanje funkcije dodje do poslednje naredbe return,
27     to znaci da broj nije ispunio nijedan od prethodnih uslova
28     (nije ni 2, ni 3, ni paran, niti ima ijednog delioca), odakle
29     sledi da je prost i zbog toga vracamo 1
30     */
31     return 1;
32 }
33
34 int main()
35 {
36     int n;
37     scanf("%d",&n);
38     int i,j;
39
40     i=1; /* kandidat za prost broj */
41     j=0; /* brojac prostih brojeva */
42     while(j<n)
```

```

41 {
    if (prost(i))          /* ako je broj prost */
43     {
        printf("%d\n", i); /* stampamo ga i */
45         j++;             /* uvecavamo brojac prostih brojeva */
    }
47     i++; /* bilo da je i prost ili ne, uvecavamo ga za 1 i
        nastavljamo sa sledecom iteracijom */
    }
49
51     return 0;
}

```

### Rešenje 2.140

```

1  /*
   Napisati funkciju koja ispituje da li se cifra c nalazi u zapisu
   celog broja x.
3  Napisati potom glavni program koji za uneti ceo broj i unetu cifru
   poziva
   napisanu funkciju i ispisuje odgovarajucu poruku.
5  */

7  #include<stdio.h>
   #include<stdlib.h>

9
11 int sadrzi(int x, int c)
   {
13     char cifra;
        x=abs(x);
        while(x)
15     {
            cifra = x%10;
17             if (cifra==c)
                 return 1;
            x/=10;
19         }
21     return 0;
   }

23 int main()
   {
25     int x;
        int c;
27     printf("Unesi jedan ceo broj i jednu cifru:");
        scanf("%d%d",&x,&c);
29     if (sadrzi(x,c))
            printf("Cifra %d se nalazi u zapisu broja %d\n",c,x);
31     else
            printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
33     return 0;
   }

```

### Rešenje 2.141

```
2  /*
3  a) Napisati funkciju sve_parne_cifre koja ispituje da li se dati ceo
4  broj sastoji isključivo iz parnih cifara. Funkcija treba
5  da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.
6
7  b) Napisati funkciju sve_cifre_jednake koja ispituje da li su sve
8  cifre datog celog broja jednake. Funkcija treba
9  da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.
10
11 c) Napisati potom glavni program koji na uneti ceo broj primenjuje
12 napisane funkcije i ispisuje odgovarajuće poruke.
13
14 Na primer, za uneti broj 222, program treba da ispise:
15 Sve cifre broja su parne.
16 Sve cifre broja su jednake.
17
18 A za uneti broj -284:
19 Sve cifre broja su parne.
20 Broj sadrži različite cifre
21
22 */
23 #include <stdio.h>
24 #include <stdlib.h>
25
26 int sve_parne_cifre(int x) /* funkcija vraća 1 ako su sve cifre broja
27 parne i 0 u suprotnom */
28 {
29     char d;
30     x=abs(x); /* uzimamo apsolutnu vrednost broja za slučaj da je
31 broj negativan */
32     while (x>0)
33     {
34         d=x%10; /* izdvajamo cifru broja */
35
36         if (d%2==1) /* u slučaju da je neparna, to znači da nisu sve
37 cifre broja parne */
38             return 0; /* vraćamo 0 */
39
40         x/=10; /* "uklanjamo" poslednju cifru broja celobrojnim
41 deljenjem sa 10 */
42     }
43
44     return 1; /* ukoliko se while petlja završila, to znači da
45 uslov d%2==1 nije
46 nijednom bio ispunjen i da su sve cifre broja
47 parne; zbog toga
```

```

40             vacamo 1
41         */
42     }
43
44     int sve_cifre_jednake(int x) /* funkcija vraca 1 ako su sve cifre
45         broja jednake i 0 u suprotnom*/
46     {
47         char d;
48         char prva_cifra;
49         x=abs(x);
50         prva_cifra = x%10; /* izdvajamo prvu cifru broja */
51         x/=10;             /* broj delimo sa 10 jer smo prvu cifru vec
52             izdvojili */
53
54         while(x)
55         {
56             d = x%10;
57
58             if (d!=prva_cifra)
59                 return 0;
60
61             x/=10;
62         }
63
64         return 1;
65     }
66
67     main()
68     {
69         int x;
70         int d;
71
72         printf("unesi ceo broj:");
73         scanf("%d", &x);
74
75         if (sve_parne_cifre(x))
76             printf("Sve cifre broja su parne\n");
77         else
78             printf("Broj sadrzi bar jednu neparnu cifru\n");
79
80         if (sve_cifre_jednake(x))
81             printf("Sve cifre broja su jednake\n");
82         else
83             printf("Broj sadrzi razlicite cifre \n");
84     }

```

## Rešenje 2.142

```

/*

```

## 2 Kontrola toka

```
2  Napisati funkciju koja za dva uneta neoznacena broja x i n utvrđuje
   da li je x neki stepen
   broja n. Ukoliko jeste, funkcija vraća izlozilac stepena, a u
   suprotnom vraća -1. Napisati
4  potom glavni program koji testira ovu funkciju.
   */
6
8  #include <stdio.h>
10
12  int je_stepen(unsigned x, unsigned n) /* funkcija vraća izlozilac
   stepena ukoliko broj x jeste neki stepen broja n */
14  {
16      int i=1;
18      int s=n;
20
22      while(s<x)
24      {
26          s=s*n;
28          i++;
30      }
32
34      if (s==x)
36          return i;
38
40      return -1;
42  }
44
46  int main()
48  {
50      unsigned x;
52      unsigned n;
54      int st;
56
58      scanf("%u%u",&x,&n);
60
62      st = je_stepen(x,n);
64
66      if (st!=-1)
68          printf("%u=%u~d\n",x,n,st);
70      else
72          printf("%u nije stepen broja %u\n",x,n);
74
76      return 0;
78  }
```

### Rešenje 2.143

```
/*
2
   Napisati funkciju
4
```

```
double e_na_x(double x, double eps)
6
    koja racuna vrednost  $e^x$  kao parcijalnu sumu reda
8    suma( $x^n/n!$ ), gde indeks n ide od
    od 0 do beskonacno, pri cemu se sumiranje vrsi dok
10   je razlika sabiraka u redu po apsolutnoj vrednosti
    manja od eps. Napisati potom program koji omogucuje
12   korisniku da unese jedan realan broj x i ispisuje
    vrednost  $e^x$ .
14
    */
16
    #include<stdio.h>
18   #include<math.h>

20   double e_na_x(double x, double eps)
    {
22       double s=1;
        double clan=1;
24       int n=1;

26       /*
        parcijalnu sumu formiramo tako sto u svakoj iteraciji petlje
28       promenljivoj s dodamo jedan sabirak sume oblika  $(x^n)/n!$  koji
        cuvamo u promenljivoj clan

        svaki sabirak mozemo da dobijemo na osnovu prethodnog tako sto
32       ga pomnozimo sa x i podelimo sa n, koje predstavlja redni broj
        sabirka u sumi

        prvi sabirak (kome odgovara  $n=0$ ) iznosi 1; zbog toga promenljive
36       s i clan inicijalizujemo na vrednost 1

        sumiranje se sprovodi dogod je sabirak po apsolutnoj vrednosti
38       veci od trazene tacnosti eps
40       */

42       do
        {
44         clan = (clan*x)/n;
         s += clan;
46         n++;
        } while(fabs(clan)>eps);
48
        return s;
50   }

52   int main()
    {
54       double x,eps;
        printf("x=");
56       scanf("%lf", &x);
```

```
printf("eps=");
58 scanf("%lf", &eps);

60 printf("e~%f=%f\n", x, e_na_x(x,eps));
return 0;
62 }
```

### Rešenje 2.144

```
/*
2 Za dati broj moze se formirati niz tako da je svaki sledeci clan niza
dobijen
kao suma cifara prethodnog clana niza. Broj je srecan ako se dati niz
zavrsava sa
4 jedinicom. Napisati program koji za uneti broj odredjuje da li je
srecan.
Na primer:
6 - broj 1234 je srecan jer je zbir njegovih cifara 10, dalje zbir
cifara broja 10 je 1.
- broj 999 nije srecan jer je njegov zbir cifara 27, zbir cifara
broja 27 je 9.
8 - broj 991 je srecan, zbir njegovih cifara je 19, zbir cifara broja
19 je 10, zbir cifara
broja 10 je 1.
10 - broj 372 nije srecan, zbir njegovih cifara je 12, zbir cifara broja
12 je 3

12 Napisati funkciju koja vraca 1 ako je broj srecan, a 0 u suprotnom.

14 Napisati program koji omogucava korisniku da unese prirodan broj,
poziva funkciju
i ispisuje da li je dati broj srecan. Potom traziti od korisnika da
unese prirodan
16 broj n i ispisati sve srecne brojeve od 1 do n.
*/
18
#include<stdio.h>
20
int zbir_cifara(int x)
22 {
    int s=0;
    char cifra;
    while(x)
    24 {
        cifra = x%10;
        s+=cifra;
        x/=10;
    }
    30 return s;
    32 }
```



```

34 int srecan(int x)
35 {
36     int s; /* promenljiva s sadrzi sumu cifara */
37
38     do
39     {
40         s=zbir_cifara(x);
41         x=s; /* kada izracunamo sumu cifara, dodeljujemo je promenljivoj
42             x jer iz te promenljive izdvajamo cifre u funkciji zbir_cifara
43             */
44     } while(x>=10);
45
46     return (x==1);
47 }
48
49 int main()
50 {
51     unsigned n;
52     int i;
53     printf("Unesi jedan neoznaceni broj:");
54     scanf("%u",&n);
55
56     for(i=1;i<=n;i++)
57         if (srecan(i))
58             printf("%d je srecan\n", i);
59
60     return 0;
61 }

```

### Rešenje 2.145

```

/*
2 . a) Napisati funkciju
3
4     int konverzija (int c)
5
6 koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.
7
8 b) Napisati program koji omogućava korisniku da unese niz karaktera
9 sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
10 Na primer, za uneti tekst "Kolokvijum iz Progi je 1.12." program
11 treba da ispise "KOLOVKIJUM IZ pROGI JE 1.12."
12
13 */
14 #include <stdio.h>
15
16 int konverzija(int c)
17 {
18     /* ključna rec return vraća povratnu vrednost funkcije (ako je ima)
19     */

```

```
/* i završava izvršavanje funkcije */
20
    if (c>='A' && c<='Z')
22         return c+'a'-'A';

    if (c>='a' && c<='z')
24         return c-'a'+'A';

26     return c;
28 }

30 int main()
{
32     int c;

34     while((c=getchar())!=EOF) /* korisnik unosi karakter po karakter
        do konstante EOF */
        putchar(konverzija(c)); /* funkcija putchar ispisuje jedan
36                                karakter na standardni izlaz */

38     return 0;
}
```

### Rešenje 2.146

```
1 #include <stdio.h>

3 /* Funkcija int zapis(int x, int y) proverava da li su dva cela broja
    napisana
    * pomocu istih cifara, kao i da li se te cifre pojavljuju
5    * isti broj puta.
    * Ideja je sledeca:
7    * iz broja x izdvajaju se redom cifra po cifra s kraja,
    * a zatim se svaka takva cifra trazi i u broju y.
9    * Ukoliko postoji, eliminiše se prvi put kada se pojavi (dakle,
        samo jednom).
    * Ukoliko su sve cifre iste (**redosled nije bitan**),
11    * na kraju će i iz x i iz y biti sve cifre eliminisane",
    * te ostaju nule u oba broja.
13    *
    * Broj novo_y formira se, zbog jednostavnosti, pomocu Heronovog
        obrasca.
15    * Ovaj postupak obradjen je u okviru funkcije int izbaci_cifru(int
        y, int cifra).
    */

17 int izbaci_cifru(int y, int cifra) {
19     int novo_y = 0;
21     int indikator = 0;
    int izdvojena_cifra;
```

```
23 while(y) {
24
25     izdvojena_cifra = y % 10;
26     /* U slucaju da se cifra razlikuje od one koju treba eliminisati,
27      * ili ukoliko je jedna cifra vec eliminisana =>
28      * tekucu cifru ukljuciti prilikom formiranja novog y
29      */
30     if(izdvojena_cifra != cifra || indikator)
31
32         /* Heronov obrazac.
33          * Menja poredak cifara, ali on u ovom slucaju i nije bitan.
34          */
35         novo_y = novo_y*10 + izdvojena_cifra;
36     else
37
38         /* U slucaju da je cifra vec eliminisana,
39          * ne treba je opet eliminisati.
40          * Za svaku pojavu cifre iz x,
41          * eliminiše se jedna odgovarajuća pojava
42          * te cifre iz y.
43          */
44         indikator = 1;
45
46     y /= 10;
47 }
48
49 return novo_y;
50 }
51
52 int zapis(int x, int y) {
53
54     /* Cifra koja se izdvaja iz x, a onda eliminiše iz y */
55     int cifra;
56
57     /* U slucaju da su prosledjeni brojevi negativni */
58     x = abs(x);
59     y = abs(y);
60
61     while(x) {
62
63         cifra = x % 10;
64         x /= 10;
65
66         y = izbaci_cifru(y, cifra);
67
68         /* otkomentarisati donju liniju radi lakseg pracenja rada
69          programa: */
70         // printf("Iz x izdvojeno: %d\n\tx = %d, y = %d\n\n", cifra, x, y
71         );
72     }
```

## 2 Kontrola toka

---

```
73     return (x == 0 && y == 0);
74 }
75
76 int main() {
77
78     int x, y;
79     printf("Unesite dva cela broja: ");
80     scanf("%d%d", &x, &y);
81
82     if(zapis(x, y))
83         printf("Uslov je ispunjen!\n");
84     else
85         printf("Uslov nije ispunjen!\n");
86
87     return 0;
88 }
```

### Rešenje 2.147

```
1  #include <stdio.h>
2
3  /* Funkcija racuna faktorijel broja.
4   * Faktorijel formiramo mnozenjem sa trenutnom vrednoscu broja x,
5   * a zatim smanjujuci tu vrednost za 1.
6   * Ukoliko je x = 5, f = 5 * 4 * 3 * 2 * 1
7   */
8  int faktorijel(int x) {
9
10     int f = 1;
11     while(x) {
12         f *= x;
13         x--;
14     }
15     return f;
16 }
17
18 int main() {
19
20     int x, y;
21
22     printf("Unesite dva broja: ");
23     scanf("%d%d", &x, &y);
24
25     /* Provera uslova.
26     *
27     * Faktorijel je veoma brza funkcija, tj.
28     * s povecanjem broja x, drasticno brzo uvecava se i vrednost x!.
29     * Tip podatka int ima ogranicenje u velicini broja koji moze da
30     * sadrzi.
31     * Za 13! i vece, int ne bi mogao da sacuva sve cifre potrebne za
32     * zapis tako velikog broja,
```

```

31  * te bi doslo do prekoracenja.
32  *
33  * Slicno, faktorijel nije definisan nad skupom negativnih celih
    brojeva.
34  */
35  if(x < 0 || y < 0 || x > 12 || y > 12) {
    printf("Greska: pogresan unos!\n");
37  }
    else{
39  printf("%d\n", faktorijel(x) + faktorijel(y));
    }
41  return 0;
}

```

### Rešenje 2.148

```

1  #include <stdio.h>
    #include <stdlib.h>
3
    /* Funkcija proverava da li se
5  * cifre u zapisu broja nalaze u rastucem poretku.
    *
7  * Situacija od interesa je kada za dve uzastopne cifre to nije
    slucaj.
    * Tada ne treba proveravati i za ostale cifre,
9  * vec odmah prekinuti izvršavanje funkcije.
    *
11 * Ukoliko funkcija nije ranije prekinuta,
    * to znaci da cifre jesu u rastucem poretku
13 * (odnosno, kako izdvajamo cifre od nazad, u stvari proveravamo
    opadajuci poredak),
    * te treba vratiti 1.
15 */
17 int rastuce(int n) {
19     int tekuca_cifra;
    int prethodna_cifra;
21
    n = abs(n);
23
    /* Prvu cifru (odnosno, poslednju u zapisu broja)
25     * izdvajamo izvan petlje
    * kako bismo mogli da je poredimo sa narednom
27     */
    tekuca_cifra = n % 10;
29     n /= 10;
31     while(n) {
33         /* Cifra koja je bila tekuca u prethodnoj iteraciji petlje,

```

```
35     * u novoj iteraciji postaje prethodna.
36     *
37     * Novoizdvojena cifra je tekuca.
38     */
39     prethodna_cifra = tekuca_cifra;
40     tekuca_cifra = n % 10;
41
42     /* Ukoliko smo naisli na cifre koje kvare rastuci poredak,
43     * prekidamo izvršavanje funkcije sa odgovarajucom povratnom
44     * vrednoscu 0.
45     */
46     if(prethodna_cifra < tekuca_cifra)
47         return 0;
48
49     /* Inace, nastavljamo sa izdvajanjem cifara */
50     n /= 10;
51 }
52
53 return 1;
54 }
55
56 int main() {
57
58     int x;
59     printf("Unesite broj: ");
60     scanf("%d", &x);
61
62     if(rastuce(x))
63         printf("Cifre su u rastucem poretku!\n");
64     else
65         printf("Cifre nisu u rastucem poretku!\n");
66
67     return 0;
68 }
```

### Rešenje 2.149

```
1 #include <stdio.h>
2
3 /* Funkcija racuna broj x na n-ti stepen */
4 int stepen(int x, int n) {
5
6     int i;
7     /* Promenljiva u kojoj se cuva proizvod broja x sa samim sobom, n
8     puta */
9     int st = 1;
10
11     for(i = 1; i <= n; i++)
12         st *= x;
13
14     return st;
15 }
```

```
15 }
16
17 /* Funkcija proverava da li je broj Armstrongov. */
18 int armstrong(int x) {
19     /* u y se cuva zbir i-tih stepena cifara */
20     int y;
21     /* stepen za koji se proverava */
22     int i = 1;
23     /* prilikom izdvajanja cifara, broj x se menja,
24      * te treba imati promenljivu koja cuva pravu vrednost x
25      */
26     int original = x;
27
28     do {
29
30         y = 0;
31         /* Racunamo i-te stepene za svaku cifru,
32          * i istovremeno te stepen sabiramo.
33          * Rezultat pamtimo u promenljivoj y.
34          */
35         while(x) {
36
37             y += stepen(x % 10, i);
38             x /= 10;
39         }
40
41         /* x je sada promenjen, pa ga treba vratiti na pravu vrednost. */
42         x = original;
43         i++;
44     } while(y < x); /* Petlju vrtimo sve dok je zbir stepena cifara
45         manji od datog broja. */
46
47     /* Ukoliko smo nasli i, takvo da je zbir i-tih stepena cifara
48      * jednak upravo broju x, takav broj je Armstrongov,
49      * te izraz x == y vraca 1.
50      *
51      * Inace, vraca 0, tj. broj nije Armstrongov.
52      */
53     return x == y;
54 }
55
56 int main() {
57
58     int x;
59     printf("Unesite broj: ");
60     scanf("%d", &x);
61
62     if(armstrong(x))
63         printf("Broj je Armstrongov!\n");
64     else
```

```
65     printf("Broj nije Armstrongov!\n");
67     return 0;
}
```

### Rešenje 2.150

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   /* Funkcija proverava da li su dve uzastopne cifre
   * razlicite parnosti.
   *
   * Interesantna situacija je ukoliko su dve uzastopne cifre
   * obe parne, odnosno obe neparne.
   * Ovaj uslov svodimo na poredjenje njihovih ostataka pri deljenju sa
   * 2:
   * ukoliko su ostaci isti, cifre su iste parnosti,
11  * te ne treba dalje proverati da li je uslov zadovoljen,
   * vec odmah prekinuti sa izvršavanjem funkcije.
13  *
   * Ukoliko dve uzastopne cifre ni u jednom slucaju nisu bile iste
   * parnosti,
15  * a izdvojene su sve cifre iz broja x,
   * uslov je ispunjen, pa funkcija vraća 1.
17  */
   int par_nepar(int x) {
19
       int prethodna_cifra;
21       int tekuca_cifra;

23       /* u slucaju da je uneti broj negativan */
       x = abs(x);

25
       /* jednu cifru izdvajamo van petlje
       * kako bismo mogli da je odmah u petlji poredimo sa narednom
       */
27
       prethodna_cifra = x % 10;
       x /= 10;
31
       while(x) {
33
           tekuca_cifra = x % 10;

35
           if(tekuca_cifra % 2 == prethodna_cifra % 2)
               return 0;
37
           /* tekuca cifra postaje prethodna cifra za narednu iteraciju */
           prethodna_cifra = tekuca_cifra;
39
           x /= 10;
41
       }
}
```



```
43     return 1;
44 }
45
46
47 int main() {
48
49     int x;
50     printf("Unesite broj: ");
51     scanf("%d", &x);
52
53     if(par_nepar(x))
54         printf("Broj ispunjava uslov!\n");
55     else
56         printf("Broj ne ispunjava uslov!\n");
57
58     return 0;
59 }
```

### Rešenje 2.151

```
1  #include <stdio.h>
2
3  /* Funkcija broji koliko puta se realan broj x
4   * javlja u nizu unetih brojeva sa tastature.
5   *
6   * Brojevi se unose sve do pojave 0,
7   * pa treba koristiti do..while petlju,
8   * kako bi bar jedan broj bio unet (makar bio i 0).
9   */
10 int prebrojavanje(float x) {
11
12     /* y prihvata uneti broj sa tastature */
13     float y;
14     /* br_pojavljivanja je broj koji broji koliko puta se broj x
15      * javlja u unetom nizu brojeva */
16     int br_pojavljivanja = 0;
17
18     printf("Unesite brojeve: ");
19     do {
20
21         /* Unosimo broj. */
22         scanf("%f", &y);
23
24         /* Poredimo uneti broj sa datim brojem.
25          * Ukoliko je unet bas trazeni broj,
26          * uvecavamo brojac.
27          */
28         if(x == y)
29             br_pojavljivanja++;
30     } while(y); /* Sve dok nije uneta 0 */
31 }
```

```
32     return br_pojavljivanja;
33 }
34
35 int main() {
36     float x;
37     int br_pojavljivanja;
38
39     printf("Unesite broj x: ");
40     scanf("%f", &x);
41
42     br_pojavljivanja = prebrojavanje(x);
43     printf("Broj pojavljivanja broja %.2f je: %d\n", x,
44           br_pojavljivanja);
45
46     return 0;
47 }
```

### Rešenje 2.152

```
1  #include <stdio.h>
2
3  /* Funkcija racuna n-ti clan Fibonacijevog niza.
4   * Clanovi ovog niza zadaju se rekurzivno tj. u zavisnosti od
5   * prethodnih clanova.
6   * Fibonacijevi brojevi od 0. do 47. se mogu smestiti u tip int, a
7   * kako n moze uzimati vrednosti
8   * od 1 do 50, povratni tip funkcije je long int.
9   */
10 long int fibonaci(int n) {
11
12     int i;
13
14     /* f0 i f1 su prva dva clana niza */
15     int f0 = 1;
16     int f1 = 1;
17     /* promenljiva u kojoj se cuvaju opsti clanovi: n+2, n+1. i n-ti
18     clan */
19     long int fn2, fn1, fn;
20
21     /* ukoliko treba vratiti multi ili prvi clan,
22     * njih ne treba racunati
23     * jer su vec dati.
24     */
25     if(n == 0 || n == 1)
26         return 1;
27
28     /* postavljamo prethodne clanove niza */
29     fn = f0;
30     fn1 = f1;
```

```

29  /* racunamo od drugog clana, pa dok ne dodjemo do n-tog */
    for(i = 2; i <= n; i++) {

31      /* izracunamo n+2-i clan niza sabiranjem prethodna dva clana */
        fn2 = fn1 + fn;
33      /* promenimo prethodne clanove niza, zbog naredne iteracije */
        fn = fn1;
35        fn1 = fn2;
    }

37    return fn2;
39 }

41 int main() {

43     int n;
    printf("Unesite broj n: ");
45     scanf("%d", &n);

47     /* Provera vrednosti za broj n */
    if(n < 0 || n > 50) {
49         printf("Greska: nedozvoljena vrednost!\n");
    }
51     else{
        printf("%ld\n", fibonaci(n));
53     }

55     return 0;
}

```

### Rešenje 2.153

```

1  #include <stdio.h>

3  /* Funkcija vraca karakter koji se u abecedi
   * nalazi k mesta pre datog karaktera c
   */
5  char sifra(char c, int k) {

7      /* Ukoliko je uneto malo slovo ... */
9      if(c >= 'a' && c <= 'z')
        /* Pri tome karakter koji je k pozicija pre datog karaktera
        ispada iz opsega malih slova ... */
11     if(c-k < 'a')
        /* Treba krenuti s drugog kraja abecede, racunajuci i
        preskocena slova.
13         *
        * Na primer, ukoliko je c = 'b' i k = 2
15         * Jedan karakter pre 'b' je 'a'.
        * Dva karaktera pre 'b' je 'z' (kruzno).
17         *

```

```
19      * Karakter iz prvog dela abecede, koji je preskocen, je 'a'.
      * Broj preskocenih karaktera iz prvog dela abecede
      * racunamo tako sto izracunamo c - 'a' (rastojanje od datog
      karaktera do malog slova a)
21      * sto je u ovom slucaju 'b' - 'a' = 1.
      *
23      * Ostatak karaktera do k ispisujemo, ali gledavsi unazad od z.
      * Zato racunamo k - (c - 'a') - 1.
25      *
      * Od k oduzimamo rastojanje izmedju c i 'a',
27      * kako bismo dobili preostali broj karaktera koji treba
      preskociti.
      */
29      return 'z' - (k - (c - 'a') - 1);
      else
31      /* U suprotnom, karakter ne ispada iz opsega malih slova, te je
      dovoljno bas njega i vratiti */
      return c-k;
33
      /* Ukoliko je uneto veliko slovo ... */
35      else if(c >= 'A' && c <= 'Z')
          if(c-k < 'A')
37              return 'Z' - (k - (c - 'A') - 1);
          else
39              return c-k;
41
      return c;
43 }

45 int main() {
47     int k;
      char c;
49
      printf("Unesite broj k: ");
51     scanf("%d", &k);

53     printf("Unesite tekst (CTRL + D za prekid): ");
      while((c = getchar()) != EOF)
55         putchar(sifra(c, k));

57     return 0;
}
```

# 3

## Predstavlanje podataka

### 3.1 Nizovi

**Zadatak 3.1** Tekst

[Rešenje [3.1](#)]

**Zadatak 3.2** Tekst

[Rešenje [3.2](#)]

**Zadatak 3.3** Tekst

[Rešenje [3.3](#)]

**Zadatak 3.4** Tekst

[Rešenje [3.4](#)]

**Zadatak 3.5** Tekst

[Rešenje [3.5](#)]

**Zadatak 3.6** Tekst

[Rešenje [3.6](#)]

**Zadatak 3.7** Tekst

[Rešenje [3.7](#)]

#### Zadatak 3.8 Tekst

[Rešenje 3.8]

#### Zadatak 3.9 Tekst

[Rešenje 3.9]

#### Zadatak 3.10 Tekst

[Rešenje 3.10]

**Zadatak 3.11** Sa standardnog ulaza se unosi dimenzija niza (broj manji od 100), a zatim i njegovi elementi. Napisati program koji kvadrira sve negativne elemente niza i ispisuje rezultujući niz.

##### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 6
Unesite elemente niza:
12.34 -6 1 8 32.4 -16
12.34 36 1 8 32.4 256
```

##### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 9
Unesite elemente niza:
-8.25 6 17 2 -1.5 1 -7 2.65 -125.2
68.0625 6 17 2 2.25 1 49 2.65 15675.04
```

##### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza:
9.53 5 1 4.89
9.53 5 1 4.89
```

[Rešenje 3.114]

**Zadatak 3.12** Sa standardnog ulaza se učitava dimenzija niza (broj manji od 100), elemente niza i jedan ceo broj  $k$ . Napisati program koji štampa indekse elemenata koji su deljivi sa  $k$ .

##### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 10 14 86 20
Unesite broj k: 5
0 3
```

##### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 6 14 8 9
Unesite broj k: 5
U nizu nema elemenata koji su deljivi brojem 5!
```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 6
|| Unesite elemente niza: 8 9 11 -4 8 11
|| Unesite broj k: 2
|| 0 3 4

```

[Rešenje 3.115]

**Zadatak 3.13** Napisati program koji sa standardnog ulaza učitava dimenziju niza (broj manji od 100) i elemente niza, a zatim štampa niz u kojem su najveći i najmanji element niza razmenili mesta.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 5
|| Unesite elemente niza: 8 -2 11 19 4
|| 8 19 11 -2 4

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 10
|| Unesite elemente niza:
|| 46 -2 51 8 -5 66 2 8 3 14
|| 46 -2 51 8 66 -5 2 8 3 14

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 145
|| Greska: pogresan unos!

```

[Rešenje 3.116]

**Zadatak 3.14** Napisati program koji učitava karaktere sa ulaza (najviše njih 100) sve do pojave karaktera \*, a zatim ih ispisuje u redosledu suprotnom od redosleda čitanja.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karakter: a
|| Unesite karakter: 8
|| Unesite karakter: 5
|| Unesite karakter: Y
|| Unesite karakter: I
|| Unesite karakter: o
|| Unesite karakter: ?
|| Unesite karakter: *
|| ? o I Y 5 8 a

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karakter: g
|| Unesite karakter: g
|| Unesite karakter: 2
|| Unesite karakter: 2
|| Unesite karakter: )
|| Unesite karakter: )
|| Unesite karakter: *
|| ) ) 2 2 g g

```

### 3 Predstavljanje podataka

---

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: U  
|| Unesite karakter: 4  
|| Unesite karakter: a  
|| Unesite karakter: u  
|| Unesite karakter: *  
|| u a 4 U
```

[Rešenje 3.117]

**Zadatak 3.15** Napisati program koji za dva cela broja  $x$  i  $y$  koja se učitavaju sa standardnog ulaza proverava da li se zapisuju pomoću istih cifara. Napomena: iskoristiti nizove za čuvanje broja pojavljivanja svake od cifara.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 251 125  
|| Brojevi se zapisuju istim ciframa!
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 8898 9988  
|| Brojevi se ne zapisuju istim ciframa!
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: -7391 1397  
|| Brojevi se zapisuju istim ciframa!
```

[Rešenje 3.118]

**Zadatak 3.16** Sa standardnog ulaza se učitava dimenzija niza (broj manji od 100), zatim i elementi dvaju nizova  $a$  i  $b$ . Napisati program koji formira i ispisuje niz  $c$  čiju prvu polovinu čine elementi niza  $b$ , a drugu polovinu elementi niza  $a$ .

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 3  
|| Unesite elemente niza a: 4 -8 32  
|| Unesite elemente niza b: 5 2 11  
|| 5 2 11 4 -8 32
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 4  
|| Unesite elemente niza a: 1 0 -1 0  
|| Unesite elemente niza b: 5 5 5 3  
|| 5 5 5 3 1 0 -1 0
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 145  
|| Greska: pogresan unos!
```

[Rešenje 3.119]



**Zadatak 3.17** Sa standardnog ulaza se unosi dimenzija niza  $a$  (broj manji od 100), a zatim i njegovi elementi. Napisati program koji od datog niza formira niz  $b$  u koji ulaze elementi niza  $a$  koji se pojavljuju tačno 3 puta.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza a:
4 11 4 6 8 4 6 6
Elementi niza b: 4 6
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 13
Unesite elemente niza a:
-8 26 7 2 1 1 7 2 2 2 7 5 1
Elementi niza b: 7 1
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 2
Unesite elemente niza a:
9 5
Elementi niza b:
```

[Rešenje 3.17]

**Zadatak 3.18** Sa standardnog ulaza se, redom, učitavaju dimenzija i elementi dva niza  $a$  i  $b$ . Napisati program koji određuje njihovu uniju, presek i razliku (redosled prikaza elemenata nije bitan). Pretpostaviti da će nizovi imati manje od 100 elemenata.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 5
Unesite elemente niza a: 2 8 1 5 2
Unesite broj elemenata niza b: 3
Unesite elemente niza b: 5 7 8
Unija: 2 8 1 5 2 5 7 8
Presek: 5
Razlika: 2 1 2
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 3
Unesite elemente niza a: 11 4 4
Unesite broj elemenata niza b: 2
Unesite elemente niza b: 18 9
Unija: 11 4 4 18 9
Presek:
Razlika: 11 4 4
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 6
Unesite elemente niza a: 12 7 9 12 5 1
Unesite broj elemenata niza b: 4
Unesite elemente niza b: 1 12 22 12
Unija: 12 7 9 12 5 1 1 12 22 12
Presek: 12 12 1
Razlika: 7 9 5
```

[Rešenje 3.18]

### 3 Predstavljanje podataka

---

**Zadatak 3.19** Napisati program koji učitava dimenziju niza (broj manji od 100) i elemente niza, a zatim formira i ispisuje niz koji se dobija izbacivanjem svih neparnih elemenata niza. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 4  
|| Unesite elemente niza: 8 9 15 12  
|| 8 12
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 6  
|| Unesite elemente niza: 21 5 3 22 19 188  
|| 22 188
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 4  
|| Unesite elemente niza: 133 129 121 101
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 8  
|| Unesite elemente niza: 15 -22 -23 13 18 46 14 -31  
|| -22 18 46 14
```

[Rešenje 3.19]

**Zadatak 3.20** Napisati program koji učitava dimenziju niza (broj manji od 100) i elemente niza, a zatim formira i ispisuje niz koji se dobija izbacivanjem svih elemenata koji su prosti brojevi. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Napomena: brojeve -1 i 1 smatrati prostim.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 5  
|| Unesite elemente niza: 11 5 6 48 8  
|| 6 48 8
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 4  
|| Unesite elemente niza: 11 5 19 21  
|| 21
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 5  
|| Unesite elemente niza: 12 18 9 31 7  
|| 12 18 9
```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 3
|| Unesite elemente niza: -31 11 -19

```

*Primer 5*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 5
|| Unesite elemente niza: -2 15 -11 8 7
|| 15 8

```

[Rešenje 3.20]

**Zadatak 3.21** Napisati funkciju *int prebrojavanje(int a[], int n)* koja izračunava broj elemenata niza celih brojeva *a* dužine *n* koji su manji od poslednjeg elementa niza. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 4
|| Unesite elemente niza: 11 2 4 9
|| 2

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 7
|| Unesite elemente niza: 7 2 1 14 65 2 8
|| 4

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 5
|| Unesite elemente niza: 25 18 29 30 14
|| 0

```

[Rešenje 3.21]

**Zadatak 3.22** Napisati funkciju *int prebrojavanje(int a[], int n)* koja izračunava broj parnih elemenata niza celih brojeva *a* dužine *n* koji prethode maksimalnom elementu niza. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 4
|| Unesite elemente niza: 11 2 4 9
|| 0

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 7
|| Unesite elemente niza: 7 2 1 14 65 2 8
|| 2

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 5
|| Unesite elemente niza: 25 18 29 30 14
|| 1

```

### 3 Predstavljanje podataka

---

[Rešenje 3.22]

**Zadatak 3.23** Napisati funkciju *int prebrojavanje\_cifre(char s[], int n)* koja izračunava broj cifara u nizu karaktera *a* dužine *n*. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
4
+
4
u
8
Broj cifara je: 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza:
J
M
a
5
5
-
2
Broj cifara je: 3
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza:
e
k
F
Broj cifara je: 0
```

[Rešenje 3.23]

**Zadatak 3.24** Napisati funkciju *int zbir(int a[], int n, int i, int j)* koja računa zbir elemenata niza celih brojeva *a* dužine *n* od pozicije *i* do pozicije *j*. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: 11 5 6 48 8
Unesite vrednosti za i i j: 0 2
Zbir je: 22
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza: -2 8 1
Unesite vrednosti za i i j: 8 12
Greska: nekorektne vrednosti granica!
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza: -2 5 9 11 6 -3 -4
Unesite vrednosti za i i j: 2 5
Zbir: 23
```

[Rešenje 3.24]

**Zadatak 3.25** Napisati funkciju *float zbir\_pozitivnih(float a[], int n, int k)* koja izračunava zbir prvih  $k$  pozitivnih elemenata realnog niza  $a$  dužine  $n$ . Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza:
2.34 1 -12.7 5.2 -8 -6.2 7 14.2
Unesite vrednost za k: 3
Zbir je: 8.54
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza:
-6.598 -8.14 -15
Unesite vrednost za k: 4
Zbir je: 0.00
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza:
-35.11 5.29 -1.98 12.1 12.2 -3.33 -4.17
Unesite vrednost za k: 15
Zbir: 29.59
```

[Rešenje 3.25]

**Zadatak 3.26** Napisati funkciju *void kvadriranje(float a[], int n)* koja kvadrira elemente realnog niza  $a$  dužine  $n$  koji se nalaze na parnim pozicijama. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza:
2.34 1 -12.7 5.2 -8 -6.2 7 14.2
5.4756 1 161.29 5.2 64 -6.2 49 14.2
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza:
-6 -8.14 -15
36 -8.14 225
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 1
Unesite elemente niza:
-35.11
1232.71
```

[Rešenje 3.26]

**Zadatak 3.27 Filip-Janicic?** Napisati funkciju (i program koji je testira) koja:

- (a) proverava da li dati niz sadrži dati broj;
- (b) pronalazi indeks prve pozicije na kojoj se u nizu nalazi dati broj (-1 ako niz ne sadrži broj).
- (c) pronalazi indeks poslednje pozicije na kojoj se u nizu nalazi dati broj (-1 ako niz ne sadrži broj).
- (d) izračunava zbir svih elemenata datog niza brojeva;
- (e) izračunava prosek (aritmetičku sredinu) svih elemenata datog niza brojeva;
- (f) izračunava najmanji element datog elemenata niza brojeva;
- (g) određuje poziciju najvećeg elementa u nizu brojeva (u slučaju više pojavljivanja najvećeg elementa, vratiti najmanju poziciju);
- (h) proverava da li je dati niz brojeva uređen neopadajuće

**Zadatak 3.28 Filip-Janicic?** Napisati funkciju (i program koji je testira) koja:

- (a) izbacuje poslednji element niza;
- (b) izbacuje prvi element niza (napisati varijantu u kojoj je bitno očuvanje redosleda elemenata i varijantu u kojoj nije bitno očuvanje redosleda);
- (c) izbacuje element sa date pozicije k ;
- (d) ubacuje element na kraj niza;
- (e) ubacuje element na početak niza;
- (f) ubacuje dati element x na datu poziciju k ;
- (g) izbacuje sva pojavljivanja datog elementa x iz niza.

Napomena: funkcija kao argument prima niz i broj njegovih trenutno popunjenih elemenata, a vraća broj popunjenih elemenata nakon izvođenja zahtevane operacije.

[Rešenje 3.37]

**Zadatak 3.29 Filip-Janicic?** Napisati funkciju (i program koji je testira) koja:

- (a) određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva;
- (b) određuje dužinu najvećeg neopadajućeg podniza datog niza celih brojeva;
- (c) određuje da li se jedan niz javlja kao podniz uzastopnih elemenata drugog;
- (d) određuje da li se jedan niza javlja kao podniz elemenata drugog (elementi ne moraju da budu uzastopni, ali se redosled pojavljivanja poštuje);
- (e) obrće dati niz brojeva;
- (f) rotira sve elemente datog niza brojeva za k pozicija ulevo;
- (g) rotira sve elemente datog niza brojeva za k pozicija udesno;
- (h) izbacuje višestruka pojavljivanja elemenata iz datog niza brojeva (napisati varijantu u kojoj se zadržava prvo pojavljivanje i varijantu u kojoj se zadržava poslednje pojavljivanje).
- (i) spaja dva niza brojeva koji su sortirani neopadajući u treći niz brojeva koji je sortiran neopadajući.

[Rešenje 3.37]

**Zadatak 3.30** Napisati funkciju `int f3(int a[], int n, int b[], int m)` i ispituje da li prvi sadrži bar dva broja koji se pojavljuju u drugom nizu. Povratna vrednost je dakle, 0, ili 1. Testirati pozivom u main-u. Maksimalna dužina niza je 100 elemenata.

[Rešenje 3.37]

**Zadatak 3.31** Napisati C funkciju koja u proslećenom nizu eliminiše sve brojeve koji nisu deljivi svojim indeksom (vrednost na indeksu 0 zadržati, jer nije dozvoljeno deljenje sa 0). Niz reorganizovati, tako da nema *rupa* koje su nastale eliminacijom elemenata. Kao rezultat funkcije vratiti novu dimenziju niza.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 10
|| Unesite elemente niza:
|| 4 2 1 6 7 8 10 2 16 3
|| 4 2 6 16

```

[Rešenje 3.37]

**Zadatak 3.32** Implementirati funkciju `int min_max(int a[], int n)` koja prihvata celobrojni niz, pronalazi indekse najmanjeg i najvećeg elementa tog niza koristeći samo jedan prolaz (jednu petlju), a zatim kao povratnu vrednost vraća manji od ta dva indeksa.

Program testirati pozivom funkcije iz main programa i ispisom rezultata na standardni izlaz, pri čemu korisnik sa standardnog ulaza unosi niz dužine 10 elemenata.

[Rešenje 3.37]

**Zadatak 3.33** Napisati funkciju `void brojanje(int a[], int brojac[], int N)` čiji su argumenti `a` i `brojac` celobrojni nizovi dimenzije `N`. Vrednosti elemenata niza `a` su između 0 i `N - 1`. Funkcija izračunava elemente niza `brojac` tako da je `brojac[i]` jednak broju pojavljivanja broja `i` u nizu `a`. Program testirati pozivom funkcije iz main programa - korsnik učitava broj `N` i potom niz `a` dužine `N`, potom poziva funkciju i potom na standardnom izlazu izpisuje dobijeni niz.

[Rešenje 3.37]

**Zadatak 3.34** Napisati funkciju `int ind(int a[], int n)` koja kao povratnu vrednost ima indeks onog elementa niza koji je po vrednosti najbliži srednjoj vrednosti onih elemenata niza brojeva koji su deljivi sa 3.

Program testirati pozivom funkcije iz main programa i ispisom rezultata na standardni izlaz, pri čemu korisnik sa standardnog ulaza unosi broj `n`, a zatim niz od `n` celih brojeva (maksimalna dimenzija niza je 100 elemenata).

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 5  
|| Unesite elemente niza:  
|| 1 2 3 4 5  
|| 2
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj elemenata niza: 5  
|| Unesite elemente niza:  
|| 3 6 2 4 7  
|| 3
```

[Rešenje 3.37]

**Zadatak 3.35** Sa standardnog ulaza se unosi jedna linija teksta. Napisati program koji prikazuje koliko puta se javilo svako od slova engleskog alfabeta (ne praviti razliku između velikih i malih slova).

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| haHJJkL  
|| a:1 b:0 c:0 d:0 e:0 f:0 g:0 h:2 i:0 j:2 k:1 l:1 m:0 n:0 o:0 p:0 q:0 r:0 s:0t:0 u:0 v:0 w:0 x:0 y:0 z:0
```



*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
```

```
|| DanaS j3 _j_utRo laBU78d
```

```
|| a:3 b:1 c:0 d:2 e:0 f:0 g:0 h:2 i:0 j:2 k:0 l:1 m:0 n:1 o:1 p:0 q:0 r:1 s:1t:1 u:2 v:0 w:0 x:0 y:0 z:0
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
```

```
|| Sao PaoLo 1998 _JuZna Amerika90
```

```
|| a:5 b:0 c:0 d:2 e:1 f:0 g:0 h:0 i:1 j:1 k:1 l:1 m:1 n:1 o:3 p:1 q:0 r:1 s:1t:0 u:1 v:0 w:0 x:0 y:0 z:0
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
```

```
|| Ixxx kk 3yyy 4qqqq
```

```
|| a:0 b:0 c:0 d:0 e:0 f:0 g:0 h:0 i:1 j:0 k:2 l:0 m:0 n:0 o:0 p:0 q:4 r:0 s:0t:0 u:0 v:0 w:0 x:3 y:3 z:0
```

[Rešenje 3.37]

**Zadatak 3.36** Napisati program koji sa standardnog ulaza učitava 50 celih brojeva i razdvaja ih na parne i neparne tako što parne brojeve upisuje na početak niza, a neparne na kraj niza. Ispisati niz dobijen na taj način. Nije dozvoljeno koristiti dodatne nizove.

[Rešenje 3.37]

**Zadatak 3.37**

- Napisati funkciju `void brojanje(int a[], int brojac[], int N)` čiji su argumenti `a` i `brojac` celobrojni nizovi dimenzije  $N$ . Vrednosti elemenata niza `a` su između  $0$  i  $N - 1$ . Funkcija izračunava elemente niza `brojac` tako da je  $i$ -ti element `brojac[i]` jednak broju pojavljivanja broja  $i$  u nizu `a`.
- Za celobrojni niz `a` dimenzije  $N$  kažemo da je *permutacija* ako sadrži sve brojeve  $i$ :  $0 \leq i \leq N$ . Sastaviti funkciju `int DaLiJePermutacija(int a[], int N)` koja vraća `1` ako je niz `a` permutacija, a `0` inače. (koristiti funkciju `brojanje`).

[Rešenje 3.37]

## 3.2 Rešenja

**Rešenje 3.1**

### 3 Predstavljanje podataka

---

```
/*
2  Napisati program koji racuna skalarni proizvod dva vektora. Svaki
   vektor
   je zadat kao celobrojni niz sa najvise 100 elemenata. Program treba
   da
4  ucita dimenziju nizova (oba niza su iste dimenzije), zatim jedan po
   jedan element niza i da ispise njihov skalarni proizvod na
   standardni
6  izlaz.
   */
8
   #include <stdio.h>
10  #define MAX 100
12
   /*
14  Pretprocesorskom direktivom define uvode se simbolicka imena (u ovom
   slucaju
   MAX) kojima se pridružuje nekakav tekst (u ovom slucaju 100). Pre
   kompilacije,
16  sva pojavljivanja simbolickog imena MAX bice zamenjena pridruzenim
   tekstom
   100. MAX nije promenljiva i za nju se tokom izvršavanja programa ne
   izdvaja
18  memorijski prostor.

20  MAX se u ovom zadatku koristi kao maksimalni broj elemenata niza.
   Ukoliko bismo zeleli
   da izmenimo ovu vrednost, npr. da povecamo sa 100 na 200, sve
22  sto bi bilo neophodno uraditi je da izmenimo tekst sa 100 na 200. Sa
   druge
   strane, da nismo koristili pretprocesorsku direktivu i da smo svaki
   put
24  umesto MAX direktno navodili vrednost 100, morali bismo da je
   izmenimo na svakom
   mestu u kodu.
26
   */
28  int main()
   {
30     int a[MAX];
     int b[MAX];
32     int n;
     int i;
34     int s;

36
     printf("Unesi dimenziju niza:");
38     scanf("%d", &n);

40     if (n<1 || n>100)
```

```

42     {
        printf("Neispravan unos\n");
        return -1;
44     }

46     /*
        prvi element niza ima indeks 0, a poslednji n-1,
48     gde je n broj elemenata niza; elementima niza pristupamo
        preko indeksa; na primer, ako niz a ima 5 elemenata, mozemo
50     im pristupiti pomocu
        a[0], a[1], a[2], a[3], a[4]
52
53     */
54
55     for (i=0; i<n; i++)
56     {
57         printf("a[%d]=",i);
58         scanf("%d", &a[i]);
59     }
60
61     for (i=0; i<n; i++)
62     {
63         printf("b[%d]=",i);
64         scanf("%d", &b[i]);
65     }
66
67     s=0;
68
69     for (i=0; i<n; i++)
70         s = s + a[i]*b[i];
71
72     printf("Skalarni proizvod: %d\n",s);
73     return 0;
74 }

```

### Rešenje 3.2

```

/*
2     Napisati program koji učitava broj elemenata niza (n<=100),
    zatim učitava elemente niza i ispisuje:
4     a) elemente niza koji se nalaze na parnim indeksima
    b) parne elemente niza
6
7     */
8
9     #include <stdio.h>
10    #define MAX 100
11
12    int main()
13    {
14        int a[MAX];

```

### 3 Predstavljanje podataka

```
16  int n;  
    int i;  
  
18  
    printf("Unesi dimenziju niza:");  
20  scanf("%d", &n);  
  
22  if (n<1 || n>MAX)  
  {  
24      printf("Nekorektan unos\n");  
      return -1;  
26  }  
  
28  
  
30  for (i=0; i<n; i++)  
  {  
32      printf("a[%d]=",i);  
      scanf("%d", &a[i]); /* učitavamo jedan po jedan element niza */  
34  }  
  
36  printf("Elementi sa parnim indeksima:\n");  
  for (i=0; i<n; i+=2)  
38      printf("a[%d]=%d\n",i,a[i]);  
  
40  printf("Parni elementi:\n");  
  for (i=0; i<n; i++)  
42      if (a[i]%2==0)  
          printf("a[%d]=%d\n",i,a[i]);  
44  
46  return 0;  
}
```

#### Rešenje 3.3

```
1  /*  
    Napisati program koji učitava jedan ceo broj a zatim ispisuje  
    koliko puta koja cifra ucestvuje  
3  u zapisu tog broja. Nije potrebno ispisivati da se neka cifra  
    pojavila 0 puta.  
  
5  Na primer, za uneti broj 4611, izlaz treba da bude:  
  
7  U zapisu broja 4611, cifra 1 se pojaviljuje 2 puta  
  U zapisu broja 4611, cifra 4 se pojaviljuje 1 puta  
9  U zapisu broja 4611, cifra 6 se pojaviljuje 1 puta  
  
11  A za uneti broj -252  
  
13  U zapisu broja -252, cifra 2 se pojaviljuje 2 puta
```

```
15      U zapisu broja -252, cifra 5 se pojaviljuje 1 puta
16
17  */
18
19  #include<stdio.h>
20  #include<stdlib.h>
21  #define MAX 100
22
23  int main()
24  {
25      int x;
26      int brojac[10];
27      char cifra;
28      int original;
29      int i;
30
31      printf("Unesi jedan ceo broj:");
32      scanf("%d",&x);
33
34      /*
35       svaki element niza brojac predstavlja
36       brojac za jednu cifru:
37       brojac[0] sadrzi broj nula
38       brojac[1] sadrzi broj jedinica
39       ...
40       brojac[9] sadrzi broj devetki
41
42       brojac se inicijalizuju na vrednost 0
43  */
44
45      for(i=0;i<10;i++)
46          brojac[i]=0;
47
48      /*
49       vrednost promenljive x ce biti unistena
50       u while petlji jer je u svakom koraku delimo
51       sa 10; njenu vrednost cuvamo u promenljivoj
52       original kako bismo mogli da je iskoristimo
53       na kraju prilikom ispisa
54  */
55      original = x;
56
57      /*
58       Uzimamo apsolutnu vrednost broja za slucaj
59       da je uneti broj negativan
60  */
61      x=abs(x);
62
63      /* Izdvajanje cifara broja */
64      do
65      {
66          cifra = x%10;
```

### 3 Predstavljanje podataka

```
        brojaci[cifra]++; /* Uvecavamo brojac odgovarajuće cifre */
67     x/=10;
    } while(x);

69     /* Ispis brojaca koji su razliciti od nule */
71     for(i=0;i<10;i++)
        if(brojaci[i])
73         printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n",
            original, i, brojaci[i]);

75     return 0;
77 }
```

#### Rešenje 3.4

```
1  /* Napisati program koji ucitava karakter po karakter do EOF i
   ispisuje koliko se puta
   u unetom tekstu pojavila svaka cifra, svako malo slovo i svako
   veliko slovo. Ispisati
3  broj pojavljivanja samo za ona mala slova, velika slova i cifre
   koji su se u unetom
   tekstu pojavili >0 puta.
5  */

7  #include <stdio.h>

9

11 int main()
12 {
13     /* Za svaku dekadnu cifru definisemo jedan brojac (tj. imamo niz
       od 10 brojaca): brojaci[0] broji koliko se puta pojavio karakter
       '0', brojaci[1] broji koliko se puta pojavio karakter '1' i tako
15     dalje. Svi brojac se inicijalizuju nulama.
       */
17     int cifre[10];
18     int mala[26];
19     int velika[26];

21     int c, i;

23     for(i=0;i<10;i++)
        cifre[i]=0;

25     for(i=0;i<26;i++)
26     {
27         mala[i]=0;
28         velika[i]=0;
29     }

31     while((c = getchar()) != EOF)
```

```

33 {
34
35     if (c>='A' && c<='Z')
36         velika[c-'A']++;
37     else if (c>='a' && c<='z')
38         mala[c-'a']++;
39     else if (c >='0' && c <= '9') /* Ako je karakter c dekadna cifra
... */
        cifre[c-'0']++;          /* Uvecavamo odgovarajuci brojca za
1 */
41
42     /*
43     Izraz c - '0' ce u slucaju da je c dekadna cifra imati
        upravo
        vrednost 0, 1, ..., 9 za karaktere '0', '1', ..., '9' respektivno,
45     a to su upravo indeksi u nizu brojac (jer niz ima 10 elemenata,
        pa su indeksi od 0 do 9). Time postizemo da brojac[0] broji
47     karaktere '0', itd. Isto vazi i za brojace za mala i velika slova.
        */
49 }
51
52 /* Prikazujemo elemente niza, tj. vrednosti brojaca: */
53 for(i = 0; i < 10; i++)
54     if (cifre[i]!=0)
55         printf("Karakter %c se pojavljuje %d puta\n", '0' + i,
            cifre[i]);
57
58 for(i = 0; i < 26; i++)
59     if (mala[i]!=0)
60         printf("Karakter %c se pojavljuje %d puta\n", 'a' + i,
            mala[i]);
61
62 for(i = 0; i < 26; i++)
63     if (velika[i]!=0)
64         printf("Karakter %c se pojavljuje %d puta\n", 'A' + i,
            velika[i]);
65
66 return 0;
67 }

```

### Rešenje 3.5

```

/*
2  Napisati program koji ucitava dimenziju n dva celobrojna niza a i b
    (oba niza su iste dimenzije),
    zatim ucitava elemente oba niza i formira treci niz c tako sto
    naizmenicno rasporedjuje
4  elemente nizova a i b unutar njega: a_0,b_0,a_1,b_1,...,a_(n-1),b_(
    n-1). Program treba
    da ispise elemente novog niza c na standardni izlaz. Mozemo
    pretpostaviti da je maksimalni

```

### 3 Predstavljanje podataka

---

```
6   broj elemenata u nizovima a i b 100.
   */
8
   #include <stdio.h>
10  #define MAX 100
12  int main()
   {
14     int a[MAX];
       int b[MAX];
16     int c[2*MAX];

18     int n;
       int i,j;
20
       printf("Unesi dimenziju niza:");
22     scanf("%d", &n);

24     if (n<1 || n>MAX)
       {
26         printf("Neispravan unos\n");
           return -1;
28     }

30
       printf("\nUnesi elemente niza a:\n");
32     for(i=0;i<n;i++)
       {
34         printf("a[%d]=",i);
           scanf("%d", &a[i]);
36     }

38     printf("\nUnesi elemente niza b:\n");
39     for(i=0;i<n;i++)
40     {
41         printf("b[%d]=",i);
42         scanf("%d", &b[i]);
43     }
44
45     /*
46      Koristimo dva indeksa:
47      1. i, sa kojim pristupamo
48         elementima niza a i b, i koji uvecavamo za 1
         nakon svake iteracije,
49      2. j, sa kojim pristupamo
50         elementima niza c; s obzirom da u svakoj
51         iteraciji dodeljujemo vrednost za dva
52         elementa niza c (c[j] i c[j+1]), indeks
53         j uvecavamo za 2 nakon svake iteracije
54     */
55     for(i=0,j=0;i<n;i++,j+=2)
56     {
```



```

58     c[j]=a[i];
    c[j+1]=b[i];
60 }

62 printf("\nNiz c:\n");
    for(i=0;i<2*n;i++)
64     printf("c[%d]=%d\n",i,c[i]);

66     return 0;
}

```

### Rešenje 3.6

```

1  /*
    Napisati program koji ucitava dimenziju n celobrojnog niza a i
3  njegove elemente, a zatim iz niza a izbacuje sve elemente
    koji nisu deljivi svojom poslednjom cifrom, izuzev elemenata
5  cija je poslednja cifra 0 koji treba zadržati. Program treba da
    ispise
    izmenjeni niz na standardni izlaz. Mozemo pretpostaviti da niz a
7  sadrzi najvise 100 elemenata.
    */
9
11 #include <stdio.h>
#define MAX 100
13
15 int main()
17 {
    int a[MAX];
19
    int n;
    int i,j;
    char poslednja_cifra;
21     int novo_n;

23     printf("Unesi dimenziju niza:");
    scanf("%d", &n);
25

    if (n<1 || n>MAX)
27     {
        printf("Neispravan unos\n");
29         return -1;
    }
31

33     printf("\nUnesi elemente niza a:\n");
    for(i=0;i<n;i++)
35     {
        printf("a[%d]=",i);
37         scanf("%d", &a[i]);
    }
}

```

```
39     }
41     /*
43     Dodatni indeks j se uvecava u slucaju da element na indeksu
45     i treba da ostane u nizu, tj da je deljiv svojim
47     indeksom i; u suprotnom, j se nece uvecati i
49     element i ce u narednoj iteraciji biti zamenjen elementom koji
51     jeste deljiv svojim indeksom
53     */
55     for(i=0,j=0;i<n;i++)
57     {
59         poslednja_cifra = a[i]%10;
61
63         /*
65         zbog lenjog izracunavanja, ako je prvi uslov
67         u disjunkciji tacan, drugi se nece ispitivati
69         (jer ce tada disjunkcija biti tacna bez obzira
71         da li je drugi uslov tacan ili ne)
73         */
75         if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
77         {
79             a[j]=a[i];
81             j++;
83         }
85     }
87     /*
89     Izbacivanjem elemenata dimenzija niza se menja, odnosno
91     smanjuje se za broj izbacenih elemenata
93     */
95     novo_n=j;
97
99     printf("Nakon izmena:\n");
101     for(i=0;i<novo_n;i++)
103         printf("a [%d]=%d\n",i,a[i]);
105
107     return 0;
109 }
```

#### Rešenje 3.7

```
2     /*
3     a) Napisati funkciju koja ucitava sadrzaj niza.
4     b) Napisati funkciju koja stampa sadrzaj niza.
5     c) Napisati funkciju koja racuna sumu elemenata niza.
6     d) Napisati funkciju koja racuna prosechnu vrednost elemenata niza.
7     e) Napisati funkciju koja izracunava minimum elemenata niza.
8     f) Napisati funkciju koja izracunava poziciju maksimalnog elementa
9     u nizu.
10    g) Napisati program koji testira prethodne funkcije.
```

```
10 */
12 #include <stdio.h>
13 #define MAX 100
14
15 /* a) */
16 void ucitaj(int a[], int n)
17 {
18     int i;
19     for(i=0;i<n;i++)
20     {
21         printf("Unesi element na poziciji %d:",i);
22         scanf("%d",&a[i]);
23     }
24 }
25
26 /* b) */
27 void stampaj(int a[], int n)
28 {
29     int i;
30     for(i=0;i<n;i++)
31         printf("%d\t",a[i]);
32     printf("\n");
33 }
34
35 /* c) */
36 int suma(int a[], int n)
37 {
38     int i;
39     int s=0;
40     for(i=0;i<n;i++)
41         s+=a[i];
42     return s;
43 }
44
45 /* d) */
46 float prosek(int a[], int n)
47 {
48     int i;
49     int s = suma(a,n);
50     return (float) s/n;
51 }
52
53
54 /* e) */
55 int minimum (int a[],int n)
56 {
57     int m;
58     int i;
59     m = a[0];
60 }
```

```
62  /*
63     minimum inicijalizujemo na prvi element niza (a[0])
64     u svakom koraku poredimo vrednost minimuma
65     sa jednim elementom niza, iduci redom; s obzirom
66     da je minimum inicijalizovan na a[0], nema potrebe
67     porediti a[0] sa a[0] i zbog toga indeksiranje kreće
68     od 1
69  */
70
71  for(i=1;i<n;i++)
72      if (m>a[i])
73          m = a[i];
74
75  return m;
76 }
77
78 /* f) */
79 int max_pozicija (int a[],int n)
80 {
81     int m;
82     int m_poz;
83     int i;
84     m = a[0];
85     m_poz=0;
86
87     for(i=1;i<n;i++)
88         if (m<a[i])
89         {
90             m = a[i];
91             m_poz=i;
92         }
93
94     return m_poz;
95 }
96
97
98
99
100 int main()
101 {
102     int a[MAX];
103     int n;
104     printf("Unesi dimenziju niza:");
105     scanf("%d",&n);
106
107     if (n<1 || n>MAX)
108     {
109         printf("Nekorektan unos\n");
110         return -1;
111     }
112 }
```

```

114     ucitaj(a,n);
115     printf("Ucitani niz:");
116     stampaj(a,n);

118     printf("Suma elemenata niza: %d\n", suma(a,n));
119     printf("Prosečna vrednost elemenata niza: %.2f\n", prosek(a,n));
120     printf("Minimumalni element niza: %d\n", minimum(a,n));
121     printf("Indeks maksimalnog elementa niza: %d\n", max_pozicija(a,n)
122           );
123     return 0;
124 }

```

### Rešenje 3.8

```

/*
2   a) Napisati funkciju koja ucitava sadrzaj niza.
3   b) Napisati funkciju koja stampa sadrzaj niza.
4   c) Napisati funkciju koja proverava da li niz sadrzi neku vrednost
5       m.
6   d) Napisati funkciju koja vraca vrednost prve pozicije na kojoj se
7       nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
8       nizu.
9   e) Napisati funkciju koja vraca vrednost poslednje pozicije na
10      kojoj se
11      nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
12      nizu.
13  f) Napisati funkciju koja proverava da li elementi niza cine
14      palindrom.
15  g) Napisati funkciju koja proverava da li su elementi niza
16      uredjeni
17      neopadajuće.
18  h) Napisati funkciju koja izracunava najduzu uzastopnu seriju
19      jednakih
20      elemenata u nizu. Na primer, za uneti niz 1 2 3 4 4 4 5 6 7 8 9 9
21      funkcija
22      treba da vrati 3.
23  i) Napisati program koji testira prethodne funkcije.
24 */
25 #include <stdio.h>
26 #define MAX 100

/* a) */
void ucitaj(int a[], int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        printf("Unesi element na poziciji %d:",i);
        scanf("%d",&a[i]);
    }
}

```

```
28     }
29 }
30
31 /* b) */
32 void stampaj(int a[], int n)
33 {
34     int i;
35     for(i=0;i<n;i++)
36         printf("%d\t",a[i]);
37     printf("\n");
38 }
39
40
41 /* c) */
42 int sadrzi(int a[], int n, int m)
43 {
44     int i;
45     /*
46      poredimo jedan po jedan element niza a sa datim m; ukoliko
47      ustanovimo jednakost, to znaci da niz sadrzi element jednak
48      m i vracamo 1
49     */
50     for(i=0;i<n;i++)
51         if (a[i]==m)
52             return 1;
53
54     /*
55      ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
56      ispunjen,
57      to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
58      vrati 0
59     */
60     return 0;
61 }
62
63 /* d) */
64 int prvo_pojavljivanje(int a[], int n, int m)
65 {
66     int i;
67     /*
68      poredimo jedan po jedan element niza a sa datim m; ukoliko
69      ustanovimo jednakost, vracamo indeks elementa niza a koji
70      je jednak sa m
71     */
72     for(i=0;i<n;i++)
73         if (a[i]==m)
74             return i;
75
76     /*
77      ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
78      ispunjen,
79      to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
```

```
        vrati -1
    */
78     return -1;
}

80
/* e) */
82     int poslednje_pojavljivanje(int a[], int n, int m)
    {
84         int i;
        /*
86         krecemo od indeksa poslednjeg elementa, n-1
        */
88         for(i=n-1; i>=0; i--)
            if (a[i]==m)
90                 return i;

92         return -1;
    }

94
/* f) */
96     int palindrom(int a[], int n)
    {
98
100         int i,j;

        /*
102         uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
104         uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
106         .
108         i tako redom dok je prva pozicija manja od druge
        */
110         for(i=0, j=n-1; i<j; i++, j--)
            if(a[i]!=a[j])
                return 0;

112         return 1;
    }

114
/* g) */
116     int neopadajuci(int a[], int n)
    {
118         int i;

120         /*
        Funkcija neopadajuci proverava da li je dati niz sortiran
        neopadajuće i vraća
122         1 ako jeste, a 0 u suprotnom

124         Sortiranost proveravamo na sledeći način: za svaki par susednih
        elemenata
        a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
```

### 3 Predstavljanje podataka

---

```
126     proveravamo
    da li vazí da je drugi član para manji od prvog. Ako naídjemo na
    par za koji
    to ne vazí, niz nije sortiran i funkcija vraća 0. Ukoliko se
    petlja završi
128     a da pritom uslov  $a[i] < a[i-1]$  nije nijednom bio ispunjen, to znači
    da je
    niz sortiran i funkcija vraća 1
130
131     */
132     for(i=1; i<n; i++)
133         if (a[i]<a[i-1])
134             return 0;
135
136     return 1;
137 }
138
139 /* h) */
140 int najduza_konstanta(int a[], int n)
141 {
142     int i; /* indeks niza */
143     int j; /* dužina intervala */
144     int dužina;
145     int max_dužina=0;
146
147
148     for(i=0, j=0; i<n-1; i++)
149     {
150         if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
151         {
152             j++;          /* uvećavamo dužinu konstantnog intervala */
153
154             /*
155              ako se niz završava konstantnim intervalom (nalazimo se u
156              poslednjoj
157              iteraciji petlje i tada je  $i==n-2$ ), ispitujemo da li je
158              taj konstantni
159              interval maksimalne dužine
160              */
161             if(i==n-2)
162             {
163                 j++;
164                 if(j>max_dužina)
165                     max_dužina=j;
166             }
167         }
168         else
169         {
170             /*
171              izašli smo iz konstantnog intervala
172
173              ukoliko smo imali bar dva elementa u konstantnom
```



```
intervalu,
172     vrednost promenljive j ce biti 1, a duzina tog intervala
    je 2;
    zbog toga je neophodno takve (pozitivne) j uvecati za 1;
174
    sa druge strane, ako su a[i] i a[i+1] razliciti,
176     duzina tog intervala je 0
    */
178
    if (j>0)
180         j++;

    /* azuriramo maksimalnu duzinu uspona */
    if(j>max_duzina)
184         max_duzina=j;
    /*
186     duzina uspona se postavlja na nulu
    kako bi mogli da je iskoristimo
188     za naredni uspon
    */
190     j=0;

192 }

194 }

196     return max_duzina;
198 }

200 int main()
202 {
    int a[MAX];
204     int n;
    int m;
206     int i;

    printf("Unesi dimenziju niza:");
    scanf("%d",&n);

210     if (n<1 || n>MAX)
    {
212         printf("Nekorektan unos\n");
214         return -1;
    }

216     ucitaj(a,n);
218     printf("Ucitani niz:");
    stampaj(a,n);

220     printf("Unesi jedan ceo broj:");
```

### 3 Predstavljanje podataka

```
222     scanf("%d",&m);

224

226     if(sadrzi(a,n,m))
227         printf("Niz sadrzi element cija je vrednost %d\n", m);
228     else
229         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

230     i = prvo_pojavljivanje(a,n,m);
231     if(i!=-1)
232         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
233         prvog pojavljivanja u nizu je %d\n", m,i);
234     else
235         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

236

237     i = poslednje_pojavljivanje(a,n,m);
238     if(i!=-1)
239         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
240         poslednjeg pojavljivanja u nizu je %d\n", m,i);
241     else
242         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

243

244     if(palindrom(a,n))
245         printf("Elementi niza cine palindrom\n");
246     else
247         printf("Elementi niza ne cine palindrom\n");

248     if(neopadajuci(a,n))
249         printf("Niz je sortiran neopadajuće\n");
250     else
251         printf("Niz nije sortiran neopadajuće\n");

252

253     printf("Duzina najduzeg konstantnog intervala: %d\n",
254           najduza_konstanta(a,n));

255     return 0;
256 }
```

#### Rešenje 3.9

```
1  /*
2
3  a) Napisati funkciju koja učitava sadržaj niza.
4  b) Napisati funkciju koja stampa sadržaj niza.
5  c) Napisati funkciju koja proverava da li niz sadrzi neku vrednost
   m.
6  d) Napisati funkciju koja vraća vrednost prve pozicije na kojoj se
   nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
   nizu.
```

```

7      e) Napisati funkciju koja vraća vrednost poslednje pozicije na
      kojoj se
      nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
      nizu.
9      f) Napisati funkciju koja proverava da li elementi niza cine
      palindrom.
      g) Napisati funkciju koja proverava da li su elementi niza
      uredjeni
11     neopadajuće.
      h) Napisati funkciju koja izracunava najduzu uzastopnu seriju
      jednakih
13     elemenata u nizu. Na primer, za uneti niz 1 2 3 4 4 4 5 6 7 8 9 9
      funkcija
      treba da vrati 3.
15     i) Napisati program koji testira prethodne funkcije.
      */
17     #include <stdio.h>
      #define MAX 100
19
      /* a) */
21     void ucitaj(int a[], int n)
      {
23         int i;
         for(i=0;i<n;i++)
25         {
             printf("Unesi element na poziciji %d:",i);
27             scanf("%d",&a[i]);
         }
29     }

31     /* b) */
      void stampaj(int a[], int n)
33     {
         int i;
35         for(i=0;i<n;i++)
             printf("%d\t",a[i]);
37         printf("\n");
     }
39

41     /* c) */
      int sadrzi(int a[], int n, int m)
43     {
         int i;
45         /*
            poredimo jedan po jedan element niza a sa datim m; ukoliko
            ustanovimo jednakost, to znaci da niz sadrzi element jednak
            m i vracamo 1
47         */
         for(i=0;i<n;i++)
49             if (a[i]==m)
51                 return 1;
     }

```

### 3 Predstavljanje podataka

---

```
53
54     /*
55         ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
56         ispunjen,
57         to znači da se broj m ne nalazi u nizu a i da funkcija treba da
58         vrati 0
59     */
60     return 0;
61 }
62
63 /* d) */
64 int prvo_pojavljivanje(int a[], int n, int m)
65 {
66     int i;
67     /*
68         poredimo jedan po jedan element niza a sa datim m; ukoliko
69         ustanovimo jednakost, vraćamo indeks elementa niza a koji
70         je jednak sa m
71     */
72     for(i=0; i<n; i++)
73         if (a[i]==m)
74             return i;
75
76     /*
77         ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
78         ispunjen,
79         to znači da se broj m ne nalazi u nizu a i da funkcija treba da
80         vrati -1
81     */
82     return -1;
83 }
84
85 /* e) */
86 int poslednje_pojavljivanje(int a[], int n, int m)
87 {
88     int i;
89     /*
90         krećemo od indeksa poslednjeg elementa, n-1
91     */
92     for(i=n-1; i>=0; i--)
93         if (a[i]==m)
94             return i;
95
96     return -1;
97 }
98
99 /* f) */
100 int palindrom(int a[], int n)
101 {
102     int i, j;
```

```
101  /*
102     uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
103     uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
104     .
105     .
106     i tako redom dok je prva pozicija manja od druge
107  */
108  for(i=0,j=n-1;i<j;i++,j--)
109      if(a[i]!=a[j])
110          return 0;
111
112  return 1;
113 }
114
115 /* g) */
116 int neopadajuci(int a[], int n)
117 {
118     int i;
119
120     /*
121     Funkcija neopadajuci proverava da li je dati niz sortiran
122     neopadajuće i vraća
123     1 ako jeste, a 0 u suprotnom
124
125     Sortiranost proveravamo na sledeci nacin: za svaki par susednih
126     elemenata
127     a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
128     proveravamo
129     da li vazi da je drugi clan para manji od prvog. Ako naidjemo na
130     par za koji
131     to ne vazi, niz nije sortiran i funkcija vraća 0. Ukoliko se
132     petlja završi
133     a da pritom uslov a[i]<a[i-1] nije nijednom bio ispunjen, to znaci
134     da je
135     niz sortiran i funkcija vraća 1
136
137     */
138     for(i=1; i<n; i++)
139         if (a[i]<a[i-1])
140             return 0;
141
142     return 1;
143 }
144
145 /* h) */
146 int najduza_konstanta(int a[], int n)
147 {
148     int i; /* indeks niza */
149     int j; /* dužina intervala */
150     int dužina;
151     int max_dužina=0;
```

```
147 for(i=0,j=0;i<n-1;i++)
149 {
151     if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
153     {
155         j++;          /* uvecavamo duzinu konstantnog intervala */
157         /*
158         ako se niz zavrшава konstantnim intervalom (nalazimo se u
159         poslednjoj
160         iteraciji petlje i tada je i==n-2), ispitujemo da li je
161         taj konstantni
162         interval maksimalne duzine
163         */
164         if(i==n-2)
165         {
166             j++;
167             if(j>max_duzina)
168                 max_duzina=j;
169         }
170     }
171     else
172     {
173         /*
174         izašli smo iz konstantnog intervala
175
176         ukoliko smo imali bar dva elementa u konstantnom
177         intervalu,
178         vrednost promenljive j ce biti 1, a duzina tog intervala
179         je 2;
180         zbog toga je neophodno takve (pozitivne) j uvecati za 1;
181
182         sa druge strane, ako su a[i] i a[i+1] razliciti,
183         duzina tog intervala je 0
184         */
185         if (j>0)
186             j++;
187
188         /* azuriramo maksimalnu duzinu uspona */
189         if(j>max_duzina)
190             max_duzina=j;
191         /*
192         duzina uspona se postavlja na nulu
193         kako bi mogli da je iskoristimo
194         za naredni uspon
195         */
196         j=0;
197     }
198 }
```

```
195     }
197     return max_duzina;
199 }
201 int main()
202 {
203     int a[MAX];
204     int n;
205     int m;
206     int i;
207
208     printf("Unesi dimenziju niza:");
209     scanf("%d",&n);
210
211     if (n<1 || n>MAX)
212     {
213         printf("Nekorektan unos\n");
214         return -1;
215     }
216
217     učitaj(a,n);
218     printf("Učitani niz:");
219     stampaj(a,n);
220
221     printf("Unesi jedan ceo broj:");
222     scanf("%d",&m);
223
224
225     if(sadrzi(a,n,m))
226         printf("Niz sadrzi element cija je vrednost %d\n", m);
227     else
228         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
229
230     i = prvo_pojavljivanje(a,n,m);
231     if(i!=-1)
232         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog prvog pojavljivanja u nizu je %d\n", m,i);
233     else
234         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
235
236
237     i = poslednje_pojavljivanje(a,n,m);
238     if(i!=-1)
239         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog poslednjeg pojavljivanja u nizu je %d\n", m,i);
240     else
241         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
242
243     if(palindrom(a,n))
244         printf("Elementi niza cine palindrom\n");
```

### 3 Predstavljanje podataka

```
245     else
246         printf("Elementi niza ne cine palindrom\n");
247
248     if(neopadajuci(a,n))
249         printf("Niz je sortiran neopadajuce\n");
250     else
251         printf("Niz nije sortiran neopadajuce\n");
252
253     printf("Duzina najduzeg konstantnog intervala: %d\n",
254           najduza_konstanta(a,n));
255
256     return 0;
257 }
```

#### Rešenje 3.10

```
1  /*
2   a) Napisati funkciju koja sve vrednosti niza uvecava za vrednost m.
3   b) Napisati funkciju koja obrce vrednosti elementima niza.
4   c) Napisati funkciju koja rotira niz ciklicno za jedno mesto u levo
5   d) Napisati funkciju koja rotira niz ciklicno za k mesta u levo.
6   e) Napisati program koji testira prethodne funkcije.
7
8   Napisati potom glavni program koji testira ovu funkciju.
9  */
10
11 #include<stdio.h>
12 #define MAX 100
13
14 void ucitaj(int a[], int n)
15 {
16     int i;
17     for(i=0;i<n;i++)
18     {
19         printf("Unesi element na poziciji %d:",i);
20         scanf("%d",&a[i]);
21     }
22 }
23
24 void stampaj(int a[], int n)
25 {
26     int i;
27     for(i=0;i<n;i++)
28         printf("%d\t",a[i]);
29     printf("\n");
30 }
31
32 void uvecaj(int a[], int n, int m)
```



```
{
35     int i;
36     for(i=0;i<n;i++)
37         a[i]+=m;
38 }
39
40
41 void obrni(int a[], int n)
42 {
43     int t;
44     int i,j;
45     /*
46      Niz obrnemo tako sto razmenimo vrednosti elemenata na pozicijama
47      0 i n-1,
48      zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
49      druge
50     */
51     for(i=0,j=n-1;i<j;i++, j--)
52     {
53         t = a[i];
54         a[i] = a[j];
55         a[j] = t;
56     }
57 }
58
59 void rotiraj1(int a[], int n)
60 {
61     int i;
62     int tmp;
63     tmp=a[0]; /* izdvajamo prvi element */
64     for(i=0;i<n-1;i++)
65         a[i]=a[i+1]; /* pomeramo preostale elemente */
66     a[n-1] = tmp; /* poslednjem elementu dodeljujemo
67                  sacuvanu vrednost prvog elementa */
68 }
69
70
71 void rotirajk(int a[], int n, int k)
72 {
73     int i;
74     /*
75      k puta rotiramo niz za jednu poziciju
76      ulevo
77     */
78     for(i=0;i<k;i++)
79         rotiraj1(a,n);
80 }
81
82 int main()
83 {
```

```
    int a[MAX];
85    int n;
    int i;
87    int k;
    int m;

89    printf("Unesi dimenziju niza:");
91    scanf("%d",&n);

93    if (n<1 || n>MAX)
    {
95        printf("Nekorektan unos\n");
        return -1;
97    }

99    učitaj(a,n);

101    printf("Unesi jedan ceo broj:");
    scanf("%d", &m);
103
    uvecaj(a,n,m);
105    printf("Elementi niza nakon uvecanja za %d:\n",m);
    stampaj(a,n);
107
    obrni(a,n);
109    printf("Elementi niza nakon obrtanja:\n");
    stampaj(a,n);
111
    printf("Unesi jedan pozitivan ceo broj:");
113    scanf("%d",&k);

115    if (k<=0)
    {
117        printf("Nekorektan unos\n");
        return -1;
119    }

121    rotiraj1(a,n);
    printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
123    stampaj(a,n);

125    rotirajk(a,n,k);
    printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n",k);
127    stampaj(a,n);

129    return 0;
}
```

#### Rešenje 3.114

---

```

1 #include <stdio.h>
3 #define MAX 100
5 int main()
6 {
7     float brojevi[MAX];
8     int n, i;
9
10    printf("Unesite broj elemenata niza: ");
11    scanf("%d", &n);
12
13    if(n<1 || n>100)
14    {
15        printf("Greska: pogresan unos!\n");
16    }else{
17
18        printf("Unesite elemente niza:\n");
19        for(i=0;i<n;i++)
20            scanf("%f", &brojevi[i]);
21
22        /*
23         Ukoliko je i element niza brojevi[i] negativan broj,
24         kvadriramo ga tako sto ga pomnozimo sa samim sobom.
25        */
26
27        for(i=0;i<n;i++)
28            if(brojevi[i]<0)
29                brojevi[i] *= brojevi[i];
30
31        /* Ispisujemo sve elemente niza. */
32
33        for(i=0;i<n;i++)
34            printf("%g ", brojevi[i]);
35    }
36    return 0;
37 }

```

### Rešenje 3.115

```

1 #include <stdio.h>
3 #define MAX 100
5 int main()
6 {
7     int brojevi[MAX];
8     int n, i, k, indikator;
9
10    printf("Unesite dimenziju niza: ");
11    scanf("%d", &n);

```

### 3 Predstavljanje podataka

---

```
13  if(n<1 || n>100)
14  {
15      printf("Greska: pogresan unos!\n");
16  }
17  else{
18      printf("Unesite elemente niza: ");
19
20      for(i=0;i<n;i++)
21          scanf("%d", &brojevi[i]);
22
23      printf("Unesite broj k: ");
24      scanf("%d", &k);
25
26      if(k == 0)
27      {
28          printf("Greska: pogresan unos!\n");
29      }
30      else{
31          /*
32           Promenljiva koja nam cuva informaciju o tome
33           da li je u nizu postojao element koji je deljiv brojem k.
34           Inicijalno je postavimo na nulu.
35          */
36
37          indikator = 0;
38
39          /*
40           Ukoliko je element niza deljiv brojem k, postavljamo indikator na
41           1 i ispisujemo indeks tog elementa.
42          */
43
44          for(i=0;i<n;i++)
45              if(brojevi[i]%k == 0)
46              {
47                  indikator = 1;
48                  printf("%d ",i);
49              }
50
51          /*
52           Ukoliko je indikator jednak nuli to znaci da ne postoji element u
53           nizu koji je deljiv brojem k.
54          */
55
56          if(indikator == 0)
57              printf("U nizu nema elemenata koji su deljivi brojem %d!\n",k);
58          }
59      return 0;
60  }
```

## Rešenje 3.116

```
1  #include <stdio.h>
3  #define MAX 100
5  int main()
6  {
7      int brojevi[MAX];
8      int n, i, poz_max, poz_min, max, min, tmp;
9
10     printf("Unesite dimenziju niza: ");
11     scanf("%d", &n);
12
13     if(n<1 || n>100)
14     {
15         printf("Greska: pogresan unos!\n");
16         return 0;
17     }
18
19     printf("Unesite elemente niza:\n");
20     for(i=0; i<n; i++)
21         scanf("%d", &brojevi[i]);
22
23     /*
24      * Maksimum tj. minimum pre ulaska u petlju postavimo da budu prvi
25      * element niza.
26      * Pozicije maksimuma tj. minimuma postavimo na 0.
27      */
28     max = brojevi[0];
29     min = brojevi[0];
30     poz_max = 0;
31     poz_min = 0;
32
33     /*
34      * Pronadjemo maksimalni tj. minimalni element tako sto u petlji
35      * prodjemo kroz sve elemente i ukoliko naletimo na element veci od
36      * maksimuma
37      * tj. manji od minimuma, promenimo tako da sada maksimum tj.
38      * minimum budu taj element
39      * i promenimo njihove pozicije.
40      */
41     for(i=1; i<n; i++)
42     {
43         if(brojevi[i] > max)
44         {
45             max = brojevi[i];
46             poz_max = i;
47         }
48
49         if(brojevi[i] < min)
50         {
51             min = brojevi[i];
52             poz_min = i;
53         }
54     }
55 }
```

### 3 Predstavljanje podataka

---

```

    min = brojevi[i];
49     poz_min = i;
    }
51 }

53 /*
    Zamenimo minimalni i maksimalni element na pozicijama poz_min i
    poz_max.
55     Koristimo pomocnu promenljivu tmp kako bismo sacuvali vrednost
    maksimalnog elementa.
    */
57 tmp = max;
    brojevi[poz_max] = min;
59     brojevi[poz_min] = tmp;

61     for(i=0;i<n;i++)
        printf("%d ", brojevi[i]);
63
    return 0;
65 }
```

#### Rešenje 3.117

```

1  #include <stdio.h>

3  #define MAX 100

5  int main()
{
7     char karakteri[MAX];
    char c;
9     int i, n;

11
    for(i=0;i<MAX;i++)
13     {
        /*
15         Ucitavamo karakter po karakter dok ne unesemo * ili ne
        prekoracimo 100 karaktera
        i upisujemo ih u niz.
        */
17         printf("Unesite karakter: ");
19         scanf("%c", &c);

21         /*
            Citamo belinu nakon unesenog karaktera.
            */
23         getchar();

25         /*
27         Ukoliko smo uneli * izlazimo iz petlje
        */
    }
}
```

```
29     */
    if(c == '*')
        break;
31
    /*
33     Stavljam karakter u niz.
    */
35     karakteri[i] = c;
    }
37
    /*
39     Broj unetih karaktera je nakon prolaska kroz petlju i-1.
    */
41     n = i-1;
43
    /*
45     Ispisujemo karaktere u obrnutom redosledu.
    */
    for(i=n;i>=0;i--)
47     {
        printf("%c ", karakteri[i]);
49     }
51     return 0;
    }
```

### Rešenje 3.118

```
1  #include <stdio.h>
3  int main()
    {
5      char c;
        int cifrex[10], cifrey[10];
7      int x, y, i, indikator;
9
        printf("Unesite dva broja: ");
        scanf("%d%d", &x, &y);
11
        /*
13         Uzmemo apsolutnu vrednost brojeva za slucaj da su negativni.
        */
15         x=abs(x);
            y=abs(y);
17
            /*
19             Niz cifrex nam predstavlja brojeve za cifre broja x, na pocetku
                ga inicijalizujemo na 0.
                Analogno za cifrey.
            */
21         for(i=0;i<10;i++)
```

### 3 Predstavljanje podataka

---

```
23 {
24     cifrex[i] = 0;
25     cifrey[i] = 0;
26 }
27
28 /*
29  Skidamo jednu po jednu cifru broja x i povecavamo njen brojac u
    nizu cifrex.
30 */
31 while(x)
32 {
33     c = x%10;
34     cifrex[c]++;
35     x /= 10;
36 }
37
38 /*
39  Isto radimo i za broj y.
40 */
41 while(y)
42 {
43     c = y%10;
44     cifrey[c]++;
45     y /= 10;
46 }
47
48 /*
49  Promenljiva koja nam služi za proveru da li su oba broja
    sastavljena od istih cifara.
    Pretpostavicemo da jesu i postaviti indikator na 1.
51  Nakon toga u petlji prolazimo kroz nizove cifrex i cifrey u
    kojima se nalaze
    brojevi pojavljivanja svih cifri 0-9 u broju x i y, i prvi put
    kada naletimo na
53  neku cifru koja se ne pojavljuje isti broj puta u oba broja x i y
    ,
    postavljamo promenljivu indikator na 0 (brojevi x i y nisu
    zapisani sa istim ciframa)
55  i izlazimo iz petlje.
    */
57 indikator = 1;
58 for(i=0;i<10;i++)
59     if(cifrey[i] != cifrex[i])
60     {
61         indikator = 0;
62         break;
63     }
64
65 /*
66  Ako je promenljiva indikator ostala 1, to znaci da u petlji nismo
    pronasli cifru
67  koja se ne pojavljuje isti broj puta u brojevima x i y, sto znaci
```



```

        da se oni zapisuju istim ciframa.
    */
69     if(indikator)
        printf("Brojevi se zapisuju istim ciframa!\n");
71     else
        printf("Brojevi se ne zapisuju istim ciframa!\n");
73
    return 0;
75 }

```

### Rešenje 3.119

```

1  #include <stdio.h>

3  #define MAX 100

5  int main()
{
7      int a[MAX], b[MAX], c[2*MAX];
      int i, n;

9      printf("Unesite broj n: ");
11     scanf("%d", &n);

13     if(n<1 || n>100)
    {
15         printf("Greska: pogresan unos!\n");
        return 0;
17     }

19     printf("Unesite elemente niza a: ");
    for(i=0;i<n;i++)
21         scanf("%d", &a[i]);

23     printf("Unesite elemente niza b: ");
    for(i=0;i<n;i++)
25         scanf("%d", &b[i]);

27     /*
        Niz c ima 2*n elemenata. Prvih n elemenata niza b, i nakon toga n
        elemenata niza a.
29     Elementi iz niza a se nalaze na pozicijama 0,1,2,...n-1, a
        elementi niza b na pozicijama
        n,n+1,...2*n. Jednim prolaskom kroz petlju na poziciju i u nizu c
        stavljamo element niza b - b[i],
31     a na poziciju n+i element niza a - a[i].
    */
33     for(i=0;i<n;i++)
    {
35         c[i] = b[i];
        c[n+i] = a[i];

```

```
37 }  
39 for(i=0;i<2*n;i++)  
    printf("%d ", c[i]);  
41  
43 return 0;  
}
```

#### Rešenje 3.17

```
1  #include <stdio.h>  
3  #define MAX 100  
5  /*  
6   Funkcija koja vraca broj pojavljivanja broja x u nizu.  
7  */  
9  int broj_pojavljivanja(int niz[], int n, int x)  
10 {  
11     int i, rezultat = 0;  
13     /*  
14      Kada naidjemo na element niza koji je jednak broju x, povecamo  
15      brojjac rezultat.  
16     */  
17     for(i=0;i<n;i++)  
18         if(niz[i] == x)  
19             rezultat++;  
21     return rezultat;  
22 }  
23 int main()  
24 {  
25     int a[MAX], b[MAX];  
26     int i, j, n, n_b;  
28     printf("Unesite broj n: ");  
29     scanf("%d", &n);  
31     if(n<1 || n>100)  
32     {  
33         printf("Greska: pogresan unos!\n");  
34         return -1;  
35     }  
37     printf("Unesite elemente niza a: ");  
38     for(i=0;i<n;i++)  
39         scanf("%d", &a[i]);
```

```

41  /*
    Brojac elemenata rezultujucega niza b.
43  */
    j = 0;
45  for(i=0;i<n;i++)
    {
47      /*
        Ukoliko se element niza pojavljuje tacno tri puta i ne postoji
        u nizu b koji trenutno ima j elemenata
49      (nismo ga jos uvek dodali) dodajemo ga u niz b i povecavamo
        brojac j.
        */
51      if(broj_pojavljivanja(a, n, a[i])==3 && broj_pojavljivanja(b, j,
        a[i])==0)
        {
53          b[j] = a[i];
            j++;
55        }
    }
57  /*
    Broj elemenata u nizu b je j.
59  */
    n_b = j;
61
    for(i=0;i<n_b;i++)
63        printf("%d ", b[i]);
65
    return 0;
}

```

### Rešenje 3.18

```

#include <stdio.h>
2
#define MAX 100
4
/*
6  Funkcija koja vraca 1 ukoliko broj x postoji u nizu, 0 inace.
*/
8
int postoji(int niz[], int n, int x)
10 {
    int i;
12
    for(i=0;i<n;i++)
14        if(niz[i] == x)
            return 1;
16
    return 0;
18 }

```

### 3 Predstavljanje podataka

---

```
20 int main()
21 {
22     int a[MAX], b[MAX], unija[2*MAX], presek[MAX], razlika[MAX];
23     int i, j, n_a, n_b, n_u, n_p, n_r, indikator;
24
25     printf("Unesite broj elemenata niza a: ");
26     scanf("%d", &n_a);
27
28     if(n_a<1 || n_a>100)
29     {
30         printf("Greska: pogresan unos!\n");
31         return -1;
32     }
33
34     printf("Unesite elemente niza a: ");
35     for(i=0;i<n_a;i++)
36         scanf("%d", &a[i]);
37
38     printf("Unesite broj elemenata niza b: ");
39     scanf("%d", &n_b);
40
41     if(n_b<1 || n_b>100)
42     {
43         printf("Greska: pogresan unos!\n");
44         return -1;
45     }
46
47     printf("Unesite elemente niza b: ");
48     for(i=0;i<n_b;i++)
49         scanf("%d", &b[i]);
50
51     /*
52      Brojaci elemenata u nizovima unija, presek i razlika.
53     */
54     n_u = 0;
55     n_p = 0;
56     n_r = 0;
57
58     for(i=0;i<n_a;i++)
59     {
60         /*
61          Ukoliko se element a[i] ne nalazi u uniji, dodajemo ga u uniju
62          i povecamo brojac elemenata u nizu unija.
63         */
64         if(postoji(unija,n_u,a[i]) == 0)
65         {
66             unija[n_u] = a[i];
67             n_u++;
68         }
69
70         /*
71          Ukoliko se element a[i] postoji u nizu b i ne postoji u nizu
```

```

    presek, dodajemo ga u presek i povecavamo brojac elemenata u nizu
    presek.
    */
72  if(postoji(b, n_b, a[i])==1 && postoji(presek, n_p, a[i])==0)
    {
74      presek[n_p] = a[i];
        n_p++;
76  }

78  /*
    Ukoliko element a[i] ne postoji u nizu b i ne postoji u nizu
    razlika, dodajemo ga u razliku i povecavamo brojac elemenata u
    nizu razlika.
    */
80  if(postoji(b, n_b, a[i])==0 && postoji(razlika, n_r, a[i])==0)
82  {
        razlika[n_r] = a[i];
84      n_r++;
    }
86  }

88  /*
    Elemente niza b koji ne postoje u uniji dodajemo u uniju.
    */
90  for(i=0;i<n_b;i++)
92      if(postoji(unija, n_u, b[i]))
    {
94          unija[n_u] = b[i];
            n_u++;
96    }

98  printf("Unija: ");
    for(i=0;i<n_u;i++)
100      printf("%d ", unija[i]);

102  printf("\nPresek: ");
    for(i=0;i<n_p;i++)
104      printf("%d ", presek[i]);

106  printf("\nRazlika: ");
    for(i=0;i<n_r;i++)
108      printf("%d ", razlika[i]);

110  return 0;
}

```

### Rešenje 3.19

```

#include <stdio.h>
2
#define MAX 100

```

```
4  int main()
6  {
    int a[MAX], b[MAX];
    int i, j, n_a, n_b;

10  printf("Unesite broj elemenata niza: ");
    scanf("%d", &n_a);

12
14  if(n_a<1 || n_a>100)
    {
        printf("Greska: pogresan unos!\n");
        return -1;
    }

18  printf("Unesite elemente niza: ");
20  for(i=0; i<n_a; i++)
        scanf("%d", &a[i]);

22
24  /*
    1. nacin

26  J nam predstavlja brojac prve slobodne pozicije na koju mozemo
    upisati element niza koji treba da ostane u nizu.
    Kada naletimo na element koji je paran, kopiramo ga na mesto a[j]
    i povecamo brojac j.
28  Ukoliko naletimo na element koji je neparan, njega samo preskocimo
    */
30
    for(i=0, j=0; i<n_a; i++)
    {
        if(a[i]%2 == 0)
        {
34            a[j] = a[i];
36            j++;
        }
    }

38
40  /*
    Na pozicijama od 0...j-1 se sada nalaze elementi koji su parni,
    te je nova dimenzija niza sada j.
42  */
    n_a=j;

44  for(i=0; i<n_a; i++)
        printf("%d ", a[i]);

48  /*
    2. nacin

50  Kada naletimo na element niza koji je paran, kopiramo ga u niz b
```

```
        i povecamo j - brojac elemenata niza b.
52
    for(i=0, j=0; i<n_a; i++)
54        if(a[i]%2 == 0)
56        {
            b[j] = a[i];
            j++;
58        }

60    n_b = j;

62    for(i=0; i<n_b; i++)
        printf("%d ", b[i]);
64

    /*
66    return 0;
}
```

### Rešenje 3.20

```
#include <stdio.h>
2 #include <math.h>

4 #define MAX 100

6 /*
    Funkcija koja proverava da li je broj prost.
    Vraca 1 ukoliko broj jeste prost, inace 0.
    */
10 int prost(int x)
12 {
    int i;

14    if(x == 2 || x == 3)
        return 1;

16    if(x%2 == 0)
18        return 0;

20    for(i=3; i<=sqrt(x); i+=2)
        if(x%i == 0)
22        return 0;

24    return 1;
}

26
28 int main()
30 {
    int a[MAX], b[MAX];
    int i, j, n_a, n_b;
```

### 3 Predstavljanje podataka

---

```
32 printf("Unesite broj elemenata niza: ");
   scanf("%d", &n_a);
34
   if(n_a<1 || n_a>100)
36   {
       printf("Greska: pogresan unos!\n");
38       return -1;
   }
40
   printf("Unesite elemente niza: ");
42   for(i=0; i<n_a; i++)
       scanf("%d", &a[i]);
44
   /*
46   1. nacin

48   J nam predstavlja brojac prve slobodne pozicije na koju mozemo
   upisati element niza koji treba da ostane u nizu.
   Kada naletimo na element koji nije prost, kopiramo ga na mesto a[
   j] i povecamo brojac j.
50   Ukoliko naletimo na element koji je prost, njega samo preskocimo.

52   */

54   for(i=0, j=0; i<n_a; i++)
   {
56       if(prost(a[i]) == 0)
       {
58           a[j] = a[i];
           j++;
60       }
   }
62
   n_a=j;

64   for(i=0; i<n_a; i++)
       printf("%d ", a[i]);
66

68   /*
70   2. nacin

   Prolazimo kroz niz a i svaki broj koji nije prost kopiramo u niz
   b i povecamo j - brojac elemenata u nizu b.

72   for(i=0, j=0; i<n_a; i++)
       if(prost(a[i]) == 0)
74       {
           b[j] = a[i];
76           j++;
78       }

80   n_b = j;
```



```
82     for(i=0;i<n_b;i++)
83         printf("%d ", b[i]);
84     */
85     return 0;
86 }
```

### Rešenje 3.21

```
/*
2   Napisati funkciju int prebrojavanje(int a[], int n) koja izracunava
   broj elemenata niza celih brojeva a duzine n
   koji su manji od poslednjeg elementa niza. Napisati i program koji
   testira rad funkcije. Pretpostaviti da duzina
4   niza nece biti veca od 100.

6   */

8   #include <stdio.h>
   #define MAX 100

10  /*
12   * Funkcija prebrojavanje vraca broj elemenata niza koji su manji od
   poslednjeg
   * NAPOMENA: Poslednji element niza se nalazi na poziciji n-1
14   */
   int prebrojavanje(int a[], int n)
16   {
       int i;
18       /*Inicijalizujemo brojac na 0*/
       int br=0;

20       /*
22       * Petljom prolazimo kroz sve clanove niza,
       * poredimo ih sa poslednjim elementom i
24       * ukoliko su manji, uvecavamo brojac
       */
26       for(i=0;i<n-1;i++){
           if(a[i]<a[n-1]){
28               br++;
           }
30       }

32       /*Vracamo izracunatu vrednost*/
       return br;
34   }

36   int main()
   {
38       int a[MAX];
       int n;
```

### 3 Predstavljanje podataka

---

```
40  int i;

42  printf("Unesite broj elemenata niza:");
    scanf("%d", &n);

44

46  /*Provera korektnosti ulaznih podataka*/
    if(n<=0 || n>100)
    {
48      printf("Greska: pogresan unos!\n");
        return 0;
50  }

52  /*Ucitavanje niza*/
    printf("Unesite elemente niza:");
54  for(i=0;i<n;i++)
        scanf("%d",&a[i]);

56

58  /*Ispis rezultata*/
    printf("%d\n", prebrojavanje(a,n));

60  return 0;
}
```

#### Rešenje 3.22

```
/*
2  Napisati funkciju int prebrojavanje(int a[], int n) koja izracunava
    broj parnih elemenata niza celih brojeva a
    duzine n koji prethode maksimalnom elementu niza. Napisati i program
    koji testira rad funkcije. Pretpostaviti
4  da duzina niza nece biti veca od 100.
    */

6

8  #include <stdio.h>
    #define MAX 100

10 /*Funkcija koja vraca broj parnih elemenata niza koji se nalaze
    ispred najveceg elementa u nizu
    * Ideja je da prvo pronadjemo najveći element niza i njegovu
    poziciju, a zatim da jos jednom
12 * prodjemo kroz niz sa ciljem da nadjemo sve parne brojeve koje
    prethode maksimalnom.
    */

14 int prebrojavanje(int a[], int n)
    {
16     int i;
        int max;
18     int max_ind;
        int br = 0;

20
```

```

22  /*Na pocetku postavljamo da je maksimalni element a[0] i da je
    odgovarajuca pozicija 0*/
max = a[0];
max_ind=0;

24  /*Pronalazimo maksimum niza i pamtimo i vrednost i poziciju*/
26  for(i=1;i<n-1;i++)
    if(a[i]>max)
28      {
        max = a[i];
30        max_ind = i;
      }

32  /* Krecemo od pocetka niza i idemo do pozicije na kojoj se nalazi
    najveći element
    * i pronalazimo sve parne brojeve
    */
36  for(i=0;i<max_ind;i++)
    if(a[i]%2==0)
38        br++;

40  return br;
}

42  int main()
44  {
    int a[MAX];
46    int n;
    int i;

48    printf("Unesite broj elemenata niza:");
50    scanf("%d", &n);

52    /*Vrsimo proveru korektnosti ulaza*/
    if(n<=0 || n>100)
54    {
        printf("Greska: pogresan unos!\n");
56        return 0;
    }

58    /*Ucitavamo elemente niza*/
60    printf("Unesite elemente niza:");
    for(i=0;i<n;i++)
62        scanf("%d",&a[i]);

64    /*Ispisujemo rezultat*/
    printf("%d\n", prebrojavanje(a,n));
66    return 0;
}

```

## Rešenje 3.23

```
1  /*
   Napisati funkciju int prebrojavanje_cifre(char s[], int n) koja
   izracunava broj cifara u nizu karaktera a duzine n.
3  Napisati i program koji testira rad funkcije. Pretpostaviti da
   duzina niza nece biti veca od 100.
   */
5
6  #include <stdio.h>
7  #include <ctype.h>
8  #define MAX 100
9
10 /* Funkcija koja prebrojava koliko ima cifara u datom nizu karaktera
   */
11 int prebrojavanje(char a[], int n)
12 {
13     int i;
14     int br = 0;
15
16     /*Prolazimo kroz niz i proveravamo da li je trenutni karakter cifra
       i ukoliko jeste, uvecavamo brojac*/
17     /*Funkcija isdigit vraca 1 ukoliko je prosledjeni karakter cifra, a
       0 u suprotnom
       i nalazi se u zaglavlju ctype.h
       */
19     for(i=0;i<n;i++)
20         if(isdigit(a[i]))
21             br++;
22
23     return br;
24 }
25
26 int main()
27 {
28     char a[MAX];
29     int n;
30     int i;
31
32     printf("Unesite broj elemenata niza:");
33     scanf("%d", &n);
34
35     /*Vrsimo proveru korektnosti ulaza*/
36     if(n<=0 || n>100)
37     {
38         printf("Greska: pogresan unos!\n");
39         return 0;
40     }
41
42     /*Ucitavamo elemente niza*/
43     printf("Unesite elemente niza:");
44     for(i=0;i<n;i++)
45     {
```

```

47  /*Kako su elementi niza karakteri, neophodno je da u svakoj
    iteraciji preskocimo karakter koji oznacava belinu ili novi red*/
    getchar();
49  /*A da zatim učitamo sam karakter u niz*/
    scanf("%c",&a[i]);
51  }

53  /*Ispisujemo rezultat*/
    printf("Broj cifara je: %d\n", prebrojavanje(a,n));
55  return 0;
}

```

### Rešenje 3.24

```

1  /*
    Napisati funkciju int zbir(int a[], int n, int i, int j) koja racuna
    zbir elemenata niza celih brojeva a duzine n od pozicije i do
    pozicije j.
3  Napisati i program koji testira rad funkcije. Pretpostaviti da
    duzina niza nece biti veca od 100.

5  */

7  #include<stdio.h>
    #define MAX 100

9  /*Funkcija koja vraca zbir elemenata koji se nalaze izmedju pozicija
    i i j*/
11 int zbir(int a[], int n, int i, int j){
    /*Na pocetku inicijalizujemo sumu na 0*/
13     int k, s=0;

15     /*Krecemo od pozicije i i idemo do pozicije j i dodajemo na sumu
        tekuci element niza*/
        for(k=i; k<=j; k++)
17         s+=a[k];

19     /*Na kraju vracamo izracunatu sumu*/
        return s;
21 }

23 int main(){

25     int n, i, j;
        int a[MAX];

27     printf("Unesite broj elemenata niza: ");
29     scanf("%d", &n);

31     /*Proveravamo korektnost ulaza*/
        if(n <=0 || n>100)

```

### 3 Predstavljanje podataka

```
33 {
34     printf("Greska: pogresan unos!\n");
35     return 0;
36 }
37
38 /*Ucitavamo elemente niza*/
39 printf("Unesite elemente niza:");
40 for(i=0; i<n; i++)
41     scanf("%d", &a[i]);
42
43 /*Ucitavamo interval [i,j]*/
44 printf("Unesite vrednosti za i i j: ");
45 scanf("%d%d", &i, &j);
46
47 /*Proveravamo korektnost zadatog intervala */
48 if(i > n-1 || j > n-1 || i > j){
49     printf("Greska: nekorektne vrednosti granica!\n");
50     return 0;
51 }
52
53 /*Ispisujemo rezultat*/
54 printf("Zbir je: %d", zbir(a,n,i,j));
55
56     return 0;
57 }
```

#### Rešenje 3.25

```
/*
2  Napisati funkciju float zbir_pozitivnih(float a[], int n, int k)
   koja izracunava zbir prvih k pozitivnih elemenata realnog niza a
   duzine n.
   Napisati i program koji testira rad funkcije. Pretpostaviti da
   duzina niza nece biti veca od 100.
4  */
6  #include<stdio.h>
   #define MAX 100
8
   /*Funkcija racuna zbir prvih k pozitivnih clanova niza a*/
10 float zbir_pozitivnih(float a[], int n, int k){
   int i;
12
   /*Na pocetku inicijalizujemo sumu na 0*/
14 float s=0;
16
   /*Prolazimo kroz niz brojeva i zaustavljamo se ili ako smo dosli do
   kraja ili ukoliko smo sabrali k brojeva */
   for(i=0; i<n && k>0; i++){
18     if(a[i] >= 0){
```

```

    /*Kada naidjemo na pozitivan element, uvecavamo sumu i smanjujemo
    k*/
20    s+=a[i];
    k--;
22 }
    }

24 /*Na kraju vracamo izracunatu sumu */
26 return s;
    }

28 int main(){
30     int n, i, k;
    float a[MAX];

32     printf("Unesite broj elemenata niza: ");
34     scanf("%d", &n);

36     /*Proveravamo korektnost ulaza*/
    if(n<=0 || n> MAX){
38         printf("Greska: pogresan unos!\n");
        return 0;
40     }

42     /*Ucitavamo elemente niza*/
    printf("Unesite elemente niza: ");
44     for(i=0; i<n; i++)
        scanf("%f", &a[i]);

46     /*Ucitavamo k*/
    printf("Unesite vrednost za k: ");
48     scanf("%d", &k);

50     /*Proveravamo korektnost za k*/
    if(k<0){
52         printf("Greska: pogresan unos!");
        return 0;
54     }

56     /*Ispisujemo rezultat*/
58     printf("Zbir je: %.2f\n", zbir_pozitivnih(a,n,k));
    return 0;
60 }

```

### Rešenje 3.26

```

1  /*
    Napisati funkciju void kvadriranje(float a[], int n) koja kvadrira
    elemente realnog niza a duzine n koji se nalaze na parnim
    pozicijama.

```

### 3 Predstavljanje podataka

---

```
3   Napisati i program koji testira rad funkcije. Pretpostaviti da
    duzina niza nece biti veca od 100.

5   */

7   #include<stdio.h>
    #define MAX 100

9   /*Funkcija kvadriraj menja niz a tako sto kvadrira sve elemente na
    parnim pozicijama. */
11  void kvadriraj(float a[], int n){
    int i;

13     /*Petljom prolazimo kroz niz i ukoliko je pozicija parna, a[i]
    postaje a[i]*a[i]*/
15     for(i=0; i<n; i++){
        if(i%2 ==0)
17         a[i]*=a[i]; //skraceno od a[i] = a[i]*a[i]
    }

19 }

21 int main(){

23     int n, i, j;
    float a[MAX];

25     printf("Unesite broj elemenata niza: ");
    scanf("%d", &n);

27     /*Proveravamo korektnost ulaza*/
31     if(n <=0 || n>100)
    {
33         printf("Greska: pogresan unos!\n");
        return 0;
35     }

37     /*Ucitavamo elemente niza*/
    printf("Unesite elemente niza:");
39     for(i=0; i<n; i++)
        scanf("%f", &a[i]);

41     /*Pozivamo funkciju koja kvadrira odgovarajuce elemente*/
43     kvadriraj(a,n);

45     /*Stampamo rezultat
    NAPOMENA: Kada koristimo %g za stampanje realnih brojeva,
47     oni ce biti istampani na najoptimalniji nacin
    (imace onoliko decimalnih mesta koliko ima i sam broj)
49     */
    for(i=0; i<n; i++)
51     printf("%g ", a[i]);
```



```
53  return 0;  
    }
```

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

## 3.3 Pokazivači

Zadatak 3.38    Tekst

[Rešenje [3.56](#)]

Zadatak 3.39    Tekst

[Rešenje [3.39](#)]

Zadatak 3.40    Tekst

[Rešenje [3.40](#)]

### 3 Predstavljanje podataka

---

#### Zadatak 3.41 Tekst

[Rešenje 3.41]

#### Zadatak 3.42 Tekst

[Rešenje 3.42]

#### Zadatak 3.43 Tekst

[Rešenje 3.43]

**Zadatak 3.44** Napisati program koji ispisuje zbir numeričkih argumenata komandne linije. Napomena: može se koristiti funkcija *atoi*.

##### Primer 1

```
POKRETANJE: ./a.out 5 mkp 9 -2 11 a 4 2
INTERAKCIJA SA PROGRAMOM:
Zbir numerickih argumenata: 29
```

##### Primer 2

```
POKRETANJE: ./a.out ab u f hj
INTERAKCIJA SA PROGRAMOM:
Zbir numerickih argumenata: 0
```

##### Primer 3

```
POKRETANJE: ./a.out 33 1 p 44
INTERAKCIJA SA PROGRAMOM:
Zbir numerickih argumenata: 78
```

##### Primer 4

```
POKRETANJE: ./a.out
INTERAKCIJA SA PROGRAMOM:
Zbir numerickih argumenata: 0
```

[Rešenje 3.44]

**Zadatak 3.45** Napisati program koji ispisuje argumente komandne linije koji počinju slovom z.

##### Primer 1

```
POKRETANJE: ./a.out zima jabuka zvezda Zrak
INTERAKCIJA SA PROGRAMOM:
zima zvezda
```

##### Primer 2

```
POKRETANJE: ./a.out bundeva pomorandza
INTERAKCIJA SA PROGRAMOM:
```

##### Primer 3

```
POKRETANJE: ./a.out sanke zapad zujanje
INTERAKCIJA SA PROGRAMOM:
zapad zujanje
```

*Primer 4*

```

|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:

```

[Rešenje 3.45]

**Zadatak 3.46** Napisati program koji ispisuje broj argumenata komandne linije koji sadrže slovo z.

*Primer 1*

```

|| POKRETANJE: ./a.out zvezda grozd jesen kisa
|| INTERAKCIJA SA PROGRAMOM:
|| 2

```

*Primer 2*

```

|| POKRETANJE: ./a.out AZBUKA deda mraz
|| INTERAKCIJA SA PROGRAMOM:
|| 2

```

*Primer 3*

```

|| POKRETANJE: ./a.out japan caj
|| INTERAKCIJA SA PROGRAMOM:
|| 0

```

*Primer 4*

```

|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| 0

```

[Rešenje 3.46]

**Zadatak 3.47** Napisati program koji na osnovu broja  $n$  koji se zadaje kao argument komandne linije ispisuje cele brojeve iz intervala  $[-n, n]$ .

*Primer 1*

```

|| POKRETANJE: ./a.out 2
|| INTERAKCIJA SA PROGRAMOM:
|| -2 -1 0 1 2

```

*Primer 2*

```

|| POKRETANJE: ./a.out 4
|| INTERAKCIJA SA PROGRAMOM:
|| -4 -3 -2 -1 0 1 2 3 4

```

*Primer 3*

```

|| POKRETANJE: ./a.out 0
|| INTERAKCIJA SA PROGRAMOM:
|| 0

```

*Primer 4*

```

|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| Greska: nedostaje argument komandne linije!

```

[Rešenje 3.47]

**Zadatak 3.48** Napisati program koji proverava da li se među zadatim argumentima komandne linije nalaze barem dva ista.

*Primer 1*

```
|| POKRETANJE: ./a.out pec zima deda mraz pec
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima ima istih.
```

*Primer 2*

```
|| POKRETANJE: ./a.out xyz abc abc abc efgh
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima ima istih.
```

*Primer 3*

```
|| POKRETANJE: ./a.out 11 15 abc 888
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima nema istih.
```

*Primer 4*

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima nema istih.
```

[Rešenje 3.48]

**Zadatak 3.49** Napisati funkciju *void modifikacija(char\* s, char\* t, int\* br\_modifikacija)* koja na osnovu niske *s* formira nisku *t* tako što svako malo slovo zamenjuje velikim. Broj izvršenih modifikacija se čuva u okviru argumenta *br\_modifikacija*. Pretpostaviti da niska *s* neće biti duža od 20 karaktera. Napisati i program koji testira rad napisane funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 123abc789XY
|| Modifikovana niska je: 123ABC789XY
|| Broj modifikacija je: 3
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: zima
|| Modifikovana niska je: ZIMA
|| Broj modifikacija je: 3
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: SNEG
|| Modifikovana niska je: SNEG
|| Broj modifikacija je: 0
```

[Rešenje 3.49]

**Zadatak 3.50** Napisati funkciju *void interpunkcija(int\* br\_tacaka, int\* br\_zareza)* koja za tekst koji se unosi sa standardnog ulaza sve do kraja ulaza

prebrojava broj tačaka i zareza. Napisati zatim program koji testira napisanu funkciju.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
a.b.c.d
a,b,,c,d,e
Broj tacaka: 3
Broj zareza: 5
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
.....789.....
Broj tacaka: 10
Broj zareza: 0
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
sunce
Broj tacaka: 0
Broj zareza: 0
```

[Rešenje 3.50]

**Zadatak 3.51** Napisati funkciju *void par\_nepar(int a[], int n, int parni[], int\* pn, int neparni[], int\* nn)* koja razbija niz *a* na niz parnih i niz neparnih brojeva. Pokazivači *pn* i *nn* redom treba da sadrže broj elemenata niza parnih tj. niza neparnih elemenata. Pretpostaviti da dužina niza *a* neće biti veća od 50. Napisati program koji testira napisanu funkciju.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza:
1 8 9 -7 -16 24 77 4
Niz parnih brojeva: 8 -16 24 4
Niz neparnih brojeva: 1 9 -7 77
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
2 4 6 8 -11
Niz parnih brojeva: 2 4 6 8
Niz neparnih brojeva: -11
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 2
Unesite elemente niza:
-15 15
Niz parnih brojeva:
Niz neparnih brojeva: -15 15
```

[Rešenje 3.51]

**Zadatak 3.52** Napisati funkciju *void min\_max(float a[], int n, float\* min, float\* max)* koja izračunava minimalni i maksimalni element niza *a* dužine *n*. Napisati zatim i program koji učitava niz realnih brojeva maksimalne dužine

### 3 Predstavljanje podataka

---

50 i ispisuje vrednosti minimuma i maksimuma na tri decimala.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
24.16 -32.11 999.25 14.25 11
Minimum: -32.110
Maksimum: 999.250
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza:
-5.126 -18.29 44 29.268
Minimum: -18.290
Maksimum: 44.000
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 1
Unesite elemente niza:
4.16
Minimum: 4.160
Maksimum: 4.160
```

[Rešenje 3.52]

#### Zadatak 3.53 Tekst

[Rešenje 3.56]

**Zadatak 3.54** Ako su celi brojevi  $a$  i  $b$  argumenti komandne linije napraviti niz  $A[0] = a$ ,  $A[1] = a+1$ ,  $A[2] = a+2$ , ...,  $A[b-a] = b$  i ispisati ga. Pretpostaviti da je maksimalna dužina niza 200 elemenata. Proveriti da li  $a < b$  i  $b - a < 200$  i ako ovi uslovi nisu ispunjeni ispisati poruku da je došlo do greške. U slučaju da je dato manje ili više argumenata komandne linije ispisati poruku o grešci.

#### Primer 1

```
POKRETANJE: ./a.out 34
INTERAKCIJA SA PROGRAMOM:
greska
```

#### Primer 2

```
POKRETANJE: ./a.out 12 20
INTERAKCIJA SA PROGRAMOM:
12 13 14 15 16 17 18 19 20
```

#### Primer 3

```
POKRETANJE: ./a.out 30 8
INTERAKCIJA SA PROGRAMOM:
greska
```

#### Primer 4

```
POKRETANJE: ./a.out -4 -1
INTERAKCIJA SA PROGRAMOM:
-4 -3 -2 -1
```

[Rešenje 3.56]

**Zadatak 3.55** Uobičajena praksa na UNIX sistemima je da se argumenti komandne linije dele na opcije i argumente u užem smislu. Opcije počinju znakom '-' nakon čega obično sledi jedan ili više karaktera koji označavaju koja je

opcija u pitanju. Ovim se najčešće upravlja funkcionisanjem programa i neke mogućnosti se uključuju ili isključuju. Argumenti naćšće predstavljaju opisne informacije poput na primer imena datoteka. Napisati program koji ispisuje sve opcije koje su navedene u komandnoj liniji.

*Primer 1*

```

|| POKRETANJE: ./a.out -abc input.txt -d -Fg output
|| INTERAKCIJA SA PROGRAMOM:
||   a b c d F g

```

*Primer 2*

```

|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:

```

*Primer 3*

```

|| POKRETANJE: ./a.out ulaz.txt
|| INTERAKCIJA SA PROGRAMOM:

```

[Rešenje 3.56]

**Zadatak 3.56** Parametri komandne linije su  $n$ ,  $a$ ,  $b$  ( $a < b$ ). Treba popuniti prvih  $n$  elemenata niza  $A$  celim slučajnim brojevima koji su između  $a$  i  $b$ . Ištampati niz  $A$  na standardni izlaz. Maksimalan broj elemenata niza  $A$  je 200. Ukoliko nisu zadati svi argumenti komandne linije ili ne zadovoljavaju potrebna svojstva ispisati poruku o grešci.

[Rešenje 3.56]

## 3.4 Rešenja

### Rešenje 3.56

```

1  /*
2  Napisati funkciju uredi koja uredjuje svoja dva
   celobrojna argumenta tako da se u prvom nalazi manji
4  a u drugom veci. Napisati potom glavni program koji
   ucitava dva cela broja i uredjuje njihove vrednosti
6  primenom napisane funkcije. Na primer, ako su učitane
   promenljive x=5 i y=2, njihove vrednosti nakon
8  primene funkcije uredi treba da budu x=2 i y=5.
9  */
10
11 #include <stdio.h>
12
13 /*
14 Argumenti funkcije uredi_pogresno, promenljive a i b,

```

```
16     predstavljaju lokalne promenljive za ovu funkciju
17     i prestaju da postoje po zavrsetku funkcije. Zbog toga
18     se efekti razmene vrednosti promenljivih a i b u slucaju
19     da je a>b vide u funkciji, ali se ne vide u glavnom programu.
20 */
21 void uredi_pogresno(int a, int b)
22 {
23     int t;
24
25     if (a>b)
26     {
27         t = a;
28         a = b;
29         b = t;
30     }
31     printf("uredi_pogresno :: a=%d, b=%d\n", a, b);
32     printf("uredi_pogresno :: &a=%p, &b=%p\n", &a, &b);
33 }
34
35 /*
36     Argumenti funkcije uredi_tacno, promenljive pa i pb,
37     takodje su lokalne promenljive za ovu funkciju i
38     prestaju da postoje kada se funkcija zavrshi.
39     Njima prosledjujemo adrese promenljivih a i b koje zelimo
40     da razmenimo u slucaju da je a>b.
41
42     Promenljivoj a pristupamo preko pokazivacke promenljive
43     pa sa *pa i slicno, promenljivoj pb pristupamo sa *pb.
44
45     Vrednosti promenljivih *pa i *pb razmenjujemo kao
46     i vrednosti bilo koje dve celobrojne promenljive.
47 */
48 void uredi_tacno(int * pa, int * pb)
49 {
50     int t;
51     if (*pa>*pb)
52     {
53         t = *pa;
54         *pa = *pb;
55         *pb = t;
56     }
57     printf("uredi_tacno :: *pa=%d, *pb=%d\n ", *pa, *pb);
58     printf("uredi_tacno :: pa=%p, pb=%p\n ", pa, pb);
59 }
60
61 int main()
62 {
63     int a,b;
64
65     printf("Unesi dve celobrojne promenljive:");
66     scanf("%d%d",&a,&b);
```



```

68 printf("main :: a=%d, b=%d\n", a,b);
printf("main :: &a=%p, &b=%p\n", &a, &b);
70 uredi_pogresno(a,b);
printf("main :: nakon uredi_pogresno, a=%d, b=%d\n", a, b);
72
/*
74 Funkcija uredi_tacno kao argument ima dve pokazivacke
promenljive
(int*,int*). Zbog toga joj je u pozivu funkcije neophodno
proslediti
76 adrese promenljivih koje zelimo da uredimo rastuce, &a i &b.
*/
78
uredi_tacno(&a, &b);
80 printf("main :: nakon uredi_tacno, a=%d, b=%d\n", a, b);
82 return 0;
}

```

### Rešenje 3.39

```

1  /*
   Napisati funkciju koja za boju datu u rgb formatu
3  racuna cmy format po formulama:
   C = 1 - ( R / 255 )
5  M = 1 - ( G / 255 )
   Y = 1 - ( B / 255 )
7
   Napisati program koji učitava boju u rgb formatu,
9  primenjuje odgovarajucu funkciju i ispisuje boju u cmy formatu.
11 */
13 #include <stdio.h>
#include <math.h>
15
void rgb_to_cmy(float* a, float* b, float* c)
17 {
   /* Zagrade su neophodne jer aritmetičke operacije
19   imaju veći prioritet od operatora dereferenciranja (*).
   */
21 *a=1-(*a)/255;
   *b=1-(*b)/255;
23 *c=1-(*c)/255;
25
   /*
   Pomocu return ne mozemo vratiti vise od jedne vrednosti.
27
   Ceste greske:
29 return a,b,c;           return vraca samo jednu vrednost

```

### 3 Predstavljanje podataka

```
31     return a; return b; return c; return ce vratiti samo a
32
33     Zato je neophodno da promenljive ciju vrednost
34     zelimo da promenimo prenesemo preko pokazivaca.
35     */
36 }
37
38 int rgb_korektno(float a)
39 {
40     if(a<0 || a>255)
41         return 0;
42     return 1;
43 }
44
45 int main()
46 {
47     float a,b,c;
48
49     /*
50     Argumenti funkcije rgb_to_cmy su
51     pokazivaci na float. Njima prosledjujemo
52     adrese promenljivih a, b i c.
53     */
54
55     printf("Unesi boju u rgb formatu (vrednosti izmedju 0 i 255:");
56     scanf("%f%f%f",&a,&b,&c);
57
58     if(rgb_korektno(a) && rgb_korektno(b) && rgb_korektno(c))
59         rgb_to_cmy(&a,&b,&c);
60     else
61     {
62         printf("Nekorektan unos\n");
63         return -1;
64     }
65
66     printf("Nakon konverzije: %.2f,%.2f,%.2f\n", a,b,c);
67
68     return 0;
69 }
```

#### Rešenje 3.40

```
2  /*
3  Napisati funkciju koja za dve prave date svojim koeficijentima
4  pravca i slobodnim clanovima odredjuje njihovu tacku preseka.
5  Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju
6  tacku preseka (ako su paralelne). Napisati glavni program
   koji ucitava podatke o pravama, poziva napisanu funkciju i
   ispisuje odgovarajucu poruku.
```

```
8  */
10 #include<stdio.h>
12 /*
14     Funkcija presek treba da izracuna tri vrednosti:
16     1. indikator da li su koeficijenti pravca jednaki ili ne
18     2. prvu koordinatu presečne tačke (ukoliko prave nisu paralelne)
20     3. drugu koordinatu presečne tačke (ukoliko prave nisu paralelne)
22
24     Indikator funkcija vraca kao povratnu vrednost, preko ključne reci
26     return.
28
29     Koordinate presečne tačke (ako postoji) funkcija vraca preko
30     liste argumenata, zbog čega promenljive kojima će koordinate
31     biti dodeljene prenosimo preko pokazivaca (promenljive px i py)
32
33     Promenljive koje sadrže podatke o pravama (k1,n1,k2,n2) se ne
34     menjaju u funkciji i zbog toga ih ne moramo prenositi preko
35     pokazivaca.
36 */
37
38 int presek(float k1, float n1, float k2, float n2, float* px, float*
39           py)
40 {
41     if (k1==k2)
42         return 0;
43
44     *px = -(n1-n2)/(k1-k2);
45     *py = k1*(px)+n1;
46     return 1;
47 }
48
49 int main()
50 {
51     float k1,k2,n1,n2;
52     float x,y;
53
54     printf("Unesi k i n za prvu pravu:");
55     scanf("%f%f",&k1,&n1);
56
57     printf("Unesi k i n za drugu pravu:");
58     scanf("%f%f",&k2,&n2);
59
60     if(presek(k1,n1,k2,n2,&x,&y))
61         printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
62     else
63         printf("Prave su paralelne\n");
64
65     return 0;
66 }
```

### 3 Predstavljanje podataka

---

#### Rešenje 3.41

```
1  /*
   *   Napisati program koji ispisuje broj navedenih argumenata komandne
   *   linije,
   *   a zatim i same argumenate i njihove redne brojeve.
   */
5
6  #include <stdio.h>
7
8  /*
   *   Argumenti komandne linije cuvaju se u nizu niski pod nazivom
   *   argv. Svaki element tog niza odgovara jednom argumentu komandne
   *   linije pri cemu prvi element predstavlja naziv programa koji
   *   pokrecemo. Celobrojna promenljiva argc predstavlja ukupan
   *   broj argumenata komandne linije ukljucujuci i argument koji
   *   odgovara nazivu programa.
   */
15 /*
16
17 int main(int argc, char *argv[])
18 {
19     int i;
20
21     printf("Broj argumenata je: %d\n",argc);
22
23     for(i=0; i<argc; i++)
24         printf("%d: %s\n",i,argv[i]);
25
26     return 0;
27 }
```

#### Rešenje 3.42

```
1  /*
   *   Napisati funkciju koja za dva data stringa str i
   *   accept odredjuje koliko se uzastopnih karaktera stringa str
   *   nalazi u stringu accept pocev od pocetka niza str. Napisati
   *   potom program koji testira napisanu funkciju za dva stringa
   *   koji se unose kao argumenti komandne linije. Primeri upotrebe:
   *
   *   1:
   *   ./a.out aladin bal
   *   3
   *
   *   2:
   *   ./a.out aladin lad
   *   4
   *
   *   3:
   *   ./a.out Aladin ala
   */
```

```
0
19
20 */
21
22 #include <stdio.h>
23 #include <string.h>
24
25 /*
26 Funkcija strspn(str,accept) je ugradjena funkcija koja vraca broj
27 karaktera
28 stringa str koji se nalaze u stringu accept, pocev od pocetka
29 stringa str.
30
31 Funkcija strspn se nalazi u zaglavlju string.h.
32
33 Funkcija strspn_klon je jedna implementacija funkcije strspn.
34
35 U zadacima cemo uvek koristiti ugradjenu funkciju strspn osim ako
36 u tekstu zadatka
37 nije naglaseno da se ona ne sme koristiti. Funkcija strspn_klon
38 sluzi da pokaze na koji
39 nacin radi ugradjena funkcija strspn.
40
41 Ugradjena funkcija strspn poziva se na isti nacin kao funkcija
42 strspn_klon:
43 strspn(s1,s2)
44
45 */
46
47 int strspn_klon(char str[], char accept[])
48 {
49     int br=0;
50     int i;
51
52     for(i=0; str[i];i++)
53         if(strchr(accept, str[i])!=NULL)
54             br++;
55         else /* ako pronadjemo karakter u stringu str koji nije */
56             break; /* u stringu accept, prekidamo petlju */
57
58     return br;
59 }
60
61 int main(int argc, char* argv[])
62 {
63     int br;
64
65     if(argc<3)
66     {
67         printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
68             arg2\n");
69     }
70 }
```

### 3 Predstavljanje podataka

---

```
        return -1;
65     }

67     br = strspn_klon(argv[1],argv[2]);
    printf("Broj karaktera stringa %s koji se nalaze u stringu %s,
        pocev od pocetka stringa %s: %d\n", argv[1],argv[2],argv[1],br);
69     return 0;
    }
```

#### Rešenje 3.43

```
/*
2   Napisati funkciju void sifruj(char s[], char c, int k) koja
    sifruje
    string s na sledeci nacin: svako malo i veliko slovo stringa s
    konvertuje u
4   slovo koje je u abecedi od njega udaljeno k pozicija, i to
    k pozicija ulevo, ako je karakter c jednak karakteru 'L' ili
    udesno
6   ako je karakter c jednak karakteru 'D'. Sifrovanje treba da bude
    kruzno. Ako string
    s sadrzi karakter koji nije alfanumericki, ostaviti ga
    nesifriranog.
8
    Napisati potom glavni program koji testira napisanu funkciju za
    string i prirodan
10   broj koji se unose kao argumenti komandne linije dok se pravac
    sifrovanja unosi
    kao opcija -p koja moze imati vrednosti 'L' ili 'D'. Ukoliko
    opcija -p nije
12   navedena, podrazumevani pravac je udesno.

14   Mozemo podrazumevati da string sadrzi najvise 30 karaktera.

16   Primeri upotrebe:

18   1:
    ./a.out abcd 2
20   cdef

22   2:
    ./a.out abcd 2 -p D
24   cdef

26   3:
    ./a.out abcd 2 -p L
28   yzab

30   4:
    ./a.out abcd -3 -p L
32   Nekorektan unos
```

```
34     5:
35     ./a.out abcd 3 -p X
36     Nekorektan unos
37
38     6:
39     ./a.out ab12cd 2 -p D
40     cd12ef
41
42 */
43
44 #include <stdio.h>
45 #include <string.h>
46 #include <stdlib.h>
47 #define MAX 31
48
49 void sifruj(char s[], char c, int k)
50 {
51     int i;
52     int znak;
53     char t;
54
55     /*
56      S obzirom da ce korektnost unosa podataka
57      biti ispitana pre poziva funkcije, promenljiva
58      c ce imati vrednost 'L' ili 'D'.
59
60      Promenljiva znak ima vrednost 1 ili -1
61      i sluzi kao pomocna promenljiva u slucaju
62      da prilikom sifriranja konvertovani
63      karakter izadje iz opsega malih ili velikih slova.
64
65      */
66     znak=1;
67     if (c=='L')
68         znak = -1;
69
70
71     for(i=0; s[i];i++)
72         if(isalpha(s[i]))
73         {
74             /*
75              Promenljiva t predstavlja sifrirani karakter s[i].
76              Ako je promenljiva t izvan opsega malih ili velikih slova
77              ,
78              dodajemo joj ili oduzimamo ukupan broj slova u abecedi
79              (26),
80              u zavisnosti od pravca sifriranja, kako bismo omogucili
81              kruzno sifriranje.
82              */
83             t = s[i]+znak*k;
84             if((islower(s[i]) && (t<'a' || t>'z')) || (isupper(s[i]) &&
```

### 3 Predstavljanje podataka

```

    (t<'A' || t>'Z'))
        s[i]=t-znak*26;
84     else
        s[i]=t;
86     }
}

88
int main(int argc, char* argv[])
90 {
92     int k;
93     char pravac;
94     char rec[MAX];
95
96     /*
97     Program mozemo pozivati na dva nacina:
98     ./a.out abcd 2
99     ili
100    ./a.out abcd 2 -p D
101
102    Zbog toga, broj argumenata moze biti 3 ili 5.
103    */
104
105    if (argc!=3 && argc!=5)
106    {
107        printf("Nekorektan unos: broj argumenata moze biti 3 ili 5\n");
108        return -1;
109    }
110
111    /*
112    Argumenti komandne linije su stringovi. Ako program pokrecemo
113    na sledeci nacin:
114    ./a.out abcd 2 -p D
115    to znaci da je argument koji odgovara dvojci u stvari
116    string "2". Da bismo string konvertovali u ceo broj,
117    koristimo ugradjenu funkciju atoi iz biblioteke stdlib.h.
118    */
119
120    k = atoi(argv[2]);
121
122    /*
123    Ispitujemo korektnost datih podataka:
124    */
125    if (k<=0)
126    {
127        printf("Nekorektan unos: broj pozicija mora biti pozitivan ceo
128        broj\n");
129        return -1;
130    }
131
132    /* Korektnost unosa je ispitana, sto znaci da
    argc moze biti 3 ili 5 */

```



```
134     if (argc==3) /* Ako je argc 3: */
135         pravac='D';
136     else /* Ako argc nije 3, tada je sigurno 5, jer je */
137     { /* korektnost unosa ispitana, a unos je korektan
138         /*
139         Ispitujemo korektnost pretposlednjeg argumenta koji mora da
140         bude u formatu "-p".
141         Ovaj argument je string argv[3]. Njegovom prvom karakteru (
142         koji treba
143         da bude '-') pristupamo sa argv[3][0] a drugom sa argv
144         [3][1].
145         */
146         if (argv[3][0] != '-')
147         {
148             printf("Nekorektan unos: pri zadavanju opcija prvi karakter
149             mora biti '-' \n");
150             return -1;
151         }
152
153         if (argv[3][1]!='p')
154         {
155             printf("Nekorektan unos: nedozvoljena opcija\n");
156             return -1;
157         }
158
159         /*
160         Nakon argumenta -p sledi argument koji zadaje vrednost ove
161         opcije. To je
162         poslednji argument kome pristupamo sa argv[4]. Ovaj argument
163         treba
164         da sadrzi samo jedan karakter - 'L' ili 'D' i njemu
165         pristupamo sa
166         argv[4][0].
167         */
168         if(argv[4][0]=='L' || argv[4][0]=='D')
169             pravac=argv[4][0];
170         else
171         {
172             printf("Nekorektan unos: pravac moze biti L ili D\n");
173             return -1;
174         }
175     }
176
177     strcpy(rec, argv[1]);
178     sifruj(rec,pravac,k);
179
180     printf("Sifrovana rec: %s\n", rec);
181
182     return 0;
183 }
```

#### Rešenje 3.44

```
1 #include <stdio.h>
3 int main(int argc, char* argv[]) {
5     int i;
6     int s = 0;
7
8     /* char *argv[] <--- niz niski koje predstavljaju argumente
9        navedene iza poziva programa
10    int argc    <--- ukupan broj niski (sa sve nazivom programa)
11    navedenih prilikom pozivanja
12
13    Ukoliko je program pozvan sa ./a.out 12 abc 6 5 3ab
14
15    argv[0] = "./a.out".
16    argv[1] = "12"
17    argv[2] = "abc"
18    argv[3] = "6"
19    argv[4] = "5"
20    argv[5] = "3ab"
21
22    argc iznosi 6
23
24    Kako je argv[] po prirodi niz,
25    koristimo tzv. brojacku odnosno
26    for petlju
27    i obradjujemo svaki od argumenata.
28    */
29
30    /* Funkcija atoi() prihvata nisku,
31    i racuna dekadnu vrednost prosledjene niske,
32    dokle god se ona moze racunati.
33    Na primer, ukoliko je niska "-123",
34    atoi() vraca broj -123.
35    Ako je, pak, niska "123abc",
36    atoi() ce vratiti 123
37    (prilikom prve pojave karaktera koji nije cifra, funkcija prekida
38    izracunavanje).
39
40    To za posledicu ima da, ukoliko je funkciji
41    prosledjeno nesto
42    sto se ne moze pretvoriti u broj,
43    na primer niska "abcd",
44    funkcija atoi() vraca 0.
45    */
46
47    for(i = 1; i < argc; i++)
48        s += atoi(argv[i]); /* Zbog nacina rada funkcije atoi(), mozemo
49        je pozvati nad svim argumentima
```

```

47         komandne linije, i sabrati odgovarajuće dekadne
vrednosti.
        Ukoliko neki argument i nije broj, to ne predstavlja
problem
49         jer će u tom slučaju odgovarajući sabirak biti 0
        */
51
53     printf("Zbir numerickih argumenata: %d\n", s);
55     return 0;
}

```

### Rešenje 3.45

```

1  #include <stdio.h>
3  int main(int argc, char* argv[]) {
5      int i;
7      /* Prolazimo for petljom kroz niz argumenata,
i trazimo one niske ciji je prvi karakter bas 'z'.
9      Ukoliko je trenutni argument koji se ispituje
argv[i],
11     kako je on sam po sebi niska,
do prvog karaktera dolazimo kao i pri dosadasnjem
13     radu sa niskama --> argv[i][0]
        ^
15         |
        index prvog karaktera u niski argv[i]
17     */
19     for(i = 1; i < argc; i++)
        if(argv[i][0] == 'z')
21         printf("%s ", argv[i]);
23     putchar('\n');
25     return 0;
}

```

### Rešenje 3.46

```

1  #include <stdio.h>
#include <string.h>
3
int main(int argc, char* argv[]) {
5

```

### 3 Predstavljanje podataka

---

```
int i;
7 int br = 0;

9 /* Da bismo proverili da li se karakter 'z' (tj. 'Z')
   nalazi u niski argv[i],
11 to mozemo uciniti koriscenjem funkcije
   strchr() koja se nalazi u string.h.

13
   Ukoliko je karakter sadržan u okviru niske,
15 strchr() vraća pokazivac na taj karakter
   unutar same niske.
17 Inace, ukoliko se karakter ne nalazi u niski,
   funkcija vraća NULL.
19 */

21 for(i = 1; i < argc; i++)
   if(strchr(argv[i], 'z') != NULL || strchr(argv[i], 'Z') != NULL)
23     br++;

25 printf("%d\n", br);

27 return 0;
}
```

#### Rešenje 3.47

```
1 #include <stdio.h>
   #include <stdlib.h>

3
   int main(int argc, char *argv[])
5 {
   int n,i;

7
   /*
9     Ispisujemo gresku ukoliko nema dovoljno argumenata komandne
       linije.
   */
11 if(argc != 2)
   {
13     printf("Greska: nedostaje argument komandne linije!\n");
       return -1;
15 }

17 /*
       Pretvaramo argument komandne linije koji je string u ceo broj
       koriscenjem funkcije atoi
19 */
   n = atoi(argv[1]);
21 n = abs(n);

23 for(i=(-1)*n;i<=n;i++)
```

```

    printf("%d ",i);
25
    return 0;
27 }

```

### Rešenje 3.48

```

1  #include <stdio.h>
   #include <string.h>
3
   int main(int argc, char *argv[])
5  {
       int indikator = 0;
       int i,j;
       /*
9      Ukoliko imamo samo jedan argument komandne linije,
       ispisujemo da nema istih i završavamo program.
11     */
       if(argc < 2)
13     {
           printf("Medju argumentima nema istih.\n");
15         return -1;
       }

17     /*
19     Prolazimo kroz niz argumenata i za svaki posebno proverimo
       da li medju ostalima postoji neki koji mu je jednak i ako postoji
21     ispisujemo poruku i završavamo program.
       Ako smo izasli iz prve petlje to znaci da nismo pronasli dva ista
       elementa
23     i ispisujemo odgovarajucu poruku.
       */
25     for(i=0;i<argc;i++)
       {
27         for(j=0;j != i && j<argc; j++)
           if(strcmp(argv[i], argv[j]) == 0)
29         {
               printf("Medju argumentima ima istih.\n");
31               return 0;
           }
33     }

35     printf("Medju argumentima nema istih.\n");
       return 0;
37 }

```

### Rešenje 3.49

```

#include <stdio.h>

```

### 3 Predstavljanje podataka

---

```
2
#define MAX 21
4
void modifikacija(char *s, char *t, int *br_modifikacija)
6 {
    int i;
8     for(i=0; s[i]; i++)
        if(s[i]>='a' && s[i]<='z')
10     {
        t[i] = toupper(s[i]);
12        (*br_modifikacija)++;
    }
14     else
        t[i] = s[i];
16 }

18 int main()
{
20     char s[MAX], t[MAX];
    int br_modifikacija = 0;
22
    printf("Unesite nisku: ");
24     scanf("%s", s);

26     modifikacija(s, t, &br_modifikacija);

28     printf("Modifikovana niska je: %s\nBroj modifikacija je: %d\n", t,
        br_modifikacija);

30     return 0;
}
```

#### Rešenje 3.50

```
1 /*
   Napisati funkciju
3   void interpunkcija(int * br_tacaka, int * br_zareza)
   koja za tekst koji se unosi sa standardnog ulaza sve do kraja ulaza
   prebrojava
5   broj tacaka i zareza. Napisati zatim program koji testira napisanu
   funkciju.
   */
7
9 #include <stdio.h>

11 void interpunkcija(int* br_tacaka, int* br_zareza){
13     int tacke=0, zarezi=0;
    char c;
15 }
```

```

17     while((c=getchar())!=EOF){
18         if(c==',')
19             tacke++;
20
21         if(c==' ')
22             zarezi++;
23     }
24
25     *br_tacaka=tacke;
26     *br_zareza=zarezi;
27 }
28
29 int main(){
30     int br_tacaka, br_zareza;
31
32     printf("Unesite tekst: \n");
33
34     interpunkcija(&br_tacaka, &br_zareza);
35
36     printf("Broj tacaka: %d\n", br_tacaka);
37     printf("Broj zareza: %d\n", br_zareza);
38
39     return 0;
40 }

```

### Rešenje 3.51

```

/*
2  Napisati funkciju
   void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
   int* nn)
4  koja razbija niz a na niz parnih i niz neparnih brojeva. Pokazivaci
   pn i nn
   redom treba da sadrže broj elemenata niza parnih tj. niza neparnih
   elemenata.
6  Pretpostaviti da dužina niza a neće biti veća od 50. Napisati program
   koji
   testira napisanu funkciju.
8  */

10 #include <stdio.h>
11 #define MAX 50
12
13 void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
14 int* nn){
15
16     int i, j, k;
17
18     /* i - brojac niza a */
19     /* j - brojac niza parnih brojeva */

```

```
/* k - brojac niza neparnih brojeva */
20
for(i=0, j=0, k=0; i<n; i++){
22     /* Ako je element niza paran */
    if(a[i]%2==0){
24         /* Smestamo ga u niz parnih brojeva i uvecavamo indeks niza
        j */
        parni[j]=a[i];
26         j++;
    }
    else{
28         /* Inace, smestamo ga u niz neparnih brojeva i uvecavamo
        indeks niza k */
        neparni[k]=a[i];
30         k++;
    }
32 }
34
*pn=j;
36 *nn=k;
38 }

40 int main(){
    int n, i, j, pn, nn;
42     int a[MAX], parni[MAX], neparni[MAX];

44     /* Ucitavamo dimenziju niza */
    printf("Unesite broj elemenata niza: ");
46     scanf("%d", &n);

48     if(n<0 || n>MAX){
        printf("Greska: pogresna dimenzija niza!\n");
50         return 0;
    }

52     /* Ucitavamo elemente niza */
    printf("Unesite elemente niza: ");
54     for(i=0; i<n; i++){
        scanf("%d", &a[i]);
56     }

58     /* Pozivamo funkciju koja razbija zadati niz na niz parnih i niz
        neparnih */
60     par_nepar(a, n, parni, &pn, neparni, &nn);

62     /* Ispisujemo dobijene nizove */
64     printf("Niz parnih brojeva: ");
    for(i=0; i<pn; i++)
        printf("%d ", parni[i]);
66     printf("\n");
```



```

68     printf("Niz neparnih brojeva: ");
70     for(i=0; i<nn; i++)
        printf("%d ", neparni[i]);
72     printf("\n");

74     return 0;
}

```

### Rešenje 3.52

```

/*
2   Napisati funkciju
   void min_max(float a[], int n, float* min, float* max)
4   koja izracunava minimalni i maksimalni element niza a duzine n.
   Napisati zatim i program koji ucitava niz realnih brojeva
   maksimalne
6   duzine 50 i ispisuje vrednosti minimuma i maksimuma na tri decimalne
   .

8  */

10 #include<stdio.h>
   #define MAX 50

12 void min_max(float a[], int n, float* min, float* max){
14     int i;

16     /* Inicijalizujemo vrednosti minimuma i maksimuma */
18     *min=a[0];
   *max=a[0];

20     /* Obilazimo preostale elemente niza */
22     for(i=1; i<n; i++){

24         /* Ako je tekuca vrednost veca od maksimalne, azuriramo maksimum
           */
26         if(a[i]>*max){
           *max=a[i];
28         }

           /* Ako je tekuca vrednost manja od minimalne, azuriramo minimum
           */
30         if(a[i]<*min){
           *min=a[i];
32         }
34     }

36 int main(){

```

### 3 Predstavljanje podataka

---

```
int i, n;
38 float a[MAX], min, max;

40 /* Ucitavamo dimenziju niza */
printf("Unesite broj elemenata niza: ");
42 scanf("%d", &n);

44 if(n<0 || n>MAX){
    printf("Greska: pogresna dimenzija niza!\n");
46     return 0;
}

48 /* Ucitavamo elemente niza */
printf("Unesite elemente niza:\n");
50 for(i=0; i<n; i++){
    scanf("%f", &a[i]);
52 }

54 /* Pozivamo funkciju za racunanje maksimuma i minimuma */
min_max(a, n, &min, &max);

56 /* Ispisujemo rezultat */
printf("Minimum: %.3f\n", min);
60 printf("Maksimum: %.3f\n", max);

62 return 0;
64 }
```

#### Rešenje 3.56

```
1 #include <stdio.h>

3 void suma(int a, int b, int *s);

5
int main()
7 {
    int a,b,s;
9
    scanf("%d%d",&a,&b);
11
    suma(a,b,&s);
13
    printf("suma: %d\n",s);
15
    return 0;
17 }

19 void suma(int a, int b, int *s)
{
```

```
21     *s = a + b;
    }
```

### Rešenje 3.56

```
1  #include <stdio.h>
3  void suma(int a, int b, int *s);
5
7  int main()
8  {
9      int a,b,s;
10     scanf("%d%d",&a,&b);
11
12     suma(a,b,&s);
13
14     printf("suma: %d\n",s);
15
16     return 0;
17 }
19 void suma(int a, int b, int *s)
20 {
21     *s = a + b;
22 }
```

### Rešenje 3.56

```
1  #include <stdio.h>
3  void suma(int a, int b, int *s);
5
7  int main()
8  {
9      int a,b,s;
10     scanf("%d%d",&a,&b);
11
12     suma(a,b,&s);
13
14     printf("suma: %d\n",s);
15
16     return 0;
17 }
19 void suma(int a, int b, int *s)
```

```
21 {  
    *s = a + b;  
}
```

#### Rešenje 3.56

```
2  #include <stdio.h>  
4  
6  void suma(int a, int b, int *s);  
8  
10 int main()  
12 {  
14     int a,b,s;  
16     scanf("%d%d",&a,&b);  
18     suma(a,b,&s);  
20     printf("suma: %d\n",s);  
22     return 0;  
24 }  
26 void suma(int a, int b, int *s)  
28 {  
30     *s = a + b;  
32 }
```

## 3.5 Niske

**Zadatak 3.57** Tekst

[Rešenje 3.57]

**Zadatak 3.58** Tekst

[Rešenje 3.58]

**Zadatak 3.59** Tekst

[Rešenje 3.59]

**Zadatak 3.60** Tekst

[Rešenje 3.60]

**Zadatak 3.61** Tekst

[Rešenje 3.61]

**Zadatak 3.62** Tekst

[Rešenje 3.62]

**Zadatak 3.63** Tekst

[Rešenje 3.63]

**Zadatak 3.64** Tekst

[Rešenje 3.64]

**Zadatak 3.65**

- Napisati funkciju *int samoglasnik(char c)* koja proverava da li je zadati karakter samoglasnik. Funkcija treba da vrati vrednost 1 ako karakter *c* jeste samoglasnik, odnosno 0 ako nije.
- Napisati funkciju *int samoglasnik\_na\_kraju(char s[])* koja proverava da li se niska *s* završava samoglasnikom (koristiti funkciju iz tačke a)).
- Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje da li završava samoglasnikom ili ne.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: abcde
|| Niska se završava samoglasnikom!
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: AaBb+cCdD
|| Niska se ne završava samoglasnikom!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: pRograMiranjE
|| Niska se završava samoglasnikom!
```

[Rešenje 3.65]

**Zadatak 3.66** Napisati funkciju *void kopiraj\_n(char t[], char s[], int n)* koja kopira najviše *n* karaktera niske *s* u nisku *t*. Napisati i program koji učitava nisku maksimalne dužine 20 karaktera i jedan ceo broj i testira rad napisane funkcije.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: abcdef  
| Unesite broj n: 3  
| Rezultujuca niska: abc
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: programiranje  
| Unesite broj n: 5  
| Rezultujuca niska: progr
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: abc  
| Unesite broj n: 15  
| Rezultujuca niska: abc
```

[Rešenje 3.66]

**Zadatak 3.67** Napisati funkciju *void dupliranje(char t[], char s[])* koja na osnovu niske *s* formira nisku *t* tako što duplira svaki karakter niske *s*. Napisati i program koji učitava nisku maksimalne dužine 20 karaktera i testira rad napisane funkcije.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: zima  
| zziimmaa
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: A+B+C  
| AA++BB++CC
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: C  
| CC
```

[Rešenje 3.67]

**Zadatak 3.68** Napisati funkciju *int heksa\_broj(char s[])* koja proverava da li je niskom *s* zadat korektan heksadekadni broj. Heksadekadni broj je korektno zadat ako počinje prefiksom *0x* ili *0X* i ako sadrži samo cifre i mala ili velika slova *A*, *B*, *C*, *D*, *E* i *F*. Funkcija treba da vrati vrednost 1 ako je niska korektan heksadekadni broj, odnosno 0 ako nije. Napisati i program koji učitava nisku maksimalne dužine 7 karaktera i ispisuje rezultat rada funkcije.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: 0x12EF  
| Korektan heksadekadni broj!
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite nisku: 0X22af  
| Korektan heksadekadni broj!
```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0xErA9
|| Nekorektan heksadekadni broj!

```

[Rešenje 3.68]

**Zadatak 3.69** Napisati funkciju *int heksa\_broj(char s[])* koja izračunava dekadnu vrednost heksadekadnog broja zadatog niskom *s*. Napisati i program koji učitava nisku maksimalne dužine 7 karaktera i ispisuje rezultat rada funkcije. Pretpostaviti da je uneta niska korektan heksadekadni broj.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0x2A34
|| 10804

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0xff2
|| 4082

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0xEIA9
|| 57769

```

[Rešenje 3.69]

**Zadatak 3.70** Napisati funkciju *int podniska(char s[], char t[])* koja proverava da li je niska *t* podniska niske *s*. Napisati i program koji učitava dve niske maksimalne dužine 10 karaktera i testira rad napisane funkcije.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: abcde
|| Unesite nisku t: bcd
|| t je podniska niske s!

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: abcde
|| Unesite nisku t: bCd
|| t nije podniska niske s!

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: abcde
|| Unesite nisku t: def
|| t nije podniska niske s!

```

[Rešenje 3.70]

### 3 Predstavljanje podataka

---

**Zadatak 3.71** Napisati funkciju *void modifikacija(char \* s)* koja modifikuje nisku *s* tako što svaki drugi karakter zameni zvezdicom. Pretpostaviti da niska *s* neće biti duža od 20 karaktera. Napisati i program koji testira rad napisane funkcije.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: 123abc789XY  
|| Modifikovana niska je: 1*3*b*7*9*Y
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: zima  
|| Modifikovana niska je: z*m*
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: SNEG  
|| Modifikovana niska je: S*E*
```

[Rešenje 3.71]

**Zadatak 3.72** Napisati funkciju *int strspn\_klon(char \* t, char \* s)* koja izračunava dužinu prefiksa niske *t* sastavljenog od karaktera niske *s*. Napisati zatim i program koji učitava dve niske maksimalne dužine 20 karaktera i ispisuje rezultat poziva napisane funkcije.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku t: programiranje  
|| Unesite nisku s: opqr  
|| 3
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku t: aaiioo124  
|| Unesite nisku s: aeiou  
|| 6
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku t: 5296abc  
|| Unesite nisku s: 0123456789  
|| 4
```

[Rešenje 3.72]

**Zadatak 3.73** Napisati implementaciju funkcije *char \* strchr\_klon(char \* s, char c)* koja vraća pokazivač na prvo pojavljivanje karaktera *c* u niski *s* ili NULL ukoliko se karakter *c* ne pojavljuje u niski *s*. Učitati potom jednu nisku maksimalne dužine 20 karaktera i jedan dodatni karakter i testirati rad napisane funkcije.



*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: programiranje
|| Unesite karakter c: a
|| Karakter se nalazi u niski!

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: 123456789
|| Unesite karakter c: y
|| Karakter se ne nalazi u niski!

```

[Rešenje 3.73]

**Zadatak 3.74**

- Napisati funkciju

```
int prepis(char a[][21], int na, char b[][21])
```

koja iz niza reči **a** dužine **na** prepisuje u niz **b** reči koje su zapisane samo malim ili samo velikim slovima. Informaciju o dužini niza **b** (broj reči koje zadovoljavaju prethodni uslov) vratiti kao povratnu vrednost funkcije.

- Napisati program koji sa standardnog ulaza učitava prvo broj reči (strogo veći od nule, manji od 50), a zatim i same reči razdvojene blanko znakom (smatrati da reči koje se unose sa ulaza neće biti duže od 20 karaktera - ovaj uslov ne proveravati). Za slučaj kada je broj reči izvan traženog opsega ispisati -1 i prekinuti izvršavanje programa. Korišćenjem prethodno definisane funkcije **prepis**, odrediti sve reči koje su zapisane samo malim ili samo velikim slovima. Rezultat ispisati na standardni izlaz. Napomena: Ukoliko se pri rešavanju zadatka ne bude koristila funkcija **prepis**, zadatak neće biti pregledan i nosiće nula poena.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| 3 abc ABC aBc
|| abc ABC

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| 2 mmB RGa

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| -3
|| -1

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| 4 2abc AVF$ abc AV4
|| abc

```

[Rešenje 3.91]

**Zadatak 3.75** Napisati funkciju `void min_razlika(char s[], char s1[], char s2[])` koja u datotoj nisci s pronalazi dve reči koje imaju minimalnu razliku između svojih

### 3 Predstavljanje podataka

---

samoglasnika. ( Reč je niz karaktera između dve praznine; razmak između samoglasnika reči *dan*as i *ju*tro je 2, a razmak između *sut*rk i *mno*zenje je 5). Testirati pozivom u main-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.76** Napisati funkciju `int pp(char s[], char t[])` koja određuje poziciju poslednjeg karaktera niske *s* sadržanog u okviru niske *t*, zanemarujući pri tom razliku između velikih i malih slova, ili -1 ako takvog karaktera nema. Testirati pozivom u main-u. Maksimalna dužina niske je 20 karaktera.

#### Primer 1

INTERAKCIJA SA PROGRAMOM:

```
a4BA3Bc A3b
5
```

[Rešenje 3.91]

**Zadatak 3.77** Napisati funkciju `int f1(char s[])` koja prihvata tu nisku i proverava da li niska sadrži veliko slovo. Funkcija vraća 1 ako sadrži veliko slovo, inače 0. Testirati pozivom u main-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.78** Napisati funkciju `void ukloniSlova(char s[])` koja iz niske *s* uklanja sva mala i velika slova. Testirati pozivom u main-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.79** Napisati funkciju `unsigned btoi(char* s, unsigned char b)` koja određuje vrednost zapisa datog neoznačenog broja *s* u datoj osnovi *b*. Napisati funkciju `void itob(unsigned n, unsigned char b, char* s)` koja datu vrednost *n* zapisuje u datoj osnovi *b* i smešta rezultat u nisku *s*. Napisati zatim program koji čita liniju po liniju sa standardnog ulaza i obrađuje ih sve dok ne naiđe na praznu liniju. Svaka linija sadrži jedan dekadni, oktalni ili heksadekadni broj (zapisan kako se zapisuju konstante u programskom jeziku C). Program za svaki uneti broj ispisuje njegov binarni zapis. Pretpostaviti da će svi uneti brojevi biti u opsegu tipa `unsigned`.

#### Primer 1

INTERAKCIJA SA PROGRAMOM:

```
0x49 0x1ABC
1001001 1101010111100
```

#### Primer 2

INTERAKCIJA SA PROGRAMOM:

```
012 435 0x64FE
1010 110110011 110010011111110
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 123 0777
|| 1111011 11111111
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 981
|| 1111010101
```

[Rešenje 3.91]

**Zadatak 3.80** Implementirati funkciju `int str_str(char s[], char t[])` koja proverava da li niska `s` sadrzi nisku `t`. Zatim napisati program koji sa standardnog ulaza učitava pet redova (svaki red ima najviše 100 karaktera) i koji ispisuje sve redne brojeve linija koje sadrže nisku `program` (linije se numerišu od broja 1). Ukoliko ne postoji red sa niskom `program` ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| novi red*nprogram
|| c prog. jezik
|| c? programskih jezik
|| Programski odbor
|| <b>program</b>
|| 1 3 5
```

[Rešenje 3.91]

**Zadatak 3.81** Napisati funkciju `void sifrat(char* rec, char* kljuc)` koja šifruje `rec` na sledeći način: za svako slovo reči `rec` i odgovarajuće slovo ključa određuje koliki je (alfabetски) razmak između njih i označimo taj broj sa `k`. Potom to slovo `reci` zamenjuje `k`-tim slovom alfabeta. Podrazumeva se da je ključ duži od reči.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| bac
|| dfge
|| bed
```

[Rešenje 3.91]

**Zadatak 3.82** Napisati funkciju `void obrni(char rec[], int k)` koja

### 3 Predstavljanje podataka

---

rotira reč za k mesta ulevo.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| sveska  
|| 2  
|| eskasv
```

[Rešenje 3.91]

**Zadatak 3.83** Napisati sledece funkcije:

```
int poredjenje(char* s1, char* s2);  
// vraca 1 ako su s1 i s2 iste niske, 0 u suprotnom
```

```
void uVelikaSlova(char* s);  
// pretvara sva slova niske s u velika, ostale znakove ne menja
```

Napisati program koji sa standardnog ulaza učitava dve reči (dužine najviše 20 znakova) i, koristeći ove dve funkcije, ispisuje da li su one jednake ako se sva slova pretvore u velika slova.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| isPit2010  
|| IsPiT2010  
|| jesu jednake
```

[Rešenje 3.91]

**Zadatak 3.84** Napisati program kojim se sadržaj unetog stirnga šifrira tako što se svako slovo zamenjuje sledećim ASCII slovom, a znakovi 'z' i 'Z' zamenjuju redom sa 'a' i 'A'. Uneta reč nije duža od 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.85** Napisati funkciju `void sifruj(char rec[], char sifra[])` koja na osnovu date reči formira šifru koja se dobija tako što se svako slovo u reči zameni sa naredna tri slova koja su mu susedna u abecedi. Na primer, reč "tamo" treba da bude zamenjena sa "uvwbcnoppqr" a reč "zec" sa "abcfghdef". Napisati program koji šifrue unetu reč sa standardnog ulaza i štampa dobijeni rezultat na standardni izlaz. Za reč pretpostaviti da nije duža od 20 karaktera. Unos reči ostvariti koristeći specifikator "

[Rešenje 3.91]

**Zadatak 3.86** Sa ulaza se unosi reč koja nije duža od 20 znakova. Napisati program koji formira i štampa rezultujuću reč koja se dobija tako što se uneta

reč kopira 4 puta pri čemu se između svakog kopiranja umeće crtica. Na primer ako je uneta reč **ana**, formirana reč treba da bude **ana-ana-ana-ana**.

Zadatak uraditi:

- (a) pisanjem odgovarajuće funkcije koja vrši nadovezivanje reči,
- (b) koristeći postojeću funkciju iz biblioteke `string.h` (`strcat`).

Napomena: voditi računa da se za rezultujuću reč odvoji odgovarajuća količina memorije.

[Rešenje 3.91]

**Zadatak 3.87** Sa ulaza se unosi reč koja nije duža od 20 znakova. Napisati funkciju koja svako pojavljivanje znaka koji se zadaje kao prvi argument funkcije udvaja a svako pojavljivanje znaka koji se zadaje kao drugi argument funkcije izbacuje. Napisati program koji poziva ovu funkciju za reč unetu sa standardnog ulaza i za znakove koji se takođe zadaju sa standardnog ulaza. Na primer, ako se unese reč **ana** i znakovi **a** i **n**, tada funkcija treba da izmeni reč tako da ona postane **aaaa**, ako se unese reč **abrakadabra** i znakovi **a** i **b**, tada funkcija treba reč da izmeni tako da ona postane **aaraakaadkkraa**.

Napomena: voditi računa da novonastala izmenjena reč može imati veći broj karaktera i u skladu sa tim rezervisati odgovarajuću količinu memorije. Dopusšteno je koristiti pomoćan niz.

[Rešenje 3.91]

**Zadatak 3.88** Napisati funkciju `void ukloni(char *s);` koja iz niske uklanja sva slova iza kojih neposredno sledi slovo koje je u abecedi nakon njih (veličina slova se zanemaruje). Testirati funkciju u programu koji učitava liniju teksta (najviše 100 karaktera).

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
||   zdRaVo svIma
||   zRVo vma
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
||   12345AbcD
||   12345D
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
||   JeD1aN D52Va.
||   JeD1N D52Va.
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
||   abcd efg
||   d g
```

[Rešenje 3.91]

#### Zadatak 3.89

- a) Napisati C funkciju `int procitaj_recenicu(char *s, int max_len)`, koja sa standardnog ulaza čita rečenicu i smešta je u nisku `s`. Čitanje rečenice se zaustavlja ako se pročita simbol `.` ili je već učitano `max_len-1` karaktera. Funkcija treba da vrati broj pročitanih karaktera.
- b) Napisati C funkciju `void prebroj(char *s, int *broj_malih, int *broj_velikih)`, koja za zadatu nisku `s` računa broj malih i velikih slova koji se u njoj pojavljuju.
- c) Napisati glavni program koji sa standardnog ulaza čita rečenice i na standardni izlaz ispisuje onu kod koje je razlika broja malih i velikih slova najveća.

[Rešenje 3.91]

#### Zadatak 3.90

- a) Uvesti tip podataka `Sifra` kojim se opisuje način šifrovanja alfanumeričkih karaktera. Svaka šifra se opisuje celobrojnomo vrednoscu `b` koja određuje broj pozicija pomeranja, kao i karakterom `'L'` ili `'D'` koji određuje smer pomeranja (levo ili desno).
- b) Napisati funkciju `void sifruj(char rec[], Sifra s)` koja transformiše zadatu reč `rec` po šifri `s`. Reč se šifruje tako što se svako slovo zamenjuje slovom za `b` mesta levo ili desno od njega u abecedi, i to ciklično, a isto tako i za cifre.  
Npr: za `b=2`, i `smer='D'` : `a` se menja sa `c`, `b` sa `d`,..., `x` sa `z`, `y` sa `a`, `z` sa `b`, `1` sa `3`, .. `8` sa `0`, `9` sa `1`
- c) Sa standardnog ulaza se zadaje način šifrovanja i to u obliku `2 D 5 L` (šifra može biti i duža). Potom se učitava `n` i `n` reči sa standardnog ulaza (maksimalna dužina reči je 20 karaktera). Ispisati reči na standardni izlaz nakon primenjenih svih zadatah načina šifrovanja.

[Rešenje 3.91]

**Zadatak 3.91** Implementirati funkciju `int strspn(char* s, char* t)` koja izračunava dužinu početnog dela niske `s` sastavljenog isključivo od karaktera sadržanih u niski `t`.

Napisati i program koji sa standardnog ulaza učitava dve niske (dužine najviše 100 karaktera, svaku u zasebnom redu) i ispisuje rezultat poziva funkcije `strspn` na standardni izlaz.

Na primer, za učitane podatke `"734a.bf62"`, `"0123456789"`) program ispisuje vrednost 3.

## 3.6 Rešenja

### Rešenje 3.57

```
2  /*
4  Napisati funkciju koja konvertuje dati string tako sto
6  mala slova menja u velika a velika u mala. Napisati
8  potom glavni program koji ucitava string, poziva napisanu
10 funkciju i ispisuje konvertovani string. Mozemo pretpostaviti
12 da string ne sadrzi vise od 10 karaktera.
14 */
16 #include <stdio.h>
18 #include <ctype.h>
20
22 /*
24 Kada je niz argument funkcije, dodatni argument je obavezno
26 njegova dimenzija. Kod stringova to nije slucaj jer svaki string
28 ima isti poslednji element - terminirajucu nulu - i to je oznaka
30 kraja stringa.
32 */
34 void konvertuj(char s[])
36 {
38     int i;
40     for(i=0; s[i]!='\0'; i++)
42         if (s[i]>='a' && s[i]<='z')
44             s[i] = toupper(s[i]); /* toupper - konvertuje malo slovo u
46             odgovarajuce veliko */
48         else if (s[i]>='A' && s[i]<='Z')
50             s[i] = tolower(s[i]); /* tolower - konvertuje veliko slovo
52             u odgovarajuce malo */
54     /*
56     Funkcije toupper i tolower se nalaze u zaglavlju ctype.h.
58
59     Konverzija malog slova u veliko bez upotrebe funkcije toupper:
60     s[i] = s[i]-'a'+'A';
61
62     Konverzija velikog slova u malo bez upotrebe funkcije tolower:
63     s[i] = s[i]+'a'-'A';
64
65     */
66 }
68
70 int main()
72 {
74     /*
76     Poslednji karakter svakog stringa je terminirajuca
```

### 3 Predstavljanje podataka

---

```
42      nula '\0', specijalni karakter ciji je ASCII kod 0.

44      Ukoliko pretpostavljamo da string sadrzi najvise 30
      karaktera, neophodno je deklarirati niz od 31 karaktera,
46      pri cemu se dodatni izdvaja za terminirajucu nulu.

48      */
      char s[31];
50      printf("Unesi string:");

52      /*
      Za razliku od nizova koji se ucitavaju i stampaju
54      element po element, stringovi se mogu ucitati i
      odstampati pomocu jedne scanf/printf naredbe koriscenjem
56      specifikatora %s.

      Funkcija scanf ucitava string do prvog pojavljivanja razmaka.
58      */
      scanf("%s", s);

60      konvertuj(s);

62      printf("Konvertovani string: %s\n", s);

64      return 0;

66
68 }
```

#### Rešenje 3.58

```
/*
2  Napisati funkciju skрати koja uklanja beline sa
   kraja datog stringa.

4

   Napisati glavni program koji testira napisanu
6   funkciju na stringu "rep belina"

8   */

10  #include <stdio.h>
   #include <ctype.h>

12

14  /*
   Funkcija koja racuna duzinu niza
   ne racunajuci '\0'.

16

   U biblioteci string.h definisan je veliki
18   broj funkcija za rad sa stringovima,
   ukljucujuci i funkciju strlen koja racunana
20   duzinu stringa.
```



```
22     Funkcija strlen_klon predstavlja jednu
    implementaciju funkcije strlen.

24
    U zadacima cemo uvek koristiti ugradjenu
26     funkciju strlen osim ako u tekstu zadatka
    nije naglaseno da se ona ne sme koristiti.
28     Funkcija strlen_klon služi da pokaze na koji
    način radi ugradjena funkcija strlen.

30
    Ugradjena funkcija strlen poziva se na
32     isti način kao funkcija strlen_klon:
    strlen(s1)

34
    */
36     int strlen_klon(char s[])
    {
38         int i=0;
        while(s[i]) /* ASCII kod karaktera '\0' je jednak nuli. */
40             i++;

42         return i;
    }

44
    void skрати(char s[])
46     {
        /*
48         Poslednji karakter stringa s(ne racunajuci '\0') ima
        indeks strlen_klon(s)-1; ideja je da pocnemo od poslednjeg
50         karaktera stringa i da smanjujemo indeks dokle god
        je karakter na poziciji i blanko znak.

52
        */
54         int i;
        for(i=strlen_klon(s)-1;i>=0;i--)
56             if (!isspace (s[i])) /* Ako s[i] nije blanko znak, prekidamo
                petlju. */
                    break;

58         s[i+1]='\0'; /* Dodajemo terminirajucu nulu iza indeksa i (prvi
                neblanko karakter gledano sdesna nalevo).*/

60
        /*
62         Ugradjena funkcija isspace nalazi se u biblioteci ctype.h i
        vraca 1 ako
        je dati karakter blanko znak a 0 u suprotnom.

64
        Unarni logicki operator ! oznacava negaciju.

66
        */
68     }

70     int main()
```

### 3 Predstavljanje podataka

```
{
72
/*
74     Ukoliko string ne zelimo da učitavamo po pokretanju programa
       vec da ga unapred zadamo, to mozemo uraditi na sledeci nacin:
76 */
char s[]="rep belina";
78 /* U ovom slucaju nije neophodno navoditi dimenziju stringa vec
       ce ona biti automatski postavljena na broj karaktera u stringu +
       1 za
80     terminirajucu nulu. */

82
printf("Pre skracivanja: %s*\n", s);
84 skрати(s);
printf("Posle skracivanja: %s*\n", s);
86
return 0;
88 }
```

#### Rešenje 3.59

```
/*
2     Napisati program koji učitava string src i formira string dst
       trostrukim nadovezivanjem stringa src. Program treba da ispise
4     string dst. Na primer, za uneti string "dan", string dst treba
       da bude "dandandan". Pretpostaviti da string src nije duzi od
6     30 karaktera.
*/
8
#include <stdio.h>
10 #include <string.h>

12 #define MAX 30
/*
14     Na stringove ne mozemo primeniti naredbu dodele.
       Ukoliko zelimo da jedan string "dodelimo" drugom,
16     mozemo koristiti ugradjenu funkciju strcpy(s,t)
       koja kopira karaktere stringa t
18     u string s zajedno za terminirajucom nulom.

20     Funkcija strcpy se nalazi u biblioteci string.h.

22     Funkcija strcpy_klon predstavlja jednu
       implementaciju funkcije strcpy.

24     Karakteri stringa original se, jedan po jedan,
26     kopiraju u string kopija. Nakon kopiranja,
       na kraj stringa kopija dodaje se terminalna
28     nula.
```

```
30     U zadacima cemo uvek koristiti ugradjenu
31     funkciju strcpy osim ako u tekstu zadatka
32     nije naglaseno da se ona ne sme koristiti.
33     Funkcija strcpy_klon služi da pokaze na koji
34     način radi ugradjena funkcija strcpy.

35
36     Ugradjena funkcija strcpy poziva se na
37     isti način kao funkcija strcpy_klon:
38     strcpy(dst,src)
39     gde karaktere stringa src kopiramo
40     u string dst.

41
42
43     */
44     void strcpy_klon(char kopija[], char original[])
45     {
46         int i;
47         for(i=0; original[i]; i++)
48             kopija[i]=original[i];
49
50         kopija[i] = '\0';
51     }
52
53     int main()
54     {
55         char src[MAX+1];    /* src, skraceno od source (izvor, odnosno sta
56                             kopiramo) */
57         char dst[3*MAX+1];  /* dst, skraceno od destination (odrediste,
58                             odnosno gde kopiramo) */
59
60         /*
61          Vazno je izdvojiti dovoljno memorijskog prostora
62          za string dst: on treba da bude tri puta veci od
63          maksimalne duzine stringa src + jedan karakter za
64          terminirajucu nulu.
65          */
66
67         printf("Unesi jedan string:");
68         scanf("%s", src);
69
70         strcpy_klon(dst,src);
71
72         /*
73          Funkcija strcat(s,t) nadovezuje karaktere stringa
74          t na kraj stringa s i novi string terminira
75          karakterom '\0' .
76
77          Funkcija strcat se nalazi u biblioteci string.h.
78          */
79         strcat(dst,src);
80         strcat(dst,src);
```

### 3 Predstavljanje podataka

---

```
80  printf("Kada nadovezemo string %s triput: %s\n",src,dst);
82  return 0;
}
```

#### Rešenje 3.60

```
1  /*
2      Napisati funkciju int ucitaj_liniju(char s[], int n)
3      koja ucitava liniju maksimalne duzine n u string s
4      i vraca duzinu učitane linije. Linija moze da sadrzi
5      blanko znakove ali ne moze da sadrzi \n ili EOF.
6
7      Napisati potom glavni program koji ucitava linije
8      do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko
9      ima vise linija maksimalne duzine, ispisati prvu. Mozemo
10     pretpostviti da svaka linija sadrzi najvise 80 karaktera,
11     zajedno sa \n.
12
13 */
14
15 #include<stdio.h>
16 #include<string.h>
17 #define MAX 81
18
19 /*
20     Ukoliko zelimo da učitamo string koji sadrzi beline
21     (npr liniju teksta), ne mozemo koristiti funkciju
22     scanf jer ona ucitava string do prvog blanko znaka.
23
24     Zbog toga je neophodno napisati funkciju koja ucitava
25     string karakter po karakter.
26
27     Ova funkcija ne dopusta unosenje vise karaktera od
28     unapred odredjene granice (argument n).
29
30     U standardnoj biblioteci stdio.h postoji definisana
31     funkcija char *gets(char *s) koja ucitava karaktere
32     dok se ne pojavi novi red ili EOF. Ova funkcija
33     dopusta unosenje vise karaktera nego sto string
34     s sadrzi, sto moze dovesti do neocekivanog ponasanja
35     programa.
36
37     Pored funkcije gets, koja vrsi učitavanje sa standardnog
38     ulaza, u standardnoj biblioteci stdio.h postoji
39     i ugradjena funkcija fgets koja vrsi učitavanje iz
40     datoteke. Nju cemo koristiti za nekoliko casova
41     kada budemo radili datoteke. Prototim funkcije fgets je
42     ovakav:
43
44     char *fgets(char *s, int size, FILE *stream);
45 */
```

```
45     Argumenti funkcije fgets su:
46     s - string u koji vrsimo ucitavanje
47     size - maksimalna duzina unetog stringa
48     stream - datoteka iz koje vrsimo ucitavanje
49
50
51     Funkcija fgets, za razliku od funkcije gets, ne dopusta
52     unos vise karaktera od date vrednosti size. Zbog toga
53     je ona sigurnija nego funkcija gets. Funkciju fgets
54     mozemo koristiti i za unos sa standardnog ulaza
55     ukoliko kao treci argument navedemo stdin.
56
57 */
58 int ucitaj_liniju(char s[], int n)
59 {
60     int i=0;
61     int c;
62
63     while((c=getchar())!='\n' && i<n-2 && c!=EOF)
64     {
65         s[i] = c;
66         i++;
67     }
68
69     /* Ucitavamo najvise n-2 karaktera jer na kraju dodajemo jos
70        dva: '\n' i '\0' */
71
72     s[i]='\n';
73     s[i+1]='\0';
74
75     return i;
76 }
77
78
79 int main()
80 {
81     char linija[MAX];
82     char najduza_linija[MAX];
83     int max_duzina=0;
84     int duzina;
85
86     /*
87      Petlja se zavrsava ukoliko je promenljiva duzina
88      jednaka nuli, sto cemo postici zadavanjem linije koja ne sadrzi
89      nijedan karakter osim EOF.
90     */
91
92     while ((duzina=ucitaj_liniju(linija, MAX))>0)
93     {
94         /*
95          Proveravamo da li je uneta linija duza od trenutnog
96          maksimuma i azuriramo promenljive max_duzina i najduza_linija
```

### 3 Predstavljanje podataka

---

```
97     */
99     if (max_duzina < duzina)
100     {
101         max_duzina = duzina;
102         strcpy(najduza_linija, linija);
103     }
104
105     printf("Najduza linija: %s duzine: %d\n", najduza_linija,
106           max_duzina);
107
108     return 0;
109 }
```

#### Rešenje 3.61

```
/*
2   Napisati program koji pretvara nisku u ceo broj.
   Npr. za ulaz "-1238" se generise rezultat -1238
4   Pogledati funkcije atoi i atof koje postoje u biblioteci stdlib.h
   */
6
8   #include <stdio.h>
9   #include <ctype.h>
10  #define MAX 10
11  /*
12   String b se sastoji od karaktera koji
13   cine jedan ceo broj, onim redom kojim
14   se karakteri pojavljuju u zapisu broja.
15
16   Ako je prvi karakter stringa b '-',
17   to znaci da je broj negativan i
18   funkcija znak_broja vraca -1
19
20   U suprotnom, broj je pozitivan i
21   funkcija znak_broja vraca 1
22  */
23
24  int znak_broja(char b[])
25  {
26      if(b[0] == '-')
27          return -1;
28      return 1;
29  }
30
31
32  /*
33   Funkcija formiraj_broj na osnovu
34   karaktera koji cine broj iz stringa
35  */
```

```
36 b vraća ceo broj koji odgovara
    zapisu datom u stringu b.

38 Ako su cifre broja a,b,c i d, tada
    broj možemo kreirati kao:
40  $a \cdot 10^3 + b \cdot 10^2 + c \cdot 10^1 + d \cdot 10^0$ 

42 Međutim, efikasnije je koristiti
    Hornerovu sumu:

44  $10 \cdot (10 \cdot (10 \cdot (10 \cdot 0 + a) + b) + c) + d$ 

46
48 */
49 int formiraj_broj(char b[])
50 {
51     int i;
52     int n=0;
53     int znak = znak_broja(b);

54     /*
55      Ako je broj negativan, cifre u nizu b
56      počinju od indeksa 1
57     */

60     i=0;
61     if(znak==-1)
62         i=1;

64     /*
65      Funkcija isdigit proverava da li je broj
66      cifra. Nalazi se u biblioteci ctype.h

68      Proveravamo da li je karakter u zapisu
        broja cifra kako bismo se osigurali
        od nekorektnog unosa, npr ako korisnik
        unese -123abc. Ovaj unos je moguć jer
        se vrsi sa scanf("%s",broj), gde unosimo
        karaktere do prvog blanko znaka

74      Ako nađemo na karakter koji nije cifra,
        prekidamo petlju

76     */

78     for(; b[i]!='\0'; i++)
79     {
80         if(isdigit(b[i]))
81             n = n*10 + b[i] - '0';
82         else
83             break;
84     }

86     /* Formirani broj množimo znakom: */
```

### 3 Predstavljanje podataka

```

    n*=znak;
88     return n;
90 }
92 int main()
93 {
94     char broj[MAX];
95     int n;
96
97     /* Ucitavamo broj: */
98     scanf("%s", broj);
99
100    /* Ispisujemo rezultat: */
101    printf("Broj zapisan kao int: %d\n", formiraj_broj(broj));
102
103    return 0;
104 }
```

#### Rešenje 3.62

```

/*
2   Napisati program koji pretvara zadatu broj u nisku.
   Npr. za broj -453 treba generisati nisku "-453"
4  */
6  #include <stdio.h>
7  #include <string.h>
8  #define MAX 10
9  /*
10
11   Funkcija transformisi_negativan vraca
12   1 ako je broj negativan i 0 u suprotnom, a
13   uz to, ako broj jeste negativan, funkcija
14   treba da ga konvertuje u njegovu apsolutnu
15   vrednost. S obzirom da funkcija treba da vrati dve
16   vrednosti, to realizujemo na sledeci nacin:
17   1. indikator da li je broj negativan
18   ce vratiti kao povratnu vrednost
19   2. apsolutnu vrednost broja ce vratiti
20   preko liste argumenata, zbog cega broj
21   prenosimo preko pokazivaca
22
23  */
24  int transformisi_negativan(int* pn)
25  {
26      if(*pn<0)
27      {
28          *pn = -(*pn);
29          return 1;
30      }
31  }
```



```
32     return 0;
33 }
34 int formiraj_niz_cifara(int n, char b[], int neg)
35 {
36     int i=0;
37     char cifra;
38
39     do
40     {
41         cifra = n%10;
42
43         /* Promenljiva b predstavlja string.
44          Da bismo na neku poziciju u stringu
45          upisali karakter koji odgovara nekoj
46          cifri, npr '2', neophodno je da
47          odgovarajucoj poziciji dodelimo vrednost
48          ASCII koda te cifre, konkretno za '2'
49          ASCII kod je '0'+2.
50
51          Greska bi bila navesti b[i]=2
52          jer 2 nije ASCII kod koji odgovara karakteru
53          '2'.
54          */
55         b[i]=cifra+'0';
56
57         n/=10;
58         i++;
59     } while(n);
60
61     /* Ako je broj negativan, dodajemo znak minus: */
62     if(neg)
63     {
64         b[i]='-';
65         i++;
66     }
67
68     /* Svaki string se zavrшава terminirajucom nulom: */
69     b[i]='\0';
70 }
71
72 void obrni(char s[])
73 {
74     char t;
75     int i,j;
76     /*
77     Karaktere stringa obrnemo tako sto razmenimo karaktere na
78     pozicijama 0 i n-1,
79     zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
80     druge
81     */
82 }
```

### 3 Predstavljanje podataka

---

```
82     for(i=0,j=strlen(s)-1;i<j;i++, j--)
83     {
84         t = s[i];
85         s[i] = s[j];
86         s[j] = t;
87     }
88 }
89
90 void broj_u_niz_cifara(int n, char broj[])
91 {
92     int negativan;
93
94     /* Odredjujemo znak broja: */
95     negativan=transformisi_negativan(&n);
96
97     /* Izdvajamo cifre broja i smestamo ih u niz: */
98     formiraj_niz_cifara(n, broj, negativan);
99
100    /* S obzirom da cifre izdvajamo sa kraja broja, u nizu ce biti u
101       obrnutom redosledu.
102       Na primer, za broj 234 niz ce sadrzati cifre 4 3 2. */
103    obrni(broj);
104 }
105
106 int main()
107 {
108     int n;
109     char broj[MAX];
110     int negativan;
111
112     /* Ucitavamo broj: */
113     scanf("%d", &n);
114
115     /* Kreiramo broj na osnovu niza cifara: */
116     broj_u_niz_cifara(n,broj);
117
118     /* Ispisujemo rezultat: */
119     printf("Broj zapisan kao string: %s\n", broj);
120
121     return 0;
122 }
```

#### Rešenje 3.63

```
2  /*
   Napisati program koji ucitava dva stringa i ispituje najpre da li
   su jednaki. Ako jesu, program
   treba da izda odgovarajucu poruku, a ako nisu, treba da ispita da
   li je drugi podstring
```

```

4   prvog. Ukoliko jeste, program treba da ispise pocev od kog indeksa
    prvog
    stringa pocinje drugi string. U suprotnom, ispisati odgovarajucu
    poruku. Mozemo
6   pretpostaviti da stringovi ne sadrze vise od 20 karaktera.
    */
8
10  #include <stdio.h>
10  #include <string.h>
12
12  /*
    Funkcija strcmp(s,t) je ugradjena funkcija koja utvrđuje da
    li su stringovi
14   s i t jednaki. Ukoliko jesu, vraca 0, a u suprotnom vraca
    razliku
    ASCII kodova prva dva razlicita karaktera na istim pozicijama
16   (npr strcmp("aa","ab") ce vratiti -1 a strcmp("ab","aa") 1).
18
    Funkcija strcmp se nalazi u zaglavlju string.h.
20
    Funkcija strcmp_klon je jedna implementacija funkcije strcmp.
22
    U zadacima cemo uvek koristiti ugradjenu funkciju strcmp osim
    ako u tekstu zadatka
    nije naglaseno da se ona ne sme koristiti. Funkcija
    strcmp_klon služi da pokaze na koji
24   nacin radi ugradjena funkcija strcmp.
26
    Ugradjena funkcija strcmp poziva se na isti nacin kao funkcija
    strcmp_klon:
    strcmp(s1,s2)
28   gde poredimo stringove s1 i s2.
30  */
32  int strcmp_klon(char s1[], char s2[])
    {
34     int i;
    for(i=0; s1[i]==s2[i];i++)
36     if (s1[i]=='\0')
        return 0;
38     return s1[i] - s2[i];
40  }
42  int main()
    {
44     char s1[21];
    char s2[21];
46     char* p;
48     printf("Unesi dva stringa:");

```

### 3 Predstavljanje podataka

```
scanf("%s%s",s1,s2);

50
/*
52 Funkcija strstr(s,t) je ugradjena funkcija koja utvrđuje da
   li je string t
   podstring stringa t i ako jeste, vraca pokazivac (char*) na
   karakter
54 stringa s odakle pocinje prvo pojavljivanje stringa t, a NULL
   u suprotnom.

56 NULL je pokazivac koji ne pokazuje ni na sta, odnosno ne
   sadrzi adresu
   nijedne promenljive.

58 Podsetimo se veze nizova(a time i stringova) i pokazivaca:
   ako je string deklarisan sa s1[21], tada je njegov naziv s1
   ekvivalentan adresi prvog karaktera stringa:
60 s1 <=> &s1[0]
   i nadalje redom:
62 s1+1 <=> &s1[1]
64 ...
66 u opstem slucaju:
   s1+i <=> &s1[i]
68

70 To znaci da se indeks elementa na koji pokazuje s1+i moze
   dobiti tako sto od s1+i oduzmemo pokazivac na pocetak niza:
   s1+i-s1 <=> i. Ovako od pokazivaca na karakter u stringu
72 dobijamo njegov indeks u stringu.

74 */

76 p = strstr(s1,s2);

78 if (strcmp_klon(s1,s2)==0)
   printf("Uneti stringovi su jednaki\n");
80 else if (p!=NULL)
   printf("%s jeste podstring od %s pocev od pozicije : %d\n", s2,
   s1, p-s1);
82 else
   printf("%s NIJE podstring od %s\n", s2,s1);
84
86 return 0;
}
```

#### Rešenje 3.64

```
/*
2 Napisati program koji za uneti string s i karakter c utvrđuje
   da li se c pojavljuje u stringu s i ukoliko se pojavljuje,
4 ispisuje indeks prvog pojavljivanja a u suprotnom ispisuje
   odgovarajucu poruku. Mozemo pretpostaviti da string ima najvise
```

```
6      20 karaktera.
7      */
8
9      #include <stdio.h>
10     #include <string.h>
11
12     int main()
13     {
14         char s[21];
15         char c;
16         char* p;
17
18         printf("Unesi karakter:");
19         c=getchar();
20         printf("Unesi string:");
21         scanf("%s", s);
22
23         /*
24          Da smo ucitavali obrnutim redom (prvo string pa karakter)
25          to bismo realizovali na sledeci nacin:
26          printf("Unesi string:");
27          scanf("%s",s);
28          getchar();
29          printf("Unesi karakter:");
30          c=getchar();
31
32          Dodatni getchar() bi sluzio da "pokupi" karakter kojim
33          razdvajamo unos stringa i karaktera (razmak, novi red ili
34          slicno).
35
36          */
37
38         /*
39          Funkcija strchr(s,c) je ugradjena funkcija koja vraca pokazivac
40          na prvi karakter u stringu s koji je jednak karakteru c, ako
41          takav
42          postoji, a NULL u suprotnom.
43
44          Indeks od pokazivaca dobijamo na isti nacin kao u prethodnom
45          zadatku
46          sa strstr.
47          */
48
49         p = strchr(s,c);
50         if(p!=NULL)
51             printf("%c se pojavljuje u %s na poziciji %d\n", c, s, p-s);
52         else
53             printf("%c se NE pojavljuje u %s\n",c, s);
54
55         return 0;
56     }
```

#### Rešenje 3.65

```
1  /*
   a) Napisati funkciju int samoglasnik(char c) koja proverava da li je
       zadati karakter samoglasnik. Funkcija
3  treba da vrati vrednost 1 ako karakter c jeste samoglasnik, odnosno 0
   ako nije.
   b) Napisati funkciju int samoglasnik_na_kraju(char s[]) koja
       proverava da li se niska s završava samoglasnikom
5  (koristiti funkciju iz tacke a)).
   c) Napisati program koji ucitava nisku maksimalne duzine 20 karaktera
       i ispisuje da li završava samoglasnikom ili ne.
7  */

9  #include <stdio.h>
   #include <ctype.h>
11  #include <string.h>
   #define MAX_DUZINA 20

13  /* Funkcija proverava da li je karakter c samoglasnik */
15  int samoglasnik(char c){

17     char C;

19     /* Konvertujemo karakter u veliko slovo kako bismo smanjili broj
       provera */
       C=toupper(c);

21     /* Samoglasnici su slova a, e, i, o i u */
23     if(C=='A' || C=='E' || C=='I' || C=='O' || C=='U')
           return 1;

25     return 0;

27 }

29 /* Funkcija koja proverava da li se niska s završava samoglasnikom */
   int samoglasnik_na_kraju(char s[]){
31     int duzina;

33     /* Odredjujemo duzinu niske */
       duzina=strlen(s);

35     /* Proveravamo da li je niska prazna */
37     if(duzina==0)
           return 0;

39     /* Ako niska nije prazna, proveravamo da li se samoglasnik nalazi
       na kraju */
41     /* Numeracija karaktera u niski pocinje nulom pa zato proveravamo
       poziciju duzina -1 */
       return samoglasnik(s[duzina-1]);

43 }
```

```

45 int main(){
47     char s[MAX_DUZINA+1];

49     /* Ucitavamo nisku */
    printf("Unesite nisku: ");
51     scanf("%s", s);

53     /* Proveravamo da li se završava samoglasnikom i ispisujemo
        odgovarajuću poruku */
    if(samoglasnik_na_kraju(s))
55         printf("Niska se završava samoglasnikom!\n");
    else
57         printf("Niska se ne završava samoglasnikom!\n");

59     return 0;
}

```

### Rešenje 3.66

```

/*
2  Napisati funkciju void kopiraj_n(char t[], char s[], int n) koja
    kopira najviše n karaktera niske s u nisku t.
    Napisati i program koji učitava nisku maksimalne dužine 20 karaktera
    i jedan ceo broj i testira rad napisane funkcije.
4  */

6  #include <stdio.h>
    #define MAX_DUZINA 20

8
    void kopiraj_n(char t[], char s[], int n){
10     int i;

12     /* Brojac i oznacava tekucu poziciju u niski */
    /* Uslov i<n je neophodan zbog kopiranja najviše n karaktera */
14     /* Uslov s[i]!='\0' (ili skraceno samo s[i]) je neophodan kako bi
        bili sigurni da na poziciji i postoji karakter u niski s */
    for(i=0; i<n && s[i]!='\0'; i++){
16         t[i]=s[i];
    }

18     /* Upisujemo terminirajucu nulu u novodobijenu nisku */
20     t[i]='\0';
}

22

24 int main(){
    int n;
26     char s[MAX_DUZINA+1], t[MAX_DUZINA+1];

```

### 3 Predstavljanje podataka

---

```
28  /* Ucitavamo nisku */
    printf("Unesite nisku: ");
30  scanf("%s", s);

32  /* Ucitavamo broj n i proveravamo korektnost unosa */
    printf("Unesite broj n: ");
34  scanf("%d", &n);
    if(n<0 || n>MAX){
36      printf("Greska: pogresan unos!\n");
        return 0;
38  }

40  /* Pozivamo funkciju za kopiranje */
    kopiraj_n(t, s, n);

42
44  /* Ispisujemo dobijenu nisku */
    printf("Rezultujuca niska: %s\n", t);

46  return 0;
}
```

#### Rešenje 3.67

```
1  /*
   Napisati funkciju void dupliranje(char t[], char s[]) koja na osnovu
   niske s formira nisku t tako sto duplira svaki
3  karakter niske s. Napisati i program koji ucitava nisku maksimalne
   duzine 20 karaktera i testira rad napisane funkcije.
   */

5
7  #include <stdio.h>
   #define MAX_DUZINA 20

9  void dupliranje(char t[], char s[]){
    int i, j;

11
13     /* Brojac i oznacava tekucu poziciju u niski s */
    /* Brojac j oznacava tekucu poziciju u niski t */
    for(i=0, j=0; s[i]!='\0'; i++, j+=2){
15         t[j]=s[i];
        t[j+1]=s[i];
17     }

19     /* Upisujemo terminirajucu nulu u novodobijenu nisku */
    t[j]='\0';
21 }

23
25 int main(){
    int n;
    char s[MAX_DUZINA+1], t[2*MAX_DUZINA+1];
```



```

27  /* Ucitavamo nisku */
29  printf("Unesite nisku: ");
   scanf("%s", s);

31

33  /* Pozivamo funkciju za dupliranje */
   dupliranje(t, s);

35  /* Ispisujemo dobijenu nisku */
   printf("%s\n", t);

37

   return 0;
39 }

```

### Rešenje 3.68

```

1  /*
   Napisati funkciju int heksa_broj(char s[]) koja proverava da li je
   niskom s zadat korektan heksadekadni broj.
3  Heksadekadni broj je korektno zadat ako pocinje prefiksom 0x ili 0X
   i ako sadrzi samo cifre i mala ili velika slova A, B, C, D, E i F
   .
   Funkcija treba da vrati vrednost 1 ako je niska korektan
   heksadekadni broj, odnosno 0 ako nije.
5  Napisati i program koji ucitava nisku maksimalne duzine 7 karaktera
   i ispisuje rezultat rada funkcije.
   */

7

   #include<stdio.h>
9   #define MAX 8

11

13  /*
   Funkcija koja proverava da li je prosledjeni karakter ispravna
   heksadekadna cifra (broj ili slovo a,b,c,d,e,f)
   Ukoliko jeste, funkcija vraca 1, u suprotnom 0.
15  */
   int heksa_cifra(char c){
17     /*Pretvaramo karakter c u veliko slovo*/
       c = toupper(c);

19     /*Proveravamo da li je u pitanju cifra ili slovo A,B,C,D,E,F i
       ukoliko jeste, vracamo 1*/
21     if(isdigit(c) || (c >= 'A' && c <= 'F'))
       return 1;

23     /*Ukoliko nije, vracamo 0*/
25     return 0;
   }
27

```

### 3 Predstavljanje podataka

```
/*Funkcija koja proverava da li prosledjena niska s predstavlja
   ispravan heksadekadni broj */
29 int heksa_broj(char s[]){
   int i;

31
   /*Kako heksadekadni brojevi pocinju sa 0x ili 0X, prvo proveravamo
      da li je taj uslov ispunjen,
33 tj. da li je s[0] jednak 0 i da li je s[1] jednak X i ako taj uslov
      nije ispunjen, onda
   niska s ne predstavlja korektan heksadekadni broj */
35 if(s[0] != '0' || toupper(s[1]) != 'X')
   return 0;

37
   /*Prolazimo kroz nisku, pocev od pozicije 2 (jer su prve dve
      pozicije 0x) i za svaki karakter do kraja
39 niske proveravamo da li je ispravna heksadekadna cifra.
   Ako naidjemo na bilo koji koji ne ispunjava taj uslov, onda niska
      s nije korektan heksadekadni broj
41 i vracamo 0. */
   for(i=2; s[i] != '\0'; i++)
43 if(!heksa_cifra(s[i]))
   return 0;

45
   /*Ako smo stigli do kraja, znaci da su svi karakteri date niske
      ispravne heksadekadne cifre
47 i zato vracamo 1 */
   return 1;
49 }

51 int main(){
   char s[MAX];

53
   /*Ucitavamo nisku*/
55 printf("Unesite nisku:");
   scanf("%s", s);

57
   /*Pozivamo funkciju i stampamo odgovarajucu poruku*/
59 if(heksa_broj(s))
   printf("Korektan heksadekadni broj!\n");
61 else
   printf("Nekorektan heksadekadni broj!\n");

63
   return 0;
65 }
```

#### Rešenje 3.69

```
1 /* Napisati funkciju int heksa_broj(char s[]) koja izracunava dekadnu
   vrednost heksadekadnog broja zadatog niskom s.
   * Napisati i program koji ucitava nisku maksimalne duzine 7
   karaktera i ispisuje rezultat rada funkcije.
```

```

3  * Pretpostaviti da je uneta niska korektan heksadekadni broj. */
5  #include<stdio.h>
6  #include<string.h>
7
8  #define MAX 8
9
10 /*Funkcija koja racuna dekadnu vrednost heksadekadnog broja*/
11 int heksa_broj(char s[]){
12     int i,k;
13     char c;
14
15     /*Racunamo duzinu niske koja predstavlja heksadekadni broj*/
16     int n = strlen(s);
17
18     /*Inicijalizujemo vrednost v na 0*/
19     int v = 0;
20
21     /*Prolazimo petljom kroz nisku, krenuvsi sa desne strane
22     npr: 1a8e = e*1 + 8*16 + a*256 + 1*4096
23     Promenljiva k ce nam biti mnozilac i ona uzima vrednosti 1, 16,
24         256, 4096, ...
25     Promenljiva c ce nam cuvati trenutnu heksadekadnu cifru (u nasem
26         primeru e, 8, a, 1)
27     U svakom koraku treba na ispravan nacin da pomnozimo c i k
28     */
29     for(i=n-1, k=1; i>=0; i--, k*=16)
30     {
31         /*U c smestamo trenutnu heksadekadnu cifru.
32         Pozivamo funkciju toupper da bi obezbedili da radimo samo sa
33         velikim slovima.
34         Ako je s[i] cifra, funkcija toupper je nece promeniti.
35         */
36         c = toupper(s[i]);
37
38         if(isdigit(c)){
39             /*Ako je c cifra, onda samu vrednost te cifre dobijamo sa c-'0'
40             NAPOMENA: Nije ispravno napisati c*k jer je c karakter!
41             */
42             v += (c-'0')*k;
43         }else{
44             /*Ako je c slovo izmedju A i F, mi treba da dobijemo odgovarajucu
45             vrednost izmedju 10 i 15.
46             Ova vrednost se dobija sa 10 + c - 'A'. npr. za A ce biti 10 + '
47             A' - 'A' = 10, za B: 10 + 'B' - 'A' = 11, ...*/
48             v += (c - 'A' + 10)*k;
49         }
50     }
51
52     /*Na kraju vracamo izracunatu vrednost */
53     return v;
54 }

```

### 3 Predstavljanje podataka

---

```
51 int main(){
    char s[MAX];

53     /*Ucitavamo nisku*/
55     printf("Unesite nisku:");
    scanf("%s", s);

57     /*Ispisujemo rezultat*/
59     printf("%d\n", hekza_broj(s));

61     return 0;
}
```

#### Rešenje 3.70

```
/*
2  Napisati funkciju int podniska(char s[], char t[]) koja proverava da
    li je niska t podniska niske s.
    Napisati i program koji ucitava dve niske maksimalne duzine 10
    karaktera i testira rad napisane funkcije.

4
    Napomena: u biblioteci string.h postoji funkcija strstr
6  char* strstr(const char* s, const char* t)
    koja vraca adresu pocetka prvog pojavljivanja niske t u niski s ili
    NULL ako se niska t ne javlja
8  u niski s.
    */

10 #include<stdio.h>
12 #define MAX 11

14 /*Funkcija koja proverava da li je t podniska od s*/
int podniska(char s[], char t[]){
16     int i, j, k;

18     /*Spoljna petlja ide redom po niski s*/
    for(i=0; s[i] != '\0'; i++){

20
        /*Unutrasnja petlja ide redom po niski t*/
22         /*Promenljiva k pamti vrednost i, služi za poredjenje s[k] i t[j] i
            *zatim se vrši pomeranje i po niski s i po niski t (k++, j++) */
24         /*Cim naidjemo na situaciju da se karakteri ne poklapaju, izlazimo
            iz unutrasnje petlje*/
        for(j=0, k = i; t[j] != '\0'; j++, k++){
26             if(s[k] != t[j])
                break;

28         }

30         /*Ako smo prosli celu unutrasnju petlju (do kraja niske t), to
            znaci da su se svi karakteri iz t poklopili
```

```
32     sa karakterima iz s i onda vracamo 1*/
33     if(t[j] == '\0')
34         return 1;
35 }
36
37 /*Na kraju vracamo 0*/
38 return 0;
39 }
40
41 int main(){
42     char s[MAX], t[MAX];
43
44     /*Ucitavamo prvu nisku*/
45     printf("Unesite nisku s: ");
46     scanf("%s", s);
47
48     /*Ucitavamo drugu nisku*/
49     printf("Unesite nisku t: ");
50     scanf("%s", t);
51
52     /*Pozivamo funkciju i ispisujemo odgovarajucu poruku*/
53     if(podniska(s,t))
54         printf("t je podniska niske s!\n");
55     else
56         printf("t nije podniska niske s!\n");
57
58     return 0;
59 }
```

Rešenje 3.71

Rešenje 3.72

Rešenje 3.73

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

## 3.7 Višedimenzioni nizovi

### Zadatak 3.92

a) Napisati funkciju

```
int reflektivna(int a[][MAX], int n)
```

kojom se za relaciju zadatom matricom *a* (matruca je kvadratna) ispitije da li je reflektivna.

b) Napisati funkciju

```
int simetricna(int a[][MAX], int n)
```

kojom se za relaciju zadatom matricom *a* (matruca je kvadratna) ispitije da li je simetricna.

c) Napisati funkciju

```
int tranzitivna(int a[][MAX], int n)
```

kojom se za relaciju zadatom matricom *a* (matruca je kvadratna) ispitije da li je tranzitivna.

Dva elementa *i* i *j* ( $i @ j$ ) su u relaciji akko  $a[i][j] = 1$

Relacija je refleksivana ako sa svako *i* važi:  $i @ i = 1$

Relacija je simetricna ako za svako *i* i *j* važi:  $i @ j = 1 \Rightarrow j @ i = 1$

Relacija je tranzitivna ako za svako *i*, *j* i *k* važi:  $i @ j = 1$  i  $j @ k = 1 \Rightarrow i @ k = 1$

Funkcija postavlja na 1 odgovarajuci indikator.

b) Sa standardnog ulaza prvo se unose dimenzija kvadratne matrice *n*, a nakon toga elementi matrice. Učitati matricu, i ispitati da li je relacija koju predstavlja relacija ekvivalencije (refleksivna, simetrična i tranzitivna).

[Rešenje 3.108]

**Zadatak 3.93** Napisati funkciju `float sumD(float a[][max], int n)` koja odreuje sumu elememenata iznad glavne dijagonale. Potom napisati funkciju `float sumd(float a[][max], int n)` koja odreuje sumu elememenata ispod glavne dijagonale. Funkciju testirati pozivom u main-u. Matrica je maksimalne dimenzije 50x50. Matrica je kvadratna.

[Rešenje 3.108]

**Zadatak 3.94** Napisati funkciju

`void transponovana(float a[][max], int m, int n, float b[][max])` koja odreuje transponovanu matricu matricu. Pozivom u main-u testirati funkciju. Matrica je maksimalne dimenzije 50x50.

[Rešenje 3.108]

**Zadatak 3.95** Napisati funkciju

```
void mnozenje(int a[][max], int n, int m, int b[][max], int k,
int t, int c[][max])
```

koja računa proizvod dve matrice. Pozivom u main-u testirati funkciju. Matrica je maksimalne dimenzije 50x50. Testirati da li su podaci korektno uneti i testirati da li je moguće matrice množiti.

[Rešenje 3.108]

**Zadatak 3.96** Napisati funkciju u kojoj se razmenjuju elementi  $k$ -te i  $t$ -te vrste matrice ( $k$  i  $t$  su argumenti funkcije). Funkciju testirati pozivom u main-u i ispisom novodobijene matrice na standardni izlaz. Sa standardnog ulaza učitavaju se dimenzije matrice, a potom i elementi matrice i brojevi  $k$  i  $t$ . Maksimalna dimenzija matrice je  $50 \times 50$ . Funkciju testirati pozivom u main-u.

[Rešenje 3.108]

**Zadatak 3.97** Sa standardnog ulaza unose se celi pozitivni brojevi  $m$  i  $n$  koji označavaju broj vrsta i broj kolona matrice. Potom se unose elementi matrice. Nakon unosa elemenata matrice, unose se još dva broja  $p$  i  $k$  ( $p \leq m$ ,  $k \leq n$ ). Na standardni izlaz ispisati sume svih podmatrica (dimenzije  $p \times k$ ) unete matrice. U slučaju greške ispisati  $-1$ .

**Napomena 1:** Ne razmatrati slučaj negativnih brojeva.

**Napomena 2:** Nije bitan redosled kojim se ispisuju sume.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
3 4
1 2 3 4
5 6 7 8
9 10 11 12
3 3
54 63
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
3 4
1 2 3 4
5 6 7 8
9 10 11 12
2 3
24 30 48 54
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
3 2
1 2
3 4
5 6
7 8
-1
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
5 3
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
2 2
7 5 20 19 18 23 4 8
```

[Rešenje 3.108]

**Zadatak 3.98** Sa standardnog ulaza zadata je dimenzija kvadratne matrice  $n$  ( $0 < n \leq 50$ ), a zatim i vrednosti pojedinačnih elemenata. Ukoliko je  $n$  izvan ovog opsega ispisati  $-1$  i prekinuti izvršavanje programa. Napisati program koji:

- (a) Učitava matricu i ispisuje je na izlaz. U slučaju greške ispisati  $-1$  i prekinuti izvršavanje programa.



- (b) Ispituje da li su elementi matrice po kolonama, vrstama i dijagonalama (glavnoj i sporednoj) sortirani strogo rastuće. Za svaki od ovih slučajeva redom ispisati 1 ako jesu i 0 ako nisu sortirani - videti primere.

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
3
1 2 3
4 5 6
7 8 9
1 2 3
4 5 6
7 8 9
1 1 1
```

*Primer 2*

INTERAKCIJA SA PROGRAMOM:

```
2
6 9
4 10
6 9
4 10
0 1 0
```

*Primer 3*

INTERAKCIJA SA PROGRAMOM:

```
4
5 5 7 9
6 10 11 13
8 12 14 15
13 15 16 20
5 5 7 9
6 10 11 13
8 12 14 15
13 15 16 20
1 0 1
```

*Primer 4*

INTERAKCIJA SA PROGRAMOM:

```
1
5
5
1 1 1
```

[Rešenje 3.108]

**Zadatak 3.99** Sa standardnog ulaza se unosi broj  $n$  ( $0 < n \leq 10$ ), a potom i elementi kvadratne matrice dimenzije  $n \times n$ . Elementi matrice su celi brojevi. Proveriti da li važi da su zbirovi elemenata kolona matrice uredjeni u strogo rastućem poretku. **Napomena 1:** Ukoliko program uvek ispisuje **da** ili uvek ispisuje **ne** smatraće se netačnim i poeni se ne mogu osvojiti.

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
4
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
ne
```

*Primer 2*

INTERAKCIJA SA PROGRAMOM:

```
3
1 2 3
4 5 6
7 8 9
da
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
3
2 -2 1
1 2 2
2 1 -2
ne
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
5
-1 0 2 0 20
0 0 0 10 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
da
```

[Rešenje 3.108]

**Zadatak 3.100** Sa standardnog ulaza unosi se broj  $n$  ( $0 < n \leq 200$ ), a potom i elementi kvadratne matrice dimenzije  $n \times n$ . Elementi matrice su celi brojevi. Proveriti da li je uneta matrica ortonormirana i na standardni izlaz ispisati **da** ako jeste ili **ne** ako nije ortonormirana. Matrica je ortonormirana ako je skalarni proizvod svakog para različitih vrsta jednak 0, a skalarni proizvod vrste sa samom sobom 1. U slučaju greške ispisati -1.

**Napomena 1:** Skalarni proizvod vektora  $a = (a_1, a_2, \dots, a_n)$  i  $b = (b_1, b_2, \dots, b_n)$  je  $a_1 \cdot b_1 + a_2 \cdot b_2 + \dots + a_n \cdot b_n$ .

**Napomena 2:** Ukoliko program uvek ispisuje **da** ili uvek ispisuje **ne** smatraće se netačnim i poeni se ne mogu osvojiti.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
4
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
da
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
3
1 2 3
4 5 6
7 8 9
ne
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
3
2 -2 1
1 2 2
2 1 -2
ne
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
5
-1 0 2 0 20
0 0 0 10 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
da
```

[Rešenje 3.108]

**Zadatak 3.101** Napisati funkciju koja kao argumente prima kvadratnu matricu celih brojeva i njenu dimenziju, a vraća 1 ako je matrica donja trougaona, odnosno 0 ako nije. Pretpostavka je da je maksimalna dimenzija matrice 100.

Matrica je donja trougaona ako se u gornjem trouglu (iznad glavne dijagonale, ne uključujući je) nalaze sve nule.

[Rešenje 3.108]

**Zadatak 3.102** Napisati program koji sa standardnog ulaza unosi prvo dimenziju matrice ( $n < 10$ ) pa zatim elemente matrice i izračunava sumu elemenata iznad sporedne dijagonale matrice.

[Rešenje 3.108]

**Zadatak 3.103** Za datu kvadratnu matricu kažemo da je *magični kvadrat* ako je suma elemenata u svakoj koloni i svakoj vrsti jednaka. Napisati program koji sa standardnog ulaza učitava prirodni broj  $n$  ( $n < 10$ ) i zatim elemente kvadratne matrice, proverava da li je ona *magični kvadrat* i ispisuje odgovarajuću poruku na standardni izlaz.

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
4
1 5 3 1
2 1 2 5
3 2 2 3
4 2 3 1
da
```

[Rešenje 3.108]

**Zadatak 3.104** Napisati program koji sa standardnog ulaza učitava prvo dimenzije matrice ( $n$  i  $m$ ) a zatim redom i elemente matrice (ne postoje pretpostavke o dimenziji matrice). Nakon toga na standardni izlaz, zapisati indekse ( $i$  i  $j$ ) onih elemenata matrice koji su jednaki zbiru svih svojih susednih elemenata (pod susednim elementima podrazumevamo okolnih 8 polja matrice ako postoje).

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
4 5
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
1 1
1 3
3 2
3 4
```

[Rešenje 3.108]

### 3 Predstavljanje podataka

---

**Zadatak 3.105** Sa standardnog ulaza se zadaje prvo dimenziju kvadratne matrice  $n$  ( $n < 100$ ), a zatim elemente matrice. Nakon toga, na standardni izlaz ispisati redni broj kolone koja ima najveći zbir elemenata.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
```

```
3
1 2 3
7 3 4
5 3 1
0
```

[Rešenje 3.108]

**Zadatak 3.106** Napisati funkciju koja treba da ispiše elemente matrice u grupama koje su paralelne sa sporednom dijagonalom matrice. Može se pretpostaviti da matrica nije dimenzije veće od  $100 \times 100$ .

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
```

```
3
1 2 3
4 5 6
7 8 9
1
2 4
3 5 7
6 8
9
```

[Rešenje 3.108]

**Zadatak 3.107** Sa standardnog ulaza učitava se broj  $n$ , a zatim i kvadratna matrica koja sadrži brojeve tipa `double` dimenzije  $n \times n$ . Napisati program koji izračunava i ispisuje razliku (na dve decimale) između zbira elemenata gornjeg trougla i zbira elemenata donjeg trougla matrice – gornji trougao čine svi elementi iznad sporedne dijagonale (ne računajući dijagonalu), a donji trougao čine svi elementi ispod sporedne dijagonale (računajući dijagonalu). U slučaju greške u datoteku upisati GRESKA.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
```

```
3
2 3.2 4
7 8.8 1
2.3 1 1
-2.10
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
```

```
4
2.3 1 12 8
4 -8.2 7 14.5
1 -2.5 9 11
3 4.3 -5.7 2
49.4
```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| -4
|| GRESKA

```

[Rešenje 3.108]

**Zadatak 3.108** Kao argumenti komandne linije zadate su dimenzije matrice  $A$  ( $m$  i  $n$ ). Element matrice se naziva sedlo ako je istovremeno najmanji u svojoj vrsti, a najveći u svojoj koloni. Ispisati indekse i vrednosti onih elemenata matrice koji su sedlo. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . Ukoliko nisu zadati svi potrebni argumenti komandne linije ispisati poruku da je došlo do greške. Ukoliko su dimenzije van opsega ispisati poruku o grešci.

*Primer 1*

```

|| POKRETANJE: ./a.out 2 3
|| INTERAKCIJA SA PROGRAMOM:
|| 1 2 3
|| 0 5 6
|| 0 0 1

```

*Primer 2*

```

|| POKRETANJE: ./a.out 3 3
|| INTERAKCIJA SA PROGRAMOM:
|| 10 3 20
|| 15 5 100
|| 30 -1 200
|| 1 1 5

```

*Primer 3*

```

|| POKRETANJE: ./a.out 3
|| INTERAKCIJA SA PROGRAMOM:
|| greska

```

*Primer 4*

```

|| POKRETANJE: ./a.out 200 3
|| INTERAKCIJA SA PROGRAMOM:
|| greska

```

[Rešenje 3.108]

## 3.8 Rešenja

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

## 3.9 Struktura

Zadatak [3.109](#)    Tekst

[Rešenje [3.109](#)]

Zadatak [3.110](#)    Tekst

[Rešenje [3.110](#)]

Zadatak [3.111](#)    Tekst

[Rešenje [3.111](#)]

**Zadatak 3.112** Tekst

[Rešenje 3.112]

**Zadatak 3.113** Tekst

[Rešenje 3.113]

**Zadatak 3.114** Definisati strukturu kojom se predstavlja kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja, a zati i program koji učitava dva kompleksna broja i ispisuje vrednost zbira, razlike, proizvoda i količnika.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite realni i imaginarni deo prvog broja:  1 2
Unesite realni i imaginarni deo drugog broja: -2 3
Zbir: -1.00+5.00*i
Razlika: 3.00-1.00*i
Proizvod: -8.00-1.00*i
Kolicnik: 0.31-0.54*i
```

[Rešenje 3.114]

**Zadatak 3.115** Definisati strukturu *Lopta* sa poljima *poluprecnik* (ceo broj u centimetrima) i *boja* (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Zatim učitati informacije o  $n$  lopti ( $0 < n < 50$ ) i ispisati ukupnu zapreminu, kao i broj crvenih lopti. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 4
Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta, 3-crvena, 4-zelena):
1.lopta: 4 1
2.lopta: 1 3
3.lopta: 2 3
4.lopta: 10 4
Ukupna zapremina: 4494.57
Broj crvenih lopti: 2
```

[Rešenje 3.115]

**Zadatak 3.116** Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vočka* sadrži ime vočke (nisku maksimalne dužine 20 karaktera) i količinu vitamina C u miligramima (realan broj). Napisati program koji sa standardnog ulaza učitava podatke o vočkama sve do unosa reči KRAJ i ispisuje ime vočke sa

### 3 Predstavljanje podataka

---

najviše vitamina C. Pretpostaviti da broj voćki neće biti veći od 50. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime voćke i njenu količinu vitamina C: jabuka 4.6
Unesite ime voćke i njenu količinu vitamina C: limun 51
Unesite ime voćke i njenu količinu vitamina C: kivi 92.7
Unesite ime voćke i njenu količinu vitamina C: banana 8.7
Unesite ime voćke i njenu količinu vitamina C: pomorandža 53.2
Unesite ime voćke i njenu količinu vitamina C: KRAJ
Voće sa najviše C vitamina je: kivi
```

[Rešenje 3.116]

**Zadatak 3.117** Deda Mraz planira kupovinu poklona za studente koji su vredno učili C u toku godine. Na njegovoj listi se nalazi ime i prezime studenta (niske dužina do 50 karaktera) i njegova želja (niska maksimalne dužine 100 karaktera). Napisati program koji će služiti Deda Mrazu kao podsetnik: na osnovu liste koju je napravio, Deda Mraz može da unese ime i prezime studenta i da proveri njegovu želju. Ako ima više studenata sa istim imenom i prezimenom ispisati sve želje. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Ime i prezime studenta:
Pera Peric
Njegova zelja:
privezak za kljuceve
Jos vrednih studenata (da/ne)?
da
Ime i prezime studenta:
Zika Zikic
Njegova zelja:
stap za pecanje
Jos vrednih studenata (da/ne)?
da
Ime i prezime studenta:
Mara Maric
Njegova zelja:
komplet Knutovih knjiga
Jos vrednih studenata (da/ne)?
ne
Za podsecanje uneti ime i prezime:
Pera Peric
Novogodisnja zelja: privezak za kljuceve
```

[Rešenje 3.117]



**Zadatak 3.118** Definirati strukturu *Grad* u kojoj se nalazi ime grada (niska dužine 20 karaktera) i prosečna temperatura u toku decembra (realan broj). Napisati program koji učitava imena  $n$  ( $0 < n < 50$ ) gradova i njihove prosečne temperature, a zatim ispisuje one gradove koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Unesite grad i temperaturu: Beograd 7
Unesite grad i temperaturu: Uzice 1.5
Unesite grad i temperaturu: Subotica 4
Unesite grad i temperaturu: Zrenjanin 9
Gradovi sa idealnom temperaturom za klizanje u decembru:
Beograd
Subotica
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 2
Unesite grad i temperaturu: Varsava 11
Unesite grad i temperaturu: Prag 2
Gradovi sa idealnom temperaturom za klizanje u decembru:
```

[Rešenje 3.118]

**Zadatak 3.119** Definirati strukturu *ParReci* koja sadrži reč na srpskom jeziku i odgovarajući prevod na engleski jezik. Zatim sa standardnog ulaza sve do kraja ulaza učitavati parove reči i, posebno, za rečenicu koja se zadaje sa ulaza ispisati prevod - ako je reč u rečenici nepoznata umesto nje ispisati odgovarajući broj zvezdica. Reči neće biti duže od 50 karaktera, ukupan broj parova reči neće biti veći od 100, a ukupna dužina rečenice neće biti veća od 100 karaktera. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
zima winter
godina year
sreca happiness
programiranje programming
caj tea
Unesite recenicu za prevod:
piti caj zimi je sreca
**** tea **** ** happiness
```

[Rešenje 3.119]

### 3 Predstavljanje podataka

---

**Zadatak 3.120** Napisati funkcije koje izračunavaju zbir, razliku i proizvod dva razlomka, `razlomak zbir(razlomak a, razlomak b)` itd. Unosi se broj  $n$  a potom i  $n$  razlomaka sa standardnog ulaza (najviše 100). Ispisati njihov zbir, razliku i proizvod na standardni izlaz.

[Rešenje 3.125]

**Zadatak 3.121** Napraviti strukturu `VOCE` koja sadrži ime (ne duže od 20 karaktera) i cenu (tipa `float`). Sa standardnog ulaza unosi se broj voćki (ne vići od 200), a potom uneti niz voća i pozvati funkciju koja izračunava prosečnu cenu voća. Potom ispisati imena onih voćki čija je cena veća od prosečne.

[Rešenje 3.125]

**Zadatak 3.122** Sa standardnog ulaza učitava se  $n$  ( $0 < n \leq 200$ ), a potom i spisak (dužine  $n$ ) engleskih reči i njihov prevod na srpski jezik. Potom se učitava jedna reč sa standardnog ulaza. Na standardni izlaz ispisati odgovarajući prevod date reči ili podatak o tome da se reč ne nalazi na spisku.

```
apple jabuka
pineapple ananas
orange narandza
pear kruska
grape grozdje
```

i reč *orange* program treba da ispiše *narandza* a za reč *cherry* program treba da ispiše poruku *Rec se ne nalazi u recniku*. U programu se mogu koristiti funkcije iz zaglavlja *string.h*.

[Rešenje 3.125]

**Zadatak 3.123** Napisati program koji sa standardnog ulaza čitava najpre broj artikala (ceo broj manji od 20) a zatim podatke o artiklima. Artikli su voćke koje imaju po dva podatka: naziv voćke i cenu (naziv voćke je karakterska niska dužine do 20 karaktera). Program potom traži od korisnika da unese neku cenu i štampa na standardni izlaz sve voćke koje imaju zadatu cenu.

Primer rada programa:

```
4
jabuka 30
kruska 40
ananas 60
limun 40
```

Unesite cenu: 40

Voce te cene je: kruska limun

[Rešenje 3.125]

**Zadatak 3.124** Definisati strukturu koja opisuje dete atributima ime deteta (ne vece od 20 karaktera) , pol deteta (m ili z) i ocena. Ocenu je svako dete dalo radu obdaništa. Maksimalan broj dece je 100. Napisati program koji:

- a) Sa standardnog ulaza se unosi  $n$ , a potom podaci o  $n$  dece. Koristiti strukturu:

```
typedef struct
{
    char ime[20];
    char pol;
    int ocena;
} DETE;
```

- b) ispisati na standardni izlaz statistiku: koliko ima dečaka, a koliko devojčica i prosečnu ocenu. Potom ispisuje imena dece brojnijeg pola.

[Rešenje 3.125]

### Zadatak 3.125

- Definisati tip podataka **TACKA** pogodan za predstavljanje tačke Dekartovske ravni (čije su  $x$  i  $y$  koordinate podaci tipa **double**).
- Definisati funkciju **double rastojanje(TACKA a, TACKA b)** koja izračunava rastojanje između dve tačke.
- Definisati funkciju **unsigned ucitaj\_poligon(TACKA\* tacke, unsigned n)** koja učitava  $n$  puta po dve vrednosti tipa **double** (koje predstavljaju koordinate temena poligona) i upisuje ih u zadati niz tačaka. Funkcija vraća broj uspešno učitanih tačaka.
- Definisati funkciju **double obim(TACKA\* poligon, unsigned n)** koja izračunava obim poligona sa  $n$  tačaka u zadatom nizu (napomena: ne zaboraviti stranicu koja spaja poslednje i prvo teme).
- Definisati funkciju **double maksimalna\_stranica(TACKA\* poligon, unsigned n)** koja izračunava dužinu najduže stranice poligona sa  $n$  tačaka u zadatom nizu (napomena: ne zaboraviti stranicu koja spaja poslednje i prvo teme).

- Definisati funkciju `main` u kojoj se sa standardnog ulaza učitava celobrojna nenegativna vrednost  $N$  ( $0 < N \leq 100$ ).

Inače, poziva se funkcija `ucitaj_poligon`. Ukoliko je uspešno učitano  $m$  tačka ( $N$  ne mora da bude jednako  $m$ ), onda se poziva funkcija `obim` za  $m$  učitanih tačaka i ispisuje njen rezultat na standardni izlaz (ukoliko ova funkcija nije implementirana — ispisati na standardni izlaz simbol `?`). Posle toga se poziva funkcija `maksimalna_stranica` za  $m$  učitanih tačaka i ispisuje njen rezultat na standardni izlaz (ukoliko ova funkcija nije implementirana — ispisati na standardni izlaz simbol `?`).

[Rešenje 3.125]

## 3.10 Rešenja

### Rešenje 3.109

```
1  /*
3   Data je struktura koja opisuje koordinate
   tacke u ravni:
5
7   typedef struct point
   {
9       float x;
       float y;
11      } POINT;
13
14  U glavnom programu date su dve tacke: tacka
15  A sa fiksiranim koordinatama (1,2) i tacka B
16  cije koordinate zadaje korisnik. Napisati
17  funkcije:
18  a) za racunanje rastojanja izmedju dve date tacke
19  b) za odredjivanje tacke koja se nalazi na
20     sredini duzi odredjene dvema datim tackama
21
22  Testirati napisane funkcije u glavnom programu.
23
24  */
25
26  #include <stdio.h>
27  #include <math.h>
28
29  typedef struct point
   {
30      float x;
31      float y;
```

```
31 } POINT;

33 /*
   Poljima strukture pristupamo pomocu
35 operatora .

37 Ako je promenljiva a tipa POINT,
   njenim koordinatama pristupamo
39 pomocu a.x i a.y
41 */

41 float rastojanje (POINT a, POINT b)
42 {
43     return sqrt(pow(a.x-b.x,2)+pow(a.y-b.y,2));
44 }

47 POINT sredina (POINT a, POINT b)
48 {
49     POINT s;
50     s.x = (a.x+b.x)/2;
51     s.y = (a.y+b.y)/2;
52     return s;
53 }

55 int main()
56 {

59     POINT a = {1,2};
60     POINT b;
61     POINT sredina_a_b;

63     /* Ispisujemo koordinate tacke a. */
64     printf("Tacka a ima koordinate %.2f,%.2f\n", a.x, a.y);

65     /* Ucitavamo koordinate tacke b. */
66     printf("Unesi prvu koordinatu tacke: ");
67     scanf("%f", &b.x);
68     printf("Unesi drugu koordinatu tacke: ");
69     scanf("%f", &b.y);
70     printf("Tacka b ima koordinate %.2f,%.2f\n", b.x, b.y);

72     /* Strukture kao argumenti funkcije - prenos po vrednosti. */
73     printf("Rastojanje izmedju tacaka a i b je %.2f\n", rastojanje(a,b)
74         );

75     /* Struktura kao povratna vrednost funkcije. */
76     sredina_a_b=sredina(a,b);
77     printf("Tacka na sredini izmedju tacaka a i b je %.2f,%.2f\n",
78         sredina_a_b.x, sredina_a_b.y);

79     return 0;
```

81 | }

#### Rešenje 3.110

```
1  /*
2  Data je struktura
3      typedef struct Student
4      {
5          char ime[MAX];
6          char prezime[MAX];
7          char smer;
8          float prosek;
9      } STUDENT;

11 I   Napisati funkciju koja učitava sa standardnog ulaza podatke o
      studentu. Mozemo pretpostaviti da
      ime i prezime studenta ne sadrže više od 30 karaktera.
13 II  Napisati funkciju koja ispisuje podatke o studentu na standardni
      izlaz.
      III Učitati niz od n studenata i :
15         a) ispisati imena i prezimena onih koji su na smeru R
            b) ispisati podatke za studenta sa najvećim prosekom; ako ima
            više takvih studenata, ispisati
17             1) sve njih
            2) prvog
19             3) poslednjeg
20 */
21
22 #include <stdio.h>
23 #define MAXST 100
24 #define MAX 31
25
26 typedef struct Student
27 {
28     char ime[MAX];
29     char prezime[MAX];
30     char smer;
31     float prosek;
32 } STUDENT;
33
34 /*
35 I
36
37 Ako je dat pokazivac na strukturnu promenljivu s,
38 poljima ove strukture pristupamo sa
39 (*s).ime, (*s).prezime, itd.
40
41 Zgrade su neophodne zbog prioriteta operatora:
42 operator * ima veći prioritet nego operator . .
43
44 Operator -> pruža skraćeni zapis za prethodno
```

```

45     navedeni pristup poljima:
46     s->ime je skraceno za (*s).ime
47     s->prezime je skraceno za (*s).prezime
48     itd.
49
50 */
51 void ucitaj(STUDENT* s)
52 {
53     /* printf("Ime:"); */
54     scanf("%s",s->ime);
55     /* printf("Prezime:"); */
56     scanf("%s",s->prezime);
57     getchar();
58     /* printf("Smer:"); */
59     scanf("%c",&s->smer);
60     /* printf("Prosek:"); */
61     scanf("%f", &s->prosek);
62 }
63
64 /* II */
65 /*
66  Kada neku promenljivu prenosimo u funkciju kao argument, obicno
67  je prenosimo po vrednosti (bez pokazivaca), ako se ona nece menjati
68  u funkciji
69  ili po adresi (preko pokazivaca), ako ce se njena vrednost
70  promeniti u funkciji.
71
72  Prilikom poziva funkcije, za svaki argument funkcije kreira se
73  promenljiva
74  koja predstavlja lokalnu kopiju argumenta i koja prestaje da
75  postoji po zavrsetku
76  funkcije. S obzirom da se strukture sastoje od vise polja,
77  zauzimaju
78  vise memorije nego nestrukturne promenljive. Zbog toga je za
79  njihovo kopiranje
80  potrebno vise vremena i vise memorijskih resursa nego za kopiranje
81  nestrukturnih
82  promenljivih.
83
84  Da bismo ucinili program efikasnijim, korisno je da strukturu uvek
85  kao
86  argument funkcije prenosimo po adresi (preko pokazivaca), bez
87  obzira
88  da li ce se struktura u toj funkciji menjati ili ne. Pokazivac na
89  strukturu
90  zauzima manje memorije nego sama struktura pa je izrada njegove
91  kopije
92  brza a kopija pokazivaca uzima manji memorijski prostor nego kopija
93  strukture.
94
95  Kada prenosimo strukturnu promenljivu u funkciju po adresi (preko
96  pokazivaca), tada

```

### 3 Predstavljanje podataka

---

```
85  imamo mogucnost da je u funkciji menjamo. Ukoliko zelimo da
    onemogucimo promenu,
    uz argument dodajemo kljucnu rec const. Ako pokusamo da promenimo
    argument
87  funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
    s->smer='X');),
    kompajler ce prijaviti gresku. Na ovaj nacin obezbedjujemo da
    promenljiva
89  koju smo preneli po adresi ne da bismo je promenili vec radi
    povecanja efikasnosti programa,
    ne bude, cak ni slucajno, izmenjena u funkciji.
91
    */
93
void ispisi(const STUDENT* s)
95 {
    printf("%s %s, %c, %.2f\n",s->ime, s->prezime, s->smer, s->prosek);
97 }
99
float najveći_prosek(STUDENT studenti[], int n)
101 {
    float m;
103     int i;
    /* Pretpostavimo da student sa indeksom 0 ima
105     maksimalni prosek. */
    m = studenti[0].prosek;
107     for(i=1;i<n;i++)
        if(m<studenti[i].prosek) /* Ako student sa indeksom i ima veci
            prosek od maksimalnog, */
109         m=studenti[i].prosek; /* menjamo maksimalni prosek */
    return m;
111 }
113
/*
    Struktura moze da bude povratna vrednost funkcije.
115 */
STUDENT prvi_student_sa_najvecim_prosekom(STUDENT studenti[], int n,
    float m)
117 {
    STUDENT s;
119     int i;
    for(i=0;i<n;i++)
121         if(m==studenti[i].prosek) /* Ako naidjemo na studenta sa
            maksimalnim prosekom, prekidamo petlju. */
            {
123                 /*
                    Na strukture se moze primenjivati
125                 naredba dodele.
                */
                s = studenti[i];
127                 break;
            }
```



```
129     }
130     return s;
131 }
132
133 /*
134     Strukturu mozemo preneti u funkciju preko pokazivaca. Strukture se
135     obavezno
136     prenose preko pokazivaca ukoliko je neophodno promeniti vrednosti
137     njihovih
138     polja u funkciji.
139 */
140 void poslednji_student_sa_najvecim_prosekom(STUDENT studenti[], int n
141     , float m, STUDENT* s)
142 {
143     int i;
144     for(i=0; i<n; i++)
145         if(m==studenti[i].prosek)
146             *s = studenti[i];
147 }
148
149 /*
150     Napomena: funkcije
151     1) prvi_student_sa_najvecim_prosekom
152     2) poslednji_student_sa_najvecim_prosekom
153     odredjuju studenta sa najvecim prosekom po odredenom kriterijumu.
154     Funkcija su realizovane na razlicite nacine kako bi ilustrovale:
155     - strukturu kao povratnu vrednost
156     - prenos strukture preko pokazivaca u funkciju, s obzirom da ce se
157       promeniti u funkciji
158
159     Prilikom izrade zadatka moze biti izabran bilo koji od opisanih
160     nacina rada, osim
161     ako neki nacin nije posebno naglasen u tekstu zadatka.
162
163 */
164 int main()
165 {
166     STUDENT studenti[MAXST];
167     int n;
168     int i;
169     float max_prosek;
170     STUDENT student_sa_max_prosekom;
171     int indeks;
172
173     /* printf("Unesi broj studenata:"); */
174     scanf("%d", &n);
175
176     if (n<0 || n>MAXST)
177     {
178         printf("Nekorektan unos\n");
179         return -1;
180     }
181 }
```

### 3 Predstavljanje podataka

```
177     }
178
179     /* printf("Unesi podatke o studentima:"); */
180     for(i=0;i<n;i++)
181     {
182         printf("%d. student:\n", i); */
183         ucitaj(&studenti[i]);
184     }
185
186     printf("Studenti sa R smerom:\n");
187     for(i=0;i<n;i++)
188     {
189         if(studenti[i].smer == 'R')
190             ispisi(&studenti[i]);
191     }
192     printf("-----\n");
193
194     /* b)1)
195
196     Stampamo podatke o svim studentima sa
197     maksimalnim prosekom.
198     */
199
200     max_prosek = najveći_prosek(studenti, n);
201     printf("Svi studenti koji imaju maksimalni prosek:");
202     for(i=0;i<n;i++)
203     {
204         if(studenti[i].prosek==max_prosek)
205             ispisi(&studenti[i]);
206     }
207
208     /* b)2) */
209     student_sa_max_prosekom = prvi_student_sa_najvecim_prosekom(
210         studenti,n,max_prosek);
211
212     printf("Prvi student u nizu sa najvećim prosekom: ");
213     ispisi(&student_sa_max_prosekom);
214
215     /* b)3) */
216     poslednji_student_sa_najvecim_prosekom(studenti,n,max_prosek,&
217         student_sa_max_prosekom);
218
219     printf("Poslednji student u nizu sa najvećim prosekom: ");
220     ispisi(&student_sa_max_prosekom);
221
222     return 0;
223 }
```

#### Rešenje 3.111

```
1 /*
3 Napisati program koji ucitava reci sa standardnog ulaza dok korisnik
  ne zada EOF i ispisiuje
```

```

ih na standardni izlaz svaku u posebnom redu, poravnatu udesno u
    odnosu
5 na poslednji karakter najduze reci. Koristiti
strukturu typedef struct rec
7 {
    char s[21];
9    int duzina;
    }REC;
11 Na primer, ako su unesene sledece reci:
Danas imamo ispit iz programiranja1.
13 Nadam se da nece biti tesko!
onda ispis izgleda ovako:
15     Danas
        imamo
17     ispit
        iz
19 programiranja1.
        Nadam
21     se
        da
23     nece
        biti
25     tesko!

27 Program realizovati kroz sledece funkcije:
a) Funkciju za učitavanje jedne reci u strukturu REC.
29 b) Funkciju za učitavanje niza struktura koja vraća dimenziju niza
c) Funkciju koja određuje maksimalnu dužinu reci u datom nizu
31 d) Funkciju koja ispisuje reci u traženom formatu

33 Mozemo pretpostaviti da nijedna rec ne sadrži više od 30 karaktera i
    da nece biti
uneto više od 1000 reci.

35 */
37 #include<stdio.h>
39 #include<string.h>
#define MAXRECI 100
41 #define MAX 31

43 typedef struct rec
{
45     char s[MAX];
    int duzina;
47 }REC;

49 void ucitaj_rec(REC* rec)
51 {
    scanf("%s", rec->s);
53     rec->duzina = strlen(rec->s);

```

```
    }
55
56     /*
57      U funkciji ucitaj_niz_reci argument n oznacava broj
58      elemenata niza reci, koji ce biti poznat tek po
59      zavrsetku funkcije. Ova promenljiva ce dobiti svoju
60      vrednost u funkciji i zbog toga mora biti prenesena
61      preko pokazivaca.
62
63     */
64
65     void ucitaj_niz_reci(REC reci[], int* pn, int granica)
66     {
67         int i=0;
68         do
69         {
70             ucitaj_rec(&reci[i]);
71             i++;
72         }
73         while(reci[i-1].duzina>0 && (i-1)<granica);
74
75         /*
76          S obzirom da se promenljiva i ucitava
77          pre ispitivanja uslova, uslov ispitujemo
78          za rec sa indeksom i-1
79         */
80
81         *pn = i-1;
82
83         /*
84          S obzirom da se vrednost promenljive i
85          ucitava i kada je unesen EOF, dimenzija
86          niza odgovarace vrednosti i-1
87         */
88     }
89
90     int max_duzina(REC reci[], int n)
91     {
92         int najveca_duzina;
93         int i;
94
95         /*
96          Najvecu duzinu inicijalizujemo na duzinu
97          prve reci.
98         */
99         najveca_duzina = reci[0].duzina;
100
101         for(i=1;i<n;i++)
102             if(reci[i].duzina>najveca_duzina) /* Ukoliko u nizu naidjemo
103                na rec duzine vece od najvece duzine, */
104                 najveca_duzina = reci[i].duzina; /* menjamo vrednost
105                promenljive najveca_duzina. */
106     }
```

```

105     return najveca_duzina;
106 }
107
108 /*
109  Da bismo realizovali ispis u trazenom formatu, pre
110  svake reci ispisujemo onoliko razmaka koliko iznosi
111  razlika maksimalne duzine i duzine date reci.
112 */
113
114 void ispis(REC reci[], int n, int max_d)
115 {
116     int i,j;
117
118     for(i=0;i<n;i++)
119     {
120         for(j=0;j<max_d-reci[i].duzina;j++)
121             printf(" ");
122         printf("%s\n", reci[i].s);
123     }
124 }
125
126 int main(int argc, char* argv[])
127 {
128     REC reci[MAXRECI];
129     int najveca_duzina;
130     int n;
131
132     ucitaj_niz_reci(reci, &n, MAXRECI);
133     najveca_duzina = max_duzina(reci,n);
134     ispis(reci, n, najveca_duzina);
135
136     return 0;
137 }

```

### Rešenje 3.112

```

/*
2  Napisati program koji izracunava prosečnu cenu jedne potrosacke
3  korpe. Potrosacka korpa se sastoji od broja kupljenih artikala i
4  niza kupljenih artikala. Svaki artikal određen je svojim nazivom,
5  kolicinom i cenom. Program treba da ucita broj potrosaca n (
6  najviše 100),
7  zatim podatke za n potrosackih korpi i da na osnovu ucitanih
8  podataka
9  izracuna prosečnu cenu potrosacke korpe. Ucitavanje se vrši sa
10 standarnog
11 ulaza pri čemu se prvo zadaje broj artikala, a zatim za svaki
12 artikal naziv,
13 kolicina i cena.  Mozemo pretpostaviti da nijedan

```

### 3 Predstavljanje podataka

---

```
10     potrosac nece kupiti vise od 20 artikala, kao i da naziv svakog
      artikla
      sadrzi maksimalno 30 karaktera.
12
13  */
14
15  #include <stdio.h>
16  #define MAXART 20
17  #define MAXPOT 100
18  #define MAXNAZIV 31
19
20  typedef struct artikal
21  {
22      char naziv[MAXNAZIV];
23      int kolicina;
24      float cena;
25  } ARTIKAL;
26
27  typedef struct korpa
28  {
29      int br_art;
30      ARTIKAL artikli[MAXART];
31  } KORPA;
32
33  /*
34   Funkcija ucitaj_artikal ucitava podatke za jedan
35   artikal i vraca 1 ako je ucitavanje bilo uspesno
36   a 0 u suprotnom. Ucitavanje je neuspesno ukoliko
37   kolicina nekog artikla ili njegova cena nisu pozitivni
38   brojevi.
39
40   S obzirom da funkcija ucitaj_artikal treba da vrati
41   dve vrednosti (ucitanu strukturu i indikator uspesnosti),
42   strukturu ARTIKAL prenosimo preko pokazivaca a
43   indikator uspesnosti vracamo kao povratnu vrednost.
44
45  */
46
47  int ucitaj_artikal(ARTIKAL* a)
48  {
49
50      scanf("%s", a->naziv);
51      scanf("%d", &a->kolicina);
52
53      if (a->kolicina<=0)
54      {
55          printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina
56          );
57          return 0;
58      }
59
60      scanf("%f",&a->cena);
```

```
60     if (a->cena<0)
61     {
62         printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
63         return 0;
64     }
65
66     return 1;
67 }
68
69 /*
70 Funkcija izracunaj_racun izracunava racun date
71 potrosacke korpe u kojoj su inicijalizovani
72 podaci o broju artikala i o svakom pojedinacnom
73 artiklu.
74 */
75 float izracunaj_racun(const KORPA* k)
76 {
77     int i;
78     float racun=0;
79     for(i=0;i<k->br_art;i++)
80         racun+=k->artikli[i].kolicina * k->artikli[i].cena;
81     return racun;
82 }
83
84 /*
85 Pri učitavanju korpe, zadaje se broj artikala a zatim
86 podaci za svaki artikal.
87
88 Funkcija učitaj_korpu vraća 1 ako je učitavanje uspesno
89 i 0 u suprotnom. Do neuspesnog učitavanja može doći
90 ako broj artikala u korpi nije pozitivan ili ako dodje
91 do neuspesnog učitavanja nekog artikla.
92 */
93
94 int učitaj_korpu(KORPA* k)
95 {
96     int i;
97     scanf("%d", &k->br_art);
98     if (k->br_art<=0)
99     {
100         printf("Nekorektan unos za broj artikala: %d\n", k->br_art);
101         return 0;
102     }
103     for(i=0; i<k->br_art;i++)
104         if (učitaj_artikal(&k->artikli[i])==0)
105             return 0;
106
107     return 1;
108 }
109
110 /*
111 Funkcija učitaj_niz_korpi učitava podatke
```

### 3 Predstavljanje podataka

---

```
112     za niz od n potrosackih korpi. Funkcija
113     vraca 1 ako je ucitavanje uspesno i 0 ako
114     nije. Ucitavanje je neuspesno ukoliko ne uspe
115     ucitavanje jedne od korpi.
116 */
117
118 int ucitaj_niz_korpi(KORPA korpe[], int n)
119 {
120     int i,j;
121     for(i=0; i<n; i++)
122         if(ucitaj_korpu(&korpe[i])==0)
123             return 0;
124
125     return 1;
126 }
127
128 /*
129 Funkcija stampaj_racun ispisuje na
130 standardni izlaz racun za datu korpu
131 tako sto za svaki artikal ispise
132 naziv, cenu i kolicinu i na kraju
133 ukupnu cenu za kupljene artikle.
134 */
135
136 void stampaj_racun(const KORPA* k)
137 {
138     int i,j;
139     for(i=0;i<k->br_art;i++)
140         printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
141             kolicina, k->artikli[i].cena);
142     printf("-----\n");
143     printf("\tukupno: %.2f\n", izracunaj_racun(k));
144 }
145
146 /*
147 Funkcija stampaj_racune_za_korpe
148 ispisuje na standardni izlaz racune
149 za svaku korpu u nizu potrosackih
150 korpi
151 */
152
153 void stampaj_racune_za_korpe(KORPA korpe[], int n)
154 {
155     int i;
156     for (i=0;i<n;i++)
157     {
158         printf("\nKorpa %d:\n",i);
159         stampaj_racun(&korpe[i]);
160     }
161 }
162 }
```



```

164  /*
165     Funkcija prosek racuna proseccnu cenu
166     potrosacke korpe za dati niz potrosackih
167     korpi
168  */
169  float prosek(KORPA korpe[], int n)
170  {
171      int i;
172      float p;
173
174      for(i=0; i<n; i++)
175          p+=izracunaj_racun(&korpe[i]);
176
177      return p/n;
178  }
179
180  int main()
181  {
182      int n;
183      KORPA korpe[MAXPOT];
184
185      printf("Unesi broj potrosackih korpi:");
186      scanf("%d", &n);
187
188      if(n<0 || n>MAXPOT)
189      {
190          printf("Nekorektan unos broja potrosackih korpi: %d\n", n);
191          return -1;
192      }
193
194      if (ucitaj_niz_korpi(korpe, n)==0)
195          return -1;
196
197      stampaj_racune_za_korpe(korpe, n);
198      printf("Proseccna cena potrosacke korpe: %.2f\n", prosek(korpe, n))
199          ;
200
201      return 0;
202  }

```

### Rešenje 3.113

```

1  /*
2     Struktura IZRAZ opisuje numericki izraz nad celim brojevima koji
3     se sastoji
4     od dva celobrojna operanda, numericke operacije nad celim
5     brojevima i
6     vrednosti izraza:
7
8     typedef struct izraz

```

### 3 Predstavljanje podataka

---

```
7      {
9      char o;
10     int x;
11     int y;
12     } IZRAZ;

13     a) Napisati funkciju koja ispituje da li je dati izraz korektno
14        zadat i vraća 1 ako jeste a 0 u suprotnom. Podrazumevamo da je
15        izraz korektno zadat ako operacija odgovara +,-,* ili / i u
16        slucaju
17        deljenja drugi operand je razlicit od 0.

18     b) Napisati funkciju koja za dati izraz odredjuje vrednost izraza.

19     c) Napisati funkciju koja ucitava dati izraz. Funkcija
20        treba da ucita sa standardnog ulaza operaciju i dva
21        operanda u polja o, x i y strukture IZRAZ. Funkcija vraća
22        1 ako je ucitavanje bilo uspesno, tj. ako je izraz bio
23        korektno zadat ili 0 u suprotnom.

24     d) Napisati funkciju koja stampa dati izraz infiksno, u obliku
25        x o y = vr. Na primer, za izraz + 4 17 ispis treba
26        da bude 4+17=21

27

28     e) Napisati glavni program koji ucitava prirodan broj n<1000 a
29        zatim n izraza
30        u notaciji
31        + 4 17
32        - 8 -16
33        Program treba da ispise maksimalnu vrednost medju unetim izrazima
34        i da ispise one
35        izraze cija je vrednost manja od polovine maksimalne vrednosti.

36

37

38

39 */

40
41 #include <stdio.h>
42 #define MAX 1000

43 typedef struct izraz
44 {
45     char o;
46     int x;
47     int y;
48 } IZRAZ;

49

50

51

52 /*
53 Funkcija korektan_izraz vraća 1 ako je izraz korektan a 0
54 u suprotnom. Izraz je korektan ukoliko se sastoji od
```

```
57     aritmetickih operacija +,-,* ili /, i ukoliko je u slucaju
    operacije deljenja drugi operand razlicit od nule.
58 */
59 int korektan_izraz(const IZRAZ* izraz)
60 {
61     if(izraz->o!='+' && izraz->o!='-' && izraz->o!='*' && izraz->o!='/'
    ')
62     {
63         printf("Nedozvoljena operacija!\n");
64         return 0;
65     }
66     if(izraz->o=='/' && izraz->y==0)
67     {
68         printf("Deljenje nulom!\n");
69         return 0;
70     }
71     return 1;
72 }
73
74 /*
75 Promenljiva izraz ce se promeniti u funkciji
76 vrednost tako sto ce njenom neinicijalizovanom
77 polju vr biti dodeljena vrednost izraza. Zbog
78 toga ovu promenljivu funkciji prosledjujemo
79 po adresi, preko pokazivaca
80 */
81 int vrednost(const IZRAZ* izraz)
82 {
83     int v;
84
85     switch (izraz->o)
86     {
87         case '+':
88             v=izraz->x+izraz->y;
89             break;
90         case '-':
91             v=izraz->x-izraz->y;
92             break;
93         case '*':
94             v=izraz->x*izraz->y;
95             break;
96         case '/':
97             v=izraz->x/izraz->y;
98             break;
99     }
100     return v;
101 }
102
103
104 /*
105 Promenljiva izraz ce se promeniti u funkciji
```

### 3 Predstavljanje podataka

---

```
107      ucitaj_izraz tako sto ce njenim neinicijalizovanim
109      poljima o,x,y biti dodeljene vrednosti ucitane
      sa standardnog ulaza. Zbog toga ovu promenljivu
      funkciji prosledjujemo po adresi, preko pokazivaca.

111
      S obzirom da ucitavanje karaktera nije prvo
113      ucitavanje koje se obavlja u programu, funkcijom
      getchar() "pokupimo" karakter kojim razdvajamo
115      unos karaktera od prethodnog unosa (najcesce blanko
      znak)

117
      */
119
      int ucitaj_izraz(IZRAZ* izraz)
121      {
          getchar();
123          scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
          if (!korektan_izraz(izraz))
125              return 0;
          return 1;
127      }

129
      void stampaj_izraz(const IZRAZ* izraz)
131      {
          printf("%d %c %d = %d\n", izraz->x, izraz->o, izraz->y, vrednost(
133              izraz));
      }

135
      int max_vr(IZRAZ izrazi[], int n)
      {
137          int i;
          int max;
139          /* Trazimo maksimalnu vrednost izraza */
          max=vrednost(&izrazi[0]);

141
          /* U petlji... */
143          for(i=1; i<n; i++)
          /* Ako je ona veca od maksimalne: */
145              if(vrednost(&izrazi[i])>max)
                  /* Azuriramo max: */
147                  max=vrednost(&izrazi[i]);
          return max;
149      }

151
      int main()
      {
153          int n;
          IZRAZ izrazi[MAX];
155          int max;
          int i;
157
```

```

159  /* Ucitavamo broj izraza: */
    scanf("%d", &n);
    if(n<0 || n>MAX)
161  {
        printf("Nekorektna vrednost broja n!\n");
163      return -1;
    }

165

167  /* U petlji učitavamo jedan po jedan izraz: */
    for(i=0; i<n; i++)
169      if(ucitaj_izraz(&izrazi[i])==0)
        {
171            printf("Nekorektan unos\n");
            return -1;
173        }

175

    printf("Svi izrazi:\n");
    for(i=0; i<n; i++)
177        stampaj_izraz(&izrazi[i]);

179

    max = max_vr(izrazi, n);
    printf("Maksimalna vrednost izraza:%d\n", max);

181

    printf("Izrazi cija je vrednost manja od polovine maksimalne
        vrednosti:\n");

183

    for(i=0; i<n; i++)
        if(vrednost(&izrazi[i])<max/2)/* Ako je vrednost tekućeg izraza
            manja od polovine maksimalne, ispisujemo ga. */
187            stampaj_izraz(&izrazi[i]);

189    return 0;
}

```

### Rešenje 3.114

```

1  #include <stdio.h>

3  /* Struktura koja opisuje kompleksni broj obuhvata polje za realni i
    polje za imaginarni deo broja. */
    typedef struct Complex {
5
        float re;
        float im;
7    } Complex;

9

    /* Funkcija kojom se izracunava zbir kompleksnih brojeva */
11  Complex saberi(Complex *a, Complex *b) {

```

```
13   Complex c;
14   c.re = a->re + b->re;
15   c.im = a->im + b->im;
16   return c;
17 }

19 /* Funkcija kojom se izracunava razlika kompleksnih brojeva */
20 Complex oduzmi(Complex *a, Complex *b) {
21
22   Complex c;
23   c.re = a->re - b->re;
24   c.im = a->im - b->im;
25   return c;
26 }
27
28 /* Funkcija kojom se izracunava proizvod kompleksnih brojeva */
29 Complex pomnozi(Complex *a, Complex *b) {
30
31   Complex c;
32   c.re = a->re * b->re - a->im * b->im;
33   c.im = b->re * a->im + a->re * b->im;
34   return c;
35 }
36
37 /* Funkcija kojom se izracunava kolicnik kompleksnih brojeva */
38 Complex podeli(Complex *a, Complex *b) {
39
40   Complex c;
41   c.re = (a->re * b->re + a->im * b->im) / (b->re*b->re + b->im*b->im
42   );
43   c.im = (b->re * a->im - a->re * b->im) / (b->re*b->re + b->im*b->im
44   );
45   return c;
46 }
47
48 int main() {
49
50   Complex a, b;
51   Complex c;
52
53   /* Ucitavamo kompleksne brojeve */
54   printf("Unesite realni i imaginarni deo prvog broja: ");
55   scanf("%f%f", &a.re, &a.im);
56
57   printf("Unesite realni i imaginarni deo drugog broja: ");
58   scanf("%f%f", &b.re, &b.im);
59
60   c = saberi(&a, &b);
61   printf("Zbir: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im);
62   /* Ukoliko je imaginarni deo negativan,
63      njegov zapis vec ukljucuje znak,
64      te to treba proveriti.
```

```

63                                     Inace, broj je oblika a+b*i
                                     */
65  c = oduzmi(&a, &b);
  printf("Razlika: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im)
  ;
67  c = pomnozi(&a, &b);
69  printf("Proizvod: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im
  );
71  if(b.re != 0 || b.im != 0) {
    c = podeli(&a, &b);
73    printf("Kolicnik: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.
      im);
  }
75  else
    printf("Kolicnik ne postoji.\n"); /* Ni u polju kompleksnih
      brojeva
77                                     nije dozvoljeno deljenje nulom
79                                     */
81  return 0;
  }

```

### Rešenje 3.115

```

1  #include <stdio.h>
  #include <math.h>
3
  #define MAX 50
5
  typedef struct lopta {
7    int poluprecnik;
    enum {plava, zuta, crvena, zelena} boja; /* tip "boja" je
      nabrajajuci tip,
9                                     a efekat nabiranja mogucih vrednosti
                                     {plava, zuta, zelena, crvena}
11                                    ekvivalentan je definisanju
                                     4 celobrojne konstante direktivom #define
13                                    */
  } LOPTA;
15
  /* Funkcija koja odredjuje zapreminu lopte */
17  float zapremina(LOPTA* l) {
    return pow(l->poluprecnik, 3)*4/3*M_PI;
19  }
21
  /* Pomocna funkcija koja racuna zapreminu svih lopti */
  float ukupna_zapremina(LOPTA lopte[], int n) {

```

### 3 Predstavljanje podataka

---

```
23     int i;
25     float z = 0;

27     for(i = 0; i < n; i++)
        z += zapremina(&lopte[i]);

29     return z;
31 }

33 /*
    Funkcija je opstija od trazene i broji sve lopte odredjene boje u
    nizu lopti.
35 U zavisnosti od prosledjene boje funkciji, funkcija vraca
    odgovarajuci broj.
37 */
int broj_lopti_u_boji(LOPTA lopte[], int n, int boja) {

39     int br = 0;
41     int i;
43     for(i = 0; i < n; i++)
        if(lopte[i].boja == boja)
            br++;
45     return br;
}

47 int main() {

49     LOPTA lopte[MAX];
51     int n;
53     int i;
55     int boja;

57     printf("Unesite broj lopti: ");
59     scanf("%d", &n);

61     if(n < 1 || n > MAX) {

        printf("Nekorektan unos.\n");
        return 0;
    }

63     printf("Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta,
        3-crvena, 4-zelena):\n");
65     for(i = 0; i < n; i++) {

        printf("%d. lopta: ", i+1);
        scanf("%d", &lopte[i].poluprecnik, &boja);

67     }

69     /* U zavisnosti od unetog
        celog broja,
71     bira se boja lopte.
```



```

73     */
    switch(boja) {
75         case 1: lopte[i].boja = plava; break;
76         case 2: lopte[i].boja = zuta; break;
77         case 3: lopte[i].boja = crvena; break;
78         case 4: lopte[i].boja = zelena; break;
79         default:
80             printf("Nekorektan unos.\n");
81             return 0;
82     }
83 }

85 printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));

87 printf("Ukupno crvenih lopti: %d\n", broj_lopti_u_boji(lopte, n,
88     crvena));

89 return 0;
90 }

```

### Rešenje 3.116

```

#include <stdio.h>
2 #include <string.h>

4 #define MAX_DUZINA 21
5 #define MAX_BR_VOCKI 50

6
7 typedef struct vocka
8 {
9     char ime[MAX_DUZINA];
10    float vitamin;
11 } VOCKA;
12

14 int main()
15 {
16     VOCKA vocke[MAX_BR_VOCKI];
17     int i = 0, n, max_vocka;
18     char ime[MAX_DUZINA];

20     /*
21      Ucitavamo podatke o vockama i smestamo ih u niz
22      sve dok ne unesemo rec KRAJ ili ucitamo MAX_BR_VOCKI vocki.
23     */
24     do
25     {
26         printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
27         scanf("%s", ime);
28         /*

```

### 3 Predstavljanje podataka

```
        Kada unesemo rec KRAJ prekidamo petlju.
30  */
    if(strcmp(ime, "KRAJ") == 0)
32      break;

34  /*
    Inace ucitavamo i kolicinu vitamina
36    i tu vrednost smestamo u vocku na poziciji i
    */
38  strcpy(vocke[i].ime, ime);
    scanf("%f", &vocke[i].vitamin);
40  i++;
}
42 while(i < MAX_BR_VOCKI);

44 n = i;

46 /*
    Pretpostaviceмо da prva vocka ima najvise vitamina.
48    Procicemo kroz niz vocki i ukoliko naidjemo na vocku koja ima
    vise vitamina
    od one koja trenutno ima najvise, azuriracemo vrednosti
    maksimalne vocke.

50    Sve vreme cuvamo indeks vocke sa najvise vitamina C.
52  */

54  max_vocka = 0;
    for(i=1; i < n; i++)
56      if(vocke[i].vitamin > vocke[max_vocka].vitamin)
      {
58        max_vocka = i;
      }

60  printf("Voce sa najvise C vitamina je: %s\n", vocke[max_vocka].ime)
    ;

62  return 0;
64 }
```

#### Rešenje 3.117

```
#include <stdio.h>
2 #include <string.h>

4 #define MAX_IME_PREZIME 51
    #define MAX_ZELJA 101
6 #define MAX_BR_STUDENATA 100

8 typedef struct student
    {
```

```
10  char imeipre[MAX_IME_PREZIME];
11  char zelja[MAX_ZELJA];
12 } STUDENT;

14
15
16 int main()
17 {
18     STUDENT studenti[MAX_BR_STUDENATA];
19     char odgovor[3];
20     char imeiprezime[MAX_IME_PREZIME];
21     int i = 0, n;
22     int broj_pronalazaka;

23
24     /*
25      Ucitavamo studente i njihove zelje i upisujemo ih u niz
26      sve dok to zelimo ili dok ne unesemo MAX_BR_STUDENATA studenata.
27     */
28     while(i < MAX_BR_STUDENATA)
29     {
30         printf("Ime i prezime studenta: \n");
31         /*
32          Funkcija fgets ucitava jednu liniju i smesta je promenljivu
33          koju zadajemo kao njen prvi argument.
34          Drugi argument je maksimalna duzina linije.
35          Treci argument je kod nas stdin sto predstavlja standardni ulaz
36          .
37         */
38         if(fgets(studenti[i].imeipre, MAX_IME_PREZIME, stdin) == NULL)
39         {
40             printf("Greska: nismo dobro ucitali ime i prezime studenta.");
41             return 0;
42         }

43         printf("Njegova zelja: \n");

44         if(fgets(studenti[i].zelja, MAX_ZELJA, stdin) == NULL)
45         {
46             printf("Greska: nismo dobro ucitali zelju studenta.");
47             return 0;
48         }

49         i++;

50         printf("Jos vrednih studenta (da/ne)?\n");

51         scanf("%s", odgovor);

52
53         /*
54          Moramo da pokupimo karakter koji unesemo nakon odgovora
55          kako ga ne bismo ucitali u sledecoj iteraciji petlje.
56         */
57     }
```

### 3 Predstavljanje podataka

```
60     */
61     getchar();
62     /*
63      *   Ukoliko je nas odgovor "ne" prekidamo petlju.
64      *   A ukoliko nije ni "da" ni "ne" onda nismo dobro uneli odgovor.
65      */
66     if(strcmp(odgovor,"ne") == 0)
67         break;
68     else if(strcmp(odgovor,"da") != 0)
69     {
70         printf("Greska: odgovor mora biti u obliku (da/ne)!");
71         return 0;
72     }
73 }
74
75
76 /*
77  *   Postavljamo dimenziju niza.
78  */
79 n = i;
80 printf("Za podsecanje uneti ime i prezime: \n");
81 if(fgets(imeiprezime, MAX_IME_PREZIME, stdin) == NULL)
82 {
83     printf("Greska: nismo dobro ucitali ime i prezime studenta.");
84     return -1;
85 }
86
87 /* Prolazimo kroz listu studenta i ispisujemo zelje studenta cije
88    ime i prezime smo uneli. */
89 broj_pronalazaka=0;
90 for(i=0;i<n;i++){
91     if(strcmp(imeiprezime, studenti[i].imeipre) == 0){
92         broj_pronalazaka++;
93         printf("Novogodisnja zelja: %s\n",studenti[i].zelja);
94     }
95 }
96
97 /* Za slucaj da nismo pronasli studenta sa trazenim imenom */
98 if(broj_pronalazaka==0){
99     printf("Trazeni student ne postoji - mozda ce mu poklon odneti
100     drugi Deda Mraz\n");
101 }
102
103 return 0;
104 }
```

#### Rešenje 3.118

```
1 /*
2  *   Definisati strukturu Grad u kojoj se nalazi ime grada (niska duzine
3  *   20 karaktera) i prosečna temperatura u
```

```

3   toku decembra (realan broj). Napisati program koji učitava imena n
   (0<n<50) gradova i njihove prosečne
   temperature, a zatim ispisuje one gradove koji imaju idealnu
   temperaturu za klizanje: od 3 do 8 stepeni.
5   Napomena: probati sa testiranjem zadatka pomocu preusmeravanja.
   */
7
9   #include <stdio.h>
10  #define MAX_DUZINA 20
11  #define MAX_BR_GRADOVA 50
12
13  typedef struct Grad{
14      char ime_grada[MAX_DUZINA+1];
15      float temperatura;
16  }Grad;
17
18  int main(){
19      int n, i;
20      Grad grad[MAX_BR_GRADOVA];
21
22      printf("Unesite broj n: ");
23      scanf("%d", &n);
24      if(n<0 || n>MAX_BR_GRADOVA){
25          printf("Greska: pogresan unos!\n");
26          return 0;
27      }
28
29      for(i=0; i<n; i++){
30          printf("Unesite grad i temperaturu: ");
31          scanf("%s %f", grad[i].ime_grada, &grad[i].temperatura);
32      }
33
34      printf("Gradovi sa idealnom temperaturom za klizanje u decembru:\n"
35            );
36      for(i=0; i<n; i++){
37          if(grad[i].temperatura>=3 && grad[i].temperatura<=8){
38              printf("%s\n", grad[i].ime_grada);
39          }
40      }
41
42      return 0;
43  }

```

### Rešenje 3.119

```

1  /*
   Definirati strukturu ParReci koja sadrži rec na srpskom jeziku i
   odgovarajući prevod na engleski jezik. Zatim
3  sa standardnog ulaza sve do kraja ulaza učitavati parove reci i,
   posebno, za rečenicu koja se zadaje sa ulaza

```

### 3 Predstavljanje podataka

---

```
    ispisati prevod - ako je rec u recenici nepoznata umesto nje
    ispisati odgovarajuci broj zvezdica. Rec i nece biti
5   duze od 50 karaktera, a ukupan broj reci nece biti veci od 100.
    Napomena: probati sa testiranjem zadataka
    pomocu preusmeravanja.
7   */

9   #include <stdio.h>
   #include <string.h>
11  #define MAX_DUZINA 20
   #define MAX_BR_REC I 100
13  #define MAX_DUZINA_RECENICE 100

15  typedef struct ParReci{
    char sr[MAX_DUZINA+1];
17    char en[MAX_DUZINA+1];
   }ParReci;
19

21  /*
    Funkcija koja u recniku koji sadrzi n reci trazi prevod reci rec i
    upisuje ga u prevod.
23    Ukoliko se rec ne nalazi u recniku, prevod se sastoji od zvezdica
    pri cemu broj zvezdica odgovara
    duzini nepoznate reci.
25  */

27  void pronadji_prevod(ParReci recnik[], int n, char rec[], char prevod
    []){
    int i;
29

    /* Pretraujemo recnik i trazimo zadatu rec */
31    for(i=0; i<n; i++){

33        /* Ukoliko se rec nalazi u recniku */
        if(strcmp(recnik[i].sr, rec)==0){
35            /* Ocitavamo njen prevod */
            strcpy(prevod, recnik[i].en);
37            /* I obustavljamo pretragu */
            return;
39        }
    }

41    /* Ukoliko rec nije pronadjena, formiramo prevod reci koji se
    sastoji od zvezdica */
43    for(i=0; rec[i]; i++){
        prevod[i]='*';
45    }
    prevod[i]='\0';
47 }

49
```

```

int main(){
51   ParReci recnik[MAX_BR_RECI];
    int n;
53   char sr[MAX_DUZINA+1];
    char en[MAX_DUZINA+1];
55   int i, j, k;
    char recenica[MAX_DUZINA_RECENICE+1];
57   char rec[MAX_DUZINA+1];
    char prevod[MAX_DUZINA+1];
59   int citamo_rec;
    char* novi_red;

61   /* Ucitavamo parove reci sa standardnog ulaza sve do kraja ulaza*/
63   i=0;
    while(scanf("%s %s", sr, en)!=EOF){
65       if(i==MAX_BR_RECI)
           break;

67       strcpy(recnik[i].sr, sr);
69       strcpy(recnik[i].en, en);

71       i++;
    }
73   /* Broj parova reci cuvamo u promenljivoj n */
    n=i;

75   /* Ucitavamo recenicu - nisku karaktera sve do pojave znaka za novi
       red */
77   printf("Unesite recenicu za prevod:\n");
    fgets(recenica, MAX_DUZINA_RECENICE, stdin);

79   /* Ako postoji, zamenjujemo znak za novi red terminirajucom nulom
       */
81   novi_red=strchr(recenica, '\n');
    if(novi_red!=NULL)
83       *novi_red='\0';

85

    /* Izdvajamo rec po rec unesene recenice */
87   /* j oznacava tekuci karakter recenice koji se obradjuje */
    j=0;
89   /* citamo_rec sa mogucim vrednostima 1 i 0 ce biti indikator koji
       pokazuje da li citamo rec ili ne */
    citamo_rec=0;

91   while(1){
93       /* Proveravamo da li smo stigli do kraja recenice */
        if(recenica[j]=='\0')
95           break;

97       /* Ukoliko smo procitali karakter koji je sastavni deo reci (nije
           belina) */

```

```
99     if(recenica[j]!=' ' && recenica[j]!='\n' && recenica[j]!='\t'){
100         /* Smestamo ga u rec */
101         if(citamo_rec==0){
102             citamo_rec=1;
103             k=0;
104         }
105         rec[k]=recenica[j];
106         k++;
107     }
108     else{
109         /* Inace, procitali smo karakter koji ne treba ukljuciti u rec */
110         /* Ako smo pre toga citali rec */
111         if(citamo_rec==1){
112             /* Prekidamo citanje */
113             citamo_rec=0;
114             rec[k]='\0';
115
116             /* I trazimo i ispisujemo odgovarajuci prevod reci */
117             pronadji_prevod(recnik, n, rec, prevod);
118             printf("%s ", prevod);
119         }
120     }
121
122     /* Prelazimo na sledeci karakter recenice */
123     j++;
124 }
125
126 /* Za slucaj da nije obradjena, obradjujemo poslednju rec i
127    ispisujemo njen prevod */
128 if(citamo_rec){
129     rec[k]='\0';
130     pronadji_prevod(recnik, n, rec, prevod);
131     printf("%s\n", prevod);
132 }
133
134 return 0;
135 }
```

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125



Rešenje [3.125](#)

Rešenje [3.125](#)



# 4

## Ulaz i izlaz programa

### 4.1 Standardni tokovi

### 4.2 Argumenti komandne linije

### 4.3 Datoteke

**Zadatak 4.1** Tekst

[Rešenje [4.1](#)]

**Zadatak 4.2** Tekst

[Rešenje [4.2](#)]

**Zadatak 4.3** Tekst

[Rešenje [4.3](#)]

**Zadatak 4.4** Tekst

[Rešenje [4.4](#)]

**Zadatak 4.5** Tekst

[Rešenje [4.5](#)]

**Zadatak 4.6** Tekst

[Rešenje [4.6](#)]

**Zadatak 4.7** Napisati program koji prebrojava mala slova u datoteci *test.txt*.

*Primer 1*

```
TEST.TXT
Abcd EFGH+ijKLMN

IZLAZ:
Broj malih slova je: 5
```

*Primer 2*

```
TEST.TXT
PrograMiranje

IZLAZ:
Broj malih slova je: 11
```

[Rešenje 4.7]

**Zadatak 4.8** Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*.

*Primer 1*

```
ULAZ.TXT
Volim programiranje.
IZLAZ.TXT
Vipgmae
```

[Rešenje 4.8]

**Zadatak 4.9** Kao argumenti komandne linije se zadaju ime datoteke i ceo broj  $k$ . Napisati program koji na standardni izlaz ispisuje sve linije zadate datoteke čija je dužina veća od  $k$ . Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera.

*Primer 1*

```
POKRETANJE: ./a.out test.txt 7
TEST.TXT
Teme koje su obradjuvane:
Petlje
Funkcije
Nizovi
Strukture

IZLAZ:
Teme koje su obradjuvane:
Funkcije
Strukture
```

*Primer 2*

```
POKRETANJE: ./a.out test.txt

IZLAZ:
Greska: Pogresan broj argumenata!
```

[Rešenje 4.9]

**Zadatak 4.10** Napisati program koji prebrojava koliko se linija datoteke *ulaz.txt* završava niskom  $s$  koja se učitava sa standardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera, kao i da dužina niske  $s$

neće biti veća od 20 karaktera.

#### Primer 1

```

ULAZ.TXT
abcde abcde
abcde aab
abcde abcde abcde
abcde abcde Aab
abcde abcde ab
abcde abcde abcde abcde

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 3

```

#### Primer 2

```

ULAZ.TXT
abcde abcde
abcde
abcde abcde AB

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 0

```

[Rešenje 4.10]

**Zadatak 4.11** Napisati program koji pronalazi maksimum brojeva zapisanih u datoteci *brojevi.txt*.

#### Primer 1

```

BROJEVI.TXT
2.36 -16.11 5.96 8.88
-265.31 54.96 38.4

IZLAZ:
Najveci broj je: 54.96

```

[Rešenje 4.11]

**Zadatak 4.12** U datoteci *studenti.txt* se nalaze informacije o studentima: prvo broj studenata, a zatim u pojedinačnim linijama korisničko ime i pet poslednjih ocena koje je student dobio. Napisati program koji pronalazi studenta koji je ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da broj studenata neće biti veći od 100.

#### Primer 1

```

STUDENTI.TXT
mr15239 10 9 9 8 10
m114005 8 8 9 8 10
m115112 9 8 8 7 10
mr15007 10 10 10 10 10
mn13208 7 7 9 6 10

IZLAZ:
korisnicko ime: mr15007, prosek ocena: 10.00

```

[Rešenje 4.12]

## 4 Ulaz i izlaz programa

**Zadatak 4.13** U datoteci *tacke.txt* se nalazi prvo broj tačaka, a zatim u pojedinačnim linijama  $x$  i  $y$  koordinate tačke. Napisati program koji u datoteku *rastojanja.txt* upisuje rastojanje svake od pročitanih tačaka od koordinatnog početka, a na standardni izlaz koordinate tačke koja je najudaljenija. Koristiti strukturu *Tacka* sa poljima  $x$  i  $y$ , kao i funkciju kojom se računa rastojanje. Pretpostaviti da broj tačaka u datoteci neće biti veći od 50.

### Primer 1

```
TACKE.TXT
4
11 -2
3 5
8 -8
0 4

RASTOJANJA.TXT
11.18
5.29
11.31
4.00

IZLAZ:
Najudaljenija je tačka: 8 -8
```

### Primer 1

```
TACKE.TXT
-2
0 0
9 -8

IZLAZ:
Greska: Nedozvoljen broj tacaka!
```

[Rešenje 4.13]

**Zadatak 4.14** Napisati program koji za reč  $s$  maksimalne dužine 20 karaktera koja se zadaje sa standardnog ulaza u datoteku *rotacije.txt* upisuje sve rotacije reči  $s$ .

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite rec: abcde

ROTACIJE.TXT
abcde
bcdea
cdeab
deabc
eabcd
```

[Rešenje 4.14]

**Zadatak 4.15** Napisati program koji linije koji se učitavaju sa standardnog ulaza sve do kraja ulaza prepisuje u datoteku *izlaz.txt* i to, ako je prilikom pokretanja zadata opcija  $-v$  ili  $-V$  samo one linije koje počinju velikim slovom, ako je zadata opcija  $-m$  ili  $-M$  samo one linije koje počinju malim slovom, a ako je opcija izostavljena sve linije. Pretpostaviti da linije neće biti duže od 80 karak-

tera.

#### Primer 1

```
POKRETANJE: ./a.out -m
INTERAKCIJA SA PROGRAMOM:
Unesite recenice:
programiranje u C-u je zanimljivo
Volim programiranje!
Kada porastem bicu programer!
u slobodno vreme programiram

IZLAZ.TXT
programiranje u C-u je zanimljivo
u slobodno vreme programiram
```

#### Primer 2

```
POKRETANJE: ./a.out -V
INTERAKCIJA SA PROGRAMOM:
Unesite recenice:
programiranje u C-u je zanimljivo
Volim programiranje!
Kada porastem bicu programer!
u slobodno vreme programiram

IZLAZ.TXT
Volim programiranje!
Kada porastem bicu programer!
```

#### Primer 3

```
POKRETANJE: ./a.out -k
INTERAKCIJA SA PROGRAMOM:
Greska: Pogresno pokretanje programa!
```

[Rešenje 4.15]

**Zadatak 4.16** Sa standardnog ulaza učitavaju se imena dve tekstualne datoteke i jedan karakter. Napisati program koji prepisuje datoteku čije se ime navodi kao prvo u datoteku čije ime se navodi kao drugo. Ukoliko je učitani karakter u program prilikom prepisivanja treba da zamenjuje sva mala slova velikim, a ukoliko je učitani karakter 1 sva velika slova se zamenjuju malim. U slučaju greške ispisati -1. Greška može biti neuspešno otvaranje datoteke ili pogrešno zadati karakter. Maksimalna dužina naziva datoteke je 20 karaktera.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
ulaz.txt izlaz.txt u
ULAZ.TXT
danas je lep dan
i Ja zelim
da postanem programer
IZLAZ.TXT
DANAS JE LEP DAN
I JA ZELIM
DA POSTANEM PROGRAMER
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
prva.dat druga.dat l
PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120
DRUGA.DAT
cena soka je 30
cena vina je 150
cena limunade je 200
cena sendvica je 120
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
  primer.c prazna.txt V
PRIMER.C
#include <stdio.h>
int main()
{
}
PRAZNA.TXT

IZLAZ:
-1
```

[Rešenje 4.33]

**Zadatak 4.17** Sastaviti program koji sa standardnog ulaza prima ime datoteke koju treba otvoriti. Ispisati (na standardnom izlazu) koja cifra (međusvim ciframa koje se pojavljuju u datoteci) ima najveći broj pojavljivanja. U slučaju greške pri otvaranju datoteke ispisati -1. Ukoliko nema cifara u datoteci ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  ulaz.txt
ULAZ.TXT
  danas je lep dan
  i Ja zelim
  da postanem programer
IZLAZ:
-1
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  prva.dat druga.dat l
PRVA.DAT
  Cena soka je 30
  Cena vina je 150
  Cena limunade je 200
  Cena sendvica je 120
IZLAZ:
0
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
  primer.c
PRIMER.C
#include <stdio.h>
int main()
{
}
PRAZNA.TXT

IZLAZ:
-1
```

[Rešenje 4.33]

**Zadatak 4.18** Prvi red datoteke `matrice.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice A. Svaki



sledeći red sadrži po jednu vrstu matrice. Napisati program koji pronalazi sve elemente matrice A koji su jednaki zbiru svih svojih susednih elemenata i štampa ih u obliku

(broj vrste, broj kolone, vrednost elementa).

U slučaju greške prilikom otvaranja datoteke ispisati -1. Pretpostaviti da je sadržaj datoteke ispravan.

#### Primer 1

```
MATRICE.TXT
1 2 3 4
7 2 15 -3
-1 3 1 3
IZLAZ:
(1, 0, 7)
(1, 2, 15)
```

[Rešenje 4.33]

**Zadatak 4.19** Napisati program koji za dve datoteke čija su imena data kao prvi i drugo na standardnom ulazu, radi sledeće: za cifru u prvoj datoteci, u drugu datoteku se upisuje 0, za slovo se upisuje 1, a za sve ostale karaktere se upisuje 2. Maksimalna dužina naziva datoteka je 20 karaktera.

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
prva.dat druga.dat
PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120
DRUGA.DAT
111121111121120021111211111211200021111211111111211200021111211111112112000
```

[Rešenje 4.33]

**Zadatak 4.20** Ako je data tekstualna datoteka `plain.txt` napraviti tekstualnu datoteku `sifra.txt` tako što se svako slovo zamenjuje svojim prethodnikom (ciklično) suprotne velicine 'b' sa 'A', 'B' sa 'a', 'a' sa 'Z', 'A' sa 'z', itd. Podrazumevati da se na sistemu koristi tabela karaktera ASCII.

[Rešenje 4.33]

**Zadatak 4.21** Sa standardnog ulaza se učitava ime tekstualne datoteke i prirodan broj k. Podrazumeva se da zadata datoteka sadrži samo slova i beline i

## 4 Ulaz i izlaz programa

da je svaka reč iz datoteke dužine najviše 100. Program treba da učitava reči iz datoteke, da svaku reč rotira za  $k$  mesta i da tako dobijenu reč upiše u datoteku čije je ime `rotirano.txt`. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.33]

**Zadatak 4.22** Napisati program koji u datoteku `izlaz.txt` prepisuje sve reči iz datoteke `ulaz.txt` čiji je zbir ascii kodova slova strogo veći od 1000. Reči su odvojene prazninama i nisu duže od 200 karaktera.

### Primer 1

```
ULAZ.TXT
Sa standardnog ulaza unosi se neoznaceni
ceo broj. Formirati novi broj koji se dobija
izbacivanjem svake druge cifre iz polaznog
broja.
IZLAZ.TXT
standardnog izbacivanjem
```

### Primer 2

```
ULAZ.TXT
i sada jedan kratak primer
p1: 1234567890
p2: ABCDEFGHIJ
p3: abcdefghij
IZLAZ.TXT
abcdefghij
```

### Primer 3

```
ULAZ.TXT
konstruisanje test-primer a
i dugackim recima kao prestolonaslednik
brojevima1234567890
IZLAZ.TXT
konstruisanje test-primer a
prestolonaslednik
brojevima1234567890
```

### Primer 4

```
ULAZ.TXT
ima jos dugackih reci: predskazanje,
potom
nelogicnosti, zanemarivati, odugovlaciti, a ima
i i malih reci koje su kratke
predosecaj
IZLAZ.TXT
predskazanje, nelogicnosti,
zanemarivati, odugovlaciti,
predosecaj
```

[Rešenje 4.33]

**Zadatak 4.23** U datoteci `razno.txt` nalazi se tekst. U datoteku `palindromi.txt` prepisati sve reči iz datoteke `razno.txt` koje su palindromi. Reč je palindrom ako se čita isto sa leve i desne strane. Za reč smatramo niz karaktera koji se nalazi između belina i koji nije duži od 200 karaktera. Dozvoljeno je korišćenje specifikatora za čitanje reči. Maksimalan broj reči nije poznat. U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

### Primer 1

```
RAZNO.TXT
Ana i melem su primeri palindroma.
PALINDROMI.TXT:
Ana i melem
```

### Primer 2

```
RAZNO.TXT
jabuka neven pomorandza kuk
Oko kapAk pero radar caj
PALINDROMI.TXT:
neven kuk\datoteka{Oko kapAk radar}
```

[Rešenje 4.33]

**Zadatak 4.24** U datoteci čije se ime navodi na standardnom ulazu programa nalazi se broj  $n$ , a zatim i  $n$  reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

(a) ispisuje ga [3],

(b) iz njega uklanja sve duplikate i u datoteku `rez.txt` ispisuje transformisani niz [4]

U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT
12 jha14 hahaha deda mraz deda
   mraz deda deda jase konj konj konj
IZLAZ:
jha14 hahaha deda mraz deda mraz deda
deda jase konj konj konj
REZ.TXT:
jha14 hahaha deda mraz jase konj
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
  dat2.txt
DAT2.TXT
14
so secer supa so ljuto secer kiselo slatko
ljuto
paprika, ljuta paprika, ljuto dete
IZLAZ:
so secer supa so ljuto secer kiselo slatko
ljuto paprika, ljuta paprika, ljuto dete
REZ.TXT:
so secer supa ljuto kiselo slatko
paprika, ljuta dete
```

[Rešenje 4.33]

**Zadatak 4.25** U datoteci čije se ime navodi na standardnom ulazu programa nalazi se broj  $n$ , a zatim i  $n$  reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

(a) ispisuje ga, [3]

(b) u datoteku `rez.txt` upisuje sve reči koje sadrže prvu reč i podvlatku. [4]

U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT
  7 rec Opet _rec Reci rec_enica
  DVa recica_
IZLAZ:
  rec Opet _rec Reci rec_enica
  DVa recica_
REZ.TXT:
  _rec rec_enica recica_
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  dat2.txt
DAT2.TXT
  11 Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
  suncanica.
IZLAZ:
  Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
  suncanica.
REZ.TXT:
  Sunce_Moje Sunce123_123
```

[Rešenje 4.33]

**Zadatak 4.26** Imena dve datoteke se zadaje na standardnom ulazu. U prvoj datoteci navedena je rec `r` i niz linija. Napisati program koji u drugu datoteku upisuje sve linije u kojima se reč `r` pojavljuje bar `n` puta, gde je `n` prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu `broj_pojavljivanja: linija`. Linije brojati počevši od 1. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.33]

**Zadatak 4.27** Napisati program koji poredi dva fajla i ispisuje redni broj linija u kojima se fajlovi razlikuju. Imena fajlova se zadaju kao argumenti komandne linije. U slučaju neuspešnog otvaranja datoteka ispisati poruku o grešci. Pretpostaviti da je maksimalna dužina reda u datoteci 200 karaktera. Ukoliko nisu zadati potrebni argumenti komadne linije ispisati poruku o grešci. Linije brojati počevši od 1.

### Primer 1

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt
ULAZ.TXT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ.TXT:
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ:
```

### Primer 2

```
POKRETANJE: ./a.out primer1.dat primer2.dat
PRIMER1.DAT
  danas vezbamo
  analizu
  ovo je primer kad
  su datoteke razlicite
PRIEMR2.DAT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke razlicite
IZLAZ:
  2 3 4
```

*Primer 3*

```

|| POKRETANJE: ./a.out prva.dat
|| IZLAZ:
|| greska

```

*Primer 2*

```

|| POKRETANJE: ./a.out prva.dat druga.dat
|| PRVA.DAT
||   ovo je primer
||   kada su
||   datoteke
||   razlicite duzine
|| DRUGA.DAT
||   kada su
||   programiranje
||   datoteke
||   razlicite
||   duzine
||   i kada treba ispisati broj
||   tih redova
|| IZLAZ:
|| 1 4 5 6 7

```

[Rešenje 4.33]

**Zadatak 4.28** Definirati strukturu

```

typedef struct{
    unsigned int a, b;
    char ime[5];
}_pravougaonik;

```

kojom se opisuje pravougaonik dužinama svojih stranica i imenom. Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava pravougaonike (nepoznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine među pravougaonicima koji nisu kvadrati. U slučaju unosa nekorektnih dužina stranica pravougaonika ili neko-rektne vrednosti broja *n*, ispisati -1 i odmah prekinuti izvršavanje programa. Maksimalan broj pravougaonika je 200.

*Primer 1*

```

|| POKRETANJE: ./a.out pravougaonici.dat
|| PRAVOUGAONICI.DAT
|| 2 4 p1
|| 3 3 p2
|| 1 6 p3
|| IZLAZ:
|| p2 8

```

*Primer 2*

```

|| POKRETANJE: ./a.out dva.dat
|| DVA.DAT
|| 5 2 pm
|| 4 7 pv
|| IZLAZ:
|| 28

```

### Primer 3

```
|| POKRETANJE: ./a.out tri.dat
|| TRI.DAT
|| 5 5 m
|| 3 3 s
|| 8 8 xl
|| IZLAZ:
|| m s xl
```

### Primer 4

```
|| POKRETANJE: ./a.out primerx.dat
|| PRIMERX.DAT
|| 9 7 p
|| IZLAZ:
|| 63
```

### Primer 5

```
|| POKRETANJE: ./a.out prazna.dat
|| PRAZNA.DAT
|| IZLAZ:
```

[Rešenje 4.33]

**Zadatak 4.29** Ime datoteke dato je kao argument komandne linije. U datoteci se nalaze otvorene i zatvorene zagrade i još nekakav tekst. Proveriti da li su zagrade pravilno uparene. Npr. `ab( cd) ..` odgovor je `jesu`, a `..)ba()` odgovor je `nisu`. Ukoliko nisu zadati svi argumenti komandne linije ispisati poruku o grešci.

### Primer 1

```
|| POKRETANJE: ./a.out
|| zagrade.txt
|| ZAGRADE.TXT
|| ab( cd) ..
|| ((3+4)*5+1)*9
|| IZLAZ:
|| jesu
```

### Primer 2

```
|| POKRETANJE: ./a.out
|| primer2.dat
|| PRIMER2.DAT
|| (7+8
|| nisu(
|| uparene
|| IZLAZ:
|| nisu
```

### Primer 3

```
|| POKRETANJE: ./a.out
|| primer3.dat
|| PRIMER3.DAT
|| )) 7 + 6 ((
|| IZLAZ:
|| nisu
```

### Primer 4

```
|| POKRETANJE: ./a.out
|| IZLAZ:
|| greska
```

[Rešenje 4.33]

**Zadatak 4.30** Napraviti strukturu `STUDENT` koja sadrži:

- ime (u polju se čuva ime i prezime studenta, napr. "Marko Markovic", maksimalna dužina polja je 100 karaktera),
- oc (sadrži najviše 10 ocena studenta)

- `br_ocena` (ukupan broj ocena za studenata)
- `pr_oc` (prosečna ocena)

U datoteci se nalaze podaci o studentima. Za svakog studenta unosi se ime i prezime razdvojeno razmakom (uputstvo: može se korisiti `strcat` da spoji ime i prezime koji se mogu pročitati sa specifikatorom `%s`), a potom ocene koje se završavaju sa 0. Pronaći studenta koji ima najveći prosek i ispisati sve njegove podatke (prosek ispisati na 2 decimale). Maksimalan broj studenta je 100. Ime datoteke se zadaje kao argument komandne linije.

*Primer 1*

```
|| POKRETANJE: ./a.out
   studenti.txt
|| STUDENTI.TXT
   Marko Markovic 5 6 7 8 9 0
   Jelena Jankovic 10 10 10 0
   Filip Viskovic 10 9 8 7 6 0
   Jana Peric 10 10 9 9 8 8 7 7
   0
|| IZLAZ:
   Jelena Jankovic 10 10 10 0
   10.00
```

*Primer 2*

```
|| POKRETANJE: ./a.out
   IZLAZ:
   greska
```

[Rešenje 4.33]

**Zadatak 4.31**

- Napisati C funkciju `int unesiSkup(char *s, FILE* f)` kojom se unosi skup elemenata iz datoteke `F`. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiđe na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspešno učitani.
- Napisati funkciju `void prebroj(char *s, int *br_slova, int *br_cifara)` kojom se određuje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.
- Napisati glavni program gde se unose podaci o skupu elemenata. Ime datoteke se zadaje kao argument komandne linije. Na standardni izlaz ispisati informacije o broju slova i cifara (koristiti funkcije pod a) i b)).

### Primer 1

```
POKRETANJE: ./a.out skup.txt
SKUP.TXT
  abc56ighj9012hjFGHH
IZLAZ:
  broj slova: 13
  broj cifara: 6
```

### Primer 2

```
POKRETANJE: ./a.out skup2.txt
SKUP2.TXT
  ovdeimamo$dolar
IZLAZ:
  broj slova: 9
  broj cifara: 0
```

### Primer 3

```
POKRETANJE: ./a.out skup3.txt
SKUP3.TXT
  broj3
  broj5
IZLAZ:
  broj slova: 4
  broj cifara: 1
```

### Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
  greska
```

[Rešenje 4.33]

### Zadatak 4.32 Definirati strukturu

```
typedef struct{
    int x;
    int y;
    int z;
} vektor;
```

kojom se opisuje trodimenzioni vektor. U datoteci **vektori.txt** nalazi se nepoznati broj vektora (maksimalno ih može biti 200). Učitati ih u niz i ispisuje na standardnom izlazu koordinate vektora sa najvećom dužinom. Dužina vektora se izračunava po formuli:

$$|v| = \sqrt{x^2 + y^2 + z^2}$$

U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

### Primer 1

```
VEKTORI.TXT
  2
  4 -1 7
  3 1 2
IZLAZ:
  4 -1 7
```

### Primer 2

```
VEKTORI.TXT
  67
IZLAZ:
  -1
```

### Primer 3

```
VEKTORI.TXT
  3
  0 0 0
  0 1 0
  1 0 0
IZLAZ:
  0 1 0
```



*Primer 4*

```

VEKTORI.TXT
4
3 0 1
4 5 2
1 0 0
2 -1 2
IZLAZ:
4 5 2

```

[Rešenje 4.33]

**Zadatak 4.33** Prvi red datoteke `ulaz.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice  $A$ . Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji nalazi i štampa sve četvorke oblika  $(A(i,j), A(i+1,j), A(i,j+1), A(i+1,j+1))$  u kojima su svi elementi međusobno različiti.

[Rešenje 4.33]

## 4.4 Rešenja

## Rešenje 4.1

```

1  /*
   Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
3  datoteku izlaz.txt karakter po karakter.
   */
5
7  #include <stdio.h>
   #include <stdlib.h>
9
11 int main()
   {
13     int c;
       FILE *ulaz, *izlaz;
15
17     /*
       Promenljive ulaz i izlaz predstavljaju
       pokazivace na ugradjenu strukturu FILE.
       Unutar ove strukture nalaze se polja neophodna
       za rad sa datotekama.
19
       Kada zelimo da radimo sa nekom datotekom,
       moramo je prvo otvoriti. Ugradjena funkcija
21

```

```
23     fopen(dat, mode) otvara datoteku sa nazivom
24     dat. Datoteka moze biti otvorena za citanje,
25     pisanje ili nadovezivanje, sto odredjuje
26     argument mode koji moze imati vrednost "r" (read),
27     "w"(write) ili "a"(append).
28 */
29
30     ulaz=fopen("ulaz.txt", "r");
31
32     /*
33     Do neuspesnog otvaranja datoteke moze doci
34     ukoliko ne postoji datoteka sa datim nazivom
35     ili je putanja do datoteke pogresna. U tom
36     slucaju, funkcija fopen vraca pokazivac na NULL
37     i tada treba prijaviti gresku. Datoteka stderr
38     predstavlja standardnu datoteku u koju se upisuju
39     greske. Stderr je podrazumevano postavljen
40     na standardni izlaz.
41
42     Ugradjena funkcija exit prouzrokuje zavrsetak programa.
43     Argument ove funkcije je jedna od konstanti definisanih
44     u biblioteci stdlib.h koje pokazuju da li se program
45     zavrrio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).
46
47     */
48     if(ulaz==NULL)
49     {
50         fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke ulaz
51         .txt za citanje.\n");
52         exit(EXIT_FAILURE);
53     }
54
55     izlaz= fopen("izlaz.txt", "w");
56     if(izlaz==NULL)
57     {
58         fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke
59         izlaz.txt za citanje.\n");
60         exit(EXIT_FAILURE);
61     }
62
63     /*
64     Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
65     Povratna vrednost ove funkcije je ascii kod unetog
66     karaktera.
67
68     Funkcija fputc ispisuje karakter c u datoteku izlaz.
69
70     */
71     while((c=fgetc(ulaz))!=EOF)
72         fputc(c, izlaz);
73
74     /*
```

```

73     Nakon zavrsetka rada sa datotekama, neophodno ih je
       zatvoriti pomocu ugradjene funkcije fclose.
75     */
       fclose(ulaz);
       fclose(izlaz);
77     return 0;
}

```

## Rešenje 4.2

```

/*
2   Napisati program koji u datoteci cije se ime navodi kao prvi
   argument komandne linije odredjuje liniju maksimalne duzine i
4   ispisuje je na standarni izlaz. Ukoliko ima vise takvih linija,
   ispisati onu koja je leksikografski prva. Mozemo pretpostaviti
6   da datoteka ne sadrzi linije duze od 80 karaktera.
   */
8   #include <stdio.h>
   #include <stdlib.h>
10  #include <string.h>
   #define MAX_LEN 81
12
   int main(int argc, char* argv[])
14  {
       char linija[MAX_LEN];
       char max_linija[MAX_LEN];
16       int duzina;
       int max_duzina;
18
       FILE *ulaz, *izlaz;
20
       /*
22        Proveravamo da li poziv programa ima dovoljan broj argumenata.
       */
24       if(argc!=2)
       {
26           fprintf(stderr, "Greska: program se pokrece sa: %s
           ime_ulazne_datoteke\n", argv[0]);
           exit(EXIT_FAILURE);
28       }
30
       ulaz=fopen(argv[1], "r");
32       if(ulaz==NULL)
       {
34           fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke %s
           za citanje.\n", argv[1]);
           exit(EXIT_FAILURE);
36       }
38       /*

```

```

    Funkcija fgets ucitava jednu liniju teksta maksimalne duzine
    MAX_LEN
40    iz datoteke ulaz u string linija. Ukoliko učitavanje ne uspe (
    na primer,
    zato sto smo dosli do kraja datoteke), povratna vrednost ove
    funkcije
42    bice prazan pokazivac (NULL).
    */
44
    max_duzina=0;
46    while(fgets(linija, MAX_LEN, ulaz)!=NULL)
    {
48        duzina = strlen(linija);
        /*
50        Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u
        petlju.
        Ovu promenljivu menjamo kada je duzina učitana linije
52        veca od max_duzina ili kada su jednake, ali je učitana
        linija
        leksikografski ispred trenutne linije sa maksimalnom duzinom
        .
54
        Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii
        kodova prva dva
56        razlicita karaktera stringova s1 i s2 na istim indeksima,
        ukoliko oni
        postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga
        osetljiva
58        na mala i velika slova (npr 'D' je leksikografski ispred 'p
        ').
        */
60
        if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija,
        max_linija)<0))
        {
64            strcpy(max_linija, linija);
            max_duzina=duzina;
66        }
    }
68
    /*
70    Funkcija fputs ispisuje string koji je njen prvi argument u
    datoteku
    koja je njen drugi argument. Sve funkcije za učitavanje iz
    datoteka i
72    upis u datoteke (fgetc, fputc, fgets, fputs, ...) mozemo
    koristiti
    i kada radimo sa standardnim ulazom i standardnim izlazom. Kao
    nazive
74    datoteka tada navodimo stdin i stdout.
    */

```

```
76     fputs(max_linija, stdout);
77
78     fclose(ulaz);
79     return 0;
80 }
```

### Rešenje 4.3

```
/*
2   U datoteci cije se ime zadaje kao prvi argument komandne linije
   nalazi se
   prirodan broj n a zatim i n celih brojeva. Napisati program koji
   prebrojava
4   koliko k-tocifrenih brojeva postoji u datoteci, pri cemu se
   prirodan broj k
   zadaje kao drugi argument komandne linije.
6 */

8 #include <stdio.h>
9 #include <stdlib.h>
10 #include <math.h>

12 /*
   Funkcija ucitaj_i_prebroj ucitava brojeve
14   iz datoteke na koju pokazuje f i prebrojava
   koliko je medju njima k-tocifrenih brojeva
16 */
17 int ucitaj_i_prebroj(FILE* f, int k)
18 {
19     int n;
20     int x;
21     int i;
22     int br;

24     /* U datoteci je prvo naveden ukupan broj brojeva. */
25     fscanf(f, "%d", &n);

26     /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
       */
27     if(n <= 0)
28     {
29         fprintf(stderr, "Greska: broj n mora biti prirodan\n");
30         exit(EXIT_FAILURE);
31     }

32     br=0;
33     for(i=0; i<n; i++)
34     {
35         fscanf(f, "%d", &x);
36         if(broj_cifara(x)==k)
37             br++;
38     }
```

```
40     }
42     return br;
43 }
44
45 int broj_cifara(int x)
46 {
47     int br_c;
48
49     br_c=0;
50
51     /*
52      Do while petlja je pogodnija od petlji sa preduslovom
53      jer tacno racuna broj cifara i za broj 0.
54     */
55     do
56     {
57         br_c++;
58         x/=10;
59     } while(x);
60
61     return br_c;
62 }
63
64 int main(int argc, char* argv[])
65 {
66     int n;
67     int k;
68     FILE* f;
69     int br;
70
71     if(argc!=3)
72     {
73         fprintf(stderr, "Greska: program se pokrece sa: %s
74         naziv_datoteke k \n", argv[0]);
75         exit(EXIT_FAILURE);
76     }
77
78     f=fopen(argv[1], "r");
79
80     if(f==NULL)
81     {
82         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
83         \n", argv[1]);
84         exit(EXIT_FAILURE);
85     }
86
87     /* Argumenti komandne linije su stringovi. Da bismo konvertovali
88     string
89     u ceo broj koristimo ugradjenu funkciju atoi. */
90     k = atoi(argv[2]);
```

```

    if (k<=0)
90  {
        fprintf(stderr, "Greska: broj k mora biti prirodan\n");
92      exit(EXIT_FAILURE);
    }

94      printf("Broj %d-cifrenih brojeva u datoteci: %d\n", k,
        ucitaj_i_prebroj(f,k));

96      fclose(f);
98      return 0;
}

```

### Rešenje 4.4

```

1  /*
    U datoteci cije se ime navodi kao prvi argument komandne
3  linije navedena je rec r i niz linija. Napisati
    program koji u datoteku cije se ime navodi kao
5  drugi argument komandne linije upisuje sve linije
    u kojima se rec r pojavljuje bar n puta, gde je
7  n prirodan broj koji se unosi sa standardnog ulaza. Ispis
    treba da bude u formatu broj_pojavljivanja: linija.
9  */

11 #include <stdio.h>
12 #include <stdlib.h>
13 #define MAXL 81
14 #define MAXR 31
15
16 /*
17  Funkcija broj_pojavljivanja broji koliko
    se puta pojavio string t u stringu s
19  */
20 int broj_pojavljivanja(char s[], char t[])
21 {
    int br;
23     int i,j;
    /*
25     i - indeks karaktera u s
        j - indeks karaktera u t
27     br - brojac koliko se puta t javlja u s
    */
29     br=0;
    for(i=0;s[i];i++)
31     {
        for(j=0;t[j];j++)
33         if(s[i+j]!=t[j]) /* Ako naidjemo na razlicite karaktere, */
            break;        /* prekidamo petlju. */
35     }

    /*
        Do prekida petlje moze doci ili zbog toga sto su pronadjeni

```

```
37         razliciti karakteri i usledio je break ili zbog toga sto
39         je prestao da vazi uslov petlje, odnosno karakter t[j] je
        jednak '\0'. Ako vazi drugi slucaj, to znaci da se string
        t nalazi u stringu s pocev od indeksa i i potrebno je
        uvecati
41         brojac br.
        */
43         if (t[j]!='\0')
            br++;
45     }

47     return br;
}
49 int main(int argc, char* argv[])
{
51     char rec[MAXR];
52     char linija[MAXL];
53     FILE* in, *out;
54     int n;
55     int br;

57     if(argc!=3)
    {
59         fprintf(stderr, "Greska: program se pokrece sa: %s
        ime_ulazne_datoteke ime_izlazne_datoteke\n", argv[0]);
        exit(EXIT_FAILURE);
61     }

63     in= fopen(argv[1], "r");
    if(in==NULL)
65     {
        fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", argv[1] );
        exit(EXIT_FAILURE);
67     }

69     out= fopen(argv[2], "w");
    if(out==NULL)
71     {
        fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", argv[2] );
        exit(EXIT_FAILURE);
73     }

75     printf("Unesi n:");
    scanf("%d", &n);

77     if(n<=0)
    {
81         fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
        exit(EXIT_FAILURE);
83     }
```



```

85     fscanf(in, "%s", rec);
87
88     while(fgets(linija, MAXL, in) != NULL)
89     {
90         br = broj_pojavljivanja(linija, rec);
91         if (br >= n)
92             fprintf(out, "%d: %s\n", br, linija);
93     }
94     fclose(in);
95     fclose(out);
96     return 0;
97 }

```

### Rešenje 4.5

```

1  /* Program se pokrece tako sto se navedu nazivi dve datoteke(ulazna i
   izlazna) i opcije.
   U datoteci cije se ime navodi kao prvi argument komandne linije
   nalaze se podaci o razlomcima:
3  u prvom redu se nalazi broj razlomaka, a u svakom sledecem redu
   brojilac i imenilac jednog razlomka.
   Potrebno je kreirati strukturu koja opisuje razlomak i ucitati niz
   razlomaka
5  iz datoteke, a potom:
   a) ukoliko je navedena opcija x, upisati u datoteku cije je ime
   drugi argument komandne linije
7   recipročni razlomak za svaki razlomak iz niza (npr. za 2/3
   treba upisati 3/2)
   b) ukoliko je navedena opcija y, upisati u datoteku cije je ime
   drugi argument komandne linije
9   realnu vrednost reciprocnog razlomka svakog razlomka iz niza
   (npr. za 2/3 treba upisati 1.5)
   Mozemo pretpostaviti da se u datoteci sa podacima o razlomcima
   nalazi najviše 100 razlomaka.
11 */
13 /*
   Prilikom pokretanja programa se, pored naziva ulazne i izlazne
   datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
   obe opcije, sto znaci da je minimalni broj argumenata 3.
17
   Moguci nacini pokretanja:
19 ./a.out ulaz.txt izlaz.txt -x
   ./a.out ulaz.txt izlaz.txt -y
21 ./a.out ulaz.txt izlaz.txt -yx
   ./a.out ulaz.txt izlaz.txt -xy
23
24 */
25 #include <stdio.h>

```

```
27 #include <stdlib.h>
   #include <ctype.h>
29
   #define MAX 100
31
   typedef struct razlomak
33 {
       int br;
35       int im;
   } RAZLOMAK;
37
   /*
39    Funkcija ucitaj_razlomke ucitava razlomke iz datoteke
       na koju pokazuje f u niz. Dimenzija niza, na koju
41    pokazuje pokazivac dim, nije poznata. Prva vrednost
       u datoteci je ukupan broj razlomaka i tu vrednost
43    ucitavamo u promenljivu dim.

45    Funkcija fscanf se koristi isto kao i funkcija scanf
       uz dodatni prvi argument koji predstavlja naziv
47    datoteke iz koje se vrsi ucitavanje.

49    */
   int ucitaj_razlomke(RAZLOMAK niz[], int* dim, FILE* f)
51 {
       int i;
53
       fscanf(f, "%d", dim);
55       for (i=0; i<*dim; i++)
       {
           fscanf(f, "%d %d", &niz[i].br, &niz[i].im);
57           if (niz[i].im==0)
59               return 0;
       }
61       return 1;
   }
63
   RAZLOMAK reciprocni(RAZLOMAK* r)
65 {
       RAZLOMAK rec;
67       rec.im = r->br;
       rec.br = r->im;
69       return rec;
   }
71
   float vrednost(RAZLOMAK* r)
73 {
       return 1.0*r->br/r->im;
75 }

77 int main(int argc, char* argv[])
   {
```

```
79 FILE *in, *out;
   char c;
81 int i;
   int j;
83 int xoption=0;
   int yoption=0;
85 int dim;
   RAZLOMAK razlomci[MAX];
87 RAZLOMAK r;

89 /*
   Prilikom pokretanja programa se, pored naziva ulazne i izlazne
91 datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
   obe opcije, sto znaci da je minimalni broj argumenata 3.

93
   Moguci nacini pokretanja:
95 ./a.out ulaz.txt izlaz.txt -x
   ./a.out ulaz.txt izlaz.txt -y
97 ./a.out ulaz.txt izlaz.txt -yx
   ./a.out ulaz.txt izlaz.txt -xy
99
101 */
   if(argc!=4)
103 {
       fprintf(stderr, "Greska: program se pokrece sa: %s
       ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n", argv[0]);
105       exit(EXIT_FAILURE);
   }

107
109 in= fopen(argv[1], "r");
   if(in==NULL)
111 {
       fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
       .\n", argv[1] );
113       exit(EXIT_FAILURE);
   }

115
   out= fopen(argv[2], "w");
117 if(out==NULL)
   {
119     fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
       .\n", argv[2] );
       exit(EXIT_FAILURE);
121 }

123 /* Ispitujemo da li je treca opcija u trazenom formatu. Prvi
   karakter mora biti '-' */

125 if (argv[3][0] != '-')
   {
```

```

127     fprintf(stderr, "Greska u zadavanju opcija: program se pokrece
sa: %s ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n",
argv[0]);
    exit(EXIT_FAILURE);
129 }

131 /* Ostali karakteri mogu biti 'x' ili 'y'. U zavisnosti od date
opcije, postavljamo
vrednosti indikatorskih promenljivih xoption i yoption. */
133
135 for(j=1;argv[3][j]!='\0';j++)
    switch(argv[3][j])
    {
137         case 'x': xoption=1;
                    break;
139         case 'y': yoption=1;
                    break;
141         default:
                    fprintf(stderr, "Greska: nedozvoljeni karakter\n"
);
                    exit(EXIT_FAILURE);
    }
143
145
147 if(ucitaj_razlomke(razlomci, &dim, in)==0)
{
149     fprintf(stderr, "Greska pri zadavanju razlomaka\n");
    exit(EXIT_FAILURE);
151 }

153 /*
    U zavisnosti od datih opcija, vrsimo upis reciprocnih
155     razlomaka u trazenom formatu.

    Funkcija fprintf se koristi na isti nacin kao
157     funkcija printf uz dodatni prvi argument koji
159     oznacava naziv datoteke u koju se vrši upis.
*/
161 for (i=0; i<dim;i++)
{
163     /*
        Ukoliko je brojilac razlomka jednak nuli,
165         nema smisla traziti njegovu reciprocnu vrednost
        */
167     if (razlomci[i].br==0)
        continue;

169     r = reciprocni(&razlomci[i]);

171     if (xoption)
173         fprintf(out,"%d/%d ", r.br, r.im);

```

```

175     if (yoption)
176         fprintf(out, "%f ", vrednost(&r));
177
178     fprintf(out, "\n");
179 }
180
181 fclose(in);
182 fclose(out);
183
184 return 0;
185 }

```

### Rešenje 4.6

```

1  /*
2   Za svaki automobil poznati su marka, model i cena. Iz datoteke cije
3   se ime zadaje sa standardnog ulaza učitava se broj automobila a
4   potom
5   i podaci za svaki automobil. Program treba da:
6   a) izracuna prosečnu cenu po marki kola
7   b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
8   zadaje
9   kao argument komandne linije, da ispise automobile u tom cenovnom
10  rang u zajednu sa prosečnom cenom odgovarajuće marke
11
12  Mozemo pretpostaviti da se model i marka sastoje od jedne reci i
13  da svaka od njih sadrži najviše 30 karaktera kao i da se u datoteci
14  nalaze podaci za najviše 100 automobila.
15
16  */
17
18  #include <stdio.h>
19  #include <stdlib.h>
20  #include <string.h>
21  #define MAX 31
22  #define MAXA 100
23
24  typedef struct automobil
25  {
26      char marka[MAX];
27      char model[MAX];
28      float cena;
29  } AUTOMOBIL;
30
31  /*
32   Struktura INFO sadrži naziv
33   marke automobila, prosek cena
34   za tu marku i broj automobila
35   te marke
36  */
37  typedef struct info

```

```

{
37     char marka[MAX];
    float vrednost;
39     int n;
} INFO;

41
int ucitaj_podatke(FILE* f, AUTOMOBIL a[], int* pn, int max)
43 {
    int i;
45
    fscanf(f, "%d", pn);
47     if (*pn <= 0 || *pn > max)
    {
49         printf("Nekorektan unos dimenzije niza automobila\n");
        return 0;
51     }
    for(i=0; i < *pn; i++)
53         fscanf(f, "%s %s %f", a[i].marka, a[i].model, &a[i].cena);

55     return 1;
}

57
/*
59     Funkcija sadrzi ispituje da li se u nizu proseka po marki
    nalazi prosek za marku m. Posto podatak o marki automobila
61     predstavlja string, poredjenje vrsimo pomocu funkcije strcmp.

63     Povratna vrednost ove funkcije je indeks pojavljivanja, ukoliko
    se marka m pojavljuje u nizu proseka, ili -1 u suprotnom.
65 */

67 int sadrzi(INFO p[], int n, char m[])
{
69     int i;
    for(i=0; i < n; i++)
71         if(strcmp(p[i].marka, m) == 0)
            return i;

73     return -1;
75 }

77
/*
79     Funkcija informacije_o_markama za niz automobila a dimenzije n
    racuna proseke cena automobila po markama i smesta ih u niz
    p. Na dimenziju niza p pokazuje pokazivac pn.

81
    Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
83     sumu cena automobila te marke (koju cemo smestiti u polje vrednost
    strukture
    INFO), i broj automobila te marke (koju cemo smestiti u polje
85     n strukture INFO) i da na kraju podelimo ove dve promenljive
    i tako dobijemo prosečnu vrednost cene.

```

```

87      Za svaki automobil a[i] proveravamo da li se njegova marka vec
89      nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
91      vredost cene automobila a[i] i uvecavamo broj automobila sa
93      tom markom. U suprotnom, dodajemo novi element u niz p. Posto
      ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
      na koju pokazuje pokazivac *pn.
*/
95 void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
96 {
97     int i,j;
98     int ind;
99     for(i=0;i<n;i++)
100     {
101         /* Proveravamo da li se marka automobila a[i] vec nalazi u
102            nizu p (niz proseka po markama) */
103         ind = sadrzi(p,*pn1,a[i].marka);
104         if(ind==-1) /* Ako se ne nalazi, uvodimo novi element niza na
105            kraj, na poziciju *pn. */
106         {
107             strcpy(p[*pn1].marka, a[i].marka);
108             p[*pn1].vrednost = a[i].cena;
109             p[*pn1].n = 1;
110             (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
111         */
112         }
113         else /* Ako se nalazi, azuriramo polja strukture. */
114         {
115             p[ind].vrednost+=a[i].cena;
116             p[ind].n++;
117         }
118     }
119     /* Na osnovu sume cena i broja automobila racunamo prosečnu
120        vrednost. */
121     for(i=0;i<*pn1;i++)
122         p[i].vrednost = p[i].vrednost/p[i].n;
123 }
124
125 void stampaj_informacije(INFO p[], int n)
126 {
127     printf("Informacije o broju automobila i prosečnoj ceni po markama
128     :\n");
129     int i;
130     for(i=0;i<n;i++)
131         printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
132 }
133
134 /*
135 Funkcija stampa automobile cija je cena manja od maksimalne
136 cene koju je korisnik naveo u komandnoj liniji da je spreman

```

```
135     da plati, zajedno sa prosecom cenom za tu marku automobila
136     */
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
    n1)
138 {
139     /*
140     S obzirom da je niz p formiran na osnovu niza a, marka svakog
141     automobila iz niza a se sigurno nalazi u nizu p. Zbog toga
142     nije neophodno proveravati da li je povratna vrednost funkcije
143     sadrzi razlicita od -1.
144     */
145     int i;
146     printf("Kola u vasem cenovnom rangu:\n");
147     for(i=0;i<n;i++)
148         if(a[i].cena<g)
149             printf("%s %s %.2f\n", a[i].marka, a[i].model, p[sadrzi(p,n1
    ,a[i].marka)].vrednost);
150 }
151
152 int main(int argc, char* argv[])
153 {
154     AUTOMOBIL kola[MAXA];
155     FILE* f;
156     char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
    ulaza. */
157     float granica; /* Maksimalna cena koju je korisnik spreman da
    plati.
158                     Zadaje se kao argument komandne linije.
159                     */
160     INFO infos[MAXA];
161     int dim_kola,dim_infos;
162     int i;
163
164     if(argc!=2)
165     {
166         fprintf(stderr,"Greska: program se pokrece sa: %s
    gornja_granica_cene \n", argv[0]);
167         exit(EXIT_FAILURE);
168     }
169
170     /* Argumenti komandne linije su stringovi. Da bismo od stringa
    dobili
171     realan broj, koristimo ugradjenu funkciju atof. */
172     granica = atof(argv[1]);
173
174     printf("Unesi naziv datoteke:");
175     scanf("%s", dat);
176
177     f=fopen(dat, "r");
178
179     if(f==NULL)
180     {
```



```

181     fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
      .\n", dat);
      exit(EXIT_FAILURE);
183 }

185 if (ucitaj_podatke(f,kola,&dim_kola,MAXA)==0)
    {
187     fprintf(stderr, "Greska pri učitavanju podataka\n");
      exit(EXIT_FAILURE);
189     }

191     informacije_o_markama(kola, dim_kola, infos, &dim_infos);
193     stampaj_informacije(infos,dim_infos);
195     stampaj_kandidate(kola, dim_kola, granica, infos, dim_infos);
197     fclose(f);
199     return 0;
    }

```

#### Rešenje 4.7

```

1  /* Napisati program koji prebrojava mala slova u datoteci test.txt */
3  #include<stdio.h>
5  int main(){
7      FILE* in;
      int c, broj_malih=0;
9
      /*Otvaramo datoteku test.txt za citanje i proveravamo da li smo je
        uspesno otvorili*/
11     in = fopen("test.txt", "r");
      if(in == NULL){
13         printf("Greska!");
        return 0;
15     }

17     /*Citamo karakter po karakter, i ukoliko je procitani
        *karakter malo slovo, uvecevamo brojac*/
19     while((c=fgetc(in))!=EOF){
        if(islower(c))
21         broj_malih++;
    }
23
25     /*Ispisujemo rezultat*/
    printf("Broj malih slova je: %d\n", broj_malih);

```

## 4 Ulaz i izlaz programa

---

```
27  /*Zatvaramo datoteku*/  
    fclose(in);  
29  
    return 0;  
31 }
```

### Rešenje 4.8

```
/* Napisati program koji prepisuje svaki treci karakter datoteke ulaz  
:txt u datoteku izlaz.txt */  
2  
#include<stdio.h>  
4  
int main(){  
6  
    FILE *in, *out;  
8    int c;  
    int rbr_karaktera;  
10  
12    /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je  
        uspesno otvorili*/  
    in = fopen("ulaz.txt", "r");  
14    if(in == NULL){  
        printf("Greska!");  
16    return 0;  
    }  
18  
    /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo dali smo je  
        uspesno otvorili*/  
20    out = fopen("izlaz.txt", "w");  
    if(out == NULL){  
22    printf("Greska!");  
        return 0;  
24    }  
26  
    /* Inicijalizujemo redni broj karaktera koji se cita */  
    rbr_karaktera=0;  
28  
    /* Citamo karakter po karakter iz datoteke sve dok ne stignemo do  
        kraja datoteke */  
30    while((c=fgetc(in)) != EOF){  
32  
        /* Ukoliko je procitani karakter na poziciji koja je deljiva sa 3  
            prepisujemo ga */  
        if(rbr_karaktera%3==0)  
34            fputc(c, out);  
36  
        /* Uvecavamo redni broj karaktera */  
        rbr_karaktera++;  
38    }  
}
```

```

40  /*Zatvaramo obe datoteke koje smo otvorili*/
    fclose(out);
42  fclose(in);
    return 0;
44  }

```

### Rešenje 4.9

```

/* Kao argumenti komandne linije se zadaju ime datoteke i ceo broj k.
   Napisati program koji na standardni izlaz
2  ispisuje sve linije zadate datoteke cija je duzina veka od k. Moze se
   pretpostaviti da duzina linije nece biti veka
   od 80 karaktera */
4
#include<stdio.h>
6  #include<string.h>
8  #define MAXL 81
10 int main(int argc, char* argv[]){
12     FILE* in;
    char linija[MAXL];
14     int k;
16     /*Proveravamo da li je program ispravno pozvan*/
    if(argc!=3){
18         printf("Greska: pogresan broj argumenata!");
        return 0;
20     }
22     /*Otvaramo za citanje datoteku koja se navodi kao prvi argument
        komandne linije*/
    in = fopen(argv[1], "r");
24     if(in == NULL){
        printf("Greska: neuspesno otvaranje datoteke!");
26         return 0;
    }
28     /*Uzimamo brojevnu vrednost drugog argumenta komandne linije*/
    k = atoi(argv[2]);
30
32     /*Citamo liniju po liniju i sve linije duze od k ispisujemo na
        standardni izlaz*/
    while(fgets(linija, MAXL, in) != NULL){
34         if(strlen(linija) > k)
            printf("%s", linija);
36     }
    printf("\n");
38

```

```
/*Zatvaramo datoteku*/
40 fclose(in);
    return 0;
42 }
```

### Rešenje 4.10

```
/* Napisati program koji prebrojava koliko se linija datoteke ulaz.
   txt završava niskom s koja se učitava sa standardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća
   od 80 karaktera, kao i da dužina niske s
   ne će biti veća od 20 karaktera */

4
#include<stdio.h>
6 #include <string.h>
#define MAXL 81
8 #define MAXS 21

10 /*Funkcija brojLinija proverava koliko linija u datoteci in se
   završava niskom s.
   Funkcija radi tako što čita jednu po jednu liniju iz datoteke,
   i zatim kraj te linije poredi sa niskom s.*/
12 int brojLinija(FILE* in, char* s){
14
    char linija[MAXL];
16     int broj_linija = 0;
    int dužina_s = strlen(s);
18     int dužina_linije;

20     while(fgets(linija, MAXL, in) != NULL){
        dužina_linije = strlen(linija);
22
        /* Ukoliko je znak za novi red bio indikacija kraja linije,
           uklanjamo ga kako bi mogli da izvršimo
           *ispravno poredjenje (jer niska s nema novi red na kraju) */
24         if(linija[dužina_linije-1]=='\n'){
26             linija[dužina_linije-1] = '\0';
            dužina_linije--;
28         }

30         /*linija + dužina_linije će nas odvesti na kraj tog stringa, a kada
           oduzmemo dužinu stringa s,
           a kada od toga oduzmemo dužinu niske s, dobićemo bas onoliko
           poslednjih karaktera, koliko
           nam i treba. U primeru uspravna crta (|) označava pokazivač
           s                ab
           dužina_s                2
           Linija:                aaabbbdfssab
36             |
           Linija + dužina linije aaabbbdfssab
38             |
```

```

        Linija + duzina linije - 2 aaabbbdfssab
40      |
        kada kazemo strcmp(linija + duzina_linije - duzina_s, s), mi
        cemo u nasem primeru zaista porediti "ab" i "ab".
42    */
    if(strcmp(linija + duzina_linije - duzina_s, s) == 0)
44        broj_linija++;
    }
46    return broj_linija;
}
48
49    int main(){
50
51        FILE* in;
52        char s[MAXS];
53
54        /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
        uspesno otvorili*/
55        in = fopen("ulaz.txt", "r");
56        if(in == NULL){
57            printf("Greska: neuspesno otvaranje datoteke!\n");
58            return 0;
59        }
60
61        /*Ucitavamo nisku*/
62        printf("Unesite nisku s: ");
63        scanf("%s", s);
64
65        /*Ispisujemo koliko linija iz datoteke se zavrшава sa niskom s*/
66        printf("Broj linija: %d\n", brojLinija(in, s));
67
68        /*Zatvaramo datoteku*/
69        fclose(in);
70
71        return 0;
72    }

```

### Rešenje 4.11

```

/* Napisati program koji pronalazi maksimum brojeva zapisanih u
   datoteci brojevi.txt */
2
#include<stdio.h>
4
int main(){
6
    FILE* in;
8    float broj, max_broj;
10
    /*Otvaramo datoteku brojevi.txt za citanje i proveravamo da li smo
    je uspesno otvorili*/

```

## 4 Ulaz i izlaz programa

```
12  in = fopen("brojevi.txt", "r");
13  if(in == NULL){
14      printf("Greska pri otvaranju datoteke!");
15      return 0;
16  }
17
18  /*
19   Kako bismo inicijalizovali promenljivu max_broj,
20   citamo jedan broj iz datoteke i smestamo ga u
21   ovu promenljivu. */
22  fscanf(in, "%f", &max_broj);
23
24  /*U petlji citamo sve ostale brojeve i poredimo ih sa trenutnim
25   maksimumom.*/
26  while(fscanf(in, "%f", &broj) != EOF){
27      if(broj > max_broj)
28          max_broj = broj;
29  }
30
31  /*Ispisujemo rezultat*/
32  printf("Najveci broj je: %.2f\n", max_broj);
33
34  /*Zatvaramo datoteku brojevi.txt*/
35  fclose(in);
36
37  return 0;
38 }
```

### Rešenje 4.12

```
1  /* U datoteci studenti. txt se nalaze informacije o studentima: prvo
2   broj studenata, a zatim u pojedinacnim linijama
3   korisnicko ime i pet poslednjih ocena koje je student dobio. Napisati
4   program koji pronalazi studenta koji je
5   ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da
6   broj studenata nece biti veci od 100. */
7
8  #include<stdio.h>
9
10 #define MAXS 100
11
12 /*Definisemo strukturu za cuvanje studenata*/
13 typedef struct st{
14     char korisnicko_ime[8];
15     float prosek;
16 }STUDENT;
17
18 int main(){
19
20     FILE *ulaz;
21     STUDENT studenti[MAXS];
```

```

19  int ocena1,ocena2,ocena3,ocena4,ocena5, i=0, i_max_prosek;
21  float max_prosek = 0;

23  /*Otvaramo datoteku studenti.txt za citanje*/
    ulaz = fopen("studenti.txt", "r");
25  if(ulaz == NULL){
    printf("Greska pri otvaranju datoteke!\n");
27  return 0;
    }

29  /*Ucitavamo liniju po liniju iz datoteke, sve dok ne dodjemo do
    kraja.
31  Korisnicko ime smestamo u niz, a ocene ucitavamo u pomocne
    promenljive ocena1,...ocena5.
    Zatim, na osnovu ocena racunamo prosek.

33  Ovdje paralelno sa ucitavanjem pronalazimo i studenta sa najvećim
    prosekom i
35  pamtimo njegov prosek i njegovu poziciju u nizu studenata,
    Nismo morali ovako. Mogli smo i prvu da ucitamo sve studente, a
    zatim da prodjemo
37  jednom kroz niz i da nadjemo onog sa najvećim prosekom.

39  */
    while(fscanf(ulaz, "%s%d%d%d%d", studenti[i].korisnicko_ime, &
        ocena1, &ocena2, &ocena3, &ocena4, &ocena5) != EOF){
41  studenti[i].prosek = (ocena1 + ocena2 + ocena3 + ocena4 + ocena5)
        /5.0;

43  if(studenti[i].prosek > max_prosek){
        max_prosek= studenti[i].prosek;
45  i_max_prosek = i;
    }

47  i++;
    }

49  /*Ispisujemo rezultat*/
51  printf("korisnicko ime: %s, prosek ocena: %.2f\n", studenti[
    i_max_prosek].korisnicko_ime, studenti[i_max_prosek].prosek);

53  /*Zatvaramo datoteku*/
    fclose(ulaz);

55  return 0;
57 }

```

### Rešenje 4.13

### Rešenje 4.14

```
1  /* Napisati program koji za rec s maksimalne duzine 20 karaktera koja
   se zadaje sa standardnog ulaza u datoteku
   rotacije.txt upisuje sve rotacije reci s */
3
4  #include<stdio.h>
5  #include<string.h>
6
7  #define MAXS 21
8
9  /*Funkcija rotira nisku za jedno mesto u desno.
   Duzina niske n nije obavezan argument. Mogli smo
11 i da je racunamo u okviru funkcije, ali kako ce sve niske
   sa kojima radimo biti iste duzine, efikasnije je da jednom
13 izracunamo tu duzinu u glavnom programu,
   pa da je prosledjujemo kao argument.*/
15 void rotiraj_zal(char* s, int n){
    int i;
17     char c = s[0];
    for(i=0; i<n-1; i++){
19         s[i] = s[i+1];
    }
21     s[n-1] = c;
    }
23
24 int main(){
25
26     char s[MAXS];
27     int n, i;
28     FILE * izlaz;
29
30     /*Otvaramo datoteku rotacije.txt za pisanje i proveravamo da li smo
       je uspesno otvorili*/
31     izlaz = fopen("rotacije.txt", "w");
    if(izlaz == NULL){
33         printf("Greska pri otvaranju fajla!");
        return 0;
35     }
36
37     /*Sa standardnog ulaza učitavamo rec koju treba da rotiramo*/
    scanf("%s", s);
39
40     /*Racunamo njenu duzinu*/
41     n = strlen(s);
42
43     /*U petlji, ispisujemo tu rec u datoteku, pa je rotiramo za jedno
       mesto u desno.*/
    for(i=0; i<n; i++){
45         fprintf(izlaz, "%s\n", s);
        rotiraj_zal(s,n);
47     }
```



```

49  /*Zatvaramo datoteku rotacije.txt*/
    fclose(izlaz);
51
    return 0;
53
}

```

### Rešenje 4.15

```

1  /* Napisati program koji linije koje se ucitavaju sa standardnog
    ulaza sve do kraja ulaza prepisuje u datoteku
    izlaz.txt i to, ako je prilikom pokretanja zadata opcija -v ili -V
        samo one linije koje pocinju velikim slovom,
3  ako je zadata opcija -m ili -M samo one linije koje pocinju malim
        slovom, a ako je opcija izostavljena sve linije.
    Pretpostaviti da linije nece biti duze od 80 karaktera.
5  */

7  #include<stdio.h>
    #include<string.h>
9  #include<ctype.h>

11 #define MAXL 81

13 int main(int argc, char* argv[]){

15     char linija[MAXL];
    FILE* izlaz;

17     /*Indikatori koji oznacavaju koja opcija je navedena kao argument
        komandne linije
19     vind - ispisuju se recenice koje pocinju velikim slovom
        mind - ispisuju se recenice koje pocinju malim slovom
21     */
    int vind=0, mind = 0;

23     /*Proveravamo da li je program ispravno pozvan*/
    if(argc > 2){
25         printf("Greska pri pozivanju programa!\n");
27         return 0;
    }

29     /*Ako opcije nisu zadate, onda treba da se ispisuju sve recenice,
        pa postavljamo oba indikatora na 1*/
31     if(argc == 1){
        vind = mind = 1;
33     }else{

35     /*Proveravamo da li je postavljena neka od opcija -v,-V,-m, -M
        Ako jeste, postavljamo odgovarajuci indikator
37     Ako nije, onda ispisujemo poruku o gresci*/

```

```
39     if(strcmp(argv[1], "-v") == 0 || strcmp(argv[1], "-V") == 0)
        vind = 1;
41     else if(strcmp(argv[1], "-m") == 0 || strcmp(argv[1], "-M") == 0)
        mind = 1;
43     else{
        printf("Greska pri zadavanju opcije!\n");
        return 0;
45     }
    }

47

49     /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo da li smo je
        uspesno otvorili*/
    izlaz = fopen("izlaz.txt", "w");
51     if(izlaz == NULL){
        printf("Greska pri otvaranju datoteke!\n");
53     return 0;
    }

55

    /*Citamo liniju po liniju sa standardnog ulaza i ispisujemo je u
        datoteku.
57     Liniju ispisujemo ukoliko je ispunjen neki od dva uslova:
        1. Izabrana je opcija za ispis malih slova i linija pocinje malim
            slovom
59     2. Izabrana je opcija za velika slova i linija pocinje velikim
            slovom
        NAPOMENA: Kada dodje do kraja ulaza, funkcija fgets vraca NULL
61     */
    while(fgets(linija, MAXL, stdin) != NULL){
63     if( mind && islower(linija[0]) || vind && isupper(linija[0]) ||
        mind && vind)
        fputs(linija, izlaz);
65     }

67     /*Zatvaramo datoteku izlaz.txt*/
    fclose(izlaz);

69     return 0;
71 }
```

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33



# 5

## Razni zadaci

### 5.1 Rešenja