PROGRAMIRANJE 1

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

PROGRAMIRANJE 1
Zbirka zadataka

Beograd
2016.

Autori:
dr Milena Vujošević Janičić, 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

Sadržaj

1

Uvodni zadaci

1.1 Naredba izraza

Zadatak 1.1.1 Napisati program koji na standardni izlaz ispisuje tekst `Zdravo svima!`.

Primer 1

```
Interakcija sa programom:
 Zdravo svima!
```

[Rešenje 1.1.1]

Zadatak 1.1.2 Napisati program za uneti ceo broj ispisuje njegov kvadrat i njegov kub.

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj: 4
 Kvadrat: 16
 Kub: 64
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj: -14
 Kvadrat: 196
 Kub: -2744
```

[Rešenje 1.1.2]

Zadatak 1.1.3 Napisati program koji za uneta dva cela broja ispisuje najpre unete vrednosti, a zatim i njihov zbir, razliku, proizvod, ceo deo pri deljenju prvog broja drugim brojem i ostatak pri deljenju prvog broja drugim brojem.

NAPOMENA: *Pretpostaviti da je unos korektan, tj. da druga uneta vrednost nije 0.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesi vrednost celobrojne promenljive x: 7
 Unesi vrednost celobrojne promenljive y: 2
 7 + 2 = 9
 7 - 2 = 5
 7 * 2 = 14
 7 / 2 = 3
 7 % 2 = 1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesi vrednost celobrojne promenljive x: -3
 Unesi vrednost celobrojne promenljive y: 8
 -3 + 8 = 5
 -3 - 8 = -11
 -3 * 8 = -24
 -3 / 8 = 0
 -3 % 8 = -3
```

[Rešenje 1.1.3]

Zadatak 1.1.4 Napisati program koji pomaže kasirki da izračuna ukupan račun ako su poznate cene dva kupljena artikla. NAPOMENA: *Pretpostaviti da su cene artikala pozitivni celi brojevi i da je unos korektan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesi cenu prvog artikla: 173
 Unesi cenu drugog artikla: 2024
 Ukupna cena iznosi 2197
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesi cenu prvog artikla: 384
 Unesi cenu drugog artikla: 555
 Ukupna cena iznosi 939
```

[Rešenje 1.1.4]

Zadatak 1.1.5 Napisati program koji za unetu količinu jabuka u kilogramima i unetu cenu po kilogramu ispisuje ukupnu vrednost date količine jabuka. NAPOMENA: *Pretpostaviti da je cena jabuka pozitivan ceo broj i da je unos korektan.*

Primer 1

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

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite kolicinu jabuka (u kg): 10
 Unesite cenu (u dinarima): 93
 Molimo platite 930 dinara.
```

[Rešenje 1.1.5]

Zadatak 1.1.6 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. NAPOMENA: *Pretpostaviti*

da su cene svih artikala pozitivni celi brojevi, kao i da su unete vrednosti ispravne, tj. da se može vratiti kusur.

Primer 1

```
Interakcija sa programom:
 Unesite cenu, kolicinu i iznos: 132 2 500
 Kusur je 236 dinara.
```

Primer 2

```
Interakcija sa programom:
 Unesite cenu, kolicinu i iznos: 59 6 2000
 Kusur je 1646 dinara.
```

[Rešenje 1.1.6]

Zadatak 1.1.7 Napisati program koji za uneta vremena poletanja i sletanja aviona ispisuje dužinu trajanja leta. Napomena: *Pretpostaviti da su poletanje i sletanje u istom danu kao i da su sve vrednosti ispravno unete.*

Primer 1

```
Interakcija sa programom:
 Unesite vreme poletanja: 8 5
 Unesite vreme sletanja: 12 41
 Duzina trajanja leta je 4 h i 36 min
```

Primer 2

```
Interakcija sa programom:
 Unesite vreme poletanja: 13 20
 Unesite vreme sletanja: 18 45
 Duzina trajanja leta je 5 h i 25 min
```

[Rešenje 1.1.7]

Zadatak 1.1.8 Date su dve celobrojne promenljive. Napisati program koji razmenjuje njihove vrednosti.

Primer 1

```
Interakcija sa programom:
 Unesi dve celobrojne vrednosti: 5 7
 pre zamene: x=5, y=7
 posle zamene: x=7, y=5
```

Primer 2

```
Interakcija sa programom:
 Unesi dve celobrojne vrednosti: 237 -592
 pre zamene: x=237, y=-592
 posle zamene: x=-592, y=237
```

[Rešenje 1.1.8]

Zadatak 1.1.9 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.*

Primer 1

```
Interakcija sa programom:
 Unesi dve celobrojne vrednosti: 5 7
 Nove vrednosti su: a=12, b=-2
```

Primer 2

```
Interakcija sa programom:
 Unesi dve celobrojne vrednosti: 237 -592
 Nove vrednosti su: a=-355, b=829
```

Zadatak 1.1.10 Napisati program koji za uneti pozitivan trocifreni broj na standardni izlaz ispisuje njegove cifre jedinica, desetica i stotina. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesi trocifreni broj: 697
 jedinica 7, desetica 9, stotina 6
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesi trocifreni broj: 504
 jedinica 4, desetica 0, stotina 5
```

[Rešenje 1.1.10]

Zadatak 1.1.11 Napisati program koji za unetu cenu proizvoda ispisuje najmanji broj novčanica koje je potrebno izdvojiti prilikom plaćanja proizvoda. Na raspolaganju su novčanice od 5000, 2000, 1000, 500, 200, 100, 50, 20, 10 i 1 dinar. NAPOMENA: *Pretpostaviti da je cena proizvoda pozitivan ceo broj.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cenu proizvoda: 8367
 8367=1*5000+ 1*2000 +1*1000 +0*500 +1*200 +1*100 +1*50 +0*20 +1*10 +7*1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cenu proizvoda: 934
 934=0*5000+ 0*2000 +0*1000 +1*500 +2*200 +0*100 +0*50 +1*20 +1*10 +4*1
```

[Rešenje 1.1.11]

Zadatak 1.1.12 Napisati program koji učitava pozitivan trocifreni broj sa standardnog ulaza i ispisuje broj dobijen obrtanjem njegovih cifara. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesi trocifreni broj: 892
 Obrnuto: 298
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesi trocifreni broj: 230
 Obrnuto: 32
```

[Rešenje 1.1.12]

Zadatak 1.1.13 Napisati program koji za uneti pozitivan 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

NAPOMENA: *Pretpostaviti da je unos ispravan.*

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
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cetvorocifreni broj: 3570
 Proizvod cifara: 0
 Razlika sume krajnjih i srednjih: -9
 Suma kvadrata cifara: 83
 Broj u obrnutom poretku: 753
 Broj sa zamenjenom cifrom jedinica i stotina: 3075
```

[Rešenje 1.1.13]

Zadatak 1.1.14 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
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 825
 Rezultat je: 85
```

Zadatak 1.1.15 Sa standardnog unosa se unosi pozitivan prirodan broj n i pozitivan dvocifreni broj m. Napisati program ispisuje broj dobijen umetanjem broja m između cifre stotina i cifre hiljada broja n. NAPOMENA: *Za neke ulazne podatke može se dobiti neočekivan rezultat zbog prekoračenja, što ilustruje test primer broj 2.*

Primer 1

```
Interakcija sa programom:
 Unesite pozitivan prirodan broj: 12345
 Unesite pozitivan dvocifreni broj: 67
 Novi broj je 1267345
```

Primer 2

```
Interakcija sa programom:
 Unesite pozitivan prirodan broj: 50000000
 Unesite pozitivan dvocifreni broj: 12
 Novi broj je 705044704
```

[Rešenje 1.1.15]

Zadatak 1.1.16 Napisati program koji učitava realnu vrednost izraženu u inčima, konvertuje tu vrednost u centimetre i ispisuje je zaokruženu na dve decimale. Uputstvo: *Jedan inč ima* 2.54 *centimetra.*

Primer 1

```
Interakcija sa programom:
 Unesi broj inca: 4.69
 4.69 in = 11.91 cm
```

Primer 2

```
Interakcija sa programom:
 Unesi broj inca: 71.426
 71.43 in = 181.42 cm
```

[Rešenje 1.1.16]

Zadatak 1.1.17 Napisati program koji učitava dužinu izraženu u miljama, konvertuje tu vrednost u kilometre i ispisuje je zaokruženu na dve decimale. Uputstvo: *Jedna milja ima* 1.609344 *kilometara.*

Primer 1

```
Interakcija sa programom:
 Unesi broj milja: 50.42
 50.42 mi = 81.14 km
```

Primer 2

```
Interakcija sa programom:
 Unesi broj milja: 327.128
 327.128 mi = 526.46 km
```

[Rešenje 1.1.17]

Zadatak 1.1.18 Napisati program koji učitava težinu izraženu u funtama, konvertuje tu vrednost u kilograme i ispisuje je zaokruženu na dve decimale. Uputstvo: *Jedna funta ima* 0.45359237 *kilograma.*

Primer 1

```
Interakcija sa programom:
 Unesi broj funti: 2.78
 2.78 lb = 1.26 kg
```

Primer 2

```
Interakcija sa programom:
 Unesi broj funti: 89.437
 89.437 lb = 40.57 kg
```

[Rešenje 1.1.18]

Zadatak 1.1.19 Napisati program koji učitava temperaturu izraženu u farenhajtima, konvertuje tu vrednost u celzijuse i ispisuje je zaokruženu na dve decimale. UPUTSTVO: *Veza između farenhajta i celzijusa je zadata narednom formulom* $F = \frac{9 \cdot C}{5} + 32$

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesi temperaturu u F: 100.93
 100.93 F = 38.29 C
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesi temperaturu u F: 25.562
 25.562 F = -3.58 C
```

[Rešenje 1.1.19]

Zadatak 1.1.20 Napisati program koji za unete realne vrednosti a_{11}, a_{12}, a_{21}, a_{22} ispisuje vrednost determinante matrice:

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}$$

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
```

Zadatak 1.1.21 Napisati program koji za unete realne vrednosti dužina stranica pravougaonika ispisuje njegov obim i površinu. Ispisati tražene vrednosti zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite duzine stranica: 4.3 9.4
 Obim: 27.40
 Povrsina: 40.42
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite duzine stranica: 10.756 36.2
 Obim: 93.91
 Povrsina: 389.37
```

[Rešenje 1.1.21]

Zadatak 1.1.22 Napisati program koji za unetu realnu vrednost dužine poluprečnika kruga ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite duzinu poluprecnika kruga: 4.2
Obim: 26.39, povrsina: 55.42
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite duzinu poluprecnika kruga: 14.932
Obim: 93.82, povrsina: 700.46
```

[Rešenje 1.1.22]

Zadatak 1.1.23 Napisati program koji za unetu realnu vrednost dužine stranice jednakostraničnog trougla ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite duzinu stranice trougla: 5
Obim: 15.00
Povrsina: 10.82
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite duzinu stranice trougla: 2
Obim: 6.00
Povrsina: 1.73
```

[Rešenje 1.1.23]

Zadatak 1.1.24 Napisati program koji za unete realne vrednosti dužina stranica trougla ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite duzine stranica trougla: 4.3 9.7 8.8
Obim: 22.80
Povrsina: 18.91
```

[Rešenje 1.1.24]

Zadatak 1.1.25 Pravougaonik čije su stranice paralelne koordinatnim osama zadat je svojim realnim koordinatama suprotnih temena (gornje levo i donje desno teme). Napisati program koji ispisuje njegov obim i površinu zaokružene na dve decimale.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite koordinate gornjeg levog temena: 4.3 5.8
 Unesite koordinate donjeg desnog temena: 6.7 2.3
 Obim: 11.80
 Povrsina: 8.40
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite koordinate gornjeg levog temena: -3.7 8.23
 Unesite koordinate donjeg desnog temena: -0.56 2
 Obim: 18.74
 Povrsina: 19.56
```

Zadatak 1.1.26 Napisati program koji za tri uneta cela broja ispisuje njihovu artimetičku sredinu zaokruženu na dve decimale.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite tri cela broja: 11 5 4
 Aritmeticka sredina unetih brojeva je 6.67
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite tri cela broja: 3 -8 13
 Aritmeticka sredina unetih brojeva je 2.67
```

[Rešenje 1.1.26]

Zadatak 1.1.27 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 unete cene usluge po kvadratnom metru program izračuna ukupnu cenu krečenja. Sve realne vrednosti ispisati zaokružene na dve decimale.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije sobe: 4 4 3
 Unesite cenu po m2: 500
 Moler treba da okreci 51.20 m2
 Cena krecenja je 25600.00
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije sobe: 13 17 3
 Unesite cenu po m2: 475
 Moler treba da okreci 320.80 m2
 Cena krecenja je 152380.00
```

[Rešenje 1.1.27]

Zadatak 1.1.28 Napisati program koji za unete pozitivne prirodne brojeve x, p i c ispisuje broj koji se dobija ubacivanjem cifre c u broj x na poziciju p.

NAPOMENA: *Podrazumevati da je unos ispravan, tj. da je broj p manji od ukupnog broja cifara broja x. Numeracija cifara počinje od nule, odnosno cifra najmanje težine nalazi se na nultoj poziciji.* UPUTSTVO: *Koristiti funkciju* `pow` *iz* `math.h` *biblioteke.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite redom x, p i c:  140 1 2
 Rezultat je: 1420
```

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite redom x, p i c:  12345 2 9
 Rezultat je: 123945
```

[Rešenje 1.1.28]

Zadatak 1.1.29 Napisati program koji za uneta dva cela broja a i b dodeljuje promenljivoj *rezultat* vrednost 1 ako važi uslov:

a) a i b su različiti brojevi

b) a i b su parni brojevi

c) a i b su pozitivni brojevi, ne veći od 100

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

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dva cela broja:  4 8
 a) rezultat=1
 b) rezultat=1
 c) rezultat=1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dva cela broja:  3 -11
 a) rezultat=1
 b) rezultat=0
 c) rezultat=0
```

[Rešenje 1.1.29]

Zadatak 1.1.30 Napisati program koji za uneta dva cela broja ispisuje njihov maksimum.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dva cela broja:  19 256
 Maksimum je 256
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dva cela broja:  -39 57
 Maksimum je 57
```

[Rešenje 1.1.30]

Zadatak 1.1.31 Napisati program koji za uneta dva cela broja ispisuje njihov minimum.

Primer 1

```
Interakcija sa programom:
 Unesite dva cela broja: 4 8
 Minimum je 4
```

Primer 2

```
Interakcija sa programom:
 Unesite dva cela broja: -3 -110
 Minimum je -110
```

[Rešenje 2.1.36]

Zadatak 1.1.32 Napisati program koji za unete realne vrednosti promenljivih x i y ispisuje vrednost sledećeg izraza:

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

zaokruženu na dve decimale.

Primer 1

```
Interakcija sa programom:
 Unesite dva realna broja: 5.7 11.2
 Rezultat je: 0.05
```

Primer 1

```
Interakcija sa programom:
 Unesite dva realna broja: -9.34 8.99
 Rezultat je: -0.11
```

[Rešenje 1.1.32]

1.2 Rešenja

Rešenje 1.1.1

```
/* Navedeni program definise funkciju koja se zove main. Program
   moze da definise vise funkcija, ali obavezno mora da definise
   funkciju koja se zove main i izvrsavanje programa uvek pocinje
   od te funkcije. Pored naziva, zapis svake funkcije cine i
   povratna vrednost funkcije (u ovom slucaju int), lista
   argumenata koje funkcija koristi (u ovom slucaju funkcija nema
   argumenata pa se navode samo prazne zagrade, ()) i telo
   funkcije koje je ograniceno viticastim zagradama ({ i }). O
   ovim pojmovima bice vise reci u narednim poglavljima.

   Unutar tela funkcije navode se naredbe. Unutar navedenog
   programa postoji jedna naredba koja predstavlja poziv funkcije
   printf. Funkcija printf sluzi za ispis teksta na standardni
```

```
14    izlaz (obicno ekran). Deklaracija ove funkcije data je u
      zaglavlju stdio.h koje je potrebno ukljuciti direktivom #include
16    na pocetku samog programa.

18    Da bismo pokrenuli program, prvo ga moramo prevesti u izvrsnu
      datoteku. Na primer, ako je navedeni program sacuvan kao
20    zdravo.c, ako koristimo gcc kompajler koji je sastavni deo
      standardnih Linux distribucija, prevodjenje iz komandne linije
22    se vrsi narednom naredbom: gcc zdravo.c Ukoliko nije bilo
      gresaka prilikom prevodjenja, bice generisana izvrsna datoteka
24    pod nazivom a.out koja se pokrece navodjenjem sledece naredbe:
      ./a.out Ukoliko je bilo gresaka prilikom prevodjenja, one se
26    moraju otkloniti a postupak prevodjenja se mora ponoviti. */

28 #include<stdio.h>

30 int main()
   {
32   /* printf: funkcija pomocu koje se vrsi ispis
        Specijalni karakter \n : prelazak u novi red
34      Svaka naredba zavrsava se karakterom ; */
     printf("Zdravo svima!\n");
36
     /* Povratna vrednost 0 se obicno koristi da oznaci da je prilikom
38      izvrsavanja programa sve proslo u redu. */
     return 0;
40 }
```

Rešenje 1.1.2

```
   #include <stdio.h>
2
   int main()
4  {
     /* Svaka promenljiva u programu mora biti deklarisana na pocetku
6       main funkcije. Deklaracija se sastoji iz naziva promenljive
        (u ovom slucaju n) ispred kog se navodi tip promenljive (u
8       ovom slucaju celobrojni tip, int). */

10   int n;

12   /* Vrednost promenljive se ucitava pomocu funkcije scanf koja
        je, kao i funkcija printf, sastavni deo standardne biblioteke.
14      Argumenti funkcije scanf koji se navode u zagradama ( i ) i
        razdvajaju zarezima, oznacavaju sledece: "%d" - format za tip
16      podatka koji ce biti ucitan (%d za int, svaki tip ima svoj
        format) &n - adresa promenljive x (o adresama ce biti vise
18      reci u narednim zadacima).

20      Ucitavanje se vrsi sa standardnog ulaza (obicno tastatura). */
     printf("Unesite ceo broj: ");
```

```
22    scanf("%d", &n);

24    /* Funkcija printf ispisuje tekst "Uneti broj: ", a nakon toga,
         umesto formata %d, ispisuje vrednost promenljive n. */
26    printf("Uneti broj: %d\n", n);
      /* Umesto formata %d, ispisuje vrednost izraza n*n. */
28    printf("Kvadrat: %d\n", n * n);
      /* Umesto formata %d, ispisuje vrednost izraza n*n*n. */
30    printf("Kub: %d\n", n * n * n);

32    return 0;
  }
```

Rešenje 1.1.3

```
  #include<stdio.h>
2
  int main()
4 {
    /* Promenljive istog tipa mogu se deklarisati jedna za drugom. */
6   int x, y, rezultat;

8   printf("Unesi vrednost celobrojne promenljive x: ");
    scanf("%d", &x);
10
    printf("Unesi vrednost celobrojne promenljive y: ");
12  scanf("%d", &y);

14  /* Dodeljujemo vrednost promenljivoj rezultat. */
    rezultat = x + y;
16  printf("%d + %d = %d\n", x, y, rezultat);

18  /* Mozemo ispisivati direktno vrednost izraza x-y i bez njegovog
       dodeljivanja posebnoj promenljivoj */
20  printf("%d - %d = %d\n", x, y, x - y);
    printf("%d * %d = %d\n", x, y, x * y);
22
    /* Kada bilo koju artimeticku operaciju primenimo na dve
24     promenljive istog tipa (u ovom slucaju dva celobrojne
       promenljive), rezultat ce biti tog istog tipa. Specijalno, za
26     operaciju deljenja: kada operator / primenimo na dva
       celobrojna argumenta x i y, kao rezultat dobijemo ceo deo pri
28     deljenju broja x brojem y, a ne kolicnik. Na primer, rezultat
       primene operatora / na 7 i 2 je 3, a ne 3.5. */
30  printf("%d / %d = %d\n", x, y, x / y);

32  /* Operator % izracunava ostatak pri celobrojnom deljenju dve
       celobrojne promenljive. Na primer, 7%2 ima vrednost 1 (jer je
34     7=3*2+1). Da bismo odstampali karakter %, u naredbi printf
       pisemo %% */
36  printf("%d %% %d = %d\n", x, y, x % y);
```

```
38    return 0;
}
```

Rešenje 1.1.4

Rešenje ovog zadatka svodi se na rešenje zadatka 1.1.3, na deo koji se odnosi na izračunavanje zbira dva broja. Zbog pretpostavke da su cene artikala pozitivni celi brojevi, tip promenljivih za artikle treba da bude `unsigned int`.

Rešenje 1.1.5

Rešenje ovog zadatka svodi se na rešenje zadatka 1.1.3, na deo koji se odnosi na izračunavanje proizvoda dva broja. Zbog pretpostavke da su cene artikala pozitivni celi brojevi, tip promenljivih za artikle treba da bude `unsigned int`.

Rešenje 1.1.6

```
1  #include <stdio.h>

3  int main()
   {
5    /* S obzirom da su sve promenljive pozitivini celi brojevi,
        koristimo tip unsigned int (skraceno unsigned) */
7    unsigned cena;
     unsigned kolicina;
9    unsigned iznos;
     unsigned kusur;
11
     /* Ucitavamo potrebne podatke. Unutar jednog scanf-a mozemo
13      ucitati vise podataka odjednom. Za svaki treba navesti
        odgovarajuci format za tip podataka koji se unosi (%u za
15      unsigned). */
     printf("Unesite cenu, kolicinu i iznos: ");
17   scanf("%u%u%u", &cena, &kolicina, &iznos);

19   /* Izracunavamo kusur: */
     kusur = iznos - kolicina * cena;
21
     /* I ispisujemo trazenu vrednost: */
23   printf("Kusur je %u dinara.\n", kusur);

25   return 0;
}
```

Rešenje 1.1.7

```c
#include <stdio.h>

int main()
{

  unsigned poletanje, poletanje_sat, poletanje_minut;
  unsigned sletanje, sletanje_sat, sletanje_minut;
  unsigned duzina, duzina_sat, duzina_minut;

  printf("Unesite vreme poletanja: ");
  scanf("%u%u", &poletanje_sat, &poletanje_minut);


  printf("Unesite vreme sletanja: ");
  scanf("%u%u", &sletanje_sat, &sletanje_minut);

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

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

  /* Izdvajamo broj sati i broj minuta. */
  duzina_sat = duzina / 3600;
  duzina_minut = (duzina % 3600) / 60;

  /* I ispisujemo rezultat */
  printf("Duzina trajanja leta je %u h i %u min\n", duzina_sat,
         duzina_minut);

  return 0;
}
```

Rešenje 1.1.8

```c
#include<stdio.h>

int main()
{
  int x, y;
  int p;

  printf("Unesi dve celobrojne vrednosti:");
  scanf("%d%d", &x, &y);

  printf("pre zamene: x=%d, y=%d\n", x, y);

  /* Pomocna promenljiva p je potrebna da sacuva vrednost
     promenljive x pre nego sto se ona izmeni i dobije vrednost
```

```
    promenljive y. */
  p = x;
  x = y;
  y = p;
  printf("posle zamene: x=%d, y=%d\n", x, y);

  return 0;
}
```

Rešenje 1.1.10

```
#include <stdio.h>

int main()
{
  /* S obzirom da broj treba da bude pozitivan, koristimo tip
     unsigned. */
  unsigned x;

  /* Promenljive koje cuvaju cifre treba da budu najmanjeg
     celobrojnog tipa jer nece sadrzati druge vrednosti osim
     jednocifrenih celih brojeva. Zbog toga za njih biramo tip
     char.   */
  char cifra_jedinice;
  char cifra_desetice;
  char cifra_stotine;

  printf("Unesi trocifreni broj:");
  scanf("%u", &x);

  /* Na primer, neka je uneti broj 374. Potrebno je da koriscenjem
     racunskih operacija za rad sa celim brojevima pristupimo
     njegovoj cifri jedinice, cifri desetice i cifri stotine.

     Primetimo najpre sledece: 374/10 = 37 374%10 = 4 Dakle,
     operacijama celobrojnog deljenja i ostatka pri deljenju mozemo
     iz svakog broja izdvojiti njegovu poslednju cifru (u ovom
     slucaju 4) i broj sastavljen od svih cifara osim poslednje (u
     ovom slucaju 37).

     Cifri jedinice sada lako pristupamo koriscenjem ostatka pri
     deljenju sa 10. Ona iznosi upravo 4.

     Pri trazenju cifre desetice mozemo ponovo primeniti princip
     izdvajanja poslednje cifre kao ostatka pri deljenju sa 10.
     Razlika je sto ne mozemo deseticu izdvojiti ako primenimo %10
     na 374 (time dobijamo 4), vec %10 primenjujemo na 37, pri cemu
     37 dobijamo kao ceo deo pri deljenju broja 374 brojem 10.
     Dakle, cifru desetice dobijamo kao (374/10)%10.

     S obzirom da znamo da je u pitanju trocifreni broj, cifru
```

```
41      stotine mozemo izdvojiti celobrojnim deljenjem sa 100: 374/100
        iznosi upravo 3. */
43    cifra_jedinice = x % 10;
      cifra_desetice = (x / 10) % 10;
45    cifra_stotine = x / 100;

47    /* Ako zelimo da odstampamo numericku vrednost promenljive tipa
        char, koristimo format %d. Ako zelimo da odstampamo karakter
49      ciji je ASCII kod jednak vrenosti te promenljive, koristimo %c
        (na primer, ako bismo promenljivu cija je vrednost 65 stampali
51      pomocu formata %d, ispis bi bio 65, ali ako bismo je stampali
        pomocu formata %c, ispis bi bio A). U ovom slucaju nam je
53      neophodna numericka vrednost. */
      printf("jedinica %d, desetica %d, stotina %d\n", cifra_jedinice,
55           cifra_desetice, cifra_stotine);

57    /* 2. nacin, bez uvodjenja dodatnih promenljivih cifra_jedinice,
        cifra_desetice i cifra_stotine:
59
        printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10,
61      x/100); */
      return 0;
63  }
```

Rešenje 1.1.11

```
1   #include <stdio.h>

3   int main()
    {
5     unsigned x;
      printf("Unesi cenu:");
7     scanf("%u", &x);

9     /* Na primer, neka je uneta cena 8347 dinara. Vrednost x/5000
        predstavlja broj novcanica od 5000 dinara pomocu kojih mozemo
11      sakupiti celokupnu sumu. 8347 celobrojno deljeno sa 5000
        (operacija / nad celim brojevima) iznosi 1. */
13    printf("%u=%u*5000+ ", x, x / 5000);
      /* Potrebna nam je 1 novcanica od 5000 dinara, a koliko nam je
15      potrebno ostalih novcanica? Za to moramo pristupiti preostaloj
        sumi. Jedan nacin je da nadjemo ostatak pri deljenju unete
17      vrednosti x (u primeru 8347) sa 5000 (operacija %). On iznosi
        3347. Ovu vrednost dodeljujemo promeljivoj x. */
19    x = x % 5000;

21    /* Nastavljamo postupak trazenjem broja novcanica od 2000 dinara
        i redom za ostale monete. */
23    printf("%u*2000 +", x / 2000);
      x = x % 2000;
25    printf("%u*1000 +", x / 1000);
```

```
  x = x % 1000;
  printf("%u*500 +", x / 500);
  x = x % 500;
  printf("%u*200 +", x / 200);
  x = x % 200;
  printf("%u*100 +", x / 100);
  x = x % 100;
  printf("%u*50 +", x / 50);
  x = x % 50;
  printf("%u*20 +", x / 20);
  x = x % 20;
  printf("%u*10 +", x / 10);
  x = x % 10;
  printf("%u*1\n", x);

  return 0;
}
```

Rešenje 1.1.12

```
#include <stdio.h>

int main()
{
  unsigned x;
  unsigned obrnuto_x;
  char cifra_jedinice;
  char cifra_desetice;
  char cifra_stotine;

  printf("Unesi trocifreni broj:");
  scanf("%u", &x);

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

  obrnuto_x = cifra_jedinice * 100 +
      cifra_desetice * 10 + cifra_stotine;

  printf("Obrnuto: %u\n", obrnuto_x);

  return 0;
}
```

Rešenje 1.1.13

```
#include <stdio.h>
```

```c
int main()
{

  unsigned n, broj_obrnuto, broj_zamena;
  char j, d, s, h;
  int proizvod_cifara, razlika_cifara, suma_kvadrata;

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

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

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

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

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

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

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


  return 0;
}
```

Rešenje 1.1.15

```c
#include <stdio.h>

int main()
{
```

```
  unsigned broj, novibroj;
  unsigned levi, desni, m;

  printf("Unesite pozitivan prirodan broj: ");
  scanf("%u", &broj);
  printf("Unesite pozitivan dvocifreni broj:");
  scanf("%u", &m);

  /* Na primer, za unete broj 12345 i 67, potrebno je ubaciti 67
     izmedju cifre hiljade (2) i cifre stotine (3). Rezultat je
     12|67|345. Potrebno je da razdvojimo uneti broj na levi i
     desni deo: 12 i 345 i izmedju njih umetnemo broj m */

  levi = broj / 1000;
  desni = broj % 1000;

  /* Kada levi deo pomnozimo sa 100 000, dobijamo 1 200 000 Kada m
     pomnozimo sa 1000, dobijamo 67 000 Dobijene vrednosti saberemo
     sa desnim delom 345 --------- Konacan rezultat: 1 267 345   */
  novibroj = levi * 100000 + m * 1000 + desni;

  printf("Novi broj je %u\n", novibroj);

  return 0;
}
```

Rešenje 1.1.16

```
#include <stdio.h>

int main()
{
  /* float - realni tip jednostruke tacnosti */
  float in;
  float cm;

  printf("Unesi broj inca: ");

  /* "%f" - format za unos/ispis float promenljivih */
  scanf("%f", &in);

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

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

  return 0;
}
```

Rešenje 1.1.17 Zadatak se rešava analogno zadatku 1.1.16.

Rešenje 1.1.18 Zadatak se rešava analogno zadatku 1.1.16.

Rešenje 1.1.19 Zadatak se rešava analogno zadatku 1.1.16.

Rešenje 1.1.21

```c
#include <stdio.h>

int main()
{
  float a, b;
  float obim, povrsina;

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

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

  /* Povrsina */
  povrsina = a * b;

  /* Ispisujemo trazene vrednosti */
  printf("Obim: %.2f\n", obim);
  printf("Povrsina: %.2f\n", povrsina);

  /* Zavrsavamo sa programom */
  return 0;
}
```

Rešenje 1.1.22

```c
#include <stdio.h>
#include <math.h>

/* Zaglavlje math.h sadrzi deklaracije velikog broja matematickih
   funkcija i konstanti. U ovom zadatku se koristi zbog konstante
   pi (M_PI)

   Ukoliko se koristi i neka funkcija matematicke biblioteke, za
   prevodjenje je neophodno ukljuciti opciju -lm npr. gcc primer.c
   -lm */
int main()
{
  float r;
```

```
  float O;
  float P;
  printf("Unesite duzinu poluprecnika kruga:");
  scanf("%f", &r);

  O = 2 * r * M_PI;
  P = r * r * M_PI;

  printf("Obim: %.2f, povrsina: %.2f\n", O, P);

  return 0;
}
```

Rešenje 1.1.23

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

int main()
{
  float a;
  float P, O;

  printf("Unesi duzinu stranice trougla:");
  scanf("%f", &a);

  O = 3 * a;
  P = (a * a * sqrt(3)) / 4;

  printf("Obim: %.2f\n", O);
  printf("Povrsina: %.2f\n", P);

  return 0;
}
```

Rešenje 1.1.24

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

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

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

```

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

  /* Povrsina - koristicemo Heronov obrazac */
  s = obim / 2;
  povrsina = sqrt(s * (s - a) * (s - b) * (s - c));

  /* Ispisujemo trazene vrednosti */
  printf("Obim: %.2f\n", obim);
  printf("Povrsina: %.2f\n", povrsina);

  return 0;
}
```

Rešenje 1.1.26

```
#include<stdio.h>

int main()
{
  int a, b, c;
  float as;

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

  /* pogresan nacin: as = (a+b+c)/3;

     Ukoliko podelimo zbir a+b+c sa 3, to ce biti primena
     operatora / na dva cela broja. Na ovaj nacin izracunacemo
     koliko iznosi a+b+c celobrojno podeljeno sa 3. To znaci da ce
     za unete vrednosti 11, 5 i 4 aritmeticka sredina biti 6.00.
     Zaista, zbir 11+5+4 iznosi 20, a kada 20 celobrojno podelimo
     sa 3 dobijamo 6. Ovu celobrojnu vrednost dodeljujemo realnoj
     promenljivoj as, cime se ona konvertuje u 6.000000 i
     ispisujemo je zaokruzenu na dve decimale. Izlaz iz programa bi
     bio pogresan: 6.00.

     Da bismo dobili kolicnik prilikom primene operatora / na dva
     cela broja, a ne celobrojno deljenje, jedan argument mora da
     bude realan broj. Jedan nacin je da umesto sa celobrojnom
     trojkom (3) deljenje izvedemo sa realnom trojkom (3.0): */
  as = (a + b + c) / 3.0;

  /* Trazeni kolicnik mozemo dobiti na razne nacine:
     as=1.0*(a+b+c)/3; ili as=(0.0+a+b+c)/3; ili
     as=((float)(a+b+c))/3; itd. */

  printf("Aritmeticka sredina unetih brojeva je %.2f\n", as);
  return 0;
}
```

Rešenje 1.1.27

```c
#include <stdio.h>

int main()
{
  unsigned duzina, sirina, visina;
  unsigned cena;
  float povrsina_za_krecenje;
  float ukupna_cena;

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

  /* Ucitavamo cenu krecenja */
  printf("Unesite cenu po m2: ");
  scanf("%u", &cena);

  /* Povrsina za krecenje odgovara povrsini kvadra - bez poda jer
     se on ne kreci */
  povrsina_za_krecenje = 0.8 * (duzina * sirina +
                                2 * duzina * visina +
                                2 * sirina * visina);
  ukupna_cena = povrsina_za_krecenje * cena;

  /* Ispisujemo trazene podatke */
  printf("Moler treba da okreci %.2f m2\n", povrsina_za_krecenje);

  printf("Cena krecenja je %.2f\n", ukupna_cena);

  /* Zavrsavamo sa programom */
  return 0;
}
```

Rešenje 1.1.28

```c
#include <stdio.h>
#include <math.h>

int main()
{
  unsigned x, p;
  char c;
  unsigned levo, desno;
  unsigned novo_x;
```

```c
    /* Ucitavamo potrebne vrednosti. Sa unosom podataka tipa char
       moramo biti pazljivi i o tome ce vise biti reci u narednim
       poglavljima kod zadataka za rad sa funkcijama getchar i
       putchar. Zbog toga cemo ovde za ucitavanje podataka zatraziti
       da podatke razdvajamo blanko znakovima (a ne znakom za novi
       red, zarezom ili nekim drugim separatorom). Ovaj zahtev
       navodimo u format stringu funkcije scanf tako sto
       specifikatore promenljivih razdvajamo blanko znakovima.

       Ukoliko specifikatore promenljivih u format stringu pisemo
       spojeno, tada ih prilikom unosa mozemo razdvojiti bilo kojim
       karakterom. Zbog toga blanko znakove u format stringu funkcije
       scanf treba izbegavati i ovo je redak slucaj kada je njihova
       upotreba opravdana.

       Ako zelimo da odstampamo znak ", u format stringu funkcije
       printf navodimo \". */
    printf("Unesite vrednosti u formatu \"x p c\": ");
    scanf("%u %u %c", &x, &p, &c);

    /* Kada ucitavamo karaktersku promenljivu, njena numericka
       vrednost je jednaka ASCII kodu unetog karaktera. Na primer,
       ako karakter '0' ucitamo u promenljivu c, njena numericka
       vrednost bice 48. Da bismo pretvorili ovu numericku vrednost u
       numericku vrednost koja odgovara cifri, od nje oduzimamo ASCII
       kod karakterske konstante '0' koji iznosi upravo 48. */
    c = c - '0';

    /* Odredjujemo deo broja koji se nalazi desno od pozicije p */
    desno = x % (unsigned) pow(10, p);

    /* Odredjujemo deo broja koji se nalazi levo od pozicije p */
    levo = x / (unsigned) pow(10, p);

    /* Odredjujemo novi broj */
    novo_x =
        levo * (unsigned) pow(10, p + 1) +
           c * (unsigned) pow(10, p) + desno;

    /* Ispisujemo dobijenu vrednost */
    printf("Rezultat je: %u\n", novo_x);

    /* Zavrsavamo sa programom */
    return 0;

}
```

Rešenje 1.1.29

```c
#include <stdio.h>
```

```c
int main()
{
  int a, b, rezultata, rezultatb, rezultatc;

  printf("Unesite dva cela broja:");
  scanf("%d%d", &a, &b);

  /* Izraz a!=b ima vrednost 1 ako je ova relacija tacna, a 0 ako
     je netacna */
  rezultata = a != b;

  /* Izraz a%2==0 && b%2==0 je konjunkcija koja se sastoji od dve
     relacije jednakosti. Izraz a%2==0 ima vrednost 1 ako je ova
     relacija tacna, a 0 u suprotnom. */

  rezultatb = (a % 2 == 0 && b % 2 == 0);

  /* Izraz a>0 && a<=100 && b>0 && b<=100 konjunkcija koja se
     sastoji od cetiri konjunkata. Svaki od konjunkata je izraz
     koji sadrzi relacioni operator i ima vrednost 1 ako relacija
     vazi a 0 ako ne vazi */

  rezultatc = (a > 0 && a <= 100 && b > 0 && b <= 100);

  printf("a) rezultat=%d\n", rezultata);
  printf("b) rezultat=%d\n", rezultatb);
  printf("c) rezultat=%d\n", rezultatc);

  return 0;
}
```

Rešenje 1.1.30

```c
#include <stdio.h>

int main()
{
  int a, b, max;
  printf("Unesite dva cela broja:");
  scanf("%d%d", &a, &b);

  /* Ternarni operator uslova :? koristi se u sledecem obliku:

     izraz1 ? izraz2 : izraz3;

     Izraz izraz1 se izracunava prvi. Ako je njegova vrednost
     razlicita od nule (tj. ako ima istinitosnu vrednost tacno),
     onda se izracunava vrednsot izraza izraz2 i to je vrednost
     citavog uslovnog izraza. U suprotnom, izracunava se vrednost
     izraz3 i to je vrednost citavog uslovnog izraza
  */
```

```
19
    max = (a > b) ? a : b;
21
    printf("Maksimum je %d\n", max);
23
    return 0;
25  }
```

Rešenje 2.1.36 Zadatak se rešava analogno zadatku 2.1.36

Rešenje 1.1.32

```
1   #include <stdio.h>

3   int main()
    {
5     float a, b, rez;
      float min, max;
7     printf("Unesite dva realna broja:");
      scanf("%f%f", &a, &b);
9
      /* Odredjujemo minimalnu i maksimalnu vrednost unetih brojeva */
11    min = (a < b) ? a : b;
      max = (a > b) ? a : b;
13
      /* Racunamo vrednost promenljive rez */
15    rez = (min + 0.5) / (1 + max * max);

17    printf("Rezultat je %.2f\n", rez);

19    return 0;
    }
```

2

Kontrola toka

2.1 Naredbe grananja

Zadatak 2.1.1 Napisati program koji za dva uneta cela broja ispisuje njihov minimum.

Primer 1

```
Interakcija sa programom:
 Unesite dva cela broja: 5 18
 Minimum je 5.
```

Primer 2

```
Interakcija sa programom:
 Unesite dva cela broja: 43 -16
 Minimum je -16.
```

[Rešenje 2.1.1]

Zadatak 2.1.2 Napisati program koji za dva uneta cela broja ispisuje njihov maksimum.

Primer 1

```
Interakcija sa programom:
 Unesite dva cela broja: 141 67
 Maksimum je 141.
```

Primer 2

```
Interakcija sa programom:
 Unesite dva cela broja: -893 -54
 Maksimum je -54.
```

Zadatak 2.1.3 Napisati program koji za uneti realan broj ispisuje njegovu apsolutnu vrednost zaokruženu na dve decimale.

Primer 1

```
Interakcija sa programom:
 Unesite jedan realan broj: 7.42
 Njegova apsolutna vrednost je: 7.42
```

Primer 2

```
Interakcija sa programom:
 Unesite jedan realan broj: -562.428
 Njegova apsolutna vrednost je: 562.43
```

Primer 3

```
Interakcija sa programom:
 Unesite jedan realan broj: 0
 Njegova apsolutna vrednost je: 0.00
```

Primer 4

```
Interakcija sa programom:
 Unesite jedan realan broj: 52
 Njegova apsolutna vrednost je: 52.00
```

[Rešenje 2.1.3]

Zadatak 2.1.4 Napisati program koji za uneti ceo broj ispisuje njegovu recipročnu vrednost zaokruženu na četiri decimale.

Primer 1

```
Interakcija sa programom:
 Unesite jedan ceo broj: 22
 Reciprocna vrednost unetog broja: 0.0455.
```

Primer 2

```
Interakcija sa programom:
 Unesite jedan ceo broj: -9
 Reciprocna vrednost unetog broja: -0.1111.
```

Primer 3

```
Interakcija sa programom:
 Unesite jedan ceo broj: 0
 Nedozvoljeno deljenje nulom.
```

Primer 4

```
Interakcija sa programom:
 Unesite jedan ceo broj: 57298
 Reciprocna vrednost unetog broja: 0.0000.
```

[Rešenje 2.1.4]

Zadatak 2.1.5 Napisati program koji učitava tri cela broja i ispisuje zbir pozitivnih.

Primer 1

```
Interakcija sa programom:
 Unesite tri cela broja: 1 3 -6
 Suma unetih pozitivnih brojeva: 4
```

Primer 2

```
Interakcija sa programom:
 Unesite tri cela broja: -15 81 0
 Suma unetih pozitivnih brojeva: 81
```

Primer 3

```
Interakcija sa programom:
 Unesite tri cela broja: -719 -48 -123
 Suma unetih pozitivnih brojeva: 0
```

Primer 4

```
Interakcija sa programom:
 Unesite tri cela broja: 16 2 576
 Suma unetih pozitivnih brojeva: 594
```

[Rešenje 2.1.5]

Zadatak 2.1.6 U prodavnici je organizovana akcija da svaki kupac dobije najjeftiniji od tri artikla za jedan dinar. Napisati program koji za unete cene tri artikla izračunava ukupnu cenu, kao i koliko dinara se uštedi zahvaljujući popustu. NAPOMENA: *Pretpostaviti da su cene artikala pozitivni celi brojevi.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cene tri artikla: 35 125 97
 Cena sa popustom: 223
 Usteda: 34
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cene tri artikla: 1034 15 25
 Cena sa popustom: 1060
 Usteda: 14
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cene tri artikla: 500 500 500
 Cena sa popustom: 1001
 Usteda: 499
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cene tri artikla: 247 133 126
 Cena sa popustom: 381
 Usteda: 125
```

[Rešenje 2.1.6]

Zadatak 2.1.7 Napisati program koji za uneti četvorocifreni broj ispisuje njegovu najveću cifru.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 6835
 Najveca cifra je: 8
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 238
 Greska: Niste uneli cetvorocifren broj!
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 7777
 Najveca cifra je: 7
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: -2002
 Najveca cifra je: 2
```

[Rešenje 2.1.7]

Zadatak 2.1.8 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.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite vreme (broj sati u itervalu [0,24),
 broj minuta u intervalu [0,60)): 18 19
 Do ponoci je ostalo 5 sati i 41 minuta.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite vreme (broj sati u itervalu [0,24),
 broj minuta u intervalu [0,60)): 23 7
 Do ponoci je ostalo 0 sati i 53 minuta.
```

Primer 3

```
Interakcija sa programom:
 Unesite vreme (broj sati u itervalu [0,24),
 broj minuta u intervalu [0,60)): 24 20
 Neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite vreme (broj sati u itervalu [0,24),
 broj minuta u intervalu [0,60)): 14 0
 Do ponoci je ostalo 10 sati i 0 minuta.
```

[Rešenje 2.1.8]

Zadatak 2.1.9 Napisati program koji za učitani karakter ispisuje uneti karakter i njegov ASCII kod. Ukoliko je uneti karakter malo (veliko) slovo, ispisati i odgovarajuće veliko (malo) slovo i njegov ASCII kod.

Primer 1

```
Interakcija sa programom:
 Unesite karakter: 0
 Uneti karakter: 0, njegov ASCII kod: 48
```

Primer 2

```
Interakcija sa programom:
 Unesite karakter: ?
 Uneti karakter: ?, njegov ASCII kod: 63
```

Primer 3

```
Interakcija sa programom:
 Unesite karakter: A
 Uneti karakter: A, njegov ASCII kod: 65
 odgovarajuce malo slovo: a, njegov ASCII kod: 97
```

Primer 4

```
Interakcija sa programom:
 Unesite karakter: v
 Uneti karakter: v, njegov ASCII kod: 118
 odgovarajuce veliko slovo: V, njegov ASCII kod: 86
```

[Rešenje 2.1.9]

Zadatak 2.1.10 Napisati program koji za unetih pet 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 2.1.10]

Zadatak 2.1.11 Program učitava pet karaktera. Napisati koliko se puta pojavilo veliko ili malo slovo `a`.

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 2.1.11]

Zadatak 2.1.12 Program učitava pet karaktera. Ispisati 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
 1
```

[Rešenje 2.1.12]

Zadatak 2.1.13 Napisati program koji za unetu godinu ispisuje da li je prestupna.

Primer 1

```
Interakcija sa programom:
 Unesite godinu: 2016
 Godina je prestupna.
```

Primer 2

```
Interakcija sa programom:
 Unesite godinu: 1997
 Godina nije prestupna.
```

Primer 3

```
Interakcija sa programom:
 Unesite godinu: 2000
 Godina je prestupna.
```

Primer 4

```
Interakcija sa programom:
 Unesite godinu: 1900
 Godina nije prestupna.
```

[Rešenje 2.1.13]

Zadatak 2.1.14 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 Amstrongov.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 111
 Broj nije Amstrongov.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 84
 Greska: Niste uneli trocifren broj!
```

Primer 4

```
Interakcija sa programom:
 Unesite broj: 371
 Broj je Amstrongov.
```

[Rešenje 2.1.14]

Zadatak 2.1.15 Napisati program koji ispisuje proizvod parnih cifara unetog četvorocifrenog broja.

Primer 1

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 8123
 Proizvod parnih cifara: 16
```

Primer 2

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 3579
 Nema parnih cifara.
```

Primer 3

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: -1234
 Proizvod parnih cifara: 8
```

Primer 4

```
Interakcija sa programom:
 Unesite broj: 288
 Broj nije cetvorocifren!
```

[Rešenje 2.1.15]

Zadatak 2.1.16 Napisati program koji učitava četvorocifreni broj i ispisuje broj koji se dobija kada se unetom broju razmene najmanja i najveća cifra. Napomena: *U slučaju da se najmanja ili najveća cifra pojavljuju na više pozicija, uzeti prvo pojavljivanje.*

Primer 1

```
Interakcija sa programom:
 Unesite broj: 2863
 8263
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 247
 Broj nije cetvorocifren!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 1192
 9112
```

Primer 4

```
Interakcija sa programom:
 Unesite broj: -4239
 -4932
```

[Rešenje 2.1.16]

Zadatak 2.1.17 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 2.1.17]

Zadatak 2.1.18 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`.

Zadatak 2.1.19 Napisati program za rad sa intervalima. Za dva intervala realne prave $[a1, b1]$ i $[a2, b2]$, program treba da odredi:

a) dužinu zajedničkog dela ta dva intervala

b) najveći interval sadržan u datim intervalima (presek), a ako on ne postoji dati odgovarajuću poruku.

c) dužinu realne prave koju pokrivaju ta dva intervala

d) 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]
```

Zadatak 2.1.20 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.

Primer 1

```
Interakcija sa programom:
 Unesite jedan ceo broj: 17
 Broj je veci od nule.
```

Primer 2

```
Interakcija sa programom:
 Unesite jedan ceo broj: 0
 Broj je jednak nuli.
```

Primer 3

```
Interakcija sa programom:
 Unesite jedan ceo broj: -586
 Broj je manji od nule.
```

Primer 4

```
Interakcija sa programom:
 Unesite jedan ceo broj: 62
 Broj je veci od nule.
```

[Rešenje 2.1.20]

Zadatak 2.1.21 Napisati program koji za unete koeficijente kvadratne jednačine ispisuje koliko realnih rešenja jednačina ima i ako ih ima, ispisuje ih zaokružene na dve decimale.

Primer 1

```
Interakcija sa programom:
 Unesite koeficijente A, B i C: 1 3 2
 Jednacina ima dva razlicita realna resenja:
     -1.00 i -2.00
```

Primer 2

```
Interakcija sa programom:
 Unesite koeficijente A, B i C: 1 1 1
 Jednacina nema resenja.
```

[Rešenje 2.1.21]

Zadatak 2.1.22 Napisati program koji za uneti četvorocifreni broj proverava da li su njegove cifre uređene rastuće, opadajuće ili nisu uređene i štampa odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 1389
 Cifre su uredjene neopadajuce.
```

Primer 2

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: -9622
 Cifre su uredjene nerastuce.
```

Primer 3

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 6792
 Cifre nisu uredjene.
```

Primer 4

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 88
 Uneti broj nije cetvorocifren.
```

[Rešenje 2.1.22]

Zadatak 2.1.23 Napisati program koji učitava karakter i:

a) ako je c malo slovo, ispisuje odgovarajuće veliko

b) ako je c veliko slovo, ispisuje odgovarajuće malo

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: >
 *>*
```

Primer 4

```
Interakcija sa programom:
 Unesite karakter: o
 O
```

[Rešenje 2.1.23]

Zadatak 2.1.24 U nizu 12345678910111213....9899 ispisani su redom brojevi od 1 do 99. Napisati program koji za uneti ceo broj k ($1 \geq k \geq 189$) 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 2.1.24]

Zadatak 2.1.25 Data je funkcija $f(x) = 2 \cdot cos(x) - x^3$. Napisati program koji za učitanu vrednost realne promenljive x i vrednost celobrojne promenljive k koje može biti 1, 2 ili 3 izračunava vrednost funkcije $F(k, x) = f(f(f(...f(x)))$ gde je funkcija f primenjena k-puta i ispisuje je zaokruženu na dve decimale. U slučaju neispravnog ulaza, odštampati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite redom x i k: 2.31 2
 F(2.31, 2)=2557.52
```

Primer 2

```
Interakcija sa programom:
 Unesite redom x i k: 12 1
 F(12, 1)=-1726.31
```

Primer 3

```
Interakcija sa programom:
 Unesite redom x i k: 2.31 0
 Greska: nedozvoljena vrednost za k
```

Primer 4

```
Interakcija sa programom:
 Unesite redom x i k: 1 3
 F(1, 3)=-8.74
```

[Rešenje 2.1.25]

Zadatak 2.1.26 Napisati program koji za uneti redni broj dana u nedelji ispisuje ime odgovarajućeg dana. U slučaju pogrešnog unosa ispisati odgovarajuću poruku.

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: nedozvoljeni unos!
```

Primer 4

```
Interakcija sa programom:
 Unesite broj:  2
 U pitanju je: utorak
```

[Rešenje 2.1.26]

Zadatak 2.1.27 Napisati program koji za uneti karakter ispituje da li je samoglasnik.

Primer 1

```
Interakcija sa programom:
 Unesite jedan karakter: A
 Uneti karakter je samoglasnik.
```

Primer 2

```
Interakcija sa programom:
 Unesite jedan karakter: i
 Uneti karakter je samoglasnik.
```

Primer 3

```
Interakcija sa programom:
 Unesite jedan karakter: f
 Uneti karakter nije samoglasnik.
```

Primer 4

```
Interakcija sa programom:
 Unesite jedan karakter: 4
 Uneti karakter nije samoglasnik.
```

[Rešenje 2.1.27]

Zadatak 2.1.28 Napisatiti program koji učitava dva cela broja i jedan od karaktera +, -, *, / ili % i ispisuje vrednost izraza dobijenog primenom date operacije na date argumente. 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!
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite operator i dva cela broja: / 19 5
Rezultat je: 3
```

[Rešenje 2.1.28]

Zadatak 2.1.29 Napisati program koji za uneti dan i mesec ispisuje godišnje doba kojem pripadaju. NAPOMENA: *Podrazumevati da je unos korektan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dan i mesec: 14 10
jesen
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dan i mesec: 2 8
leto
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dan i mesec: 27 2
zima
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dan i mesec: 19 5
prolece
```

[Rešenje 2.1.29]

Zadatak 2.1.30 Napisati program koji za unetu godinu i mesec ispisuje naziv meseca kao i koliko dana ima u tom mesecu te godine.

[Rešenje 2.1.30]

Zadatak 2.1.31 Napisati program koji za uneti datum u formatu *dan.mesec.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 2.1.31]

Zadatak 2.1.32 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 2.1.32]

Zadatak 2.1.33 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.
```

Zadatak 2.1.34 Korisnik unosi tri cela broja: P, Q i R. Nakon toga unosi i dva karaktera, $op1$ i $op2$. Ovi karakteri predstavljaju operacije nad unetim brojevima i imaju naredno značenje:

- karakter **k** predstavlja logičku konjukciju
- karakter **d** predstavlja logičku disjunkciju
- karakter **m** predstavlja relaciju manje
- karakter **v** predstavlja relaciju veće

Program treba da sračuna vrednost izraza `P op1 Q op2 R` i da ga ispiše.

Primer 1

```
Interakcija sa programom:
 Unesite tri cela broja: 0 1 2
 Unesite dva karaktera cela broja: k m
 1
```

Primer 2

```
Interakcija sa programom:
 Unesite tri cela broja: -3 -1 0
 Unesite dva karaktera cela broja: d k
 0
```

*** Zadatak 2.1.35** Program učitava jedan karakter i osam 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:

- ukoliko je uneti karakter 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 unosi 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.

- ukoliko je uneti karakter 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.
- ukoliko je uneti karakter 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 2.1.35]

Zadatak 2.1.36 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) ugrožava polje (m, n)

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

2.2 Rešenja

Rešenje 2.1.1

```c
#include <stdio.h>

int main()
{
  int a, b, min;
  printf("Unesite dva cela broja: ");
  scanf("%d%d", &a, &b);

  /* Promenljiva min dobija vrednost promenljive a. */
  min = a;
```

```
11
   /* Ako je b<a, promenljiva min ce promeniti vrednost tj. bice joj
13    dodeljena vrednost promenljive b. U suprotnom, vrednost ostaje
      ista. */
15
   if (b < a)
17   min = b;

19 printf("Minimum je %d\n", min);

21 return 0;
}
```

Rešenje 2.1.2 Rešenje je analogno rešenju broj 2.1.1.

Rešenje 2.1.3

```
1 #include<stdio.h>

3 int main()
{
5   float x;
   float apsolutno_x;
7
   printf("Unesite jedan realan broj:");
9  scanf("%f", &x);

11 apsolutno_x = x;
   if (x < 0)
13   apsolutno_x = -x;

15 printf("Njegova apsolutna vrednost je %.2f\n", apsolutno_x);

17 /* 2. nacin, pomocu funkcije fabs za koju je neophodno ukljuciti
      zaglavlje math.h: apsolutno_x=fabs(x); */
19 return 0;
}
```

Rešenje 2.1.4

```
1 #include <stdio.h>

3 int main()
{
5   int x;
   float rx;
7
   printf("Unesite jedan ceo broj:");
```

```
  scanf("%d", &x);

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

  /* Proveravamo da li je uneti broj jednak nuli. Ako jeste,
     prekidamo sa daljim izvrsavanjem programa navodjenjem naredbe
     return. Argument -1 u naredbi return oznacava da program nije
     uspesno zavrsen */
  if (x == 0) {
    printf("Nedozvoljeno deljenje nulom\n");
    return -1;
  }

  /* Primenom operatora / na argumente 1 i x dobijamo rezultat
     celobrojnog deljenja ovih argumenata. Da bismo dobili
     kolicnik, koji je realna vrednost, neophodno je da jedan od
     argumenata zapisemo kao realnu vrednost, npr celobrojnu
     vrednost 1 zapisemo kao realnu vrednost 1.0. Ovakav postupak
     se naziva implicitna konverzija. */

  rx = 1.0 / x;
  printf("Reciprocna vrednost unetog broja:%.4f\n", rx);

  return 0;
}
```

Rešenje 2.1.5

```
#include<stdio.h>

int main()
{
  int a, b, c;
  int s;
  printf("Unesite tri cela broja:");
  scanf("%d%d%d", &a, &b, &c);

  /* inicijalizujemo promenljivu s na nulu */
  s = 0;

  /* U naredbi dodele s=s+a vrednost izraza sa desne strane znaka
     jednakosti dodeljujemo promenljivoj sa leve strane znaka
     jednakosti. Staru vrednost promenljive s saberemo sa vrednoscu
     promenljive a i dobijenu vrednost upisemo u promenljivu s. */

  if (a > 0)
    s = s + a;

  /* s+=b je skraceni zapis za s=s+b */
```

```
23    if (b > 0)
        s += b;
25
      if (c > 0)
27      s += c;

29    printf("Suma unetih pozitivnih brojeva: %d\n", s);
      return 0;
31  }
```

Rešenje 2.1.6

```
    #include <stdio.h>
2
    int main()
4   {
      unsigned a, b, c;
6     unsigned min;
      unsigned cena_bez_popusta, cena_sa_popustom;
8
      printf("Unesite cene tri artikla:");
10    scanf("%u%u%u", &a, &b, &c);

12    /*
         Racunamo minimum tri broja. Dodeljujemo promenljivoj min
14       vrednost prvog broja. */
      min = a;
16
      /*
18       Ako je drugi broj manji od minimuma, to znaci da promenljiva
         min ne sadrzi najmanji broj. Dodeljujemo joj vrednost drugog
20       broja. */
      if (min > b)
22      min = b;

24    /*
         Ako je treci broj manji od minimuma, to znaci da promenljiva
26       min ne sadrzi najmanji broj. Dodeljujemo joj vrednost treceg
         broja. */
28    if (min > c)
        min = c;
30
      cena_bez_popusta = a + b + c;
32    cena_sa_popustom = cena_bez_popusta - min + 1;

34    printf("Cena sa popustom: %u\nUsteda: %u\n",
             cena_sa_popustom, cena_bez_popusta - cena_sa_popustom);
36
      return 0;
38  }
```

Rešenje 2.1.7

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int n, j, d, s, h, max;

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

  /* Za slucaj da je broj negativan, uzimamo apsolutnu vrednost
     unetog broja */
  n = abs(n);

  /* Ako uneti broj nije cetvorocifren, ispisujemo poruku o gresci
     i prekidamo izvrsavanje programa. */
  if (n < 1000 || n > 9999) {
    printf("Greska: Niste uneli cetvorocifren broj!\n");
    return -1;
  }

  /* Ako je broj cetvorocifren, izdvajamo cifre broja: j -jedinice,
     d - desetice, s - stotine i h - hiljade */
  j = n % 10;
  d = (n / 10) % 10;
  s = (n / 100) % 10;
  h = n / 1000;

  /* Odredjujemo maksimalnu cifru */
  max = j;

  if (d > max)
    max = d;

  if (s > max)
    max = s;

  if (h > max)
    max = h;

  /* II nacin: if(j>d && j>s && j>h) max=j; if(d>j && d>s && d>h)
     max=d; if(s>j && s>d && s>h) max=s; if(h>j && h>d && h>s)
     max=h; */

  /* Ispisujemo rezultat */
  printf("Najveca cifra je: %d\n", max);
```

```
  return 0;
}
```

Rešenje 2.1.8

```
#include<stdio.h>

int main()
{
  int sati;
  int minuti;
  int preostali_sati;
  int preostali_minuti;

  /* Ukoliko naredbu printf zelimo da napisemo u dva reda, i tom
     prilikom prekidamo deo pod navodnicima, to mozemo uraditi
     navodjenjem navodnika na kraju prvog i na pocetku narednog
     reda: */

  printf("Unesite vreme (broj sati u itervalu [0,24),\n"
         "broj minuta u intervalu [0,60)):");
  scanf("%d%d", &sati, &minuti);

  /* U slucaju da je unos neispravan, ispisujemo poruku o gresci i
     prekidamo dalje izvrsavanje programa.

     Uslov u if naredbi je disjunkcija (operator ||) sastavljena od
     4 disjunkata. Svaki od njih je izraz sa relacionim operatorom
     i ima vrednost 1 ako je izraz tacan i 0 u suprotnom. Da bi
     disjunkcija bila tacna, bar jedan od disjunkata mora da bude
     tacan. Zbog lenjog izracunavanja, vrednost disjunkata ce biti
     racunata do vrednosti prvog disjunkta koji je tacan. To je
     znak da je uslov u if naredbi ispunjen i nema potrebe racunati
     vrednosti drugih disjunkata. */

  if (sati > 24 || sati < 0 || minuti > 59 || minuti < 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  preostali_sati = 24 - sati - 1;
  preostali_minuti = 60 - minuti;

  if (preostali_minuti == 60) {
    preostali_sati++;
    preostali_minuti = 0;
  }

  /* Ukoliko naredbu printf zelimo da napisemo u dva reda i pritom
     ne prekidamo deo pod navodnicima, to mozemo uraditi bez
```

```
46      navodjenja dodatnih karaktera: */
  printf("Do ponoci je ostalo %d sati i %d minuta\n",
48          preostali_sati, preostali_minuti);

50  return 0;
}
```

Rešenje 2.1.9

```
#include <stdio.h>
2
int main()
4 {
  char c;
6  printf("Unesite jedan karakter:");
  scanf("%c", &c);
8
  printf("Uneti karakter: %c, njegov ASCII kod: %d\n", c, c);
10
  /* Razlika izmedju ASCII koda svakog malog i odgovarajuceg
12     velikog slova je konstanta koja se moze sracunati izrazom
     'a'-'A' (i iznosi 32) */
14
  if (c >= 'A' && c <= 'Z')
16    printf("odgovarajuce malo slovo: %c, njegov ASCII kod: ",
          "%d\n", c + ('a' - 'A'), c + ('a' - 'A'));
18
  if (c >= 'a' && c <= 'z')
20    printf("odgovarajuce veliko slovo: %c, njegov ASCII kod: ",
          "%d\n", c - ('a' - 'A'), c - ('a' - 'A'));
22
  return 0;
24 }
```

Rešenje 2.1.10

```
1 #include <stdio.h>

3 int main()
{
5  char c1, c2, c3, c4, c5;
  int broj_malih_slova = 0;
7
  /* Citamo karaktere */
9  printf("Unesite karaktere: ");
  scanf("%c %c %c %c %c", &c1, &c2, &c3, &c4, &c5);
11
  /* Proveravamo da li je prvi karakter malo slovo */
13 if (c1 >= 'a' && c1 <= 'z') {
```

```
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

  /* Proveravamo da li je drugi karakter malo slovo */
  if (c2 >= 'a' && c2 <= 'z') {
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

  /* Proveravamo da li je treci karakter malo slovo */
  if (c3 >= 'a' && c3 <= 'z') {
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

  /* Proveravamo da li je cetvrti karakter malo slovo */
  if (c4 >= 'a' && c4 <= 'z') {
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

  /* Proveravamo da li je peti karakter malo slovo */
  if (c5 >= 'a' && c5 <= 'z') {
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

  /* Ispisujemo rezultat */
  printf("Broj malih slova: %d\n", broj_malih_slova);

  return 0;
}
```

Rešenje 2.1.11

```
#include <stdio.h>
#include <ctype.h>

int main()
{
  /* Broj pojavljivanja slova a i A se inicijalizuje na 0 */
  int br_a = 0;

  /* Funkcija getchar ucitava jedan karakter. Njena povratna
     vrednost je ASCII kod ucitanog karaktera.

     Funkcija tolower za dati karakter vraca: - odgovarajuce malo
     slovo, ako je dati karakter veliko slovo - taj isti karakter,
     u suprotnom Ova funkcija je definisana u biblioteci ctype.h

```

```
    U slucaju da je uslov ispunjen, uvecavamo brojac br_a za jedan
    pomocu operatora inkrementacije ++ */
  if (tolower(getchar()) == 'a')
    br_a++;
  if (tolower(getchar()) == 'a')
    br_a++;
  if (tolower(getchar()) == 'a')
    br_a++;
  if (tolower(getchar()) == 'a')
    br_a++;
  if (tolower(getchar()) == 'a')
    br_a++;

  printf("%d\n", br_a);

  return 0;
}
```

Rešenje 2.1.12

```
#include <stdio.h>
#include <ctype.h>

int main()
{
  int br_cif = 0;

  /* Funkcija isdigit vraca 1 ako je dati karakter cifra i 0 u
     suprotnom. Nalazi se u biblioteci ctype.h. */
  if (isdigit(getchar()))
    br_cif++;
  if (isdigit(getchar()))
    br_cif++;
  if (isdigit(getchar()))
    br_cif++;
  if (isdigit(getchar()))
    br_cif++;
  if (isdigit(getchar()))
    br_cif++;

  printf("%d\n", br_cif);

  return 0;
}
```

Rešenje 2.1.13

```
#include <stdio.h>
```

```c
int main()
{
  int x;
  printf("Unesite 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;
}
```

Rešenje 2.1.14

```c
#include <stdio.h>
#include <stdlib.h>             /* abs */

int main()
{
  int n, j, d, s;

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

  /* Uzimamo apsolutnu vrednost broja za slucaj da je uneti broj
     negativan */
  n = abs(n);

  /* Ako broj nije trocifren, izdajemo poruku o gresci i prekidamo
     dalje izvrsavanje programa */
  if (n < 100 || n > 999) {
    printf("Greska: Niste uneli trocifren broj!\n");
    return -1;
  }

  /*
     Izdvajamo cifre broja: j -jedinice, d - desetice, s - stotine */
  j = n % 10;
  d = (n / 10) % 10;
  s = n / 100;

  /* Proveravamo da li je broj Amstrongov */
  if (n == j * j * j + d * d * d + s * s * s)
    printf("Broj je Amstrongov.\n");
  else
    printf("Broj nije Amstrongov.\n");

  return 0;
```

```
}
```

Rešenje 2.1.15

```
#include <stdio.h>

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

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

  n = abs(n);

  if (n < 1000 || n > 9999) {
    printf("Broj nije cetvorocifren.\n");
    return -1;
  }

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

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

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

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

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

```

```
  /* Proveravamo da li je cifra hiljada parna */
  if (h % 2 == 0) {
    proizvod_parnih = proizvod_parnih * h;
    broj_parnih++;
  }

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

  return 0;
}
```

Rešenje 2.1.16

```
#include <stdio.h>

int main()
{
  int broj;
  scanf("%d", &broj);

  // Da bismo lakse odredili da li je cetvorocifren
  int absBroj = broj < 0 ? -broj : broj;
  if (absBroj <= 999 || absBroj >= 10000) {
    printf("Broj nije cetvorocifren.");
    return -1;
  }

  int a = absBroj % 10;
  int b = (absBroj / 10) % 10;
  int c = (absBroj / 100) % 10;
  int d = absBroj / 1000;

  int max = a, min = a;
  // cuvamo i stepen da bismo lakse zamenili cifre
  /* Ideja: 4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1, ^ ^
     odnosno oduzemo 100 i dodamo 900 */
  int stepenMax = 1, stepenMin = 1;

  if (b > max) {
    max = b;
    stepenMax = 10;
  }
  if (b < min) {
    min = b;
    stepenMin = 10;
```

```
33    }

35    if (c > max) {
        max = c;
37      stepenMax = 100;
      }
39    if (c < min) {
        min = c;
41      stepenMin = 100;
      }
43
      if (d > max) {
45      max = d;
        stepenMax = 1000;
47    }
      if (d < min) {
49      min = d;
        stepenMin = 1000;
51    }

53    int rez;
      /* Ideja: 4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1, ^ ^
55       odnosno oduzemo 100 i dodamo 900 */
      if (broj > 0)
57      rez = broj - max * stepenMax + min * stepenMax
            - min * stepenMin + max * stepenMin;
59    else
        rez = broj + max * stepenMax - min * stepenMax
61          + min * stepenMin - max * stepenMin;

63    printf("%d\n", rez);

65    return 0;
}
```

Rešenje 2.1.20

```
1  #include <stdio.h>

3  int main()
   {
5    int x;
     printf("Unesite jedan ceo broj:");
7    scanf("%d", &x);

9    if (x == 0)
       printf("Broj je jednak nuli\n");
11   else if (x < 0)
       printf("Broj je manji od nule\n");
13   else
       printf("Broj je veci od nule\n");
```

```
15
  return 0;
17 }
```

Rešenje 2.1.21

```
1  #include <stdio.h>
   #include <math.h>
3
   int main()
5  {
     float a, b, c;
7    float D;
     float x1, x2;
9    printf("Unesite koeficijente A, B i C:");
     scanf("%f%f%f", &a, &b, &c);
11
     /* Proveravamo da li je kvadratna jednacina korektno zadata. */
13   if (a == 0)
       if (b == 0)
15       /* slucaj a==0 && b==0 && c==0 */
         if (c == 0)
17         printf("Jednacina ima beskonacno mnogo resenja\n");
     /* slucaj a==0 && b==0 && c!=0 */
19       else
           printf("Jednacina nema resenja\n");
21   /* slucaj a==0 && b!=0 */
       else {
23       x1 = -c / b;
         printf("Jednacina ima jedinstveno realno resenje %.2f\n", x1);
25     }
     /* slucaj a!=0 */
27   else {
       D = b * b - 4 * a * c;
29     if (D < 0)
         printf("Jednacina nema realnih resenja\n");
31     else if (D > 0) {
         /* funkcija sqrt nalazi se u biblioteci math.h (prevodjenje
33          sa -lm opcijom) */
         x1 = (-b + sqrt(D)) / (2 * a);
35       x2 = (-b - sqrt(D)) / (2 * a);
         printf("Jednacina ima dva razlicita realna resenja %.2f ",
37              "i %.2f\n", x1, x2);
       } else {
39       x1 = (-b) / (2 * a);
         printf("Jednacina ima jedinstveno realno resenje %.2f\n", x1);
41     }
     }
43
     return 0;
45 }
```

Rešenje 2.1.22

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int x;
  char c1;
  char c10;
  char c100;
  char c1000;

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

  /* Uzimamo apsolutnu vrednost unetog broja kako u slucaju da je
     negativan ne bismo za cifre dobili negativne brojeve. Funkcija
     abs nalazi se u zaglavlju stdlib.h */
  x = abs(x);

  if (x < 1000 || x > 9999) {
    printf("Uneti broj nije cetvorocifren\n");
    return -1;
  }

  /* Izdvajamo cifre broja. */
  c1 = x % 10;
  c10 = (x / 10) % 10;
  c100 = (x / 100) % 10;
  c1000 = (x / 1000) % 10;

  if (c1000 <= c100 && c100 <= c10 && c10 <= c1)
    printf("Cifre su uredjene neopadajuce \n");
  else if (c1000 >= c100 && c100 >= c10 && c10 >= c1)
    printf("Cifre su uredjene nerastuce \n");
  else
    printf("Cifre nisu uredjene\n");

  return 0;
}
```

Rešenje 2.1.23

```c
#include <stdio.h>

int main()
```

```
{
  char c;

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

  if (c >= 'a' && c <= 'z')
    printf("%c\n", c - 'a' + 'A');
  else if (c >= 'A' && c <= 'Z')
    printf("%c\n", c - 'A' + 'a');
  else if (c >= '0' && c <= '9')
    printf("cifra\n");
  /* Ako nijedan od prethodnih uslova nije ispunjen, bice izvrsena
     naredba u else grani */
  else
    printf("*%c*\n", c);

  return 0;
}
```

Rešenje 2.1.24

```
#include <stdio.h>

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

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

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

    /* Odredjujemo broj dvocifrenih brojeva koji se mogu zapisati
       pomocu k cifara */

    if (k % 2 != 0) {
      /* Ako je k neparan broj, zapisan je ceo broj dvocifrenih
         brojeva

         9 oduzimamo jer je 9 broj cifara potrebnih za zapis
         jednocifrenih brojeva */
      n = (k - 9) / 2;

      /* Broj o kojem se radi je */
      broj = 9 + n;
```

```
31        /* Ujedno, za neparno k se trazi cifra jedinica izdvojenog
             broja */
33        printf("Na %d-toj poziciji je broj %d.\n", k, broj % 10);

35      } else {
          /* Ako je k paran broj, zapisan je ceo broj dvocifrenih
37           brojeva i zapoceto je sa zapisom sledeceg

39           9 oduzimamo jer je 9 broj cifara potrebnih za zapis
             jednocifrenih brojeva */
41        n = (k - 9) / 2 + 1;

43        /* Broj o kojem se radi je */
          broj = 9 + n;
45
          /* Ujedno, za parno k se trazi cifra desetica izdvojenog
47           broja */
          printf("Na %d-toj poziciji je broj %d.\n", k, broj / 10);
49
      }
51  } else {
      printf("Greska: Nedozvoljena vrednost broja k!\n");
53  }

55  return 0;
}
```

Rešenje 2.1.25

```
1  #include <stdio.h>
   #include <math.h>
3
   int main()
5  {
     float x;
7    int k;
     float F;
9
     printf("Unesite redom x i k: ");
11   scanf("%f %d", &x, &k);

13   /* Proveravaju se vrednosti za k */
     if (k < 1 || k > 3) {
15     printf("Greska: nedozvoljena vrednost za k!\n");
       return 0;
17   }
     printf("F(%f,%d)=", x, k);
19
     /* Analiziraju se moguci slucajevi */
21   if (k == 1) {
```

```
    F = 2 * cos(x) - x * x * x;
  } else {
    if (k == 2) {
      x = 2 * cos(x) - x * x * x;
      F = 2 * cos(x) - x * x * x;
    } else {
      x = 2 * cos(x) - x * x * x;
      x = 2 * cos(x) - x * x * x;
      F = 2 * cos(x) - x * x * x;
    }
  }

  /* Ispisuje se rezultat */
  printf("%f\n", F);

  return 0;
}
```

Rešenje 2.1.26

```
#include <stdio.h>

int main()
{
  int dan;

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

  switch (dan) {
  case 1:
    printf("ponedeljak\n");
    break;
  case 2:
    printf("utorak\n");
    break;
  case 3:
    printf("sreda\n");
    break;
  case 4:
    printf("cetvrtak\n");
    break;
  case 5:
    printf("petak\n");
    break;
  case 6:
    printf("subota\n");
    break;
  case 7:
    printf("nedelja\n");
    break;
```

```
  default:
33    printf("Greska: nedozvoljeni unos!\n");
  }
35
  return 0;
37}
```

Rešenje 2.1.27

```
1#include <stdio.h>

3int main()
{
5  char c;
  printf("Unesite jedan karakter:");
7  scanf("%c", &c);

9  /* Da bi se utvrdilo da li je karakter samoglasnik, neophodno je
      proveriti da li odgovara nekom od sledecih karaktera:
11    A,E,I,O,U,a,e,i,o,u */
  switch (c) {
13  case 'A':
  case 'E':
15  case 'I':
  case 'O':
17  case 'U':
  case 'a':
19  case 'e':
  case 'i':
21  case 'o':
  case 'u':
23    printf("Uneti karakter je samoglasnik\n");
    break;
25  default:
    printf("Uneti karakter nije samoglasnik\n");
27    break;
  }
29
  return 0;
31}
```

Rešenje 2.1.28

```
1#include <stdio.h>

3int main()
{
5  char op;
  int x, y;
```

```
7
  printf("Unesite operator i dva cela broja: ");
9 scanf("%c %d %d", &op, &x, &y);

11 switch (op) {
   case '+':
13   printf("Rezultat je: %d\n", x + y);
     break;
15 case '-':
     printf("Rezultat je: %d\n", x - y);
17   break;
   case '*':
19   printf("Rezultat je: %d\n", x * y);
     break;
21 case '/':
     if (y == 0)
23     printf("Greska: deljenje nulom nije dozvoljeno!\n");
     else
25     printf("Rezultat je: %f\n", x * 1.0 / y);
     break;
27 case '%':
     printf("Rezultat je: %d\n", x % y);
29   break;
   default:
31   printf("Greska: nepoznat operator!\n");
   }
33
   return 0;
35 }
```

Rešenje 2.1.29

```
1 #include <stdio.h>

3 int main()
  {
5   int d, m;
    printf("Unesite dan i mesec");
7   scanf("%d%d", &d, &m);

9   /* Argument u naredbi switch mora biti celobrojna promenljiva,
       dok argument u naredbi case mora biti celobrojna konstanta. */
11  switch (m) {
      /* Ispitujemo da li vazi m==1 ili m==2 */
13  case 1:
    case 2:
15    printf("zima\n");
      break;
17  case 3:
      if (d < 21)
19      printf("zima\n");
```

```
    else
      printf("prolece\n");
    break;
  case 4:
  case 5:
    printf("prolece\n");
    break;
  case 6:
    if (d < 21)
      printf("prolece\n");
    else
      printf("leto\n");
    break;
  case 7:
  case 8:
    printf("leto\n");
    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 2.1.30

```
#include <stdio.h>

int main()
{
  int godina;
  int mesec;
  int prestupna;

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

  if (godina < 0) {
```

```
13      printf("Lose uneta godina!\n");
        return -1;
15    }

17    /* Provera da li je godina prestupna, zbog februara */
      if ((godina % 4 == 0 && godina % 100 != 0)
19        || godina % 400 == 0)
        prestupna = 1;
21    else
        prestupna = 0;
23
      printf("Unesite redni broj meseca: ");
25    scanf("%d", &mesec);

27    switch (mesec) {
      case 1:
29      printf("Januar, 31 dan\n");
        break;
31    case 2:
        if (prestupna)
33        printf("Februar, 29 dana\n");
        else
35        printf("Februar, 28 dana\n");
        break;
37    case 3:
        printf("Mart, 31 dan\n");
39      break;
      case 4:
41      printf("April, 30 dana\n");
        break;
43    case 5:
        printf("Maj, 31 dan\n");
45      break;
      case 6:
47      printf("Jun, 30 dana\n");
        break;
49    case 7:
        printf("Jul, 31 dan\n");
51      break;
      case 8:
53      printf("Avgust, 31 dan\n");
        break;
55    case 9:
        printf("Septembar, 30 dana\n");
57      break;
      case 10:
59      printf("Oktobar, 31 dan\n");
        break;
61    case 11:
        printf("Novembar, 30 dana\n");
63      break;
      case 12:
```

```
65      printf("Decembar, 31 dan\n");
        break;
67    default:
        printf("Lose unet redni broj meseca!\n");
69    }

71    return 0;
}
```

Rešenje 2.1.31

```
1  #include <stdio.h>

3  int main()
   {
5    int dan, mesec, godina, dozvoljen_broj_dana;

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

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

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

23   /* Ako je mesec korektan, proveravamo broj dana */
     switch (mesec) {
25   case 1:
     case 3:
27   case 5:
     case 7:
29   case 8:
     case 10:
31   case 12:
       /* Dozvoljeni broj dana za januar, mart, maj, jul, avgust,
33        oktobar i decembar je 31 */
       dozvoljen_broj_dana = 31;
35     break;
     case 2:
37     /* Proveravamo da li je godina prestupna */
       if (godina % 4 == 0 && godina % 100 != 0 || godina % 400 == 0)
39       /* Ako jeste, dozvoljeni broj dana za februar je 29 */
         dozvoljen_broj_dana = 29;
```

```
    else
      /* Ako nije, dozvoljeni broj dana za februar je 28 */
      dozvoljen_broj_dana = 28;
    break;
  case 4:
  case 6:
  case 9:
  case 11:
    /* Dozvoljeni broj dana za april, jun, septembar i novembar je
       30 */
    dozvoljen_broj_dana = 30;
    break;
  }
  /* Proveravamo dan */
  if (dan < 0 || dan > dozvoljen_broj_dana) {
    printf("Datum nije korektan (neispravan dan)!\n");
    return 0;
  }

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

  return 0;
}
```

Rešenje 2.1.32

```
#include <stdio.h>

int main()
{
  int dan, mesec, godina;
  int prethodni_dan, prethodni_mesec, prethodni_godina;

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

  /* Racunamo dan, mesec i godinu prethodnog dana */
  prethodni_dan = dan - 1;
  prethodni_mesec = mesec;
  prethodni_godina = godina;

  /* I po potrebi vrsimo korekcije */

  /* Ako je u pitanju prvi u mesecu */
  if (prethodni_dan == 0) {
    /* Treba korigovati mesec */
    prethodni_mesec = mesec - 1;
    /* Ako je u pitanju januar */
    if (prethodni_mesec == 0) {
```

```
25        /* Treba korigovati i godinu */
          prethodni_mesec = 12;
27        prethodni_godina = godina - 1;
      }
29
      /* Analiziramo redni broj meseca kako bi odredili tacan dan */
31    switch (prethodni_mesec) {
      case 1:
33    case 3:
      case 5:
35    case 7:
      case 8:
37    case 10:
      case 12:
39      prethodni_dan = 31;
        break;
41    case 2:
        if ((prethodni_godina % 4 == 0 && prethodni_godina % 100 != 0)
43          || prethodni_godina % 400 == 0)
          prethodni_dan = 29;
45      else
          prethodni_dan = 28;
47      break;
      case 4:
49    case 6:
      case 9:
51    case 11:
        prethodni_dan = 30;
53    }
  }
55
  /* Ispisujemo datum koji smo izracunali */
57 printf("Prethodni datum: %d.%d.%d\n",
         prethodni_dan, prethodni_mesec, prethodni_godina);
59
  return 0;
61}
```

Rešenje 2.1.33 Rešenje je analogno rešenju zadatka 2.1.32.

Rešenje 2.1.35

```
#include<stdio.h>
2 #include<math.h>

4 int main()
  {
6   char c;
    float x1, y1, x2, y2, x3, y3, x4, y4;
8   float kab, kbc, kad;
```

```
  float dab, dad;
  float delta, deltap, deltaq;
  float O;
  float k, n;

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

  printf("Unesi realne koordinate 4 tacke:");
  scanf("%f%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));
      dad = sqrt(pow(x1 - x4, 2) + pow(y1 - y4, 2));
      O = 2 * dab + 2 * dad;
      printf("Obim pravougaonika je %f\n", O);
    } else
      printf("Tacke ne cine pravougaonik sa stranicama ",
             "koje su paralelne koordinatnim osama\n");
    break;
  case 'h':
    /*
       Ukoliko se tacke A(x1,y1) i B(x2,y2) ne nalaze na pravoj
       koja je paralelna x osi, izracunamo k,n za pravu odredjenu
       tackama A(x1,y1) i B(x2,y2) */
    if ((x1 - x2) != 0) {
      k = (y1 - y2) / (x1 - x2);
      n = y1 - k * x1;
      /*
         Proverimo da li tacke C(x3,y3) i D(x4,y4) nalaze na toj
         pravoj */
      if (y3 == x3 * k + n && y4 == x4 * k + n)
        printf("Tacke su kolinearne, pripadaju pravoj ",
               "y=%f*x+%f\n", k, n);
      else
        printf("Tacke nisu kolinearne\n");
    }
    /*
       Ukoliko se A i B nalaze na pravoj koja je paralelna x osi,
       proverimo da li tacke C(x3,y3) i D(x4,y4) nalaze na toj
       pravoj */
    else if (x3 == x1 && x4 == x1)
      printf("Tacke su kolinearne, pripadaju pravoj ",
             "x=%f\n", x1);
    else
      printf("Tacke nisu kolinearne\n");
    break;
  case 'j':
    delta = x1 * y2 - x2 * y1;
    deltap = x2 * (y4 - y3) - y2 * (x4 - x3);
```

```
    deltaq = x1 * (y4 - y3) - y1 * (x4 - x3);
    if (delta != 0)
      printf("Sistem ima jedinstveno resenje %.2f, %.2f\n",
             deltap / delta, deltaq / delta);
    else if (deltap == 0 && deltaq == 0)
      printf("Sistem je neodredjen ili nema resenja.\n");
    else
      printf("Sistem nema resenja\n");
    break;
  default:
    printf("Nekorektan unos\n");
  }

  return 0;
}
```

2.3 Petlje

Zadatak 2.3.1 Napisati program koji 5 puta ispisuje tekst `Mi volimo da programiramo`.

Primer 1

```
Interakcija sa programom:
 Mi volimo da programiramo.
 Mi volimo da programiramo.
 Mi volimo da programiramo.
 Mi volimo da programiramo.
 Mi volimo da programiramo.
```

[Rešenje 2.3.1]

Zadatak 2.3.2 Napisati program koji učitava ceo broj n i ispisuje n puta tekst `Mi volimo da programiramo`.

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj: 6
 Mi volimo da programiramo.
 Mi volimo da programiramo.
 Mi volimo da programiramo.
 Mi volimo da programiramo.
 Mi volimo da programiramo.
 Mi volimo da programiramo.
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj 0
```

[Rešenje 2.3.2]

Zadatak 2.3.3 Napisati program koji učitava pozitivan ceo broj n a potom ispisuje sve cele brojeve od 0 do n.

Primer 1

```
Interakcija sa programom:
 Unesite ceo pozitivan broj: 4
 0 1 2 3 4
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo pozitivan broj: -10
 Neispravan unos. Promenljiva mora biti
     pozitivna!
```

[Rešenje 2.3.3]

Zadatak 2.3.4 Napisati program koji učitava dva cela broja n i m ispisuje sve cele brojeve iz intervala $[n, m]$.

(a) Koristiti `while` petlju.

(b) Koristiti `for` petlju.

(c) Koristiti `do-while` petlju.

Primer 1

```
Interakcija sa programom:
 Unesite dva cela broja: -2 4
 -2 -1 0 1 2 3 4
```

Primer 2

```
Interakcija sa programom:
 Unesite dva cela broja: 10 6
 Neispravan unos. Nisu dobro zadate granice
     intervala!
```

[Rešenje 2.3.4]

Zadatak 2.3.5 Napisati program koji učitava ceo pozitivan broj i izračunava njegov faktorijel. U slučaju neispravnog unosa ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite pozitivan broj: 18
 Faktorijel = 640237370572800
```

Primer 2

```
Interakcija sa programom:
 Unesite pozitivan broj: 8
 Faktorijel = 40320
```

Primer 3

```
Interakcija sa programom:
 Unesite pozitivan broj: 40
 Broj je veliki, dolazi do
     prekoracenja.
```

[Rešenje 2.3.5]

Zadatak 2.3.6 Sa standradnog ulaza unose se realan broj x i ceo pozitivan broj n. Napisati program koji izračunava n-ti stepen broja x, tj. 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 2.3.6]

Zadatak 2.3.7 Sa standradnog ulaza unose se realan broj x i ceo broj n. Napisati program koji izračunava n-ti stepen broja x.

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 2.3.7]

Zadatak 2.3.8 Pravi delioci celog broja su svi delioci sem jedinice i samog tog broja. Napisati program za uneti ceo pozitivan broj x ispisuje sve njegove prave delioce. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj veci od 0: 100
 2 4 5 10 20 25 50
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj: -6
 neispravan unos.
```

[Rešenje 2.3.8]

Zadatak 2.3.9 Napisati program koji za uneti prirodan broj uklanja sve nule sa njegove desne strane. Ispisati novodobijeni broj.

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 2.3.9]

Zadatak 2.3.10 Napisati program koji učitava ceo broj i ispisuje njegove cifre u obrnutom poretku.

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj: 6789
 9 8 7 6
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj: -892345
 5 4 3 2 9 8
```

[Rešenje 2.3.10]

Zadatak 2.3.11 Napisati program koji za uneti prirodan broj ispisuje da li je on deljiv sumom svojih cifara.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 12
 Deljiv je sumom svojih
     cifara.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 2564
 Nije deljiv sumom svojih
     cifara.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -4
 Neispravan ulaz.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj: 0
 Neispravan ulaz.
```

[Rešenje 2.3.11]

Zadatak 2.3.12 Napisati program koji učitava pozitivan ceo broj n, a zatim učitava n celih brojeva i ispisuje sumu pozitivnih i sumu negativnih unetih brojeva.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 7
 Unesite brojeve:
 8 -50 45 2007 -67 -123 14
 Suma pozitivnih: 2074
 Suma negativnih: -240
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 5
 Unesite brojeve:
 -5 -20 -4 -200 -8
 Suma pozitivnih: 0
 Suma negativnih: -237
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -6
 Neispravan unos.
```

[Rešenje 2.3.12]

Zadatak 2.3.13 Program unosi 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:
 5 8 13 17
 0
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: -4
 Neispravan unos.
```

[Rešenje 2.3.13]

Zadatak 2.3.14 Program učitava 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 pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite brojeve: :2 35 5 -175 -20
 Suma je -15.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: -3
 Neispravan unos.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 10
 Unesite brojeve:
 -5 6 175 -20 -25 -8 42 245 1 6
 Suma je -50.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 6
 Unesite brojeve:
 2205 -1904 2 7 -540 5
 Suma je -535.
```

[Rešenje 2.3.14]

Zadatak 2.3.15 Nikola želi da obraduje baku i da joj kupi jedan poklon u radnji. On na raspolaganju ima m novaca. U radnji se nalazi n artikala i zanima ga koliko ima artikala u radnji čija cena je manja ili jednaka m. Napisati program koji pomaže Nikoli da brzo odrediti broj atikala. Program učitava realan pozitivan broj m, ceo pozitivan broj n i n realnih pozitivnih brojeva različitih od 0. Ispisati koliko artikala ima manju ili jednaku cenu od m. U slučaju greške ispisati odgovarajuću poruku.

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
```

Primer 3

```
Interakcija sa programom:
 Unesite broj m: 2
 Unesite broj n: -4
 Broj artikala ne moze biti negativan.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj m: 30
 Unesite broj n: 4
 Unesite n brojeva: 67 -100 23 98
 Cena ne moze biti negativna.
```

[Rešenje 2.3.15]

Zadatak 2.3.16 Napisati program koji učitava cele brojeve sve dok se ne unese nula. Nakon toga ispisati proizvod onih unetih brojeva koji su pozitivni.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve:
 -87 12 -108 -13 56 0
 Proizvod pozitivnih unetih
     brojeva je 672.
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve:
 -5 -200 -43 0
 Nisu uneseni pozitivni
     brojevi.
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: 0
 Nisu uneseni brojevi.
```

[Rešenje 2.3.16]

Zadatak 2.3.17 Napisati program koji za pozitivan ceo broj proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu.

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!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -235515
 Cifra 5 se nalazi u zapisu!
```

[Rešenje 2.3.17]

Zadatak 2.3.18 Program učitava cele brojeve sve do unosa broja nula 0. Napisati program koji izračunava i ispisuje aritmetičku sredinu unetih brojeva na četiri decimale.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 8 5 6 3 0
 Aritmeticka sredina: 5.5000
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: 762 -12 800 2010 -356 899 -101
     0
 Aritmeticka sredina: 571.7143
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve: 0
 Nisu uneseni brojevi.
```

[Rešenje 2.3.18]

Zadatak 2.3.19 U prodavnici se nalaze artikala čije cene su realni pozitivni brojevi. Program unosi cene artikala sve do unosa broja nula 0. Napisati program koji izračunava i ispisuje prosečnu vrednost cena u radnji.

Primer 1

```
Interakcija sa programom:
 Unesite cene: 8 5.2 6.11 3 0
 Prosecna cena je: 5.5775
```

Primer 2

```
Interakcija sa programom:
 Unesite cene: 6.32 -9
 Cena ne moze biti negativana.
```

Primer 3

```
Interakcija sa programom:
 Unesite cene: 0
 Nisu unesene cene.
```

[Rešenje 2.3.19]

Zadatak 2.3.20 Program učitava ceo pozitivan broj n, a potom n realnih brojeva. Odrediti koliko puta je prilikom unosa došlo do promene znaka. Ispisati dobijenu vrednost.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 10
 Unesite brojeve:
 7.82 4.3 -1.2 56.8 -3.4 -72.1 8.9 11.2 -11.2
     -102.4
 Broj promena je 5.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite brojeve:
 -23.8 -11.2 0 5.6 7.2
 Broj promena je 1.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: -6
 Neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 0
 Broj promena je 0.
```

[Rešenje 2.3.20]

Zadatak 2.3.21 U prodavnici se nalazi n artikala čije cene su realni brojevi. Napisati program koji učitava n, a potom i cenu svakog od n artikala i određuje i ispisuje najmanju cenu.

Primer 1

```
Interakcija sa programom:
 Unesite broj artikla: 6
 Unesite artikle:
 12 3.4 90 100.53 53.2 12.8
 Minimalna cena je: 3.400000
```

Primer 2

```
Interakcija sa programom:
 Unesite broj artikla: 3
 Unesite artikle: 4 -8 92
 Cena ne moze biti negativna.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj artikla: -9
 Neispravan unos.
```

[Rešenje 2.3.21]

Zadatak 2.3.22 Program učitava 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 brojeve:
 18 365 25 1 78
 Broj sa najvecom cifrom desetica je 78.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 8
 Unesite brojeve:
 14 1576 -1267 -89 109 122 306 918
 Broj sa najvecom cifrom desetica je -89.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: -12
 Neispravan unos.
```

[Rešenje 2.3.22]

Zadatak 2.3.23 Program učitava 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
 Najvise cifara ima broj 365.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 7
 Unesite n brojeva:
 3 892 18 21 639 742 85
 Najvise cifara ima broj 892.
```

[Rešenje 2.3.23]

Zadatak 2.3.24 Program učitava ceo pozitivan broj n, a zatim i n celih brojeva. Napisati program koji ispisuje broj sa najvećom vodećom cifrom. Vodeća cifra je cifra najveće težine u zapisu broja. Ukoliko ima više takvih, ispisati prvi.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite n brojeva: 8 964 32 511 27
 Broj sa najvecom vodecom cifrom je 964.
```

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 Unesite n brojeva: 41 669 8
 Broj sa najvecom vodecom cifrom je 8.
```

[Rešenje 2.3.24]

Zadatak 2.3.25 Vršna su merenja nadmorskih visina na određenom delu teritorije i naučnike zanima razlika između najveće i najmanje nadmorske visine. Napisati program koji učitava realne brojeve sve do unosa 0 koji označavaju nadmorske visine i ispisuje razliku najveće i najmanje nadmorske visine.

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 2.3.25]

Zadatak 2.3.26 Program učitava cele pozitivane brojeve 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 $d(x, y) = |y - x|$. Ispisati rezultat.

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 2.3.26]

Zadatak 2.3.27 Napisati program koji uneti prirodan broj transformiše tako što svaku parnu cifru u zapisu broja uveća za jedan. Ispisati novodobijeni broj.

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 2.3.27]

Zadatak 2.3.28 Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre polaznog celog broja, počevši od krajnje desne cifre.

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
```

Primer 4

```
Interakcija sa programom:
 Unesite broj: -67123
 -613
```

[Rešenje 2.3.28]

*** Zadatak 2.3.29** Napisati program koji na osnovu unetog prirodnog broja formira i ispisuje broj koji se dobija izbacivanjem cifara koje su jednake zbiru svojih suseda.

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: -5
 Neispravan unos.
```

[Rešenje 2.3.29]

*** Zadatak 2.3.30** Broj je *palindrom* ukoliko se isto čita i sa leve i sa desne strane. Napisati program koji učitava prirodan broj i proverava da li je učitani broj palindrom.

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 2.3.30]

Zadatak 2.3.31 Fibonačijev niz počinje ciframa 1 i 1, a svaki član se dobija zbirom prethodna dva. Napisati program koji učitava ceo prirodan broj n i određuje i ispisuje n-ti član Fibonačijevog niza.

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj: 10
 Trazeni broj je: 55
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj: -100
 Neispravan unos. Pozicija u Fibonacijevom
 nizu mora biti pozitivan broj koji nije 0!
```

Primer 3

```
Interakcija sa programom:
 Unesite ceo broj: 78
 Trazeni broj je: 375819880
```

Primer 4

```
Interakcija sa programom:
 Unesite ceo broj: 20
 Trazeni broj je: 6765
```

[Rešenje 2.3.31]

*** Zadatak 2.3.32** Niz prirodnih brojeva formira se prema sledećem pravilu:

$$a_{n+1} = \begin{cases} \frac{a_n}{2} & \text{ako je } a_n \text{ parno} \\ \frac{3 \cdot a_n + 1}{2} & \text{ako je } a_n \text{ neparno} \end{cases}$$

Napisati program koji za uneti početni član niza a_0 (ceo pozitivan broj) štampa niz brojeva sve do onog člana niza koji je jednak 1.

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj: 56
 56 28 14 7 11 17 26 13 20 10
 5 8 4 2 1
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj: -48
 Nekorektan unos. Broj mora biti pozitivan.
```

Primer 3

```
Interakcija sa programom:
 Unesite ceo broj: 67
 67 101 152 76 38 19 29 44 22 11
 17 26 13 20 10 5 8 4 2 1
```

Primer 4

```
Interakcija sa programom:
 Unesite ceo broj: 33
 33 50 25 38 19 29 44 22
 11 17 26 13 20 10 5 8 4 2 1
```

[Rešenje 2.3.32]

*** Zadatak 2.3.33** Papir A_0 ima površinu $1m^2$ i odnos stranica $1 : \sqrt{2}$. Papir A_1 dobija se podelom papira A_0 po dužoj ivici. Papir A_2 dobija se podelom A_1 papira po dužoj ivici itd. Napisati program koji za uneti prirodan broj k ispisuje dimenzije papira A_k u milimetrima. Rezultat ispisati kao celobrojne vrednosti.

Primer 1

```
Interakcija sa programom:
 Unesite format papira: 4
 210 297
```

Primer 2

```
Interakcija sa programom:
 Unesite format papira: 3
 297 420
```

Primer 3

```
Interakcija sa programom:
 Unesite format papira: 7
 74 105
```

Primer 4

```
Interakcija sa programom:
 Unesite format papira: 9
 37 52
```

[Rešenje 2.3.33]

Zadatak 2.3.34 Napisati program koji učitava karaktere dok se ne unese karakter tačka, i ako je karakter malo slovo ispisuje odgovarajuće veliko, ako je karakter veliko slovo ispisuje odgovarajuće malo, a u suprotnom ispisuje isti karakter kao i uneti.

Primer 1

```
Interakcija sa programom:
 Danas je Veoma Lep DAN.
 dANAS JE vEOMA lEP dan
```

Primer 2

```
Interakcija sa programom:
 PROGRAMIRANJE 1 je zanimljivo!.
 programiranje 1 JE ZANIMLJIVO!
```

[Rešenje 2.3.34]

Zadatak 2.3.35 Napisati program koji učitava karaktere sve do kraja ulaza, a potom ispisuje broj velikih slova, broj malih slova, broj cifara, broj belina i zbir unetih cifara.

Primer 1

```
Interakcija sa programom:
 Tekst sa brojevima: 124, -8900, 23...
 velika: 1, mala: 15, cifre: 9, beline: 5
 suma cifara: 29
```

Primer 2

```
Interakcija sa programom:
 NEMA cifara!
 velika: 4, mala: 6, cifre: 0, beline: 1
 suma cifara: 0
```

[Rešenje 2.3.35]

Zadatak 2.3.36 Program učitava ceo pozitivan broj n, a potom i n karaktera. Za svaki od samoglasnika ispisati koliko puta se pojavio među unetim karakterima. Ne praviti razliku između malih i velikih slova.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite n karaktera: uAbao
 Samoglasnik a: 2
 Samoglasnik e: 0
 Samoglasnik i: 0
 Samoglasnik o: 1
 Samoglasnik u: 1
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 7
 Unesite n karaktera: jk+EEae
 Samoglasnik a: 1
 Samoglasnik e: 3
 Samoglasnik i: 0
 Samoglasnik o: 0
 Samoglasnik u: 0
```

[Rešenje 2.3.36]

Zadatak 2.3.37 Program učitava 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 2.3.37]

Zadatak 2.3.38 Napisati program koji učitava ceo pozitivan broj n i ispisuje vrednost sume kubova brojeva od 1 do n, odnosno $s = 1 + 2^3 + 3^3 + \ldots + n^3$. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite pozitivan ceo broj: 14
 Suma kubova od 1 do 14: 11025
```

Primer 2

```
Interakcija sa programom:
 Unesite pozitivan ceo broj 25
 Suma kubova od 1 do 25: 105625
```

[Rešenje 2.3.38]

Zadatak 2.3.39 Napisati program koji učitava ceo pozitivan broj n i ispisuje sumu kubova, $s = 1 + 2^3 + 3^3 + \ldots + k^3$, za svaku vrednost $k = 1, \ldots, n.$. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite pozitivan ceo broj: 5
 i=1, s=1
 i=2, s=9
 i=3, s=36
 i=4, s=100
 i=5, s=225
```

Primer 2

```
Interakcija sa programom:
 Unesite pozitivan ceo broj 8
 i=1, s=1
 i=2, s=9
 i=3, s=36
 i=4, s=100
 i=5, s=225
 i=6, s=441
 i=7, s=784
 i=8, s=1296
```

[Rešenje 2.3.39]

Zadatak 2.3.40 Program učitava realan broj x i ceo pozitivan broj n. Napisati program koji izračunava i ispisuje sumu $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \ldots + n \cdot x^n$.

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 2.3.40]

Zadatak 2.3.41 Program učitava realan broj x i ceo pozitivan broj n. Napisati program koji izračunava i ispisuje sumu $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \ldots \frac{1}{x^n}$.

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 2.3.41]

*** Zadatak 2.3.42** Napisati program koji učitava realane brojeve x i *eps* i sa zadatom tačnošću *eps* izračunava i ispisuje sumu $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$. Izračunati sumu u odnosu na tačnost *eps* znači uporediti poslednji član sume sa *eps* i ukoliko je taj poslednji član manji od *eps* prekinuti dalja izračunavanja. Uputstvo: *Prilikom računanja sume koristiti prethodni izračunati član sume u*

računanju sledećeg člana sume. Naime, ako je izračunat član sume $\frac{x^n}{n!}$ *na osnovu njega se lako može dobiti član* $\frac{x^{n+1}}{(n+1)!}$. *Nikako ne računati stepen i faktorijel odvojeno zbog neefikasnosti takvog rešenja i zbog mogućnosti prekoračenja.*

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 2.3.42]

*** Zadatak 2.3.43** Napisati program koji učitava realane brojeve x i *eps* i sa zadatom tačnošću *eps* izračunava i ispisuje sumu $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} \ldots$. Napomena: *Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.*

Primer 1

```
Interakcija sa programom:
 Unesite x: 3
 Unesite tacnost eps: 0.000001
 S=0.049787
```

Primer 2

```
Interakcija sa programom:
 Unesite x: 3.14
 Unesite tacnost eps: 0.01
 S=0.049072
```

[Rešenje 2.3.43]

Zadatak 2.3.44 Napisati program koji učitava realan broj x i prirodan broj n izračunava sumu $S = (1+\cos(x)) \cdot (1+\cos(x^2)) \cdot \ldots \cdot (1+\cos(x^n))$. Napomena: *Voditi računa o efikasnosti rešenja.*

Primer 1

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 3.4 5
 Proizvod = 0.026817
```

Primer 2

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 12 8
 Proizvod = 2.640565
```

[Rešenje 2.3.44]

*** Zadatak 2.3.45** Napisati program koji učitava ceo prirodan broj n i ispisuje vrednost razlomka

$$\cfrac{1}{1+\cfrac{1}{2+\cfrac{1}{3+\cfrac{1}{4+\cfrac{1}{\ldots+\cfrac{1}{(n-1)+\frac{1}{n}}}}}}}.$$

Primer 1

```
Interakcija sa programom:
 Unesite prirodan broj: 4
 Razlomak = 0.697674
```

Primer 2

```
Interakcija sa programom:
 Unesite prirodan broj: 20
 Razlomak = 0.697775
```

Primer 3

```
Interakcija sa programom:
 Unesite prirodan broj: 0
 Neispravan unos.
```

[Rešenje 2.3.45]

*** Zadatak 2.3.46** Napisati program koji računa sumu

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

za unete cele brojeve x i n. Napomena: *Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.*

Primer 1

```
Interakcija sa programom:
 Unesite x i n: 5.6 8
 S=0.735084
```

Primer 2

```
Interakcija sa programom:
 Unesite x i n: 14.32 11
 S=17273.136719
```

Primer 3

```
Interakcija sa programom:
 Unesite prirodan broj: -6
 Neispravan unos.
```

[Rešenje 2.3.46]

*** Zadatak 2.3.47** Program učitava ceo pozitivan broj n veći od 0. Napisati program koji računa proizvod

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

U slučaju greške pri unosu podataka ispisati odgovarajuću poruku. Napomena: *Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.*

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
 Neispravan unos.
```

Primer 4

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

[Rešenje 2.3.47]

*** Zadatak 2.3.48** Program učitava ceo pozitivan neparan broj n. Napisati program koji za uneto n izračunava:

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

U slučaju greške pri unosu podataka ispisati odgovarajuću poruku. NAPOMENA: *Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.*

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
 Neispravan unos
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: -3
 Neispravan unos.
```

[Rešenje 2.3.48]

Zadatak 2.3.49 Program učitava realne brojeve x i a i ceo pozitivan broj n veći od 0. Napisati program koji izračunava:

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

U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dva relana broja x i a:: 3.2 0.2
 Unesite prirodan broj: 5
 Izraz = 135380494030332048.000000
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dva relana broja x i a:: 2 1
 Unesite prirodan broj: 3
 Izraz = 10201.000000
```

Primer 3

```
Interakcija sa programom:
 Unesite dva relana broja x i a:: 2.6 0.3
 Unesite prirodan broj: 3
 Izraz = 5800.970129
```

Primer 4

```
Interakcija sa programom:
 Unesite dva relana broja x i a:: 5.4 7
 Unesite prirodan broj: -2
 Neispravan unos.
```

[Rešenje 2.3.49]

Zadatak 2.3.50 Za unetu pozitivnu celobrojnu vrednost n napisati programe koji ispisuju odgovarajuće brojeve. Pretpostaviti da je unos korektan.

(a) Napisati program koji za unetu pozitivnu celobrojnu vrednost n ispisuje tablicu množenja.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 1
 1
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 2
 1   2
 2   4
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 3
 1   2   3
 2   4   6
 3   6   9
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 4
 1   2   3   4
 2   4   6   8
 3   6   9   12
 4   8   12  16
```

(b) Napisati program koji za uneto n ispisuje sve brojeve od 1 do n^2 pri čemu se ispisuje po n brojeva u jednoj vrsti.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 1   2   3
 4   5   6
 7   8   9
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 1   2   3   4
 5   6   7   8
 9   10  11  12
 13  14  15  16
```

(c) Napisati program koji za uneto n ispisuje tablicu brojeva tako da su u prvoj vrsti svi brojevi od 1 do n, a svaka naredna vrsta dobija se rotiranjem prethodne vrste za jedno mesto u levo.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 1 2 3
 2 3 1
 3 1 2
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 1 2 3 4
 2 3 4 1
 3 4 1 2
 4 1 2 3
```

(d) Napisati program koji za uneto n iscrtava pravougli „trougao" sačinjen od „koordinata" svojih tačaka. „Koordinata" tačke je oblika (i, j) pri čemu $i,\ j = 0, \ldots, n$. Prav ugao se nalazi u gornjem levom uglu slike i njegova koordinata je $(0, 0)$. Koordinata i se uvećava po vrsti, a koordinata j po koloni, pa je zato koordinata tačke koja je ispod tačke $(0, 0)$ jednaka $(1, 0)$, a koordinata tačke koja je desno od tačke $(0, 0)$ jednaka $(0, 1)$.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 1
 (0,0)
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 2
 (0,0) (0,1)
 (1,0)
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 3
 (0,0) (0,1) (0,2)
 (1,0) (1,1)
 (2,0)
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 4
 (0,0) (0,1) (0,2) (0,3)
 (1,0) (1,1) (1,2)
 (2,0) (2,1)
 (3,0)
```

[Rešenje 2.3.50]

Zadatak 2.3.51 Napisati program koji za unet prirodan broj n zvezdicama iscrtava odgovarajuću sliku. Pretpostaviti da je unos korektan.

(a) Slika sadrži kvadrat stranice n sastavljen od zvezdica.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 ***
 ***
 ***
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 ****
 ****
 ****
 ****
```

(b) Slika sadrži rub kvadrata dimenzije n.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 *****
 *   *
 *   *
 *   *
 *****
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 2
 **
 **
```

(c) Slika sadrži rub kvadrata dimenzije n koji i na glavnoj dijagonali ima zvezdice.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 *****
 **  *
 * * *
 *  **
 *****
```

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 4
 ****
 ** *
 * **
 ****
```

[Rešenje 2.3.51]

*** Zadatak 2.3.52** Napisati program koji za uneti prirodan broj n zvezdicama iscrtava slovo X dimenzije n. Pretpostaviti da je unos korektan.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 *   *
  * *
   *
  * *
 *   *
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 3
 * *
  *
 * *
```

[Rešenje 2.3.52]

*** Zadatak 2.3.53** Napisati program koji za uneti prirodan neparan broj n korišćenjem znaka + iscrtava veliko + dimenzije n. Pretpostaviti da je unet prirodan broj.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
   +
   +
 +++++
   +
   +
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 3
  +
 +++
  +
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 4
 Pogresan unos.
```

[Rešenje 2.3.53]

Zadatak 2.3.54 Napisati program koji učitava prirodan broj n, a potom iscrtava odgovarajuću sliku. Pretpostaviti da je unos korektan.

(a) Slika sadrži pravougli trougao sastavljen od zvezdica. Kateta trougla je dužine n, a prav ugao se nalazi u gornjem levom uglu slike.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 ***
 **
 *
```

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 4
 ****
 ***
 **
 *
```

(b) Slika sadrži pravougli trougao sastavljen od zvezdica. Kateta trougla je dužine n, a prav ugao se nalazi u donjem levom uglu slike.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 *
 **
 ***
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 *
 **
 ***
 ****
```

(c) Slika sadrži pravougli trougao sastavljen od zvezdica. Kateta trougla je dužine n, a prav ugao se nalazi u gornjem desnom uglu slike.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 ***
  **
   *
```

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 4
 ****
  ***
   **
    *
```

(d) Slika sadrži pravougli trougao sastavljen od zvezdica. Kateta trougla je dužine n, a prav ugao se nalazi u donjem desnom uglu slike.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
    *
   **
  ***
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
     *
    **
   ***
  ****
```

(e) Slika sadrži trougao sastavljen od zvezdica. Trougao se dobija spajanjem dva pravougla trougla čija kateta je dužine n, pri čemu je prav ugao prvog trougla u njegovom donjem levom uglu, dok je prav ugao drugog trougla u njegovom gornjem levom uglu, a spajanje se vrši po horiznotalnoj kateti.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 *
 **
 ***
 **
 *
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 *
 **
 ***
 ****
 ***
 **
 *
```

(f) Slika sadrži rub jednakokrakog pravouglog trougla čije su katete dužine n. Program učitava karakter c i taj karakter koristi za iscrtavanje ruba trougla.

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 2.3.54]

Zadatak 2.3.55 Napisati program koji učitava ceo broj n, a potom iscrtava odgovarajuću sliku.

(a) Slika sadrži jednakostranični trougao stranice n koji je sastavljen od zvezdica.

Primer 1

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

Primer 2

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

(b) Slika sadrži jednakostranični trougao stranice n koji je sastavljen od zvezdica pri čemu je vrh trougla na dnu slike.

Primer 1

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

Primer 2

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

(c) Slika sadrži trougao koji se dobija spajanjem dva jednakostranična trougla stranice n koji su sastavljeni od zvezdica.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
   *
  ***
 *****
  ***
   *
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 5
     *
    ***
   *****
  *******
 *********
  *******
   *****
    ***
     *
```

(d) Slika sadrži rub jednakostraničnog trougla čija stranica je dužine n.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
   *
  * *
 * * *
```

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
     *
    * *
   *   *
  *     *
 * * * * *
```

(e) Slika se dobija spajanjem dva jednakostranična trougla čija stranica je dužine n. Iscrtavati samo rub trouglova.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
   *
  * *
 * * *
  * *
   *
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 5
     *
    * *
   *   *
  *     *
 * * * * *
  *     *
   *   *
    * *
     *
```

*** Zadatak 2.3.56** Napisati program koji za uneti prirodan broj n iscrtava strelice dimenzije n. Pretpostaviti da je unos korektan.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 *
  *
 ***
  *
 *
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 5
 *
  *
   *
    *
 *****
    *
   *
  *
 *
```

[Rešenje 2.3.56]

*** Zadatak 2.3.57** Napisati program koji učitava ceo broj n, i iscrtava sliku koja se dobija na sledeći način: u prvom redu je jedna zvezdica, u drugom redu su dve zvezdice razdvojene razmakom, treći red je sastavljen od zvezdica i iste je dužine kao i drugi red, četvrti red se sastoji od tri zvezdice razdvojene razmakom, a peti red je sastavljen od zvezdica i iste je dužine kao i četvrti red itd. Ukupna visina slike je n. Pretpostaviti da je unos korektan.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 7
 *
 * *
 ***
 * * *
 *****
 * * * *
 *******
```

[Rešenje 2.3.57]

*** Zadatak 2.3.58** Program učitava prirodne brojeve m i n. Napisati program koji iscrtava jedan do drugog stranice n kvadrata čija je svaka strana sastavljena od m zvezdica razdvojenih prazninom. Podrazumevati da je unos korektan.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5 3
 * * * * * * * * * * * * * * *
 *             *             *
 *             *             *
 *             *             *
 * * * * * * * * * * * * * * *
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4 4
 * * * * * * * * * * * * * * * *
 *       *       *       *       *
 *       *       *       *       *
 * * * * * * * * * * * * * * * *
```

[Rešenje 2.3.58]

*** Zadatak 2.3.59** Program učitava prirodan broj n. Napisati program koji štampa romb sastavljen od minusa u pravougaoniku sastavljenom od zvezdica. Podrazumevati da je unos korektan.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 6
 ************
 *****--*****
 ****----****
 ***------***
 **--------**
 *----------*
 **--------**
 ***------***
 ****----****
 *****--*****
 ************
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 2
 ****
 *--*
 ****
```

[Rešenje 2.3.59]

Zadatak 2.3.60 Napisati program koji učitava ceo broj n ($n \geq 2$) i koji iscrtava sliku kuće sa krovom: kuća je kocka stranice n, a krov jednakostranični trougao stranice n. Pretpostaviti da je unos korektan.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 4
    *
   * *
  *   *
 * * * *
 *     *
 *     *
 * * * *
```

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
   *
  * *
 * * *
 *   *
 * * *
```

[Rešenje 2.3.60]

*** Zadatak 2.3.61** Program učitava ceo pozitivan broj n. Napisati program koji ispisuje brojeve od 1 do n, zatim od 2 do $n-1$, 3 do $n-2$, itd. Ispis se završava kada nije moguće ispisati ni jedan broj. Za neispravan unos, program ispisuje odgovarajuću poruku. Pretpostaviti da je unos korektan.

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 3

```
Interakcija sa programom:
 Unesite broj n: 7
 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 2.3.61]

*** Zadatak 2.3.62** Napisati program koji učitava ceo pozitivan broj n i 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 pri unosu podataka odštampati ogovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 1 2 3
 1 3
 1
```

Primer 2

```
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 3

```
Interakcija sa programom:
 Unesite broj n: 1
 1
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: -23
 Neispravan unos.
```

[Rešenje 2.3.62]

2.4 Rešenja

Rešenje 2.3.1

```c
#include <stdio.h>

int main()
{
  /* Promenljiva i kontrolise koliko puta ce se petlja izvrsiti i
     naziva se brojac petlje. Njenu pocetnu vrednost postavljamo na
     0 jer se u pocetku petlja nije ni jednom izvrsila. */
  int i = 0;

  /* Pre ulaska u telo petlje proverava se da li je ispunjen uslov
     ulaska u petlju. */
  while (i < 5) {
    /* Ukoliko uslov ulaska u petlju jeste ispunjen, izvrsava se
       telo petlje. */
    /* Ispisujemo trazeni tekst. */
    printf("Mi volimo da programiramo.\n");

    /* Uvecava se brojac za jedan jer je jednom izvrseno telo
       petlje. Ako bi ova vrednost ostala nepromenjena, petlja bi
       se izvrsavala beskonacno. */
    i++;

    /* Nakon poslednje naredbe tela petlje ponovo se ispituje uslov
       petlje. */
  }

  return 0;
}
```

Rešenje 2.3.2

```c
#include<stdio.h>

int main()
{
  /* Brojac u petlji. */
  int i = 0;
  /* Promenljiva koja oznacava koliko puta cemo ispisati trazeni
     tekst. */
  int n;

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

  /* Pre ulaska u telo petlje proverava se da li je ispunjen uslov
```

```
    ulaska u petlju. */
  while (i < n) {
    printf("Mi volimo da programiramo.\n");
    i++;
  }

  return 0;
}
```

Rešenje 2.3.3

```
#include <stdio.h>

int main()
{
  /* Promenljivu x koristimo u dve svrhe. Prvo, ova promenljiva
     kontrolise koliko puta se petlja izvrsila. Drugo, ovu
     promenljivu koristimo za ispis potrebnih vrednosti. */
  int x;
  /* Promenljiva n se unosi i odredjuje koliko brojeva ispisujemo. */
  int n;

  printf("Unesi pozitivan ceo broj: ");
  scanf("%d", &n);

  /* U slucaju neispravnih podataka ispisujemo odgovarajucu poruku
     i izlazimo iz programa. */
  if (n < 0) {
    printf("Neispravan unos. Promenljiva mora biti pozitivna!\n");
    return -1;
  }

  /* Ispis pocinjemo od 0, zato promenljivu x postavljamo na 0. */
  x = 0;
  while (x <= n) {
    /* Ispisujemo broj. */
    printf("%d\n", x);
    /* Uvecavamo promenljivu za jedan jer smo broj ispisali i sada
       zelimo da ispisemo sledeci broj. */
    x++;
  }

  return 0;
}
```

Rešenje 2.3.4

```
/* Resenje pod a). */
```

```c
#include <stdio.h>

int main()
{
  /* Promenljive koje oznacavaju granice intervala. */
  int n, m;
  /* Promenljiva koja oznacava trenutno ispisani broj intervala. */
  int i;

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

  if (m < n) {
    printf
        ("Neispravan unos. Nisu dobro zadate granice intervala!\n");
    return -1;
  }

  /* Na pocetku ispisujemo prvi broj intervala, a to je n. */
  i = n;
  /* uslov petlje se proverava pre ulaska u telo petlje */
  while (i <= m) {
    printf("%d ", i);
    i++;
  }

  printf("\n");


  return 0;
}
```

```c
/* Resenje pod b). */

#include <stdio.h>

int main()
{

  /* Promenljive koje oznacavaju granice intervala. */
  int n, m;
  /* Promenljiva koja oznacava trenutno ispisani broj intervala. */
  int i;

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

  if (m < n) {
    printf
        ("Neispravan unos. Nisu dobro zadate granice intervala!\n");
    return -1;
  }
```

```
21
23   /* naredba i=n se izvrsava jednom, pre prve iteracije */
     for (i = n; i <= m; i++)      /* uslov petlje i<=m se proverava
25                                    pre svake iteracije */
       printf("%d ", i);           /* naredba i++ se izvrsava nakon
27                                    svake iteracije */

29   printf("\n");

31
     return 0;
33 }
```

```
1  /* Resenje pod c). */
   #include <stdio.h>
3
   int main()
5  {
     /* Promenljive koje oznacavaju granice intervala. */
7    int n, m;
     /* Promenljiva koja oznacava trenutno ispisani broj intervala. */
9    int i;

11   printf("Unesi dva cela broja: ");
     scanf("%d%d", &n, &m);
13
     if (m < n) {
15     printf
           ("Neispravan unos. Nisu dobro zadate granice intervala!\n");
17     return -1;
     }
19
     /* Uslov petlje se proverava na kraju svake iteracije. */
21   /* Zbog toga se do while petlja izvrsava bar jednom, cak i u
        slucaju da uslov petlje nikada nije ispunjen. */
23   i = n;
     do {                          /* Petlja se zapocinje bez provere
25                                    uslova. */
       printf("%d ", i);           /* Stampa se vrednost promenljive
27                                    i. */
       i++;                        /* Uvecava se vrednost promenljive
29                                    i. */
     }
31   while (i <= m);               /* Proverava se uslov i ukoliko je
                                      ispunjen, nastavlja se sa
33                                    sledecom iteracijom. */
     /* U suprotnom, petlja se zavrsava i program se nastavlja od prve
35      naredbe koja sledi za petljom. */
     printf("\n");
37
     return 0;
```

```
}
```

Rešenje 2.3.5

```
#include<stdio.h>

int main()
{
  int x;
  /* U promenljivoj f se pamti izracunati faktorijel. Kako
     faktorijel je jako veliki broj, za tip podataka se uzima
     unsigned long, da bi mogla da se upise sto veca vrednost. */
  unsigned long f;
  int i;
  int original;

  printf("Unesite pozitivan broj: ");
  scanf("%d", &x);

  if (x < 0) {
    printf("Nekorektan unos\n");
    return -1;
  }

  if (x >= 22) {
    printf("Broj je veliki, dolazi do prekoracenja.\n");
    return -1;
  }

  original = x;
  f = 1;

  while (x > 1) {
    f = f * x;
    x--;
  }

  printf("Faktorijel = %lu\n", f);

  return 0;
}
```

Rešenje 2.3.6

```
#include <stdio.h>

int main()
{
  int n;
```

```
  float x;
  float vrednost;
  unsigned exp;

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

  if (n < 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

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

  for (exp = 1; exp <= n; exp++)
    vrednost = vrednost * x;

  printf("%f\n", vrednost);

  return 0;
}
```

Rešenje 2.3.7

```
#include <stdio.h>

int main(void)
{
  int n, n_abs;
  float x;
  float vrednost;
  unsigned exp;

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

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

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

  /* Ukoliko je stepen bio negativan treba odrediti 1/x^n, sto je
     zapravo 1/vrednost. */
  if (n < 0) {
    printf("%.3f\n", 1 / vrednost);
  } else {
    printf("%.3f\n", vrednost);
```

```
27    }

29    return 0;
}
```

Rešenje 2.3.8

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

int main()
{
  int x;
  /* Brojac u petlji. */
  int i;

  /* Ucitavamo broj. */
  printf("Unesi ceo broj veci od 0: ");
  scanf("%d", &x);

  if (x <= 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  /* 1. nacin */
  printf("------------ 1. nacin ------------\n");
  for (i = 2; i < x; i++) {
    /* Proverava se da li i deli broj x i ako je to slucaj ispusje
       se i. */
    if (x % i == 0)
      printf("%d \n", i);
  }

  /* 2. nacin (brzi) -- Ne proveravaju se svi brojevi od 2 do x,
     vec se petlja izvrsava dok ne stignemo do korena broja. */
  printf("------------ 2. nacin ------------\n");
  for (i = 2; i <= sqrt(x); i++) {
    /* Proveravamo da li i deli broj x. */
    if (x % i == 0)
      /* U slucaju kada je delilac koren broja, npr. 4 za 16,
         ispisujemo ga jednom. */
      if (i == x / i)
        printf("%d \n", i);
    /* U suprotnom, npr. 2 za 16, ispisujemo i 2 i 8. */
      else
        printf("%d %d \n", i, x / i);
  }

  return 0;
}
```

Rešenje 2.3.9

```c
#include <stdio.h>

int main()
{
  int n;

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

  if (n == 0) {
    printf("0\n");
  } else {
    /* Sve dok je poslednja cifra u zapisu broja n nula */
    while (n % 10 == 0) {
      /* Broj delimo sa 10 tj. uklanjamo mu nulu sa kraja */
      n = n / 10;
    }

    /* Ispisujemo rezultat */
    printf("%d\n", n);
  }

  return 0;
}
```

Rešenje 2.3.10

```c
#include<stdio.h>
#include<stdlib.h>

int main()
{
  int x;
  char cifra;
  printf("Unesi ceo broj:");
  scanf("%d", &x);

  /* Pretvaranje u apsolutnu vrednost se vrsi za slucaj kada je
     unet negativan broj kako bismo osigurali da ce nam izdvojene
     cifre biti pozitivne. */
  x = abs(x);

  /* Kako uklanjamo cifre broja (pogledati telo petlje) u nekom
     trenutku broj ce postati 0 jer smo uklonili sve njegove cifre.
```

```
     Tada prekidamo rad petlje. */
  while (x > 0) {
    /* Izdvajamo poslednju cifru broja x. */
    cifra = x % 10;
    printf("%d\n", cifra);
    /* Uklanjamo poslednju cifru broja x. */
    x /= 10;
  }

  return 0;
}
```

Rešenje 2.3.11

```
#include <stdio.h>

int main()
{
  /* Prirodni broj koji se unosi. */
  int n;
  /* Promenljiva u koju se smesta suma cifara broja. */
  int suma = 0;
  /* Pomocna promenljiva u koju se smesta unesen broj. */
  int pom_n;

  printf("Unesi broj ");
  scanf("%d", &n);

  /* U zadatku pise da se unosi prirodan broj, sto znaci da treba
     proveriti da li je veci od 0 */

  if (n <= 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  /* Potrebno je koristiti pomocnu promenljivu jer u telu petlje se
     odstranjuju cifre broja i na taj nacin uneseni broj se menja.
     Nakon rada petlje potrebno je ponovo koristiti uneseni broj, a
     to znaci da treba sacuvati neizmenjen broj. */
  pom_n = n;

  while (pom_n != 0) {
    /* Na sumu dodajemo poslednju cifru. */
    suma += pom_n % 10;
    /* Sa broja skidamo poslednju cifru. */
    pom_n /= 10;
  }

  if (n % suma == 0)
    printf("Deljiv je sumom svojih cifara.\n");
```

```
  else
    printf("Nije deljiv sumom svojih cifara.\n");

  return 0;
}
```

Rešenje 2.3.12

```
#include<stdio.h>

int main()
{
  int n;
  /* Oznaka broja koji unosimo u jednoj iteraciji petlje. */
  int x;
  int suma_poz;
  int suma_neg;
  /* Brojac. */
  int i;

  printf("Unesi pozitivan ceo broj:");
  scanf("%d", &n);

  if (n < 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  /* Promenljivama koje ce sadrzati sume se pre ulaska u petlju
     dodeljuje 0 (neutral za sabiranje). */
  suma_poz = 0;
  suma_neg = 0;
  i = 0;

  printf("Unesite brojeve: ");
  while (i < n) {
    scanf("%d", &x);

    if (x < 0)
      suma_neg += x;
    else
      suma_poz += x;

    i++;
  }

  printf(" Suma pozitivnih: %d\n Suma negativnih: %d\n", suma_poz,
         suma_neg);
  return 0;
}
```

Rešenje 2.3.13

```c
#include <stdio.h>

int main()
{
  /* Promenljiva x oznacava tekuci uneti broj. */
  int n, x;
  /* Brojac. */
  int i;
  int zbir = 0;

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

  if (n < 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  printf("Unesite n brojeva: ");

  /* Inicijalizuje se brojac sa kojim se kontrolise broj *
     ucitavanja - treba da ih bude tacno n. */
  i = 0;
  while (i < n) {
    /* Ucitava se broj. */
    scanf("%d", &x);

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

    /* Uvecava se brojac iteracija. */
    i++;
  }

  /* Ispisuje se rezultat. */
  printf("%d\n", zbir);

  return 0;
}
```

Rešenje 2.3.14

```c
#include <stdio.h>

int main()
```

```
{
  int n, broj;
  int suma = 0;
  /* Brojac. */
  int i;

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

  if (n < 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  printf("Unesite brojeve: ");
  for (i = 0; i < n; i++) {
    scanf("%d", &broj);

    if (broj % 5 == 0 && broj % 7 != 0)
      suma += broj;
  }

  printf("Suma je %d.\n", suma);

  return 0;
}
```

Rešenje 2.3.15

```
#include <stdio.h>

int main()
{
  /* Promenljiva cena oznacava trenutno unesenu cenu. */
  float cena;
  float m;
  int n, i;
  int broj_brojeva = 0;

  printf("Unesite koliko novaca ima Nikola: ");
  scanf("%f", &m);

  if (m < 0) {
    printf("Nikola ne moze imati negativno novaca.\n");
    return -1;
  }

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

  if (n < 0) {
```

```
23      printf("Broj artikala ne moze biti negativan.\n");
        return -1;
25    }

27    printf("Unesite cene artikala: ");

29    i = 0;
      while (i < n) {
31      /* Ucitava se cena artikla. */
        scanf("%f", &cena);
33
        if (cena <= 0) {
35        printf("Cena ne moze biti negativna.\n");
          return -1;
37      }

39      /* Provera da li je cena manji od zadatog broja m. */
        if (cena < m) {
41        /* Ako jeste, uvecava se brojac brojeva za 1. */
          broj_brojeva++;
43      }

45      i++;
      }
47
      printf("%d\n", broj_brojeva);
49
      return 0;
51  }
```

Rešenje 2.3.16

```
1   #include <stdio.h>
    int main()
3   {
      int x;
5
      /* U promenljivoj p se cuva prozivod. */
7     int p;

9     /* Promenljiva u sluzi za proveru da li su brojevi uopste
         uneseni. Na pocetku se pretpostavlja da nisu i postavlja se na
11       0. */
      int u = 0;
13
      /* Promenljiva unesen_pozitivan sluzi za proveru da li su
15       pozitivni brojevi uopste uneseni. Na pocetku se pretpostavlja
         da nisu i postavlja se na 0. */
17    int unesen_pozitivan = 0;

19    p = 1;
```

```
  /* Izraz 1 je konstantan, razlicit je od nule sto znaci da je to
     tacan izraz. Uslov petlje je uvek tacan! */
  printf("Unesite brojeve:");

  while (1) {
    scanf("%d", &x);

    /* Proveravanje da li je uneta nula. */
    if (x == 0)

      /* Naredba break prekida petlju. Izvrsavanje se nastavlja od
         prve naredbe nakon petlje. */
      break;

    /* Ako je makar 1 broj razlicit od 0 promenljiva u ce biti
       postavljena na 1. */
    u = 1;

    /* Ako je unet negativan broj, taj broj se ne mnozi sa ukupnim
       proizvodom p; zato se nastavlja dalje. */
    if (x < 0)
      /* Naredba continue prekida trenutnu iteraciju petlje tako
         sto preskace sve naredbe koje nakon njega slede.
         Izvrsavanje se nastavlja od provere uslova petlje. */
      continue;

    /* Ako je makar jedan broj pozitivan, promenljiva
       unesen_pozitivan se postavja na 1. */
    unesen_pozitivan = 1;
    p = p * x;
  }

  if (u == 0)
    printf("Nisu uneseni brojevi.\n");
  else if (unesen_pozitivan == 0)
    printf("Nisu uneseni pozitivni brojevi. \n");
  else
    printf("Proizvod pozitivnih unetih brojevi je %d.\n", p);

  return 0;
}
```

Rešenje 2.3.17

```
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int n, cifra;
```

```
  int indikator = 0;

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

  if (n < 0)
    n = abs(n);

  /* Sve dok imamo cifara u zapisu broja. */
  while (n > 0) {

    /* Izdvajamo posledjnju cifru broja. */
    cifra = n % 10;

    /* Proveravamo da li je bas ona jednaka broju 5 */
    if (cifra == 5) {
      /* Ako jeste postavljamo indikator na vrednost 1 tako da
         znamo da smo pronasli peticu i prekidamo sa izvrsavanjem
         petlje. */
      indikator = 1;
      break;
    }
    /* Ako izvdvojena cifra nije jednaka broju 5, broj delimo sa 10
     * kako bi mogli da izdvojimo i preostale cifre broja na isti
     * nacin.
     */
    n = n / 10;
  }

  /* Ispisujemo rezultat */
  if (indikator == 0) {
    printf("Cifra 5 se ne nalazi u zapisu!\n");
  } else {
    printf("Cifra 5 se nalazi u zapisu!\n");
  }

  return 0;
}
```

Rešenje 2.3.18

```
#include <stdio.h>

int main()
{
  int x;
  int broj_brojeva;
  int suma;

  broj_brojeva = 0;
```

```
  suma = 0;

  printf("Unesite brojeve: ");

  while (1) {
    /* Ucitavanje broja. */
    scanf("%d", &x);

    /* Ako je unesena 0, prekida se petlja. */
    if (x == 0)
      break;

    /* Procitani broj dodaje se na sumu. */
    suma += x;

    /* I uvecava se broj ucitanih brojeva. */
    broj_brojeva++;
  }

  if (broj_brojeva == 0)
    printf("Nisu uneseni brojevi.\n");
  else
    /* Prilikom deljenja celih brojeva kao rezultat se dobija ceo
       broj. Kako je aritmeticka sredina realan broj, potrebno je
       izvrsiti konverziju prilikom deljenja da bi se dobio
       ispravan rezultat. */
    printf("Aritmeticka sredina: %.4f\n",
           (double) suma / broj_brojeva);

  return 0;
}
```

Rešenje 2.3.19

```
#include <stdio.h>

int main()
{
  float cena;
  int broj_artikla;
  float suma;

  broj_artikla = 0;
  suma = 0;

  printf("Unesite cene: ");

  while (1) {
    scanf("%f", &cena);

    if (cena == 0)
```

```
      break;

    if (cena < 0) {
      printf("Cena ne moze biti negativna.\n");
      return -1;
    }

    suma += cena;

    /* I uvecava se broj ucitanih brojeva. */
    broj_artikla++;
  }

  if (broj_artikla == 0)
    printf("Nisu unesene cene.\n");
  else
    printf("Aritmeticka sredina: %.4f\n", suma / broj_brojeva);

  return 0;
}
```

Rešenje 2.3.20

```
#include <stdio.h>

int main()
{
  int n;
  /* Ucitavaju se dva broja, broj i sledbenik, i proverava se da li
     su razlicitog znaka. */
  double broj, sledbenik;
  /* Brojac. */
  int i;
  int broj_promena = 0;

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

  if (n < 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  /* Prvo se proveara da li uopste ima unosa, i ako unosa nema,
     ispisuje se odgovarajuca poruka i izlazi iz programa. */
  if (n == 0) {
    printf("Broj promena je 0.\n");
    return 0;
  }

  printf("Unesite brojeve: ");
```

```
29    /* Pre petlje ucitava se jedan broj, a u petlji se ucitava njegov
         sledbenik i proverava se da li su razlicitog znaka. */
31    scanf("%lf", &broj);

33    /* Kako je vec jedan broj unesen, brojac se postavlja na 1, a ne
         na 0. */
35    for (i = 1; i < n; i++) {
        /* Ucitava se sledbenik. */
37      scanf("%lf", &sledbenik);

39      /* Ako su razlicitog znaka proizvod je manji od 0. */
        if (sledbenik * broj < 0)
41        broj_promena++;
        /* Problem je ako je proizvod jednak 0. Tada mora provera da li
43         je jedan od brojeva negativan jer tada postoji promena
           znaka. */
45      else if (sledbenik * broj == 0 && (sledbenik < 0 || broj < 0))
          broj_promena++;
47
        /* Tekuci sledbenik postaje tekuci broj, a u sledecoj iteraciji
49         petlje se ucitava sledeci sledbenik. */
        broj = sledbenik;
51    }

53    printf("Broj promena je %d.\n", broj_promena);

55    return 0;
  }
```

Rešenje 2.3.21

```
1   #include <stdio.h>

3   int main()
    {
5     /* Broj artikala. */
      int n;
7     /* Brojac. */
      int i;
9     /* Cena trenutno unetnog artikla. */
      float cena;
11    /* Minimalna cena. */
      float min_cena;
13
      printf("Unesite broj artikala:");
15    scanf("%d", &n);

17    if (n <= 0) {
        printf("Neispravan unos\n");
19      return -1;
      }
```

```
21
   /* Prva cena se unosi iznad petlje kako bi bio njegova vrednost
23    bila dodeljena promenljivoj min_cena. Neophodno je da
      promenljiva min bude inicijalizovana pre ulaska u petlju da bi
25    uslov x<min mogao da bude ispitan u prvoj iteraciji. */
   printf("Unesite cenu artikala:");
27 scanf("%f", &cena);
   /* Proveravamo da li je cena isprano uneta vrednost. */
29 if (cena <= 0) {
     printf("Cena ne moze biti negativna.\n");
31   return -1;
   }
33
   min_cena = cena;
35 i = 0;
   while (i < n - 1) {
37   scanf("%f", &cena);

39   if (cena <= 0) {
       printf("Cena ne moze biti negativna.\n");
41     return -1;
     }
43
     /* Provera da li je uneta cena manja od tekuce minimalne cene. */
45   if (cena < min_cena)
       min_cena = cena;
47   i++;
   }
49
   printf("Minimalna cena je: %f\n", min_cena);
51
   return 0;
53 }
```

Rešenje 2.3.22

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   int main()
5  {
     int n;
7    int x, x_desetica;
     int max_desetica, broj;
9    int i;

11   printf("Unesite broj n: ");
     scanf("%d", &n);
13
     if (n < 0) {
15     printf("Neispravan unos.\n");
```

```
    return -1;
  }

  if (n == 0) {
    printf("Nisu uneseni brojevi.\n");
    return 0;
  }

  /* Maksimalna cifra desetice se postavlja na na 0 jer 0 je
     svakako najmanja cifra pa je pocetna vrednost neutralna tj. ne
     moze da utice na izracunavanje maksimuma. Ipak, treba biti
     pazljiv jer nije uvek zgodno pretpostaviti da je maksimalna
     vrednost 0. Na primer, ako je zadatak naci maksimum celih
     brojeva, a korisnik unese -32 -7 i -22, maksimalni je broj -7,
     sto je manje od 0. */
  max_desetica = 0;

  printf("Unesite brojeve: ");
  for (i = 0; i < n; i++) {
    scanf("%d", &x);

    /* Izdvajanje cifre desetica procitanog broja. */
    x_desetica = (abs(x) / 10) % 10;

    /* Proverava da li je izdvojena cifra veca od trenutne
       maksimalne cifre desetica. */
    if (x_desetica > max_desetica) {
      /* Ako jeste vece, pamti se nova najveca cifra, kao i broj u
         kom se pojavila. */
      max_desetica = x_desetica;
      broj = x;
    }
  }

  printf("Broj sa najvecom cifrom desetica je %d\n", broj);

  return 0;
}
```

Rešenje 2.3.23

```
#include <stdio.h>
#include <stdlib.h>

int main()
{

  int n;
  int x, x_kopija, br_cifara;
  int max_br_cifara, broj;
  int i;
```

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

15 if (n < 0) {
     printf("Neispravan unos.\n");
17   return -1;
   }
19
   if (n == 0) {
21   printf("Nisu uneseni brojevi.\n");
     return 0;
23 }

25 /* Maksimalan broj cifara se postavlja na 0, svaki broj ima vise
      od 0 cifara pa je ova vrednost neutralna. */
27 max_br_cifara = 0;

29 printf("Unesite n brojeva: ");
   for (i = 0; i < n; i++) {
31   scanf("%d", &x);

33   /* Odredjivanje broja cifara unetog broja x. */
     x_kopija = abs(x);
35   br_cifara = 0;
     while (x_kopija != 0) {
37     x_kopija = x_kopija / 10;
       br_cifara++;
39   }

41   /* Ako je broj cifara unetog broja veci od maksimalnog */
     if (br_cifara > max_br_cifara) {
43     /* Cuvamo ga */
       max_br_cifara = br_cifara;
45     /* I zbog ispisa rezultata, cuvamo i originalni broj */
       /* Zbog ovoga smo morali i da racunamo broj cifara nad
47        kopijom broja x kako ne bismo promenili njegovu vrednost */
       broj = x;
49   }
   }
51
   printf("Najvise cifara ima broj %d\n", broj);
53
   return 0;
55 }
```

Rešenje 2.3.24

```
#include <stdio.h>
2 #include <math.h>
```

```c
int main()
{

  int n;
  int x, x_kopija;
  int broj;
  int vodeca_cifra, max_vodeca_cifra;
  int i;

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

  /* Postavljamo maksimalnu vodecu cifru na 0 - cifre broja su vece
     ili jednake od 0 pa je ova vrednost neutralna */
  max_vodeca_cifra = 0;

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

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

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

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

  return 0;

}
```

Rešenje 2.3.25

```c
#include <stdio.h>
```

```c
int main()
{
  int x;
  int min, max;

  printf("Unesite brojeve: ");

  /* Prvi broj se ucitava izvan petlje zbog inicijalizacije
     maksimuma i minimuma. */
  scanf("%d", &x);
  max = x;
  min = x;

  /* Sve dok se ne unese 0, ucitavaju se brojevi u petlji. */
  while (x != 0) {

    /* Provera da li je procitani broj veci od aktuelnog maksimuma.
     */
    if (x > max)
      max = x;
    /* Provera da li je procitani broj manji od aktuelnog minimuma.
     */
    if (x < min)
      min = x;

    /* Ucitavanje narednog broja. */
    scanf("%d", &x);
  }

  printf("Razlika: %d\n", max - min);

  return 0;
}
```

Rešenje 2.3.26

```c
#include <stdio.h>
#include <stdlib.h>

int main()
{
  int n;
  int d;
  /* Uzastopni brojevi za koje se racuna rastojanje. */
  int x, y;
  int broj_parova;
  int i;


  printf("Unesite brojeve n i d: ");
  scanf("%d %d", &n, &d);
```

```
  if (n < 0 || d < 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  broj_parova = 0;

  printf("Unesite n brojeva: ");

  /* Prvi broj se ucitava pre petlje. */
  scanf("%d", &x);

  for (i = 1; i < n; i++) {
    scanf("%d", &y);

    /* Provera da li su brojevi na rastojanju d. */
    if (abs(y - x) == d)
      broj_parova++;

    /* Broj iz tekuce iteracije se cuva kako bi mogao da se
       upotrebljava u narednoj iteraciji. */
    x = y;
  }

  printf("Broj parova: %d\n", broj_parova);

  return 0;
}
```

Rešenje 2.3.27

```
#include <stdio.h>

int main()
{
  int x;
  /* Tezina trenutne pozicije u broju. Moze biti 1, 10, 100, 1000
     itd. */
  int pozicija;
  /* Trenutna izdvojena cifra iz broja x. */
  int cifra;
  /* Broj dobijen nakon transformacije. */
  unsigned int y;

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


  if (x <= 0) {
    printf("Nekorektan unos.\n");
```

```
    return -1;
  }

  /* Posto pocinjemo sa izdvajanjem cifara od cifre jedinica,
     postavlja se tezinu (stepen) pozicije na 1. */
  pozicija = 1;
  y = 0;

  /* Provera da li ima cifara u zapisu broja. */
  while (x > 0) {

    /* Izdvaja se poslednja cifra iz zapisa. */
    cifra = x % 10;

    /* Provera da li je cifra parna. */
    if (cifra % 2 == 0) {
      /* I ako jeste, uvecava se. */
      cifra++;

    }

    /* Novi broj se formira tako sto se izdvojena cifra pomnozi
       odgovarajucom tezinom (stepenom) pozicije. */
    y += cifra * pozicija;

    /* Priprema se broj za izdvajanje naredne cifre, uklanja se
       poslednja cifra broja. */
    x /= 10;

    /* Uvecava se tezinu (stepen) pozicije. */
    pozicija *= 10;
  }

  /* Ispisuje se izracunatu vrednost. */
  printf("%d\n", y);

  return 0;
}
```

Rešenje 2.3.28

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>

int main()
{
  int x;
  /* Tezina trenutne pozicije u broju. Moze biti 1, 10, 100, 1000
     itd. */
  int stepen_deset;
```

```
11  /* Trenutna izdvojena cifra iz broja x. */
    int cifra;
13  /* Redni broj cifre koja se trenutno obradjuje, gledano s desna
       na levo. */
15  int rbr;
    /* Broj dobijen nakon transformacije. */
17  int y;
    /* Promenljiva znak cuva znak unesenog broja. Moze biti -1 za
19     negativnu vrednost ili 1 za poziivnu vrednost. */
    int znak = 1;
21
    /* Ucitavanje broja. */
23  printf("Unesite broj: ");
    scanf("%d", &x);
25
    if (x <= 0) {
27    x = abs(x);
      znak = -1;
29  }
    /* Postavlja se vrednost stepena na 0 - to znaci da se prvo mnozi
31     sa 10^0=1. */
    stepen_deset = 0;
33
    /* Postavlja se vrednost broja koji se formira na 0. */
35  y = 0;
    /* Postavlja se redni broj pozicije na 0. */
37  rbr = 0;

39  /* Provera da li ima cifara u zapisu broja. */
    while (x > 0) {
41
      /* Izdvajanje cifre. */
43    cifra = x % 10;

45    /* Proverava se da li je pozicija izdvojene cifre parna - cifre
         na parnim pozicijama se zadrzavaju. */
47    if (rbr % 2 == 0) {
        /* Ako jeste parna izdvojena cifra se dodaje novom broju.
49         Neophodno je izvrsiti promenu tipova, jer je double
           povratni tip funkcije pow. */
51      y += cifra * ((int) pow(10, stepen_deset));

53      /* Uvecava se stepen zbog naredne cifre. */
        stepen_deset++;
55    }

57    /* Azurira se redni broj cifre. */
      rbr++;
59    /* I priprema se broj za naredno izdvajanje. */
      x /= 10;
61  }
```

```
63    y = znak * y;

65    /* Ispisuje se rezultat. */
      printf("%d\n", y);
67
      return 0;
69  }
```

Rešenje 2.3.29

```
1   #include <stdio.h>

3   int main()
    {
5     int n, novo_n;
      int stepen;
7     int cifra_levo, cifra_sredina, cifra_desno;

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

13    if (n <= 0) {
        printf("Neispravan unos.\n");
15      return -1;
      }
17
      /* Stepen broja 10 sa kojim se mnoze cifre izdvojenog broja. */
19    stepen = 1;

21    /* Nova vrednost broja. */
      novo_n = 0;
23
      /* Provera da li u zapisu broja postoje barem tri cifre.  */
25    while (n > 99) {
        /* Izdvaja se srednja cifra, cifra desno od nje i cifra levo od
27         nje: npr. za trojku 583 8 je srednja cifra, 3 je cifra
           desno, a 5 cifra levo. */
29      cifra_desno = n % 10;
        cifra_sredina = (n / 10) % 10;
31      cifra_levo = (n / 100) % 10;

33      /* U novi broj se smesta desna cifra. */
        novo_n += cifra_desno * stepen;
35
        /* Azurira se vrednost stepena. */
37      stepen = stepen * 10;

39      /* Provera da li je srednja cifra jednaka zbiru leve i desne
           cifre. */
41      if (cifra_levo + cifra_desno == cifra_sredina) {
```

```
43        /* Treba izbaciti srednju cifru, pa broj n se azurira tako
             sto se podeli sa 100. */
45        n = n / 100;
      } else {
47
        /* Inace, zadrzava se srednja cifra i odbacuje se samo
49         poslednja. */
        n = n / 10;
51    }
  }
53
  /* Na novi broj se dodaje preostali dvocifreni ili jednocifreni
55     broj. */
  novo_n = n * stepen + novo_n;
57
  /* Ispisivanje rezultata. */
59  printf("%d\n", novo_n);


61
  return 0;
63
}
```

Rešenje 2.3.30

```
1  #include <stdio.h>
   #include <math.h>
3
   int main()
5  {
     int x;
7    int broj_cifara;
     int min_stepen, max_stepen;
9    int pom;
     int leva_cifra, desna_cifra;
11   int indikator;

13   printf("Unesite broj: ");
     scanf("%d", &x);
15
     /* Ako je korisnik uneo negativan broj, analizira se njegova
17      apsolutna vrednost. */
     if (x < 0)
19     x = -x;

21   /* Odredjuje se broj cifara u zapisu broja x da bi moglo da se
        izdvajaju istovremeno cifre i sa leve i sa desne strane. */
23   broj_cifara = 0;
     pom = x;
25   while (pom > 0) {
```

```
    pom /= 10;
    broj_cifara++;
  }

  /* Odredjuje se stepen koji stoji uz krajnju levu cifru broja. */
  max_stepen = (int) pow(10, broj_cifara - 1);

  /* Indikator je promenljiva koja ukazuje da li je broj palindrom
     ili ne. */
  indikator = 1;
  while (x != 0 && indikator == 1) {
    /* Izdvaja se leva cifra. */
    leva_cifra = x / max_stepen;
    /* Izdvaja se desna cifra. */
    desna_cifra = x % 10;
    /* Ako su cifre razlicite, odmah se moze zakljuciti da broj
       nije palindrom i prekida se izvrsavanje petlje. */
    if (leva_cifra != desna_cifra) {
      indikator = 0;
      break;
    }
    /* Formira se nova vrednost broja x tako sto se odbacuje
       krajnja leva i krajnja desna cifra. */
    x = (x % max_stepen - x % 10) / 10;
    /* Koriguje se maksimalan stepen tako dobijenog broja - deli se
       sa 100 jer su odbacene 2 cifre. */
    max_stepen = max_stepen / 100;
  }

  /* Ispisuje se rezultat. */
  if (indikator == 1)
    printf("Broj je palindrom!\n");
  else
    printf("Broj nije palindrom!\n");

  return 0;
}
```

Rešenje 2.3.31

```
#include <stdio.h>

int main()
{
  /* Pamtimo uzastopna dva Fibonacijeva broja i na osnovu njih
     racunamo sledeci. */
  /* Promenljive prvi i drugi su brojevi koje pamtimo i na osnovu
     njih racunamo treci. */
  /* Na osnovu teksta zadatka, promenljive prvi i drugi postavljamo
     na 1. */
  int prvi = 1;
```

```
  int drugi = 1;
  int treci;
  /* Promenljiva pozicija je podatak koji ucitavamo i odnosi se na
     poziciju u Fibonacijevom nizu za koju treba izracunati
     vrednost. */
  int pozicija;
  /* Promenljiva i oznacava do koje pozicije smo izracunali
     vrednosti. Kako imamo prve dve vrednosti, ovu promenljivo
     postavljamo na 2. */
  int i = 2;

  printf("Unesite poziciju u Fibonacijevom nizu: ");
  scanf("%d", &pozicija);

  /* Pozicija ne moze biti 0 i ne moze biti negativan broj. */
  if (pozicija < 1) {
    printf
        ("Neispravan unos. Pozicija u Fibonacijevom nizu mora biti
    pozitivan broj koji nije 0!\n");
    return -1;
  }

  while (i < pozicija) {
    /* Na osnovu dva uzastopna racunamo treci. */
    treci = prvi + drugi;

    /* Potom razmenjujemo vrednosti. Uzastopna dva koja pamtimo
       postaju sledeca uzastopna dva broja Fibonacijevog niza. */
    prvi = drugi;
    drugi = treci;

    /* Prelazimo na racunanje sledeceg broja na sledecoj poziciji. */
    i++;
  }

  printf("Trazeni broj je: %d\n", drugi);

  return 0;
}
```

Rešenje 2.3.32

```
#include<stdio.h>

int main()
{
  int a0;
  int an, an1;
  printf("Unesi pocetni clan niza brojeva:");
  scanf("%d", &a0);
  if (a0 <= 0) {
```

```
    printf("Nekorektan unos. Broj mora biti pozitivan.\n");
    return -1;
  }
  printf("%d\n", a0);
  an = a0;
  while (an != 1) {
    if (an % 2) {
      /* Ukoliko je vrednost izraza an%2 razlicita od nule, izraz
         se tumaci kao tacan i izvrsavaju se naredbe iz if grane. */
      an1 = (3 * an + 1) / 2;
    } else {
      /* U suprotnom, ukoliko je vrednost izraza an%2 jednaka nuli,
         izraz se tumaci kao netacan i izvrsavaju se naredbe iz
         else grane. */
      an1 = an / 2;
    }
    printf("%d\n", an1);
    an = an1;
  }
  return 0;
}
```

Rešenje 2.3.33

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

int main()
{
  int format;
  /* Pomocna promenljiva koja sluzi kao brojac u petlji. */
  int i;
  /* Trenutne vrednosti za sirinu i visinu i pomocna promenljiva za
     promene u petlji. */
  double sirina, duzina, nova_duzina;
  unsigned int konacna_sirina, konacna_duzina;

  printf("Uneti format papira: ");
  scanf("%d", &format);

  if (format <= 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  /* duzina/sirina = 1 : sqrt(2) duzina*sirina = 1000x1000mm^2 Na
     osnovu ovih odnosa dobijamo pocetnu vrednost za sirinu i
     duzinu, odnosno vrednosti za papir A0. */
  duzina = sqrt(1000 * 1000 / sqrt(2));
  sirina = sqrt(2) * duzina;

```

```
  /* Kako vec imamo odredjenu sirinu i duzinu za papir A0, petlju
     krecemo od izracunavanja za papir A1, pa brojac i postavljamo
     na 1. */
  for (i = 1; i <= format; i++) {
    nova_duzina = sirina / 2;
    sirina = duzina;
    duzina = nova_duzina;
  }

  /* Duzina i sirina celi brojevi. */
  konacna_sirina = (unsigned int) sirina;
  konacna_duzina = (unsigned int) duzina;

  printf("%u %u\n", konacna_duzina, konacna_sirina);

  return 0;
}
```

Rešenje 2.3.34

```
#include <stdio.h>

int main()
{
  char c;

  /* Funkcija getchar ucitava jedan karakter. Naredbom dodele
     (c=getchar()) promenljivoj c bice dodeljena vrednost ascii
     koda unetog karaktera. Obratiti posebnu paznju na zagrade. */

  while ((c = getchar()) != '.') {
    if (c >= 'A' && c <= 'Z')
      /* Razlika izmedju ascii koda svakog malog i odgovarajuceg
         velikog slova je konstanta koja se moze sracunati izrazom
         'a'-'A' (i iznosi 32). */
      putchar(c + 'a' - 'A');
    else if (c >= 'a' && c <= 'z')
      putchar(c - 'a' + 'A');
    else
      putchar(c);
  }

  return 0;
}
```

Rešenje 2.3.35

```
#include <stdio.h>

```

```c
int main()
{
  char c;

  /* Inicijalizacija brojaca na 0. */
  int br_v = 0;
  int br_m = 0;
  int br_c = 0;
  int br_b = 0;
  int br_k = 0;
  int suma = 0;

  /* Petlja se zavrsava kada korisnik ne unese karakter, vec zada
     konstantu EOF . Ova konstanta se zadaje kombinacijom tastera
     CTRL+D. U tom slucaju, getchar() vraca -1. */
  while ((c = getchar()) != EOF) {
    if (c >= 'A' && c <= 'Z')
      br_v++;
    else if (c >= 'a' && c <= 'z')
      br_m++;
    else if (c >= '0' && c <= '9') {
      br_c++;
      /* Kada od promenljive tipa char oduzimamo karakter (ili neku
         drugu promenljivu tipa char), zapravo se vrsi oduzimanje
         njihovih ascii vrednosti i dobija se broj. */
      suma = suma + c - '0';
    } else if (c == '\t' || c == '\n' || c == ' ')
      br_b++;

    br_k++;
  }

  printf("velika: %d, mala: %d, cifre: %d, beline: %d \n", br_v,
         br_m, br_c, br_b);
  printf("suma cifara: %d\n", suma);

  return 0;
}
```

Rešenje 2.3.36

```c
#include <stdio.h>

int main()
{
  /* Promenljiva i je brojac. */
  int n, i;
  /* Brojaci za svaki od samoglasnika. */
  int broj_a = 0, broj_e = 0, broj_i = 0, broj_o = 0, broj_u = 0;
  /* Promenljiva c je tekuci ucitani karakter. */
  char c, belina;
```

```
12    printf("Unesite broj n: ");
      scanf("%d", &n);
14
      if (n < 0) {
16      printf("Neispravan unos.\n");
        return -1;
18    }

20    for (i = 0; i < n; i++) {
        /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
22         pa tek posle procitane beline se cita uneseni karakter. */
        scanf("%c%c", &belina, &c);
24
        /* Provera da li je ucitani karakter samoglasnik. */
26      switch (c) {
        case 'a':
28      case 'A':
          broj_a++;
30        break;
        case 'e':
32      case 'E':
          broj_e++;
34        break;
        case 'i':
36      case 'I':
          broj_i++;
38        break;
        case 'o':
40      case 'O':
          broj_o++;
42        break;
        case 'u':
44      case 'U':
          broj_u++;
46        break;
        }
48    }

50    printf("samoglasnik a: %d\n", broj_a);
      printf("samoglasnik e: %d\n", broj_e);
52    printf("samoglasnik i: %d\n", broj_i);
      printf("samoglasnik o: %d\n", broj_o);
54    printf("samoglasnik u: %d\n", broj_u);

56    return 0;
  }
```

Rešenje 2.3.37

```c
/* 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. */

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

int main()
{
  int n;
  int broj_Z, broj_i, broj_m, broj_a;
  char novi_red, c;
  int i;

  broj_Z = 0;
  broj_i = 0;
  broj_m = 0;
  broj_a = 0;

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

  /* Ucitavanje karakter po karakter. */
  for (i = 0; i < n; i++) {
    printf("Unestite %d. karakter: ", i + 1);
    /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
       pa tek posle procitane beline se cita uneseni karakter. */
    scanf("%c%c", &novi_red, &c);

    /* Analiziramo karakter */
    switch (c) {
    case 'Z':
      broj_Z++;
      break;
    case 'i':
      broj_i++;
      break;
    case 'm':
      broj_m++;
      break;
    case 'a':
      broj_a++;
      break;
    }
  }

  /* Ako u unosu ima barem jedno veliko slovo z i barem po jedno
     malo slovo i, m i a, rec se moze napisati. A u suprotnom ne
     moze. */
  if (broj_Z && broj_i && broj_m && broj_a) {
    printf("Moze se napisati rec Zima.\n");
  } else {
```

```
53      printf("Ne moze se napisati rec Zima.\n");
    }
55
    return 0;
57  }
```

Rešenje 2.3.38

```
1   #include <stdio.h>
    int main()
3   {
      int n;
5     /* Brojac. */
      int i;
7     /* Promenljiva u kojoj se cuva suma kubova. */
      int s;
9
      printf("Unesite pozitivan ceo broj:");
11    scanf("%d", &n);
      if (n < 0) {
13      printf("Neispravan unos.\n");
        return -1;
15    }

17    for (s = 0, i = 1; i <= n; i++)
        s += i * i * i;
19
      printf("Suma kubova od 1 do %d: %d\n", n, s);
21
      return 0;
23  }
```

Rešenje 2.3.39

```
    #include <stdio.h>
2
    int main()
4   {
      int n;
6     /* Brojac. */
      int i;
8     /* Promenljiva u kojoj se cuva suma kubova. */
      int s;
10
      printf("Unesite pozitivan ceo broj:");
12    scanf("%d", &n);

14    if (n < 0) {
        printf("Neispravan unos.\n");
```

```
16      return -1;
    }
18
    i = 1;
20  s = 0;

22  for (i = 1; i <= n; i++) {
      s += i * i * i;
24    printf("i=%d, s=%d\n", i, s);
    }
26
    return 0;
28 }
```

Rešenje 2.3.40

```
1  #include <stdio.h>

3  int main()
   {
5    int n, i;
     float x, S, stepen;
7
     printf("Unesite redom brojeve x i n: ");
9    scanf("%f %d", &x, &n);

11   if (n < 0) {
       printf("Neispravan unos.\n");
13     return -1;
     }
15
     /* Inicijalizacija sume. */
17   S = 0;

19   /* Stepen promenljiva ce sadrzati vrednosti stepena x^n. Pocetna
        vrednost joj je 1 jer je x^0 = 1. */
21   stepen = 1;

23   for (i = 1; i <= n; i++) {
       stepen = stepen * x;
25     S = S + i * stepen;
     }
27
     printf("S=%f\n", S);
29
     return 0;
31 }
```

Rešenje 2.3.41

```c
#include <stdio.h>

int main()
{
  unsigned n, i;
  float x, S, stepen;

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

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

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

  return 0;
}
```

Rešenje 2.3.42

```c
#include <stdio.h>
#include <math.h>

int main()
{
  /* Promenljiva i je brojac, promenljiva S cuva izracunatu sumu, a
     promenljiva clan je tekuci clan niza. */
  int i;
  float S;
  float x, eps;
  float clan;

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

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

  S = 0;
  /* Prvi clan sume je 1. */
  clan = 1;
  i = 1;
  while (clan > eps) {
    S = S + clan;
    clan = clan * x / i;
    i++;
```

```
  }

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

  return 0;
}
```

Rešenje 2.3.43

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

int main()
{
  /* Promenljiva i je brojac, promenljiva S cuva izracunatu sumu,
     promenljiva znak moze bito 1 ili -1 i odredjuje znak trenutnog
     clana sume, a promenljiva clan je tekuci clan niza. */
  int i, znak;
  float S;
  float x, eps, clan;

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

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


  S = 0;
  clan = 1;
  i = 1;
  znak = -1;

  /* Kako clanovi sume mogu biti negativni, potrebno je posmatrati
     apsolutnu vrednost clana. */
  while (fabs(clan) > eps) {
    S = S + clan;

    /* Promena znaka. */
    clan = clan * x / i;
    clan *= znak;

    i++;
  }

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

  return 0;
}
```

Rešenje 2.3.44

```c
#include <stdio.h>
#include <math.h>

int main()
{
  int n, i;
  double x;
  double stepen = 1;
  double proizvod = 1;

  printf("Unesite redom brojeve x i n: ");
  scanf("%lf %d", &x, &n);

  if (n <= 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  for (i = 0; i < n; i++) {
    stepen *= x;
    proizvod *= 1 + cos(stepen);
  }

  printf("Proizvod = %lf\n", proizvod);

  return 0;
}
```

Rešenje 2.3.45

```c
#include <stdio.h>

int main()
{
  int n, i;
  double Razlomak;

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

  if (n <= 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  Razlomak = n;

  /* Razlomak se izracunava "od nazad", odnosno, krece se od
```

```
19      najnizeg razlomka 1/n i od njega se nadalje formira sledeci,
        "visi" razlomak itd. Zavrsava se kada se stigne do koraka 0 +
21      1/R. */
    for (i = n - 1; i >= 0; i--)
23    Razlomak = i + 1 / Razlomak;

25  printf("Razlomak = %lf\n", Razlomak);

27  return 0;
}
```

Rešenje 2.3.46

```
1  #include <stdio.h>

3  int main () {
       /* Promenljiva i je brojac , promenljiva S cuva izracunatu sumu ,
5      promenljiva znak moze bito 1 ili -1 i odredjuje znak trenutnog
       clana sume , a promenljiva clan je tekuci clan niza. */
7
       int i, znak , n;
9      float S;
       float x, clan;
11     printf(" Unesite x i n: ");
       scanf("%f%d", &x, &n);
13
       if (n <= 0) {
15         printf(" Neispravan unos .\n");
           return -1;
17     }

19     S = 1;
       clan = 1;
21     i = 1;
       znak = -1;
23
       /* Kako clanovi sume mogu biti negativni , potrebno je posmatrati
25     apsolutnu vrednost clana. */
       while (i <= 2 * n - 1) {
27
           /* Promena znaka. */
29         /* Svaki clan suma se od prethodnog clana razlikuje za
           x^2/(i*(i+1)). */
31         clan = clan * x * x / (i * (i + 1));
           clan *= znak;
33
           S = S + clan;
35
           i += 2;
37     }
```

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

41      return 0;
}
```

Rešenje 2.3.47

```
#include <stdio.h>

int main()
{
  int n, i;
  /* Promenljiva clan je deo proizvoda i predstavlja 1/i!. */
  double clan;
  double S = 1;

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

  if (n <= 1) {
    printf("Neispravan unos.\n");
    return -1;
  }

  clan = 1;
  for (i = 2; i <= n; i++) {
    clan = clan / i;
    S *= 1 + clan;
  }

  printf("S = %lf\n", S);

  return 0;
}
```

Rešenje 2.3.48

```
#include <stdio.h>

int main()
{
  int n, i, znak = -1;
  /* Promenljiva clan je deo proizvoda i predstavlja 1*3*5*...*i. */
  long int clan;
  long int S = 0;

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

```
13    if (n < 5 || n % 2 == 0) {
        printf("Neispravan unos.\n");
15      return -1;
      }
17
      clan = 1 * 3;
19    for (i = 5; i <= n; i += 2) {
        clan = znak * clan * i;
21      S += clan;
      }
23
      printf("S = %ld\n", S);
25
      return 0;
27  }
```

Rešenje 2.3.49

```
#include <stdio.h>
2
int main()
4  {
     int n, i;
6    double P;
     double x, a;
8
     printf("Unesite dva relana broja x i a: ");
10   scanf("%lf%lf", &x, &a);

12   printf("Unesite prirodan broj: ");
     scanf("%d", &n);
14
     if (n <= 0) {
16     printf("Neispravan unos.\n");
       return -1;
18   }

20   P = x;
     for (i = 0; i < n; i++)
22     P = (P + a) * (P + a);

24   printf("Izraz = %lf\n", P);

26   return 0;
  }
```

Rešenje 2.3.50

Rešenje (a)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= n; i++) {
    for (j = 1; j <= n; j++)
      /* U tablici mnozenja vrednost svakog polja je proizvod vrste
         i kolone u kojoj se nalazi. */
      printf("%3d ", i * j);

    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Brojac koji broji koliko brojeva je ispisano u jednom redu. */
  j = 0;
  for (i = 1; i <= n * n; i++) {
    printf("%3d ", i);
    /* Uvecavamo brojac */
    j++;

    /* Kada je ispisano n brojeva u jednom redu, ispisuje se znak
       za novi red, da bi ispis krenuo u novom redu i vrednost
       brojaca j se postavlja na 0 jer u novom redu jos ni jedan
       broj nije ispisan. */
    if (j == n) {
      j = 0;
      printf("\n");
    }
  }

  return 0;
```

```
28 }
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= n; i++) {
    for (j = 0; j < n; j++)
      if ((j + i) % n == 0)
        printf("%3d", n);
      else
        printf("%3d", (j + i) % n);

    printf("\n");
  }

  return 0;
}
```

Rešenje (d)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    for (j = 0; j < n - i; j++)
      printf("(%d, %d)", i, j);

    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.51

Rešenje (a)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Kvadrat predstavlja tabelu sa n vrsta gde svaka vrsta sadrzi
       n polja, a svako polje je isto i predstavlja karakter *. */
    for (j = 0; j < n; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter
       *, a unutrasnjost kvadrata je karakter blanko. */
    for (j = 0; j < n; j++)
      /* Provera da li je ivica. */
      if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje (c)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter
       *, a unutrasnjost kvadrata je karakter blanko osim na
       mestima na kojima je glavna dijagonala. */
    for (j = 0; j < n; j++)
      /* Provera da li je ivica ili glavna dijagonala. */
      if (j == 0 || j == n - 1 || i == 0 || i == n - 1 || i == j)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.52

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Veliko slovo X se moze posmatrati kao dijagonale kvadrata
       (glavna i sporedna). Zato, treba ispisivati blanko na
       mestima gde nije dijagonala, a karakter * na mestima gde je
       neka od dijagonala. */
    for (j = 0; j < n; j++)
      /* Provera da li je mesto glavne ili sporedne dijagonale. */
      if (i == j || i + j == n - 1)
        printf("*");
      else
        printf(" ");
    printf("\n");
```

```
  }

  return 0;
}
```

Rešenje 2.3.53

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  if (n % 2 == 0) {
    printf("Pogresan unos.\n");
    return -1;
  }

  for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++)
      if (i == n / 2 || j == n / 2)
        printf("+");
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.54

Rešenje (a)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    for (j = 0; j < n - i; j++)
```

```
12        printf("*");
      printf("\n");
14    }

16    return 0;
}
```

Rešenje (b)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
11    for (j = 0; j <= i; j++)
        printf("*");
13    printf("\n");
  }

  return 0;
17}
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
11    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < i; j++)
13      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
15    for (j = 0; j < n - i; j++)
        printf("*");
17    printf("\n");
  }

19
  return 0;
```

```
}
```

Rešenje (d)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j <= i; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (e)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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


  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j <= i; j++)
      printf("*");
    printf("\n");
  }
```

```
  /* Potrebno je iscrtati i donji deo slike, odnosno donji trougao.
     Brojac i odredjuje koji red donjeg trougla se trentno iscrtava.
     * Kako je prvi red dodnjeg trougla vec iscrtan (to je poslednji
     red gornjeg trougla), brojac se postavlja na 1. */
  for (i = 1; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < i; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < n - i; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (f)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;
  char c, blanko;

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

  printf("Unesite karakter c: ");
  /* Zbog pritiskanja tastera ENTER nakon unosa promenljive broj
     potrebno je ucitati karakter za novi red u promenljivu blanko
     pre ucitavanja karaktera kojim se iscrtava trougao. */
  scanf("%c%c", &blanko, &c);

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Iscrtavaju se samo ivice trougla, ostalo se popunjava
       belinama. */
    for (j = 0; j <= i; j++)
      if (i == n - 1 || j == 0 || j == i)
        printf("%c", c);
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.55

Rešenje (a)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < 2 * i + 1; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
     izracunavanja koliko zvezdica i praznina je potrebno ispisati
     u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
     iteraciji petlje. */
  for (i = n - 1;; i--) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i + 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < 2 * i + 1; j++)
      printf("*");
    printf("\n");
```

```
22
     /* Posebna paznja mora da se obrati na cinjenicu da su brojaci
24      tipa unsigned int. Problem nastaje kada je i==0 i pokusa se
        oduzimanje (i--). Posto su brojevi unsigned, nova vrednost
26      nece biti -1, vec pozitivan ceo broj. Imajuci to na umu,
        uslov i>=0 ne moze da se stavi u uslov za for petlju. Mnogo
28      sigurnije je brojace deklarisati da budu tipa int i izbeci
        ovakvu vrstu problema. */
30    if (i == 0)
        break;
32  }

34  return 0;
}
```

Rešenje (c)

```
1  #include <stdio.h>

3  int main()
   {
5    unsigned int n;
     int i, j;
7
     printf("Unesite broj n: ");
9    scanf("%u", &n);

11   /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
     for (i = 0; i < n; i++) {
13     /* Prvo se ispisuju beline koje prethode karakterima *. */
       for (j = 0; j < n - i - 1; j++)
15       printf(" ");
       /* Posle belina se ispisuje potreban broj karaktera *. */
17     for (j = 0; j < 2 * i + 1; j++)
         printf("*");
19     printf("\n");
     }
21
     /* Sada se ispisuje donji trougao. Kako je prvi red donjeg
23      trougla vec ispisan (poslednji red gornjeg trougla), potrebno
        je naciniti jednu iteraciju manje. */
25
     /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
27      izracunavanja koliko zvezdica i praznina je potrebno ispisati
        u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
29      iteraciji petlje. */
     for (i = n - 2; i >= 0; i--) {
31     /* Prvo se ispisuju beline koje prethode karakterima *. */
       for (j = 0; j < n - i - 1; j++)
33       printf(" ");
       /* Posle belina se ispisuje potreban broj karaktera *. */
```

```
35      for (j = 0; j < 2 * i + 1; j++)
          printf("*");
37      printf("\n");
    }
39
    return 0;
41  }
```

Rešenje (d)

```
1   #include <stdio.h>

3   int main()
    {
5     unsigned int n, i, j;

7     printf("Unesite broj n: ");
      scanf("%u", &n);
9
      /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
11    for (i = 0; i < n; i++) {
        /* Prvo se ispisuju beline koje prethode karakterima *. */
13      for (j = 0; j < n - i - 1; j++)
          printf(" ");
15      /* Posle belina se ispisuje sam trougao. Ako je brojac na
           ivici onda se ispisuje karakter *, a inace praznina.
17         Takodje, proverava se da li se ispisuje poslednji red (i==n)
           i u njemu se ispisuje svaki drugi put *, a inace praznina.
19         Kako se ispisuje svaki drugi put vrsi se provera j%2 == 0. */
        for (j = 0; j < 2 * i + 1; j++)
21        if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
            printf("*");
23        else
            printf(" ");
25      printf("\n");
    }
27
    return 0;
29  }
```

Rešenje (c)

```
1   #include <stdio.h>

3   int main()
    {
5     unsigned int n;
      int i, j;
7
```

```
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje sam trougao. Ako je brojac na
       ivici onda se ispisuje karakter *, a inace praznina.
       Takodje, proverava se da li se ispisuje poslednji red (i==n)
       i u njemu se ispisuje svaki drugi put *, a inace praznina.
       Kako se ispisuje svaki drugi put vrsi se provera j%2 == 0. */
    for (j = 0; j < 2 * i + 1; j++)
      if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  /* Sada se ispisuje donji trougao. Kako je prvi red donjeg
     trougla vec ispisan (poslednji red gornjeg trougla), potrebno
     je naciniti jednu iteraciju manje. */

  for (i = n - 2; i >= 0; i--) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < 2 * i + 1; j++)
      if (j == 0 || j == 2 * i)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.56

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

  printf("Unesite broj n: ");
  scanf("%u", &n);
```

```
9
   /* Strelica se moze posmatrati kao spojena dva pravougla trougla
11    kojima se ispisuje hipotenuza i jedna, donja kateta. */

13 /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
   for (i = 0; i < n; i++) {
15   for (j = 0; j <= i; j++)
       /* Proverava se da li se ispisuje karakter na hipotenuzi (j
17        == i-1) ili da se ispisuje poslednji red (i == n-1). */
       if (j == i || i == n - 1)
19       printf("*");
       else
21       printf(" ");
     printf("\n");
23 }

25 /* Potrebno je iscrtati i donji deo slike, odnosno donji trougao.
      Brojac i odredjuje koji red donjeg trougla se trentno iscrtava.
27    * Kako je prvi red dodnjeg trougla vec iscrtan (to je poslednji
      red gornjeg trougla), brojac se postavlja na 1. */
29 for (i = 1; i < n; i++) {
     for (j = 0; j < n - i; j++)
31     /* Provera da li se ispisuje hipotenuza. */
       if (j == n - i - 1)
33       printf("*");
       else
35       printf(" ");
     printf("\n");
37 }

39 return 0;
}
```

Rešenje 2.3.57

```
1 #include <stdio.h>

3 int main()
{
5   unsigned int n;
    int i, j, k;
7

    printf("Unesite broj n: ");
9   scanf("%u", &n);

11  /* Brojac j odredjuje koliko ukupno karaktera (praznina i
       karaktera *) u svakom redu se ispisuje. U svakom drugom redu
13     ovaj broj se povecava za 2. Na pocetku je 1 (jer se ispisuje
       samo jedna zvezda). */
15  j = 1;
```

```
  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 1; i <= n; i++) {
    /* U svakom drugom redu broj ispisanih karaktera se uvecava za
       2. */
    if (i % 2 == 0)
      j += 2;
    for (k = 0; k < j; k++)
      /* U svakom drugom redu se naiazmenicno ispisuje * ili
         praznina. */
      if (i % 2 == 0) {
        if (k % 2 == 0)
          printf("*");
        else
          printf(" ");
      } else
        printf("*");

    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.58

```
#include <stdio.h>

int main()
{
  unsigned int n, m;
  int i, j, k;

  printf("Unesite brojeve n i m: ");
  scanf("%u%u", &n, &m);

  for (i = 1; i <= m; i++) {
    /* Za svaki kvadrat se racuna duzina bez poslednje ivice.
       Kvadrat je sastavljen od (m-1) zvezdice i (m-1) praznine
       (praznine se nalaze izmedju zvezdica). Znaci ukupna duzina
       je 2*(m-1) karakter, a kako ima n kvadrata, duzina je
       n*2*(m-1). */
    for (j = 0; j <= n * 2 * (m - 1); j++)
      /* Provera da li se ispisuje prvi ili poslednji red. */
      if (i == 1 || i == m)
        /* Naizmenicno se ispisuje * i praznina. */
        if (j % 2 == 0)
          printf("*");
        else
          printf(" ");
      else
        /* Na kraju svakog kvadrata (nakon svake (m-1) zvezdice i
```

```
            (m-1) praznine se ispisuje ivica kvadrata. */
      if (j % (2 * (m - 1)) == 0)
        printf("*");
      else
        printf(" ");

    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.59

```
#include <stdio.h>

int main()
{
  unsigned int n;
  int i, j;

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

  /* Potrebno je spojiti sve slike u jednu, sliku gornjeg dela
     romba i sliku donjeg dela romba. */

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju * koje prethode karakterima -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    /* Posle * se ispisuje potreban karakter -. */
    for (j = 0; j < 2 * i; j++)
      printf("-");
    /* Potom se ispisuju * koje su nakon karaktera -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    printf("\n");
  }

  /* Sada se ispisuje donji trougao. Kako je prvi red donjeg
     trougla vec ispisan (poslednji red gornjeg trougla), potrebno
     je naciniti jednu iteraciju manje. */

  /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
     izracunavanja koliko zvezdica i praznina je potrebno ispisati
     u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
     iteraciji petlje. */
  for (i = n - 2; i >= 0; i--) {
    /* Prvo se ispisuju * koje prethode karakterima -. */
```

```
    for (j = 0; j < n - i; j++)
      printf("*");
    /* Posle * se ispisuje potreban karakter -. */
    for (j = 0; j < 2 * i; j++)
      printf("-");
    /* Potom se ispisuju * koje su nakon karaktera -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.60

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Slika se sastoji iz dva dela, trougla i kvadrata i svaki deo
     se nezavisno iscrtava. */

  /* Prvo se iscrtava krov, odnosno trougao. */
  for (i = 0; i < n - 1; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    for (j = 0; j < 2 * i + 1; j++)
      if (j == 0 || j == 2 * i)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  /* Potom se iscrtava kvadat. Da bi iscrtavanje bilo lakse
     istovremeno se ispisuju dva karaktera. */
  for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++)
      /* Provera da li je ivica. */
      if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
        printf("* ");
      else
        printf("  ");
    printf("\n");
```

```
  }

  return 0;
}
```

Rešenje 2.3.61

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= (n + 1) / 2; i++) {
    for (j = i; j <= n + 1 - i; j++)
      printf("%d ", j);
  }

  return 0;
}
```

Rešenje 2.3.62

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= n; i++) {
    for (j = 1; j <= n; j++)
      if (j % i == 1 || i == 1)
        printf("%d ", j);

    printf("\n");
  }

  return 0;
}
```

2.5 Funkcije

Zadatak 2.5.1 Napisati funkciju `int kvadrat(int x)` koja računa kvadrat datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 15
 Kvadrat broja 15 je 225
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: -89
 kvadrat broja -89 je 7921
```

[Rešenje 2.5.1]

Zadatak 2.5.2 Napisati funkciju `int kub(int x)` koja računa kub datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 15
 Kub broja 15 je 3375
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: -89
 Kub broja -89 je -704969
```

[Rešenje 2.5.2]

Zadatak 2.5.3 Napisati funkciju `unsigned int apsolutna_vrednost(int x)` koja izračunava apsolutnu vrednost broja x. Napisati program koji 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 2.5.3]

Zadatak 2.5.4 Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji 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 2.5.4]

Zadatak 2.5.5 Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja x. Napisati program koji 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 2.5.5]

Zadatak 2.5.6 Napisati funkciju `float stepen(float x, int n)` koja računa vrednost n-tog stepena realnog broja x. Napisati program koji učitava relan broj x i ceo broj n i ispisuje rezultat rada funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite jedan realan i jedan ceo broj:
 4.5 3
 4.5000003=91.125000
```

Primer 2

```
Interakcija sa programom:
 Unesite jedan realan i jedan ceo broj:
 -33.2 5
 -33.2000015=-40335776.000000
```

[Rešenje 2.5.6]

Zadatak 2.5.7 Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva 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 učitava dva cela pozitivna broja i ispisuje rezultat poziva funkcije. Napomena: *Pretpostaviti da je unos korektan.*

Primer 1

```
Interakcija sa programom:
 Unesite dva broja: 81 3
 Jeste: 81 = 34
```

Primer 2

```
Interakcija sa programom:
 Unesite dva broja: 162 11
 Broj 162 nije stepen broja 11.
```

[Rešenje 2.5.7]

Zadatak 2.5.8 Napisati funkciju `int faktorijel(int n)` koja računa faktorijel broja n. Napisati i program koji učitava dva cela broja x i y iz intervala $[0, 12]$ 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 2.5.8]

Zadatak 2.5.9 Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najveći zajednički delilac primenom Euklidovog algoritma. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite dva cela broja: 1024 832
 Najveci zajednicki delilac je 64
```

Primer 2

```
Interakcija sa programom:
 Unesite dva cela broja: -900 112
 Najveci zajednicki delilac je -4
```

[Rešenje 2.5.9]

Zadatak 2.5.10 Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato n vraća zbir recipročnih vrednosti brojeva od 1 do n. Napisati program koji učitava ceo broj i ispisuje rezultat rada funkcje zaokružen na dve decimale.

Primer 1

```
Interakcija sa programom:
 Unesi jedan pozitivan ceo broj: 10
 Zbir reciprocnih je 2.93
```

Primer 2

```
Interakcija sa programom:
 Unesi jedan pozitivan ceo broj: 100
 Zbir reciprocnih je 5.19
```

[Rešenje 2.5.10]

Zadatak 2.5.11 Napisati funkciju `void ispis(float x, float y, unsigned n)` koja za dva realna broja x i y i jedan pozitivan ceo broj n ispisuje vrednosti sinusne funkcije u n ravnomerno raspoređenih tačaka intervala $[x, y]$. Napisati program koji učitava odgovarajuće vrednosti i testira rad ove funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite dva realna broja: 7 32
 Unesite jedan prirodan broj: 10
 0.6570 -0.3457 -0.0108 0.3659 -0.6731
 0.8922 -0.9945 0.9666 -0.8122
```

Primer 2

```
Interakcija sa programom:
 Unesite dva realna broja: 20.5 -8.32
 Unesite jedan prirodan broj: 5
 -0.8934 -0.8979 -0.1920 0.6658 0.9968
```

[Rešenje 2.5.11]

Zadatak 2.5.12 Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji učitava ceo broj i ispisuje rezultat 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 2.5.12]

Zadatak 2.5.13 Napisati funkciju `int sadrzi(int x, int c)` koja ispituje da li se cifra c nalazi u zapisu celog broja x. Funkcija treba da vrati 1 ako se cifra nalazi u broju, a 0 inače. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj i cifru: 17890 7
 Cifra se nalazi u broju.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj i cifru: 1982 6
 Cifra se ne nalazi u broju.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj i cifru: 17890 26
 Neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj i cifru: -1982 9
 Cifra se nalazi u broju.
```

[Rešenje 2.5.13]

Zadatak 2.5.14 Napisati funkciju `int broj_neparnih_cifara(int x)` koja određuje broj neparnih 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.

Primer 1

```
Interakcija sa programom:
 Unesite cele brojeve:
 2341
 Broj neparnih cifara je 2
 78
 Broj neparnih cifara je 1
 800
 Broj neparnih cifara je 0
 -99761
 Broj neparnih cifara je 4
 0
```

Primer 2

```
Interakcija sa programom:
 Unesite cele brojeve:
 987611
 Broj neparnih cifara je 4
 135
 Broj neparnih cifara je 3
 -701
 Broj neparnih cifara je 2
 602
 Broj neparnih cifara je 0
 -884
 Broj neparnih cifara je 0
 79901
 Broj neparnih cifara je 4
 0
```

[Rešenje 2.5.14]

Zadatak 2.5.15 Napisati program za ispitivanje svojstava cifara datog celog broja.

(a) 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.

(b) 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.

Napisati program koji učitava ceo broj i ispisuje da li su sve cifre parne i da li su sve cifre jednake.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 86422
 Sve cifre broja su parne.
 Cifre broja nisu jednake.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 55555
 Sve cifre broja nisu parne.
 Cifre broja su jednake.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -88
 Sve cifre broja su parne.
 Cifre broja su jednake.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj i cifru: -342
 Sve cifre broja nisu parne.
 Cifre broja nisu jednake.
```

[Rešenje 2.5.15]

Zadatak 2.5.16 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, a 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!
```

Primer 3

```
Interakcija sa programom:
 Unesite dva broja: -7391 1397
 Uslov je ispunjen!
```

[Rešenje 2.5.16]

Zadatak 2.5.17 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 poruku da li su cifre unetog broja u rastućem poretku.

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 2.5.17]

Zadatak 2.5.18 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 2.5.18]

Zadatak 2.5.19 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 sve do pojave broja 0 ispisuje odgovarajuće brojeve kojima je uklonjena cifra stotine.

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 2.5.19]

Zadatak 2.5.20 Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sve do pojave broja 0 ispisuje odgovarajuće rotirane brojeve.

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 2.5.20]

Zadatak 2.5.21 Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja n. Napisati program koji 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 2.5.21]

Zadatak 2.5.22 Broj je prost ako je deljiv samo sa 1 i sa samim sobom. 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. Napisati program koji za uneti ceo broj n ispisuje prvih n prostih brojeva.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 17
 Broj je prost!
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 24
 Broj nije prost!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -11
 Broj nije prost!
```

[Rešenje 2.5.22]

Zadatak 2.5.23 Napisati funkciju `void prosti_brojevi(int m)` koja ispisuje sve proste brojeve manje od broja m. Napisati program koji učitava ceo broj veći od 1 i ispisati rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 15
 2 3 5 7 11 13
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 9
 2 3 5 7
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 1
 Greska: pogresan unos!
```

Zadatak 2.5.24 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 ε. Napisati program koji učitava dva realna broja x i *eps* i ispisuje izračunatu vrednost e^x.

Primer 1

```
Interakcija sa programom:
 Unesite broj x: 5
 Unesite eps: 0.001
 Rezultat: 148.412951
```

Primer 2

```
Interakcija sa programom:
 Unesite broj x: -3
 Unesite eps: 0.0001
 Rezultat: 0.049796
```

[Rešenje 2.5.24]

Zadatak 2.5.25 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 srecan(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. Napomena: *Pretpostaviti da je unos korektan.*

Primer 1

```
Interakcija sa programom:
 Unesite broj: 100
 Srecni brojevi:
 1 10 19 28 37 46 55 64 73 82 91 100
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 0
 Nema srecnih brojeva.
```

[Rešenje 2.5.25]

Zadatak 2.5.26 Broj a je Armstrongov ako je jednak sumi n-tih stepena svojih cifara, pri čemu je n broj cifara broja a. Napisati funkciju `int armstrong(int x)` koja vraća 1 ako je broj Armstrongov, odnosno 0 ako nije. Napisati program koji za učitani ceo broj proverava da li je Armstrongov.

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 2.5.26]

Zadatak 2.5.27 Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj x pojavljuje u nizu brojeva koji se unose sve do pojave broja 0. Napisati program koji učitava vrednost broja x i testira 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 2.5.27]

Zadatak 2.5.28 Napisati funkciju `long unsigned fibonaci(int n)` koja računa n-ti element Fibonačijevog niza. 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 2.5.28]

Zadatak 2.5.29 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 do pojave znaka `EOF`.

Primer 1

```
Interakcija sa programom:
Unesite karaktere: ZDRAVO
zdravo
```

Primer 2

```
Interakcija sa programom:
Unesite karaktere: Dobro jutro, R2D2!
dOBRO JUTRO, r2d2!
```

[Rešenje 2.5.29]

Zadatak 2.5.30 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 ispred njega u abecedi. Karakteri koji nisu slova se ne šifruju. Šifrovanje treba da bude kružno, što znači da je, na primer, šifra za karakter b i pomeraj 2 karakter z. Napisati program koji učitava karakter po karakter do kraja ulaza i ispisuje šifrovani tekst.

Primer 1

```
Interakcija sa programom:
Unesite broj k: 2
Unesite tekst (CTRL+D za prekid):
c
a
8
8
+
+
Z
X
```

[Rešenje 2.5.30]

Zadatak 2.5.31 Napisati funkciju `int prestupna(int godina)` koja za zadatu 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 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 2.5.31]

Zadatak 2.5.32 Napisati funkciju `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu. Napisati program koji učitava dva cela broja (mesec i godinu) i ispisuje broj dana u datom mesecu. U slučaju nekorektnog unosa ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite mesec i godinu: 8
     1998
 Broj dana je: 31
```

Primer 2

```
Interakcija sa programom:
 Unesite mesec i godinu: 2
     2004
 Broj dana je: 29
```

Primer 3

```
Interakcija sa programom:
 Unesite mesec i godinu: 24
     2004
 Neispravan unos.
```

Zadatak 2.5.33 Napisati funkciju `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan. Napisati program koji učitava tri cela broja (dan, mesec, godinu) i ispisuje da li je datum ispravan ili ne.

Primer 1

```
Interakcija sa programom:
 Unesite datum: 24.8.1998.
 Datum je ispravan.
```

Primer 2

```
Interakcija sa programom:
 Unesite datum: 31.4.1789.
 Datum nije ispravan.
```

Primer 3

```
Interakcija sa programom:
 Unesite datum: 29.2.2004.
 Datum je ispravan.
```

Primer 4

```
Interakcija sa programom:
 Unesite datum: 29.14.2004.
 Datum nije ispravan.
```

Zadatak 2.5.34 Napisati funkciju `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum određuje datum sledećeg dana. Napisati program koji učitava tri cela broja i ispisuje datum sledećeg dana.

Primer 1

```
Interakcija sa programom:
 Unesite datum: 24.8.1998.
 Datum sledeceg dana je: 25.8.1998.
```

Primer 2

```
Interakcija sa programom:
 Unesite datum: 31.12.1789.
 Datum sledeceg dana je: 1.1.1790.
```

Primer 3

```
Interakcija sa programom:
 Unesite datum: 28.2.2003.
 Datum sledeceg dana je: 1.3.2004.
```

Primer 4

```
Interakcija sa programom:
 Unesite datum: 31.4.2004.
 Datum nije ispravan.
```

Zadatak 2.5.35 Napisati funkciju `int od_nove_godine(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do datog datuma. Napisati program koji učitava tri cela broja i ispisuje koliko dana je proteklo od Nove godine.

Primer 1

```
Interakcija sa programom:
 Unesite datum: 24.8.1998.
 Broj dana od Nove godine je: 235
```

Primer 2

```
Interakcija sa programom:
 Unesite datum: 31.12.1680.
 Broj dana od Nove godine je: 366
```

Primer 3

```
Interakcija sa programom:
 Unesite datum: 28.2.2003.
 Broj dana od Nove godine je: 58
```

Primer 4

```
Interakcija sa programom:
 Unesite datum: 31.4.2004.
 Datum nije ispravan.
```

Zadatak 2.5.36 Napisati funkciju `int do_kraja_godine(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati program koji učitava tri cela broja i ispisuje broj dana do krja godine.

Primer 1

```
Interakcija sa programom:
 Unesite datum: 24.8.1998.
 Broj dana do Nove godine je: 129
```

Primer 2

```
Interakcija sa programom:
 Unesite datum: 31.12.1680.
 Broj dana od Nove godine je: 0
```

Primer 3

```
Interakcija sa programom:
 Unesite datum: 28.2.2004.
 Broj dana od Nove godine je: 307
```

Primer 4

```
Interakcija sa programom:
 Unesite datum: 31.4.2004.
 Datum nije ispravan.
```

[Rešenje 2.5.37]

Zadatak 2.5.37 Napisati funkciju `int broj_dana_između(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati program koji učitava šest celih brojeva, koji označavaju dva datuma, i na standarni izlaz ispisuje broj dana između ta dva datuma. U slučaju pogrešnog unosa ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite prvi datum: 12.3.2008.
 Unesite drugi datum: 5.12.2008.
 Broj dana izmedju dva datuma je: 268
```

Primer 2

```
Interakcija sa programom:
 Unesite prvi datum: 26.9.1986.
 Unesite drugi datum: 2.2.1701.
 Broj dana izmedju dva datuma je: 104319
```

Primer 3

```
Interakcija sa programom:
 Unesite prvi datum: 24.8.1998.
 Unesite drugi datum: 12.10.2010.
 Broj dana izmedju dva datuma je: 4432
```

Primer 4

```
Interakcija sa programom:
 Unesite prvi datum: 24.8.1998.
 Unesite drugi datum: 31.4.2004.
 Ulaz nije ispravan.
```

[Rešenje 2.5.37]

Zadatak 2.5.38 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.

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 2.5.38]

Zadatak 2.5.39 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 2.5.39]

Zadatak 2.5.40 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 2.5.40]

2.6 Rešenja

Rešenje 2.5.1

```
1 #include <stdio.h>

3 int kvadrat(int x)
```

```
{
    /* promenljive u listi argumenata funkcije, kao i one
       deklarisane u samoj funkciji, lokalne su za tu funkciju
       sto znaci da se promenljive x i y nece "videti" nigde izvan
       funkcije kvadrat (ni u funkciji main ni u funkciji kub)
    */

    int y;
    y = x*x;
    return y;
}

int main()
{
    int a,kv, kb;
    printf("Unesi ceo broj:");
    scanf("%d",&a);

    kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
     funkcije kvadrat */

    printf("Kvadrat broja %d je %d\n", a, kv);
    return 0;
}
```

Rešenje 2.5.2

```
#include <stdio.h>

int kub(int a)
{
    /*
       u listi argumenata funkcije mozemo, a ne moramo, imati
     promenljivu
       istog naziva kao promenljiva koja je deklarisana u main
     funkciji
       (u ovom slucaju promenljiva a); ova promenljiva se razlikuje
       od promenljive a deklarisane u main funkciji i vidljiva je
       samo unutar funkcije kub
    */
    return a*a*a;
}

int main()
{
    int a, kb;
    printf("Unesi ceo broj:");
    scanf("%d",&a);

    kb = kub(a);      /* promenljivoj kb dodeljujemo povratnu vrednost
     funkcije kub */
```

```
23    printf("Kub broja %d je %d\n", a, kb);
      return 0;
25  }
```

Rešenje 2.5.3

```
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 2.5.4

```
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)
15      min=z;

17    return min;
    }
```

```
19
int main(){
21   int x,y,z;

23   /* Ucitavamo brojeve */
     printf("Unesite brojeve: ");
25   scanf("%d%d%d", &x, &y, &z);

27   /* Pozivamo funkciju i ispisujemo rezultat */
     printf("Minimum je: %d\n", min(x,y,z));
29
     return 0;
31 }
```

Rešenje 2.5.5

```
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
9     * 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(){
     float n;
19
     /* Ucitavamo broj */
21   printf("Unesite broj:");
     scanf("%f", &n);
23
     /* Ispisujemo rezultat */
25   printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));

27   return 0;
   }
```

Rešenje 2.5.6

```c
#include <stdio.h>
#include <stdlib.h>

/* Funkcija za realan broj x i ceo broj n odredjuje vrednost stepena
    x^n. */
float stepen(float a, int b)
{
  float s=1;
  int i, abs_b = abs(b);

  for(i=0;i<abs_b;i++)
    s=s*a;

/* ukoliko je izlozilac b negativan, izracunamo a^|b| i vracamo
    reciprocnu vrednost izracunatog stepena */
  return b>0 ? s : 1/s;
}

int main()
{
  int n;
  float x;
  float s;

  printf("Unesi jedan realan i jedan ceo broj:");
  scanf("%f%d",&x,&n);

  s = stepen(x,n);

  printf("%f^%d=%f\n",x,n,s);

  return 0;
}
```

Rešenje 2.5.7

```c
#include <stdio.h>

/* Funkcija za dva uneta neoznacena broja x i n utvrdjuje da li je x
    neki stepen
broja n. Ukoliko jeste, funkcija vraca izlozilac stepena, a u
    suprotnom vraca -1. */
int je_stepen(unsigned x, unsigned n)
{
   int i=1;
   int s=n;

   while(s<x)
   {
      s=s*n;
```

```
13          i++;
    }
15
    if (s==x)
17          return i;

19      return -1;
}
21
int main()
23 {
  unsigned x;
25  unsigned n;
  int st;
27
  scanf("%u%u",&x,&n);
29
  st = je_stepen(x,n);
31
  if (st!=-1)
33    printf("%u=%u^%d\n",x,n,st);
  else
35    printf("%u nije stepen broja %u\n",x,n);

37  return 0;
}
```

Rešenje 2.5.8

```
1 #include <stdio.h>

3 /* Funkcija racuna faktorijel broja.
  * Faktorijel formiramo mnozenjem sa trenutnom vrednoscu broja x,
5 * a zatim smanjujuci tu vrednost za 1.
  * Ukoliko je x = 5, f = 5 * 4 * 3 * 2 * 1
7 */
int faktorijel(int x) {
9
  int f = 1;
11  while(x) {
    f *= x;
13    x--;
  }
15  return f;
}
17
int main() {
19
  int x, y;
21
  printf("Unesite dva broja: ");
```

```
23    scanf("%d%d", &x, &y);

25    /* Provera uslova.
       *
27     * Faktorijel je veoma brza funkcija, tj.
       * s povecanjem broja x, drasticno brzo uvecava se i vrednost x!.
29     * Tip podatka int ima ogranicenje u velicini broja koji moze da
        sadrzi.
       * Za 13! i vece, int ne bi mogao da sacuva sve cifre potrebne za
        zapis tako velikog broja,
31     * te bi doslo do prekoracenja.
       *
33     * Slicno, faktorijel nije definisan nad skupom negativnih celih
        brojeva.
       */
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 2.5.9

```
1  #include <stdio.h>

3  int euklid(int x, int y)
   {
5    int r;
     /* Euklidov algoritam */
7    while(y)    /* algoritam se zaustavlja kada vrednost */
     {           /* promenljive y postane nula */
9      r=x%y;
       x=y;
11     y=r;
     }
13
     return x; /* nzd je sacuvan u promenljivoj x */
15 }

17 int main()
   {
19   int a,b;
     int nzd;
21
     printf("Unesite dva cela broja:");
23   scanf("%d%d", &a,&b);
```

```
25    nzd = euklid(a,b);  /* promenljivoj nzd dodeljujemo povratnu
        vrednost funkcije euklid */

27    printf("Najveci zajednicki delilac je %d\n", nzd);

29    return 0;
}
```

Rešenje 2.5.10

```
#include <stdio.h>

float zbir_reciprocnih(int n)
{
  float z=0;
  int i;
  for(i=1;i<=n;i++)
    z+=1.0/i;  /* da bismo dobili reciprocnu vrednost broja, vazno je
      da izbegnemo celobrojno deljenje dva cela broja */
  return z;    /* tako sto ce npr deljenik biti 1.0 umesto 1 */
}

int main()
{
  int n;
  printf("Unesi jedan pozitivan ceo broj:\n");
  scanf("%d", &n);
  printf("Zbir je %.2f\n", zbir_reciprocnih(n));
  /* povratna vrednost funkcije zbir_reciprocnih je float; funkciju
    mozemo pozvati u okviru
      naredbe printf i umesto specifikatora %.2f bice ispisana
    povratna vrednost funkcije
      zbir_reciprocnih zaokruzena na dve decimale */
  return 0;
}
```

Rešenje 2.5.11

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

void ispis(float x, float y, int n) /* funkcija nema povratnu
    vrednost; zbog toga je povratni tip void */
{
  float i;
  float korak=(y-x)/(n-1);

  for(i=x;i<=y;i+=korak)
    printf("%.4f ", sin(i));
```

```
12    printf("\n");

14  }

16  int main()
    {
18    float a,b;
      int n;
20    float t;
      printf("Unesi dva realna broja:");
22    scanf("%f%f",&a,&b);
      printf("Unesi jedan prirodan broj:");
24    scanf("%u",&n);

26    if (n<=1 || a==b)
      {
28        printf("Nekorektan unos\n");
          return -1;
30    }
      if (b<a) /* u slucaju da je desni kraj intervala manji od levog */
32    {         /* zamenimo im mesta */
         t=a;
34       a=b;
         b=t;
36    }


38


40    ispis(a,b,n);


42
      return 0;
44  }
```

Rešenje 2.5.12

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    /* Funkcija racuna aritmeticku sredinu cifara datog celog broja */
5   float aritmeticka_sredina(int x) {

7     /* Proveravamo slucaj broja 0 */
      if(x==0)
9       return 0;

11    /* Brojaci sa svojim pocetnim vrednostima */
      int zbir_cifara = 0;
13    int broj_cifara = 0;
```

```
15    /* U slucaju da je broj negativan,
       * ostaci pri deljenju sa 10 bi takodje bili negativni
17     * sto bi se moralo dodatno proveravati.
       * Stoga je zgodnije posmatrati samo apsolutnu vrednost broja.
19     */
      x = abs(x);
21
      /* Izdvajamo cifru po cifru s kraja zapisa broja x.
23     * Uslov while(x > 0) ekvivalentan je uslovu while(x)
       * Broj x je pozitivan (apsolutna vrednost), pa cifre izdvajati
25     * sve dok ima cifara izdvajanje, tj. broj x nije deljenjem sa 10
         postao 0
       */
27    while(x) {

29      zbir_cifara += x % 10;
        broj_cifara++;
31
        x /= 10;
33    }

35    /* Da nije izvrsena implicitna konverzija (kastovanje),
       * operator / obavljao bi celobrojno deljenje.
37     * Zato je potrebno da bar jedan operand bude ceo broj,
       * sto je u ovom slucaju deljenik.
39     * Operator kastovanja je unaran, pa ima veci prioritet od /
       */
41    return (float) zbir_cifara / broj_cifara;
}
43
int main() {
45
      int x;
47
      printf("Unesite ceo broj: ");
49    scanf("%d", &x);

51    printf("Arimeticka sredina cifara broja %d je %.3f\n", x,
        aritmeticka_sredina(x));

53    /* Iako smo u funkciji aritmeticka_sredina menjali broj x,
       * prosledjivanjem argumenata funkciji pravi se lokalna kopija
         promenljive x za tu funkciju,
55     * tako da je ona menjana unutar funkcije aritmeticka_sredina,
       * a promenljiva x iz funkcije main() ostaje netaknuta.
57     */

59    return 0;
}
```

Rešenje 2.5.13

```c
#include<stdio.h>
#include<stdlib.h>

/* Funkcija ispituje da li se cifra c nalazi u zapisu celog broja x.
    */
int sadrzi(int x, int c)
{
   char cifra;
   x=abs(x);
   while(x)
   {
      cifra = x%10;
      if (cifra==c)
         return 1;
      x/=10;
   }
   return 0;
}
int main()
{
  int x;
  int c;
  printf("Unesi jedan ceo broj i jednu cifru:");
  scanf("%d%d",&x,&c);
  if (sadrzi(x,c))
     printf("Cifra %d se nalazi u zapisu broja %d\n",c,x);
  else
     printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
  return 0;
}
```

Rešenje 2.5.14

```c
#include<stdio.h>
#include<stdlib.h>

int broj_ncifara(int x)
{
   int s=0;
   char cifra;
   x = abs(x);

   while(x)
   {
      cifra = x%10;
      s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
    neparna,
                       a 0 kada je cifra parna */
      x/=10;
   }
```

```
17
    return s;
19 }

21 int main()
   {
23   int x;
     do
25   {
        scanf("%d",&x);
27      printf("%d\n", broj_ncifara(x));
     } while(x!=0);
29
     return 0;
31 }
```

Rešenje 2.5.15

```
   /*
2
   a) Napisati funkciju sve_parne_cifre koja ispituje da li se dati ceo
       broj sastoji iskljucivo iz parnih cifara. Funkcija treba
4  da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.

6  b) 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.
8
   c) Napisati potom glavni program koji na uneti ceo broj primenjuje
       napisane funkcije i ispisuje odgovarajuce poruke.
10
   Na primer, za uneti broj 222, program treba da ispise:
12 Sve cifre broja su parne.
   Sve cifre broja su jednake.
14
   A za uneti broj -284:
16 Sve cifre broja su parne.
   Broj sadrzi razlicite cifre
18
   */
20 #include <stdio.h>
   #include <stdlib.h>
22
   int sve_parne_cifre(int x) /* funkcija vraca 1 ako su sve cifre broja
        parne i 0 u suprotnom*/
24 {
     char d;
26   x=abs(x);       /* uzimamo apsolutnu vrednost broja za slucaj da je
        broj negativan */
     while (x>0)
28   {
```

```
    d=x%10;        /* izdvajamo cifru broja */

    if (d%2==1)    /* u slucaju da je neparna, to znaci da nisu sve
    cifre broja parne */
      return 0;    /* vracamo 0 */

    x/=10;         /* "uklanjamo" poslednju cifru broja celobrojnim
    deljenjem sa  10 */
  }

  return 1;       /* ukoliko se while petlja zavrsila, to znaci da
    uslov d%2==1 nije
                    nijednom bio ispunjen i da su sve cifre broja
    parne; zbog toga
                    vracamo 1
                 */

}

int sve_cifre_jednake(int x) /* funkcija vraca 1 ako su sve cifre
    broja jednake i 0 u suprotnom*/
{
  char d;
  char prva_cifra;
  x=abs(x);
  prva_cifra = x%10; /* izdvajamo prvu cifru broja */
  x/=10;             /* broj delimo sa 10 jer smo prvu cifru vec
    izdvojili */

  while(x)
  {
     d = x%10;

     if (d!=prva_cifra)
        return 0;

     x/=10;
  }

  return 1;
}
main()
{
  int x;
  int d;

  printf("unesi ceo broj:");
  scanf("%d", &x);

  if (sve_parne_cifre(x))
    printf("Sve cifre broja su parne\n");
  else
```

```
    printf("Broj sadrzi bar jednu neparnu cifru\n");

  if (sve_cifre_jednake(x))
    printf("Sve cifre broja su jednake\n");
  else
    printf("Broj sadrzi razlicite cifre \n");

}
```

Rešenje 2.5.16

```
#include <stdio.h>

/* 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
 * isti broj puta.
 * Ideja je sledeca:
 *  iz broja x izdvajaju se redom cifra po cifra s kraja,
 *  a zatim se svaka takva cifra trazi i u broju y.
 *  Ukoliko postoji, eliminise se prvi put kada se pojavi (dakle,
    samo jednom).
 *  Ukoliko su sve cifre iste (***redosled nije bitan***),
 *  na kraju ce i iz x i iz y biti sve cifre eliminisane",
 *  te ostaju nule u oba broja.
 *
 *  Broj novo_y formira se, zbog jednostavnosti, pomocu Heronovog
    obrasca.
 *  Ovaj postupak obradjen je u okviru funkcije int izbaci_cifru(int
    y, int cifra).
 */

int izbaci_cifru(int y, int cifra) {

  int novo_y = 0;
  int indikator = 0;
  int izdvojena_cifra;

  while(y) {

    izdvojena_cifra = y % 10;
    /* U slucaju da se cifra razlikuje od one koju treba eliminisati,
     * ili ukoliko je jedna cifra vec elimisana =>
     * tekucu cifru ukljuciti prilikom formiranja novog y
     * */
    if(izdvojena_cifra != cifra || indikator)

      /* Heronov obrazac.
       * Menja poredak cifara, ali on u ovom slucaju i nije bitan.
       */
      novo_y = novo_y*10 + izdvojena_cifra;
```

```
37      else

39        /* U slucaju da je cifra vec eliminisana,
           * ne treba je opet eliminisati.
41         * Za svaku pojavu cifre iz x,
           * eliminise se jedna odgovarajuca pojava
43         * te cifre iz y.
           */
45        indikator = 1;

47      y /= 10;
    }
49
    return novo_y;
51  }

53  int zapis(int x, int y) {

55    /* Cifra koja se izdvaja iz x, a onda eliminise iz y */
      int cifra;
57
    /* U slucaju da su prosledjeni brojevi negativni */
59    x = abs(x);
      y = abs(y);
61
    while(x) {
63
        cifra = x % 10;
65      x /= 10;

67      y = izbaci_cifru(y, cifra);

69      /* otkomentarisati donju liniju radi lakseg pracenja rada
        programa: */
        // printf("Iz x izdvojeno: %d\n\tx = %d, y = %d\n\n", cifra, x, y
        );
71    }

73    return (x == 0 && y == 0);
  }
75
  int main() {
77
    int x, y;
79  printf("Unesite dva cela broja: ");
    scanf("%d%d", &x, &y);
81
    if(zapis(x, y))
83    printf("Uslov je ispunjen!\n");
    else
85    printf("Uslov nije ispunjen!\n");
```

```
87    return 0;
}
```

Rešenje 2.5.17

```
#include <stdio.h>
#include <stdlib.h>

/* Funkcija proverava da li se
 * cifre u zapisu broja nalaze u rastucem poretku.
 *
 * Situacija od interesa je kada za dve uzastopne cifre to nije
    slucaj.
 * Tada ne treba proveravati i za ostale cifre,
 * vec odmah prekinuti izvrsavanje funkcije.
 *
 * Ukoliko funkcija nije ranije prekinuta,
 * to znaci da cifre jesu u rastucem poretku
 * (odnosno, kako izdvajamo cifre od nazad, u stvari proveravamo
    opadajuci poredak),
 * te treba vratiti 1.
 */

int rastuce(int n) {

  int tekuca_cifra;
  int prethodna_cifra;

  n = abs(n);

  /* Prvu cifru (odnosno, poslednju u zapisu broja)
   * izdvajamo izvan petlje
   * kako bismo mogli da je poredimo sa narednom
   */
  tekuca_cifra = n % 10;
  n /= 10;

  while(n) {

    /* Cifra koja je bila tekuca u prethodnoj iteraciji petlje,
     * u novoj iteraciji postaje prethodna.
     *
     * Novoizdvojena cifra je tekuca.
     */
    prethodna_cifra = tekuca_cifra;
    tekuca_cifra = n % 10;

    /* Ukoliko smo naisli na cifre koje kvare rastuci poredak,
     * prekidamo izvrsavanje funkcije sa odgovarajucom povratnom
    vrednoscu 0.
     */
```

```
    if(prethodna_cifra < tekuca_cifra)
45    return 0;

47    /* Inace, nastavljamo sa izdvajanjem cifara */
      n /= 10;
49  }

51  return 1;
}
53
int main() {
55
  int x;
57  printf("Unesite broj: ");
  scanf("%d", &x);
59
  if(rastuce(x))
61    printf("Cifre su u rastucem poretku!\n");
  else
63    printf("Cifre nisu u rastucem poretku!\n");

65  return 0;
}
```

Rešenje 2.5.18

```
1 #include <stdio.h>
  #include <stdlib.h>
3
  /* Funkcija proverava da li su dve uzastopne cifre
5  * razlicite parnosti.
   *
7  * Interesantna situacija je ukoliko su dve uzastopne cifre
   * obe parne, odnosno obe neparne.
9  * 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 izvrsavanjem 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 vraca 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);

  /* jednu cifru izdvajamo van petlje
   * kako bismo mogli da je odmah u petlji poredimo sa narednom
   */
  prethodna_cifra = x % 10;
  x /= 10;

  while(x) {

    tekuca_cifra = x % 10;

    if(tekuca_cifra % 2 == prethodna_cifra % 2)
      return 0;

    /* tekuca cifra postaje prethodna cifra za narednu iteraciju */
    prethodna_cifra = tekuca_cifra;
    x /= 10;
  }

  return 1;
}

int main() {

  int x;
  printf("Unesite broj: ");
  scanf("%d", &x);

  if(par_nepar(x))
    printf("Broj ispunjava uslov!\n");
  else
    printf("Broj ne ispunjava uslov!\n");

  return 0;
}
```

Rešenje 2.5.19

```
#include <stdio.h>

/* Funkcija koja uklanja broj stotina iz broja n */
int ukloni_stotine(int n){

  /* Ako je broj izmedju -100 i 100 nema cifru desetica pa onda
     vracamo isti taj broj */
  if(n>-100 && n<100)
  return n;
  else
  {
  /* U suprotnom vracamo broj sa uklonjenom cifrom stotina */
```

```
12
   /* Odredjujemo znak broja */
14 int znak=(n<0)? -1 : 1;

16 /* I nadalje radimo sa apsolutnom vrednoscu broja */
   n=abs(n);
18
   return znak*((n/1000)*100 + n%100);
20 }
}
22
/* Funkcija koja vraca znak broja */
24 int znak(int broj){
   return broj<0?-1:1;
26 }

28 int main(){

30 int broj;

32 while(1){

34 /* Ucitavamo broj sa standardnog ulaza */
   printf("Unesite broj: ");
36 scanf("%d", &broj);

38 /* Broj 0 oznacava kraj rada */
   if(broj == 0)
40   break;

42 /* Ispisujemo rezultat, vodeci racuna da program treba da radi
     ispravno i za negativne brojeve */
   printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
44 }


46
   return 0;
48 }
```

Rešenje 2.5.20

```
#include<stdio.h>
2 #include<math.h>

4 int rotacija(int n){

6 /* U promenljivoj broj pamtimo originalnu vrednost n */
   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;
30      br++;
      }
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
      return znak* ((broj%(int)pow(10, br))*10 + n);
40  }

42  int main(){

44    int n;
      while(1){
46
      /* Ucitavamo broj */
48    printf("Unesite broj: ");
      scanf("%d", &n);
50
      /* Ako je uneta 0, izlazimo iz petlje */
52    if(n == 0)
        break;
54
      /* Stampamo broj rotiran za jedno mesto u levo */
56    printf("%d\n", rotacija(n));
      }
58
```

```
60    return 0;
}
```

Rešenje 2.5.21

```
#include <stdio.h>

/* Funkcija koja racuna zbir delilaca broja x */
int zbir_delilaca(int x){
  int i=0;

  /* Na pocetku zbir inicijalizujemo na 0 */
  int zbir = 0;

  /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
  for(i=1; i<=x; i++){
  if(x % i == 0)
    zbir += i;
  }

  /* Vracamo dobijeni zbir */
  return zbir;
}

int main(){

  int k, i;

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

  /* Proveravamo korektnost ulaza */
  if(k <= 0)
    printf("Greska: pogresan unos!\n");
  else{

    /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
    for(i=1; i<=k; i++)
    printf("%d ", zbir_delilaca(i));

    printf("\n");
  }


  return 0;
}
```

Rešenje 2.5.22

```c
#include <stdio.h>
#include <math.h>

/* Funkcija vraca 1 ako je broj prost i 0 u suprotnom.  */
int prost (int x) /* 1-broj je prost, 0-broj nije prost */
{
  int i;

  if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
    return 1;

  if (x%2==0)       /* parni brojevi nisu prosti */
    return 0;

  for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
    if (x%i==0) /* ako je pronadjen, to znaci da broj nije prost */
      return 0; /* zavrsavamo funkciju */

  /* ukoliko izvrsavanje funkcije dodje do poslednje naredbe return,
     to znaci da broj nije ispunio nijedan od prethodnih uslova
     (nije ni 2, ni 3, ni paran, niti ima ijednog delioca), odakle
     sledi da je prost i zbog toga vracamo 1
  */
  return 1;
}

int main()
{
  int n;
  scanf("%d",&n);
  int i,j;

  i=1; /* kandidat za prost broj */
  j=0; /* brojac prostih brojeva */
  while(j<n)
  {
     if (prost(i))          /* ako je broj prost */
     {
        printf("%d\n", i); /* stampamo ga i */
        j++;                /* uvecavamo brojac prostih brojeva */
     }
     i++; /* bilo da je i prost ili ne, uvecavamo ga za 1 i
    nastavljamo sa sledecom iteracijom */
  }

  return 0;
}
```

Rešenje 2.5.24

```c
#include<stdio.h>
#include<math.h>

/* Funkcija racuna vrednost e^x kao parcijalnu sumu reda
 suma(x^n/n!), gde indeks n ide od
 od 0 do beskonacno, pri cemu se sumiranje vrsi dok
 je razlika sabiraka u redu po apsolutnoj vrednosti
 manja od eps. */
double e_na_x(double x, double eps)
{
  double s=1;
  double clan=1;
  int n=1;

  /*
     parcijalnu sumu formiramo tako sto u svakoj iteraciji petlje
     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
     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
     s i clan inicijalizujemo na vrednost 1

     sumiranje se sprovodi dogod je sabirak po apsolutnoj vrednosti
     veci od trazene tacnosti eps
  */

  do
  {
     clan = (clan*x)/n;
     s += clan;
     n++;
  } while(fabs(clan)>eps);

  return s;
}

int main()
{
  double x,eps;
  printf("x=");
  scanf("%lf", &x);
  printf("eps=");
  scanf("%lf", &eps);

  printf("e^%f=%f\n", x, e_na_x(x,eps));
  return 0;
```

```
}
```

Rešenje 2.5.25

```
/*
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
jedinicom. Napisati program koji za uneti broj odredjuje da li je srecan.
Na primer:
- 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.
- broj 991 je srecan, zbir njegovih cifara je 19, zbir cifara broja 19 je 10, zbir cifara
broja 10 je 1.
- broj 372 nije srecan, zbir njegovih cifara je 12, zbir cifara broja 12 je 3

Napisati funkciju koja vraca 1 ako je broj srecan, a 0 u suprotnom.

Napisati program koji omogucava korisnuku da unese prirodan broj, poziva funkciju
i ispisuje da li je dati broj srecan. Potom traziti od korisnika da unese prirodan
broj n i ispisati sve srecne brojeve od 1 do n.
*/

#include<stdio.h>

int zbir_cifara(int x)
{
   int s=0;
   char cifra;
   while(x)
   {
      cifra = x%10;
      s+=cifra;
      x/=10;
   }
   return s;
}

int srecan(int x)
{
   int s; /* promenljiva s sadrzi sumu cifara */

   do
```

```
39    {
        s=zbir_cifara(x);
41      x=s; /* kada izracunamo sumu cifara, dodeljujemo je promenljivoj
        x jer iz te promenljive izdvajamo cifre u funkciji zbir_cifara
       */
      } while(x>=10);
43
      return (x==1);
45
}
47
int main()
49 {
      unsigned n;
51    int i;
      printf("Unesi jedan neoznacen broj:");
53    scanf("%u",&n);

55    for(i=1;i<=n;i++)
         if (srecan(i))
57          printf("%d je srecan\n", i);

59   return 0;
}
```

Rešenje 2.5.26

```
1  #include <stdio.h>

3  /* Funkcija racuna broj x na n-ti stepen */
   int stepen(int x, int n) {
5
     int i;
7    /* Promenljiva u kojoj se cuva proizvod broja x sa samim sobom, n
       puta */
     int st = 1;
9
     for(i = 1; i <= n; i++)
11     st *= x;

13   return st;
   }
15
   /* Funkcija proverava da li je broj Armstrongov. */
17 int armstrong(int x) {

19   /* u y se cuva zbir i-tih stepena cifara */
     int y;
21   /* stepen za koji se proverava */
     int i = 1;
23   /* prilikom izdvajanja cifara, broj x se menja,
```

```
   * te treba imati promenlju koja cuva pravu vrednost x
   */
  int original = x;

  do {

    y = 0;
    /* Racunamo i-te stepene za svaku cifru,
     * i istovremeno te stepen sabiramo.
     * Rezultat pamtimo u promenljivoj y.
     */
    while(x) {

      y += stepen(x % 10, i);
      x /= 10;
    }

    /* x je sada promenjen, pa ga treba vratiti na pravu vrednost. */
    x = original;
    i++;

  } while(y < x); /* Petlju vrtimo sve dok je zbir stepena cifara
    manji od datog broja. */

  /* Ukoliko smo nasli i, takvo da je zbir i-tih stepena cifara
   * jednak upravo broju x, takav broj je Armstrongov,
   * te izraz x == y vraca 1.
   *
   * Inace, vraca 0, tj. broj nije Armstrongov.
   */
  return x == y;
}

int main() {

  int x;
  printf("Unesite broj: ");
  scanf("%d", &x);

  if(armstrong(x))
    printf("Broj je Armstrongov!\n");
  else
    printf("Broj nije Armstrongov!\n");

  return 0;
}
```

Rešenje 2.5.27

```
#include <stdio.h>

```

```c
/* Funkcija broji koliko puta se realan broj x
 * javlja u nizu unetih brojeva sa tastature.
 *
 * Brojevi se unose sve do pojave 0,
 * pa treba koristiti do..while petlju,
 * kako bi bar jedan broj bio unet (makar bio i 0).
 */
int prebrojavanje(float x) {

  /* y prihvata uneti broj sa tastature */
  float y;
  /* br_pojavljivanja je brojac koji broji koliko puta se broj x
     javlja u unetom nizu brojeva */
  int br_pojavljivanja = 0;

  printf("Unesite brojeve: ");
  do {

    /* Unosimo broj. */
    scanf("%f", &y);

    /* Poredimo uneti broj sa datim brojem.
     * Ukoliko je unet bas trazeni broj,
     * uvecavamo brojac.
     * */
    if(x == y)
      br_pojavljivanja++;

  } while(y); /* Sve dok nije uneta 0 */

  return br_pojavljivanja;
}

int main() {

  float x;
  int br_pojavljivanja;

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

  br_pojavljivanja = prebrojavanje(x);
  printf("Broj pojavljivanja broja %.2f je: %d\n", x,
    br_pojavljivanja);

  return 0;
}
```

Rešenje 2.5.28

```c
#include <stdio.h>

/* Funkcija racuna n-ti clan Fibonacijevog niza.
 * Clanovi ovog niza zadaju se rekurzivno tj. u zavisnosti od
    prethodnih clanova.
 * Fibonacijevi brojevi od 0. do 47. se mogu smestiti u tip int, a
    kako n moze uzimati vrednosti
 * od 1 do 50, povratni tip funkcije je long int.
 */
long int fibonaci(int n) {

  int i;

  /* f0 i f1 su prva dva clana niza */
  int f0 = 1;
  int f1 = 1;
  /* promenljiva u kojoj se cuvaju opsti clanovi: n+2, n+1. i n-ti
    clan */
  long int fn2, fn1, fn;

  /* ukoliko treba vratiti nulti ili prvi clan,
   * njih ne treba racunati
   * jer su vec dati.
   */
  if(n == 0 || n == 1)
    return 1;

  /* postavljamo prethodne clanove niza */
  fn = f0;
  fn1 = f1;
  /* racunamo od drugog clana, pa dok ne dodjemo do n-tog */
  for(i = 2; i <= n; i++) {

    /* izracunamo n+2-i clan niza sabiranjem prethodna dva clana */
    fn2 = fn1 + fn;
    /* promenimo prethodne clanove niza, zbog naredne iteracije */
    fn =  fn1;
    fn1 = fn2;
  }

  return fn2;
}

int main() {

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

  /* Provera vrednosti za broj n */
  if(n <0 || n > 50) {
    printf("Greska: nedozvoljena vrednost!\n");
```

```
  }
  else{
    printf("%ld\n", fibonaci(n));
  }

  return 0;
}
```

Rešenje 2.5.29

```
/*
. a) Napisati funkciju

  int konverzija (int c)

koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.

b) Napisati program koji omogucava korisniku da unese niz karaktera
sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
Na primer, za uneti tekst "Kolokvijum iz Prog1 je 1.12." program
treba da ispise "kOLOVKIJUM IZ pROG1 JE 1.12."

*/
#include <stdio.h>

int konverzija(int c)
{
  /* kljucna rec return vraca povratnu vrednost funkcije (ako je ima)
     */
  /* i zavrsava izvrsavanje funkcije */

  if (c>='A' && c<='Z')
    return c+'a'-'A';

  if (c>='a' && c<='z')
    return c-'a'+'A';

  return c;
}

int main()
{
  int c;

  while((c=getchar())!=EOF)  /* korisnik unosi karakter po karakter
    do konstante EOF */
    putchar(konverzija(c));  /* funkcija putchar ispisuje jedan
                                karakter na standardni izlaz */

  return 0;
}
```

Rešenje 2.5.30

```c
#include <stdio.h>

/* Funkcija vraca karakter koji se u abecedi
 * nalazi k mesta pre datog karaktera c
 */
char sifra(char c, int k) {

  /* Ukoliko je uneto malo slovo ... */
  if(c >= 'a' && c <= 'z')
    /* Pri tome karakter koji je k pozicija pre datog karaktera
    ispada iz opsega malih slova ... */
    if(c-k < 'a')
      /* Treba krenuti s drugog kraja abecede, racunajuci i
    preskocena slova.
       *
       * Na primer, ukoliko je c = 'b' i k = 2
       * Jedan karakter pre 'b' je 'a'.
       * Dva karaktera pre 'b' je 'z' (kruzno).
       *
       * 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)
       * sto je u ovom slucaju 'b' - 'a' = 1.
       *
       * Ostatak karaktera do k ispisujemo, ali gledavsi unazad od z.
       * Zato racunamo k - (c - 'a') - 1.
       *
       * Od k oduzimamo rastojanje izmedju c i 'a',
       * kako bismo dobili preostali broj karaktera koji treba
    preskociti.
       */
      return 'z' - (k - (c -  'a') - 1);
    else
      /* U suprotnom, karakter ne ispada iz opsega malih slova, te je
     dovoljno bas njega i vratiti */
      return c-k;

  /* Ukoliko je uneto veliko slovo ... */
  else if(c >= 'A' && c <= 'Z')
    if(c-k < 'A')
      return 'Z' - (k - (c -  'A') - 1);
    else
      return c-k;


  return c;
```

```
}

int main() {

  int k;
  char c;

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

  printf("Unesite tekst (CTRL + D za prekid): ");
  while((c = getchar()) != EOF)
    putchar(sifra(c, k));

  return 0;
}
```

Rešenje 2.5.31

```
#include<stdio.h>

/* Funkcija koja proverava da li je godina prestupna */
int prestupna(int godina){
  if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
  return 1;
  else
  return 0;
}

/* Funkcija koja proverava da li postoji prestupna godina u datom
    intervalu */
int postoji_prestupna(int g1, int g2){
  for(; g1<=g2; g1++){
  if(prestupna(g1))
    return 1;
  }
  return 0;
}

int main(){

  int g1, g2;

  /* Ucitavamo godine */
  printf("Unesite dve godine: ");
  scanf("%d%d", &g1, &g2);

  /* Proveravamo korektnost ulaza */
  if(g1 < 0 || g2 < 0 || g1>g2){
  printf("Greska: pogresan unos!\n");
  }
```

```
  else{

  /* Proveravamo da li uopste postoji prestupna godina u datom
     intervalu */
  if(postoji_prestupna(g1,g2)){
    /* Ako postoje, ispisujemo ih */
    printf("Prestupne godine su: ");
    for(; g1<=g2; g1++){
    if(prestupna(g1))
      printf("%d ", g1);
    }
    printf("\n");
  }else{
    /* U suprotnom, stampamo odgovarajucu poruku */
    printf("Nema prestupnih godina u ovom intervalu!\n");
  }
  }
  return 0;
}
```

Rešenje 2.5.37

Rešenje 2.5.37

Rešenje 2.5.38

```
#include<stdio.h>

/* Funkcija koja iscrtava romb */
void romb(int n){
  int i, j;

  /* U svakoj liniji */
  for(i=0; i<n; i++){

  /* Prvo ispisujemo n-i-1 razmaka */
  for(j=0; j<n-i-1; j++)
    printf(" ");

  /* Zatim ispisujemo n zvezdica */
  for(j=0; j<n; j++)
    printf("*");

  /* Na kraju svake linije stoji oznaka za novi red */
  printf("\n");
  }

}

```

```
int main(){
  int n;

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

  /* Proveravamo korektnost ulaza i ispisujemo rezultat */
  if(n<=0)
  printf("Greska: pogresna dimenzija!\n");
  else
  romb(n);

  return 0;
}
```

Rešenje 2.5.39

```
#include<stdio.h>

/* Funkcija koja stampa n zvezdica za kojima sledi znak za novi red
    */
void stampaj_zvezdice(int n){
  int i;
  for(i=0; i<n; i++)
    printf("*");

  printf("\n");
}

/* Funkcija koja crta grafikon */
void grafikon_h(int a, int b, int c, int d)
{
  int i;

  /* Prvo ispisujemo a zvezdica */
  stampaj_zvezdice(a);

  /* Zatim u sledecem redu b zvezdica */
  stampaj_zvezdice(b);

  /* Zatim u sledecem redu c zvezdica */
  stampaj_zvezdice(c);

  /* Zatim u poslednjem redu d zvezdica */
  stampaj_zvezdice(d);

}

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 2.5.40

```
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  void stampaj_znak(int polje, int granica) {
      if(polje<granica)
19      printf(" ");
      else
21      printf("*");
    }
23
    /* Funkcija koja iscrtava vertikalni grafikon */
25  void grafikon_v(int a, int b, int c, int d){
      int i, max;
27
      /* Na pocetku je potrebno pronaci najvecu od ove cetiri vrednosti
        */
29    max=maksimum(a, b, c, d);

31    /* Grafikon ukupno ima max horizontalnih linija */
      for(i=0; i<max; i++){
33
```

```
    /* U svakoj od horizontalnih linija se nalazi po 4 polja:
35    polje za a,b,c i d uspravnu liniju.
    U svako od polja treba da se upise ili * ili belina,
37    u zavisnosti od vrednosti i toga u kojoj liniji se trenutno
      nalazimo
     */
39
  /* Stampamo znak za polje a */
41    stampaj_znak(i, max-a);

43  /* Proveravamo uslov za polje b */
      stampaj_znak(i, max-b);
45
  /* Proveravamo uslov za polje c */
47    stampaj_znak(i, max-c);

49  /* Proveravamo uslov za polje d */
      stampaj_znak(i, max-d);
51
  /* Na kraju svake horizontalne linije stampamo novi red */
53  printf("\n");
  }
55 }

57 int main(){
  int a,b,c,d;
59
  /* Ucitavamo vrednosti a,b,c,d */
61  printf("Unesite vrednosti: ");
  scanf("%d%d%d%d", &a, &b, &c, &d);
63
  /* Proveravamo korektnost ulaza i stampamo grafikon */
65  if(a <0 || b<0 || c<0 || d<0)
  printf("Greska: pogresan unos!\n");
67  else
  grafikon_v(a,b,c,d);
69
  return 0;
71 }
```

3

Predstavljanje podataka

3.1 Nizovi

Zadatak 3.1.1 Ako su $a = (a_1, \ldots, a_n)$ i $b = (b_1, \ldots, b_n)$ vektori dimenzije n, njihov skalarni proizvod je $a \cdot b = a_1 \cdot b_1 + \ldots + a_n \cdot b_n$. Napisati program koji računa skalarni proizvod dva vektora. Vektori se zadaju kao celobrojni nizovi sa najviše 100 elemenata. Program učitava dimenziju i elemente nizova, a na izlaz ispisuje vrednost skalarnog proizvoda.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 5
 Unesite koordinate vektora a:
 8 -2 0 2 4
 Unesite koordinate vektora b:
 35 12 5 -6 -1
 Skalarni proizvod vektora a i b:
 240
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 3
 Unesite koordinate vektora a:
 -1 0 1
 Unesite koordinate vektora b:
 5 5 5
 Skalarni proizvod vektora a i b:
 0
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 120
 Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.1]

Zadatak 3.1.2 Napisati program koji za učitani niz ispisuje:

(a) elemente niza koji se nalaze na parnim pozicijama.

(b) parne elemente niza.

Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 6
 Unesite elemente niza:
 1 8 2 -5 -13 75
 Elementi niza na parnim pozicijama:
 1 2 -13
 Parni elementi niza:
 8 2
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 3
 Unesite elemente niza:
 11 81 -63
 Elementi niza na parnim pozicijama:
 11 -63
 Parni elementi niza:
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza:
 -4
 Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.2]

Zadatak 3.1.3 Napisati program koji za učitani ceo broj, ispisuje broj pojavljivanja svake od cifara u zapisu tog broja. Uputstvo: *Za evidenciju broja pojavljivanja svake cifre pojedinačno, koristiti niz.*

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj:
 2355623
 U zapisu broja 2355623, cifra 2 se pojavljuje 2 puta
 U zapisu broja 2355623, cifra 3 se pojavljuje 2 puta
 U zapisu broja 2355623, cifra 5 se pojavljuje 2 puta
 U zapisu broja 2355623, cifra 6 se pojavljuje 1 puta
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj:
 -39902
 U zapisu broja -39902, cifra 0 se pojavljuje 1 puta
 U zapisu broja -39902, cifra 2 se pojavljuje 1 puta
 U zapisu broja -39902, cifra 3 se pojavljuje 1 puta
 U zapisu broja -39902, cifra 9 se pojavljuje 2 puta
```

[Rešenje 3.1.3]

Zadatak 3.1.4 Napisati program koji za dva cela broja x i y koja se učitavaju sa standardnog ulaza proverava da li se zapisuju pomoću istih 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 3.1.4]

Zadatak 3.1.5 Napisati program koji učitava karaktere sa standardnog ulaza sve do kraja ulaza i izračunava koliko se puta u unetom tekstu pojavila svaka od cifara, svako malo slovo i svako veliko slovo. Ispisati broj pojavljivanja samo za karaktere koji su se u unetom tekstu pojavili barem jednom. Uputstvo: *Za evidenciju broja pojavljivanja cifara, malih i velih slova korisiti pojedinačne nizove.*

Primer 1

```
Interakcija sa programom:
 123 abcabcabc 123
 Karakter 1 se pojavljuje 2 puta
 Karakter 2 se pojavljuje 2 puta
 Karakter 3 se pojavljuje 2 puta
 Karakter a se pojavljuje 3 puta
 Karakter b se pojavljuje 3 puta
 Karakter c se pojavljuje 3 puta
```

[Rešenje 3.1.5]

Zadatak 3.1.6 Sa standardnog ulaza se unosi jedna linija teksta. Napisati program koji izračunava i ispisuje koliko puta se pojavilo svako od slova engleskog alfabeta u unetom tekstu. Ne praviti razliku između malih i velikih 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:0 t:0 u:0 v:0 w:0 x:0 y:0 z:0
```

Primer 2

```
Interakcija sa programom:
 DanaS j3 _j_utRo laBU78d
  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:0 t:0 u:0 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:1 t:0 u:1 v:0 w:0 x:0 y:0 z:0
```

Zadatak 3.1.7 Napisati program koji za dva učitana niza a i b dimenzije n formira i na izlaz ispisuje niz c koji se dobija naizmeničnim raspoređivanjem elemenata nizova a i b, tj. $c = [a_0, b_0, a_1, b_1, \ldots, a_{n-1}, b_{n-1}]$. Pretpostaviti da dimenzija učitanih nizova nije veća od 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju nizova:
 5
 Unesite elemente niza a:
 2 -5 11 4 8
 Unesite elemente niza b:
 3 3 9 -1 17
 Rezultujuci niz:
 2 3 -5 3 11 9 4 -1 8 17
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju nizova:
 105
 Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.7]

Zadatak 3.1.8 Sa standardnog ulaza se učitava ceo broj n (manji od 100) i elementi dvaju nizova a i b dimenzije n. 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: Nedozvoljena vrednost!
```

[Rešenje 3.1.8]

Zadatak 3.1.9 Napisati program koji sa standardnog ulaza učitava 10 celih brojeva i razdvaja ih na parne i neparne tako što parne brojeve upisuje na početak niza, a neparne brojeve na kraj niza. Ispisati niz dobijen na ovaj način. NAPOMENA: *Nije dozvoljeno koristiti pomoćne nizove.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 10 brojeva:
 -2 8 11 53 59 20 17 -8 3 14
 Rezultujuci niz:
 -2 8 20 -8 14 3 17 59 53 11
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 10 brojeva:
 9 142 -9 -278 -69 33 34 28 -6 14
 Rezultujuci niz:
 142 -278 34 28 -6 14 33 -69 -9 9
```

Zadatak 3.1.10 Napisati program koji učitava dimenziju n celobrojnog niza a i njegove elemente, i iz niza a izbacuje sve elemente koji nisu deljivi svojom poslednjom cifrom. Izuzetak su elementi čija je poslednja cifra 0 koje treba zadržati. Program treba da ispiše izmenjeni niz na standardni izlaz. Niz a sadrži najviše 100 elemenata.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 9
 Unesite elemente niza a:
 173 -25 23 7 17 25 34 61 -4612
 Niz a nakon izmene:
 -25 7 25 61 -4612
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 0
 Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.10]

Zadatak 3.1.11 Napisati program koji u nizu dužine n (broj manji od 100) čiji se elementi učitavaju sa ulaza eliminiše sve brojeve koji nisu deljivi svojim indeksom. Niz reorganizovati tako da nema *rupa* koje su nastale eliminacijom elemenata i ispisati na standardni izlaz. NAPOMENA: *Nulti element niza treba zadržati jer nije dozvoljeno deljenje nulom.*

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
```

Zadatak 3.1.12
Napisati funkcije za rad sa nizovima celih brojeva.

(a) Napisati funkciju `void ucitaj(int a[], int n)` koja učitava elemente niza a dimenzije n.

(b) Napisati funkciju `void stampaj(int a[], int n)` koja štampa elemente niza a dimenzije n.

(c) Napisati funkciju `int suma(int a[], int n)` koja računa i vraća sumu elemenata niza a dimenzije n.

(d) Napisati funkciju `int prosek(int a[], int n)` koja računa i vraća prosečnu vrednost (aritmetičku sredinu) elemenata niza a dimenzije n.

(e) Napisati funkciju `int minimum(int a[], int n)` koja izračunava i vraća minimum elemenata niza a dimenzije n.

(f) Napisati funkciju `int pozicija_maksimuma(int a[], int n)` koja izračunava i vraća poziciju maksimalnog elementa u nizu a dimenzije n. U slučaju više pojavljivanja maksimalnog elementa, vratiti najmanju poziciju.

Napisati program koji testira rad zadatih funkcija. Sa standardnog ulaza učitati dimenziju niza (broj ne veći od 100).

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 5
 2 5 -2 8 11
 Ucitani niz: 2 5 -2 8 11
 Suma elemenata niza: 24
 Prosecna vrednost elemenata niza: 4.80
 Minimumalni element niza: -2
 Indeks maksimalnog elementa niza: 4
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 -5
 Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.12]

Zadatak 3.1.13
Napisati funkcije za rad sa nizovima celih brojeva.

(a) Napisati funkciju koja proverava da li niz sadrži zadatu vrednost m. Povratna vrednost funkcije je 1 ako je vrednost sadržana u nizu ili 0 ako nije.

(b) Napisati funkciju koja vraća vrednost prve pozicije na kojoj se nalazi element koji ima vrednost m ili -1 ukoliko element nije u nizu.

(c) Napisati funkciju koja vraća vrednost poslednje pozicije na kojoj se nalazi element koji ima vrednost m ili -1 ukoliko element nije u nizu.

(d) Napisati funkciju koja proverava da li elementi niza čine palindrom.

(e) Napisati funkciju koja proverava da li su elementi niza uređeni neopadajuće.

Napisati i program koji testira rad napisanih funkcija za uneti broj m i niz čija dimenzija nije veća od 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 7
 8 11 -2 14 -2 11 8
 Ucitani niz: 8 11 -2 14 -2 11 8
 Unesite jedan ceo broj:
 11
 Niz sadrzi element cija je vrednost 11.
 Niz sadrzi element cija je vrednost 11.
 Indeks njegovog prvog pojavljivanja u nizu je 1.
 Niz sadrzi element cija je vrednost 11.
 Indeks njegovog poslednjeg pojavljivanja u nizu je 5.
 Elementi niza cine palindrom.
 Niz nije sortiran neopadajuce.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 -5
 Greska: Nedozvoljena vrednost!
```

[Rešenje **??**]

Zadatak 3.1.14 Napisati funkcije za rad sa nizovima celih brojeva.

(a) Napisati funkciju koja sve vrednosti niza uvećava za zadatu vrednost m.

(b) Napisati funkciju koja obrće elemente niza.

(c) Napisati funkciju koja rotira niz ciklično za jedno mesto u levo.

(d) Napisati funkciju koja rotira niz ciklično za k mesta u levo.

Napisati i program koji testira rad napisanih funkcija za uneti broj m i niz čija dimenzija nije veća od 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 6
 7 -3 11 783 26 -19
 Unesite jedan ceo broj:
 2
 Elementi niza nakon uvecanja za 2:
 9 -1 13 785 28 -17
 Elementi niza nakon obrtanja:
 -17 28 785 13 -1 9
 Elementi niza nakon rotiranja za 1 mesto ulevo:
 28 785 13 -1 9 -17
 Unesite jedan pozitivan ceo broj:
 3
 Elementi niza nakon rotiranja za 3 mesto ulevo:
 -1 9 -17 28 785 13
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 252
 Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.14]

Zadatak 3.1.15 Napisati program koji transformiše uneti niz tako što kvadrira sve negativne elemente niza. Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

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 3.1.15]

Zadatak 3.1.16 Sa standardnog ulaza se učitava dimenzija niza, elementi niza i jedan ceo broj k. Napisati program koji štampa indekse elemenata koji su deljivi sa k. Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

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 3.1.16]

Zadatak 3.1.17 Napisati program koji učitava dimenziju i elemente niza i štampa niz u kojem su najveći i najmanji element niza razmenili mesta. Ukoliko se najmanji ili najveći element više puta pojavljuju u nizu, uzeti u obzir njihova prva pojavljivanja. Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

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: Nedozvoljena vrednost!
```

[Rešenje 3.1.17]

Zadatak 3.1.18 Napisati funkciju `int min_max(int a[], int n)` koja pronalazi indekse najmanjeg i najvećeg elementa u nizu a dimenzije n koristeći samo jedan prolaz kroz niz. Funkcija kao povratnu vrednost vraća manji od ta dva indeksa. Napisati program koji testira ovu funkciju za učitane nizove celih brojeva maksimalne dužine 100 elemenata.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza:
 7
 Unesite elemente niza:
 5 8 -4 11 17 89 1
 2
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza:
 3
 Unesite elemente niza:
 9 11 6
 1
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza:
 -45
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.19 Napisati program koji učitane karaktere (najviše njih 100, učitavaju se sve do pojave karaktera *) 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
```

Primer 3

```
Interakcija sa programom:
 Unesite karakter: U
 Unesite karakter: 4
 Unesite karakter: a
 Unesite karakter: u
 Unesite karakter: *
 u a 4 U
```

[Rešenje 3.1.19]

Zadatak 3.1.20

Sa standardnog ulaza se unosi broj elemenata niza a i njegovi elementi. Napisati program koji od datog niza formira niz b u koji ulaze elementi niza a koji se pojavljuju tačno tri puta. Pretpostaviti da će uneti niz imati najviše 100 elemenata. Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

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 3.1.20]

Zadatak 3.1.21 Napisati funkciju `int sadrzi_bar_dva(int a[], int na, int b[], int nb)` koja proverava da li niz a dužine na sadrži barem dva broja koja se pojavljuju u nizu b dužine nb. Napisati i program koji učitava redom dimenzije i elemente nizova a i b i ispisuje da li uneti nizovi ispunjavaju traženo svojstvo. Pretpostaviti da će uneti nizovi imati najviše 100 elemenata.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza a:
 5
 Unesite elemente niza a:
 5 8 7 -2 6
 Unesite broj elemenata niza b:
 6
 Unesite elemente niza b:
 11 -11 7 -7 6
 Svojstvo je ispunjeno.
```

Zadatak 3.1.22 Sa standardnog ulaza se, redom, učitavaju dimenzije i elementi dva niza, a i b. Napisati program koji određuje i ispisuje njihovu uniju, presek i razliku (redosled prikaza elemenata nije bitan). Pretpostaviti da će uneti nizovi imati najviše 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 3.1.22]

Zadatak 3.1.23 Napisati program koji za učitani niz formira i ispisuje niz koji se dobija izbacivanjem svih neparnih elemenata niza. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Pretpostaviti da će uneti niz imati najviše 100 elemenata.

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 3.1.23]

Zadatak 3.1.24 Napisati program koji za učitani niz ispisuje niz koji se dobija izbacivanjem svih elemenata koji su prosti brojevi. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Pretpostaviti da će uneti niz imati najviše 100 elemenata. NAPOMENA: *Broj* 1 *nije prost.*

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 3.1.24]

Zadatak 3.1.25 Napisati funkciju `int prebrojavanje(int a[], int n)` koja izračunava broj elemenata celobrojnog niza 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 3.1.25]

Zadatak 3.1.26 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
```

[Rešenje 3.1.26]

Zadatak 3.1.27 Napisati funkciju `int cifre(char s[], int n)` koja izračunava broj cifara u nizu karaktera a dužine n. Napisati program koji za karaktere koji se unose u zasebnim redovima ispisuje broj unetih cifara. Pretpostaviti da dužina niza neće biti veća od 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 5
 Unesite elemente niza:
 4
 +
 A
 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 3.1.27]

Zadatak 3.1.28 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 3.1.28]

Zadatak 3.1.29 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 3.1.29]

Zadatak 3.1.30 Napisati funkciju `void kvadriranje(float a[], int n)` koja kvadrira elemente realnog niza a dužine n koji se nalaze na parnim pozicijama. Napisati program koji tranformiše na ovaj način uneti niz. 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 3.1.30]

Zadatak 3.1.31 Napisati funkciju `int blizu_3(int a[],int n)` koja pronalazi i vraća indeks elementa niza koji je po vrednosti najbliži aritmetičkoj sredini onih elemenata niza koji su deljivi brojem tri. Napisati 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:
 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
```

Zadatak 3.1.32 Napisati funkcije za rad sa nizovima celih brojeva.

(a) Napisati funkciju koja izbacuje poslednji element niza.

(b) Napisati funkciju koja izbacuje prvi element niza. Zadatak rešiti na dva načina: čuvanjem redosleda elemenata i premeštanjem poslednjeg elementa niza na upražnjenu poziciju.

(c) Napisati funkciju koja izbacuje element sa date pozicije k.

(d) Napisati funkciju koja izbacuje sva pojavljivanja datog elementa x iz niza.

(e) Napisati funkciju koja ubacuje dati element x na kraj niza.

(f) Napisati funkciju koja ubacuje dati element x na početak niza.

(g) Napisati funkciju koja ubacuje dati element x na datu poziciju k.

Napisati program koji testira rad zadatih funkcija. Sa standardnog ulaza učitati dimenziju niza (broj ne veći od 100).

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 8
 2 5 -2 16 33 19 8 11
 Niz posle izbacivanja poslednjeg elementa: 2 5 -2 16 33 5 8
 Niz nakon izbacivanja prvog elementa: 5 -2 16 33 5 8
 Unesite poziciju elementa za izbacivanje:
 3
 Niz nakon izbacivanja 3. elementa: 5 -2 16 5 8
 Unesite element cije pojavljivanje treba izbaciti:
 5
 Niz nakon izbacivanja elementa 5: -2 16 8
 Unesite element koji treba ubaciti u niz:
 19
 Niz nakon ubacivanja elementa 19 na kraj: -2 16 8 19
 Niz nakon ubacivanja elementa 19 na pocetak: 19 -2 16 8 19
 Unesite poziciju na koju treba ubaciti element:
 2
 Niz nakon ubacivanja elementa 19 na poziciju 2: 19 -2 19 16 8 19
```

Zadatak 3.1.33 Napisati funkcije za rad sa nizovima celih brojeva.

(a) Napisati funkciju koja određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva.

(b) Napisati funkciju koja određuje dužinu najvećeg neopadajućeg podniza datog niza celih brojeva.

(c) Napisati funkciju koja određuje da li se jedan niz javlja kao podniz uzastopnih elemenata drugog niza.

(d) Napisati funkciju koja određuje da li se jedan niz javlja kao podniz elemenata drugog niza (elementi ne moraju da budu uzastopni, ali je redosled pojavljivanja isti).

(e) Napisati funkciju koja izbacuje višestruka pojavljivanja elemenata iz datog niza brojeva. Zadatak rešiti na dva načina: zadržavnjem prvog pojavljivanje elementa i zadržavanjem poslednjeg pojavljivanje elementa.

Zadatak 3.1.34 Napisati funkciju koja iz zadatog niza izbacuje sve elemente koji su deljivi svojim indeksom. Niz reorganizovati tako da nema *rupa* koje su

nastale izbacivanjem elemenata. Povratna vrednost funkcije je nova dimenzija niza. Napisati program koji za učitni niz (dimenzije manje od 100) ispisuje niz dobijen nakon poziva funkcije. NAPOMENA: *Element na nultoj poziciji niza zadržati jer nije dozvoljeno deljenje nulom.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 10
 Unesite elemente niza:
 4 2 1 6 7 10 8 2 16 27
 4 1 7 8 2 16
```

Zadatak 3.1.35 Za celobrojni niz a dimenzije n kažemo da je *permutacija* ako sadrži sve brojeve od 1 do n.

(a) Napisati funkciju `void brojanje(int a[], int b[], int n)` koja na osnovu celobrojnog niza a dimenzije n formira niz b tako što i-ti element niza b odgovara broju pojavljivanja vrednosti i u nizu a.

(b) Napisati funkciju `int permutacija(int a[], int n)` koja proverava da li je zadati niz permutacija. Funkcija vraća vrednost 1 ako je svojstvo ispunjeno, odnosno 0 ako nije. UPUTSTVO: *Koristiti funkciju brojanje iz tačke (a).*

Napisati program koji sa standardnog ulaza učitava dimenziju niza (broj manji od 100) i elemente niza i ispisuje da li je uneti niz permutacija ili ne.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 5
 Unesite elemente niza:
 1 5 4 3 2
 Uneti niz je permutacija.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 6
 Unesite elemente niza:
 2 3 3 1 1 5
 Uneti niz nije permutacija.
```

[Rešenje 3.1.35]

3.2 Rešenja

Rešenje 3.1.1

```
#include <stdio.h>
2
```

```c
/* Predprocesorska direktiva kojom se definise maksimalni broj
    elemenata niza */
#define MAX 100

int main()
{
  int a[MAX];
  int b[MAX];
  int n;
  int i;
  int skalarni_proizvod;


  /* Ucitava se dimenzija vektora i proverava njena ispravnost */
  printf("Unesite dimenziju vektora: ");
  scanf("%d", &n);
  if (n<1 || n>100)
  {
    printf("Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se koordinate vektora */
  printf("Unesite koordinate vektora a: ");
  for (i=0; i<n; i++)
  {
    scanf("%d", &a[i]);
  }

  printf("Unesite koordinate vektora b: ");
  for (i=0; i<n; i++)
  {
    scanf("%d", &b[i]);
  }

  /* Izracunava se skalarni proizvod po zadataj formuli */
  skalarni_proizvod=0;

  for (i=0; i<n; i++)
    skalarni_proizvod = skalarni_proizvod + a[i]*b[i];

  /* I ispisuje se njegova vrednost */
  printf("Skalarni proizvod vektora a i b: %d\n",skalarni_proizvod);

  return 0;
}
```

Rešenje 3.1.2

```c
#include <stdio.h>

```

```c
#define MAX 100

int main()
{
  int a[MAX];
  int n;
  int i;

  /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
  printf("Unesi dimenziju niza:\n");
  scanf("%d", &n);
  if (n<1 || n>MAX)
  {
    printf("Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesi elemente niza:\n");
  for (i=0; i<n; i++)
  {
    scanf("%d", &a[i]);
  }

  /* Ispisuju se elementi niza na parnim pozicijama */
  printf("Elementi niza na parnim pozicijama:\n");
  for (i=0; i<n; i+=2)
  {
    printf("%d ", a[i]);
  }
  printf("\n");

  /* Ispisuju se parni elementi niza */
  printf("Parni elementi niza:\n");
  for (i=0; i<n; i++){
    if (a[i]%2==0){
      printf("%d ", a[i]);
    }
  }
  printf("\n");


  return 0;
}
```

Rešenje 3.1.3

```c
#include<stdio.h>
#include<stdlib.h>

int main()
```

```
{
    int x;
    int brojaci[10];
    char cifra;
    int original;
    int i;

    /* Ucitava se ceo broj sa standardnog ulaza */
    printf("Unesite ceo broj:\n");
    scanf("%d",&x);

    /* Cuva se njegova originalna vrednost zbog finalnog ispisa */
    original = x;

    /* I nadalje posmatra apsolutna vrednost */
    x = abs(x);

    /* Svaki element niza brojaci predstavljace brojac za jednu od
     cifara:
       brojac[0] predstavljace broj nula u zapisu broja x
       brojac[1] predstavljace broj jedinica u zapisu broja x
       ...
       brojac[9] predstavljace broj devetki u zapisu broja x
    */


    /* Brojaci se na pocetku inicijalizuju nulama */
    for(i=0;i<10;i++){
       brojaci[i]=0;
    }

    /* Sve dok ima cifara u zapisu broja x */
    do
    {
      /* Izdvaja se krajnja desna cifara */
      cifra = x%10;

      /* Uvecava se njen broj pojavljivanja */
       brojaci[cifra]++;

       /* I prelazi se na analiziranje sledece cifre */
       x/=10;

    } while(x);

    /* Ispisuju se informacije o ciframa koje se nalaze u zapisu broja
      x */
    for(i=0; i<10; i++){
       if(brojaci[i]){
          printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n
     ", original, i, brojaci[i]);
       }
```

```
    }
55
57  return 0;
}
```

Rešenje 3.1.4

```
#include <stdio.h>

#define BROJ_CIFARA 10

int main()
{
  char c;
  int cifrex[BROJ_CIFARA], cifrey[BROJ_CIFARA];
  int x, y, i, indikator;

  /* Ucitavaju se brojevi x i y */
  printf("Unesite dva broja: ");
  scanf("%d%d", &x, &y);

  /* Za slucaj da su unete vrednosti negativne, posmatra se njihova
     apsolutna vrednost.
   Ovo je opravdano iz razloga sto se brojevi x i -x zapisuju istim
     ciframa. */
  x=abs(x);
  y=abs(y);

  /* Niz cifrex predstavlja brojace za cifre broja x.
     Niz cifrey predstavlja brojace za cifre broja y.
     Na pocetku se ovi nizovi inicijalizuju nulama. */
  for(i=0;i<BROJ_CIFARA;i++)
  {
      cifrex[i] = 0;
      cifrey[i] = 0;
  }

  /* Analiziraju se cifre broja x */
  while(x)
  {
    c = x%10;
    cifrex[c]++;
    x /= 10;
  }

  /* Analiziraju se cifre broja y */
  while(y)
  {
    c = y%10;
    cifrey[c]++;
```

```
    y /= 10;
  }

  /* Promenljiva indikator sluzi za pracenje da li su oba broja
     sastavljena od istih cifara. */
  indikator = 1;

  for(i=0;i<BROJ_CIFARA;i++){
    /* Ako se broj pojavljivanja cifre i u zapisu broja x razlikuje
    od broja pojavljivanja cifre i u
     zapisu broja y, brojevi se ne zapisuju istim ciframa. Zato se
    vrednost indikatora moze postaviti na
     0 i prekinuti dalje uporedjivanje broja pojavljivanja. */
    if(cifrey[i] != cifrex[i])
    {
     indikator = 0;
     break;
    }
  }

    /* Ako je vrednost promenljive indikator ostala 1, to znaci da u
    petlji nije pronadjena cifra
    koja se ne pojavljuje isti broj puta u zapisima brojeva x i y.
    Zato se moze zakljuciti da se brojevi
    zapisuju istim ciframa. */
  if(indikator)
    printf("Brojevi se zapisuju istim ciframa!\n");
  else
    printf("Brojevi se ne zapisuju istim ciframa!\n");


  return 0;
}
```

Rešenje 3.1.5

```
#include <stdio.h>

int main()
{
  /* Niz u kojem ce se cuvati informacije o broju pojavljivanja
     cifara */
  int cifre[10];

  /* Niz u kojem ce se cuvati informacije o broju pojavljivanja malih
     slova */
  int mala_slova[26];

  /* Niz u kojem ce se cuvati informacije o broju pojavljivanja
     velikih slova */
  int velika_slova[26];
```

```
14  int c, i;

16  /* Brojaci karaktera se na pocetku inicijalizuju nulama */
    for(i=0;i<10;i++){
18    cifre[i]=0;
    }
20
    for(i=0;i<26;i++)
22  {
      mala_slova[i]=0;
24    velika_slova[i]=0;
    }
26
    /* Ucitavaju se karakteri sve do kraja ulaza */
28  while((c = getchar()) != EOF)
    {
30    /* Ako je procitani karakter veliko slovo ... */
      if (c>='A' && c<='Z'){
32      velika_slova[c-'A']++;
      }
34    else{
        /* Ako je procitani karakter malo slovo ... */
36      if (c>='a' && c<='z'){
          mala_slova[c-'a']++;
38      }
        else{
40        /* Ako je procitani karakter cifra ... */
          if(c >='0' && c <= '9'){
42          cifre[c-'0']++;
          }
44      }
      }
46  }

48  /* Ispisuju se trazene informacije */
    for(i = 0; i < 10; i++){
50    if (cifre[i]!=0)
        printf("Karakter %c se pojavljuje %d puta\n", '0' + i, cifre[i
      ]);
52  }

54   for(i = 0; i < 26; i++){
      if (mala_slova[i]!=0)
56      printf("Karakter %c se pojavljuje %d puta\n", 'a' + i,
     mala_slova[i]);
     }
58
      for(i = 0; i < 26; i++){
60      if (velika_slova[i]!=0)
          printf("Karakter %c se pojavljuje %d puta\n", 'A' + i,
     velika_slova[i]);
```

```
62      }
64
    return 0;
66  }
```

Rešenje 3.1.6

Rešenje 3.1.7

```
#include <stdio.h>
2
#define MAX 100
4
int main()
6  {
    int a[MAX];
8   int b[MAX];

10  /* Rezultujuci niz ima najvise 2*MAX elemenata */
    int c[2*MAX];
12
    int n;
14  int i,j;


16
    /* Ucitava se dimenzija nizova i proverava njena ispravnost */
18  printf("Unesite dimenziju nizova:\n");
    scanf("%d", &n);
20  if (n<1 || n>MAX)
    {
22    printf("Nedozvoljena vrednost!\n");
      return -1;
24  }

26  /* Ucitavaju se elementi prvog niza */
    printf("Unesite elemente niza a:\n");
28  for(i=0;i<n;i++)
    {
30    scanf("%d", &a[i]);
    }
32
    /* Ucitavaju se elementi drugog niza */
34  printf("Unesite elemente niza b:\n");
    for(i=0;i<n;i++)
36  {
      scanf("%d", &b[i]);
38  }

40  /*
```

```
    Formira se treci niz.
    Koriste se dva indeksa:

      - indeks i pomocu kojeg se pristupa elementima nizova a i b i
      koji treba uvecati za 1 nakon svake iteracije

      - indeks j pomocu kojeg se pristupa elementima rezultujuceg
    niza c;
      s obzirom da se u svakoj iteraciji u niz c smestaju dva
    elementa, jedan iz niza a i jedan iz niza b,
      indeks j se uvecava za 2 nakon svake iteracije
  */
  for(i=0,j=0;i<n;i++,j+=2)
  {
    c[j]=a[i];
    c[j+1]=b[i];
  }

  /* Ispisuju se elementi rezultujuceg niza */
  printf("Rezultujuci niz:\n");
  for(i=0;i<2*n;i++)
     printf("%d ",c[i]);
  printf("\n");

  return 0;
}
```

Rešenje 3.1.8

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX], b[MAX], c[2*MAX];
  int i, n;

  /* Ucitava se broj elemenata nizova i proverava se njegova
     ispravnost */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<1 || n>MAX)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi nizova */
  printf("Unesite elemente niza a: ");
  for(i=0;i<n;i++)
```

```
    scanf("%d", &a[i]);

  printf("Unesite elemente niza b: ");
  for(i=0;i<n;i++)
    scanf("%d", &b[i]);

  /*
    Niz c ima 2*n elemenata: prvih n elemenata su elementi niza b, a
    narednih n elemenata elementi niza a.
    Elementi niza b se nalaze na pozicijama 0,1,2,...n-1, a elementi
    niza a na pozicijama
    n,n+1,...2*n-1. Jednim prolaskom kroz petlju na poziciju i u nizu
     c se postavlja element b[i] niza b,
    a na poziciju n+i element a[i] niza a.
  */
  for(i=0;i<n;i++)
  {
    c[i] = b[i];
    c[n+i] = a[i];
  }

  /* Ispisuju se elementi niza c */
  for(i=0;i<2*n;i++)
    printf("%d ", c[i]);
  printf("\n");

  return 0;
}
```

Rešenje 3.1.9

Rešenje 3.1.10

```
#include <stdio.h>

#define MAX 100

int main()
{

  int a[MAX];
  int n;
  int i,j;
  char poslednja_cifra;
  int novo_n;

  /* Ucitava se dimenzija niza i proverava njena ispravnost */
  printf("Unesite dimenziju niza:\n");
  scanf("%d", &n);
  if (n<1 || n>MAX)
```

```
18    {
        printf("Nedozvoljena vrednost!\n");
20      return -1;
      }
22
      /* Ucitavaju se elementi niza a */
24    printf("Unesite elemente niza a:\n");
      for(i=0;i<n;i++)
26    {
        scanf("%d", &a[i]);
28    }


30
      /* Obilaze se svi elementi niza a */
32    for(i=0, j=0; i<n; i++)
      {
34      /* Izdvaja se poslednja cifra tekuceg elementa */
        poslednja_cifra = a[i]%10;
36
        /* Ako je poslednja cifra 0 ili je element deljiv svojom
        poslednjom cifrom,
38         zadrzavamo ga i smestamo na poziciju j */
        if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
40      {
            a[j]=a[i];
42          j++;
        }
44    }

46    /* Dimenzija novog niza odgovara posledjoj vrednosti brojaca j */
      novo_n=j;
48
      /* Ispisuje se rezultujuci niz */
50    printf("Niz a nakon izmena:\n");
      for(i=0; i<novo_n;i++)
52      printf("%d ", a[i]);
      printf("\n");
54


56    return 0;
}
```

Rešenje 3.1.11

Rešenje 3.1.12

```
#include <stdio.h>
2

#define MAX 100
4
```

```c
/* a) Napisati funkciju koja ucitava elemente niza. */
void ucitaj(int a[], int n)
{
   int i;
   for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }
}

/* b) Napisati funkciju koja stampa elemente niza. */
void stampaj(int a[], int n)
{
   int i;
   for(i=0;i<n;i++)
      printf("%d ",a[i]);
   printf("\n");
}

/* c) Napisati funkciju koja racuna sumu elemenata niza. */
int suma(int a[], int n)
{
   int i;
   int s=0;
   for(i=0;i<n;i++)
      s+=a[i];
   return s;
}


/* d) Napisati funkciju koja racuna prosecnu vrednost elemenata niza.
     */
float prosek(int a[], int n)
{
   int i;
   int s = suma(a,n);
   return (float) s/n;
}


/* e) Napisati funkciju koja izracunava minimum elemenata niza.*/
int minimum (int a[],int n)
{
   int m;
   int i;

   /* Minimum inicijalizujemo prvim elementom niza (a[0]), a zatim
    prolazimo kroz ostatak niza.
      U svakom koraku poredimo vrednost minimuma sa tekucim elementom
     niza. */
   m = a[0];
   for(i=1;i<n;i++)
```

```
54        if (a[i] < m)
             m = a[i];
56
      /* Vraca se izracunata vrednost minimuma */
58    return m;
   }
60

62 /* f) Napisati funkciju koja izracunava poziciju maksimalnog elementa
        u nizu. */
   int pozicija_maksimuma (int a[],int n)
64 {
      int m;
66    int m_pozicija;
      int i;
68
      /* Maksimum inicijalizujemo prvim elementom niza (a[0]), a zatim
       prolazimo kroz ostatak niza.
70       U svakom koraku poredimo vrednost maksimuma sa tekucim
       elementom niza. */

72    m = a[0];
      m_pozicija=0;
74    for(i=1;i<n;i++)
         if (a[i] > m)
76       {
            m = a[i];
78          m_pozicija=i;
         }
80
      /* Vraca se izracunata pozicija */
82    return m_pozicija;
   }
84


86
   int main()
88 {
      int a[MAX];
90    int n;

92    /* Ucitava se dimenzija niza i proverava njena ispravnost */
      printf("Unesite dimenziju niza:");
94    scanf("%d",&n);
      if (n<1 || n>MAX)
96    {
         printf("Nedozvoljena vrednost!\n");
98       return -1;
      }
100


102   /* Testira se funkcija kojom se ucitavaju elementi niza */
```

```
    ucitaj(a,n);

    /* Testira se funkcija kojom se ispisuju elementi niza */
    printf("Ucitani niz: ");
    stampaj(a,n);

    /* Testira se funkcija kojom se izracunava suma elemenata niza */
    printf("Suma elemenata niza: %d\n", suma(a,n));

    /* Testira se funkcija kojom se racuna prosek elemenata niza */
    printf("Prosecna vrednost elemenata niza: %.2f\n", prosek(a,n));

    /* Testira se funkcija kojom se izracunava minimum niza */
    printf("Minimumalni element niza: %d\n", minimum(a,n));

    /* Testira se funkcija kojom se izracunava pozicija maksimalnog
     elementa */
    printf("Indeks maksimalnog elementa niza: %d\n",
     pozicija_maksimuma(a,n));

    return 0;
}
```

Rešenje 3.1.13

```
#include <stdio.h>

#define MAX 100

/* Funkcija koja ucitava elemente niza */
void ucitaj(int a[], int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        scanf("%d",&a[i]);
    }
}

/* Funkcija koja ispisuje elemente niza */
void stampaj(int a[], int n)
{
    int i;
    for(i=0;i<n;i++)
        printf("%d ",a[i]);
    printf("\n");
}


/* a) Funkcija koja proverava da li niz sadrzi zadatu vrednost m */
int sadrzi(int a[], int n, int m)
```

```
{
   int i;

   /* Poredi se element po element niza a sa zadatim brojem m */

   for(i=0;i<n;i++){
     /* Ukoliko je tekuci element niza jednak trazenom broju */
     if (a[i]==m){
        /* Funkcija vraca vrednost 1 */
        return 1;
     }
   }

   /* Ako se stigne do kraja niza i ne naidje na vrednost koja je
    jednaka broju m,
     to znaci da se broj ne nalazi u nizu i da funkcija treba da
    vrati 0. */
   return 0;
}

/* b) Funkcija koja vraca vrednost prve pozicije na kojoj se nalazi
    element koji ima vrednost m,
  ili -1 ukoliko element nije u nizu */
int prvo_pojavljivanje(int a[], int n, int m)
{
   int i;

   /* Poredi se element po element niza a sa zadatim brojem m */

   for(i=0;i<n;i++){
    /* Ukoliko je tekuci element niza jednak trazenom broju */
     if (a[i]==m){
        /* Vraca se njegov indeks */
        return i;
     }
   }

   /* Ako se stigne do kraja niza i ne naidje na vrednost koja je
    jednaka broju m,
     to znaci da se broj ne nalazi u nizu i da funkcija treba da
    vrati -1. */
   return -1;
}

/* c) Funkcija koja vraca vrednost poslednje pozicije na kojoj se
    nalazi element koji ima vrednost m,
 ili -1 ukoliko element nije u nizu */
int poslednje_pojavljivanje(int a[], int n, int m)
{
   int i;

   /* Polazi se od kraja niza i poredi se element po element sa
```

```
    zadatim brojem m*/

   for(i=n-1;i>=0;i--){
     /* Ukoliko je tekuci element niza jednak trazenom broju */
      if (a[i]==m){
       /* Vraca se njegov indeks */
        return i;
      }
   }

   /* Ako se stigne do pocetka niza i ne naidje na vrednost koja je
    jednaka broju m,
     to znaci da se broj ne nalazi u nizu i da funkcija treba da
    vrati -1. */
   return -1;
}

/* d) Funkcija koja proverava da li elementi niza cine palindrom */
int palindrom(int a[], int n)
{

   int i,j;

   /*
    Uporedjuje se element na poziciji 0 sa elementom na poziciji n-1
    Uporedjuje se element na poziciji 1 sa elementom na poziciji n-2
    Uporedjuje se element na poziciji 2 sa elementom na poziciji n-3
    .
    .
    i tako redom dok je pozicija prvog elementa manja od pozicije
    drugog elementa
   */

   for(i=0,j=n-1;i<j;i++,j--){
     /* Ako element na poziciji i nije jednak odgovarajucem elementu
    na poziciji j */
      if(a[i]!=a[j]){
        /* Moze se odmah zakljuciti da niz nije palindrom */
         return 0;
      }
   }

   /* AKo su svi parovi elemenata jednaki, niz je palindrom i
    funkcija vraca vrednost 1 */
   return 1;
}

/* e) Funkcija koja proverava da li su elementi niza uredjeni
    neopadajuce */
int neopadajuci(int a[], int n)
{
   int i;
```

```
118
    /* Ako je niz uredjen neopadajuce vaze nejednakosti:
120    a[0]<=a[1], a[1]<=a[2], ..., a[i]<=a[i+1], ..., a[n-2]<=a[n-1]
       Zato je dovoljno proveriti da li za parove susednih elemenata
     vazi ovi svojstvo.
122 */
    for(i=0; i<n-1; i++)
124    if (a[i]>a[i+1])
          return 0;
126
    return 1;
128 }

130
/* f) Funkcija koja izracunava najduzu uzastopnu seriju jednakih
132 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. */
134 int najduza_serija_jednakih(int a[], int n)
    {
136    int i;
       int j;
138    int duzina;
       int max_duzina=0;
140

142    for(i=0,j=0;i<n-1;i++)
       {
144       if(a[i]==a[i+1])
          {
146          j++;

148          /*
               ako se niz zavrsava konstantnim intervalom (nalazimo se u
     poslednjoj
150            iteraciji petlje i tada je i==n-2), ispitujemo da li je
     taj konstantni
               interval maksimalne duzine
152          */
             if(i==n-2)
154          {
                j++;
156             if(j>max_duzina)
                   max_duzina=j;
158          }
          }
160       else
          {
162          /*
                izasli smo iz konstantnog intervala
164

                ukoliko smo imali bar dva elementa u konstantnom
```

```
    intervalu,
166             vrednost promenljive j ce biti 1, a duzina tog intervala
    je 2;
                zbog toga je neophodno takve (pozitivne) j uvecati za 1;
168
                sa druge strane, ako su a[i] i a[i+1] razliciti,
170             duzina tog intervala je 0
            */
172
            if (j>0)
174             j++;

176         /* azuriramo maksimalnu duzinu uspona */
            if(j>max_duzina)
178             max_duzina=j;
            /*
180              duzina uspona se postavlja na nulu
                 kako bi mogli da je iskoristimo
182              za naredni uspon
            */
184         j=0;

186       }


188
    }
190
    return max_duzina;
192 }


194
int main()
196 {
    int a[MAX];
198 int n;
    int m;
200 int i;

202 /* Ucitava se dimenzija niza i proverava se njena ispravnost */
    printf("Unesite dimenziju niza:");
204 scanf("%d",&n);
    if (n<1 || n>MAX)
206 {
        printf("Nedozvoljena vrednost!\n");
208     return -1;
    }
210
    /* Ucitavaju se i ispisuju elementi niza */
212 ucitaj(a,n);
    printf("Ucitani niz:");
214 stampaj(a,n);
```

```
    /* Ucitava se vrednost za pretragu */
    printf("Unesi jedan ceo broj:");
    scanf("%d",&m);

    /* I proverava se rad napisanih funkcija */
    if(sadrzi(a,n,m))
       printf("Niz sadrzi element cija je vrednost %d\n", m);
    else
       printf("Niz ne sadrzi element cija je vrednost %d\n", m);

    i = prvo_pojavljivanje(a,n,m);
    if(i!=-1)
       printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
         prvog pojavljivanja u nizu je %d\n", m,i);
    else
       printf("Niz ne sadrzi element cija je vrednost %d\n", m);


    i = poslednje_pojavljivanje(a,n,m);
    if(i!=-1)
       printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
         poslednjeg pojavljivanja u nizu je %d\n", m,i);
    else
       printf("Niz ne sadrzi element cija je vrednost %d\n", m);

    if(palindrom(a,n))
       printf("Elementi niza cine palindrom\n");
    else
       printf("Elementi niza ne cine palindrom\n");

    if(neopadajuci(a,n))
       printf("Niz je sortiran neopadajuce\n");
    else
       printf("Niz nije sortiran neopadajuce\n");

    printf("Duzina najduzeg konstantnog intervala: %d\n",
     najduza_konstanta(a,n));


    return 0;
}
```

Rešenje 3.1.14

```
#include<stdio.h>

#define MAX 100

/* Funkcija kojom se ucitavaju elementi niza a dimenzije n */
void ucitaj(int a[], int n)
{
```

```
8     int i;
      for(i=0;i<n;i++)
10    {
            scanf("%d",&a[i]);
12    }
}
14
/* Funkcija kojom se ispisuju elementi niza a dimenzije n */
16 void stampaj(int a[], int n)
{
18    int i;
      for(i=0;i<n;i++)
20          printf("%d ",a[i]);
      printf("\n");
22 }

24 /* a) Funkcija koja sve vrednosti niza uvecava za zadatu vrednost m
       */
   void uvecaj(int a[], int n, int m)
26 {
      int i;
28    for(i=0;i<n;i++)
          a[i]+=m;
30 }

32 /* b) Funkcija koja obrce elemente niza */
   void obrni(int a[], int n)
34 {

36    int t;
      int i,j;
38

40    /*
       Za niz a[0], a[1], ...., a[n-2], a[n-1] obrnuti niz je a[n-1], a[
       n-2], ...., a[1], a[0]
42     Zato je potrebno razmeniti vrednosti elemenata a[0] i a[n-1], a
       [1] i a[n-2], itd. i zaustaviti se
       kada je vrednost indeksa prvog elementa veca od vrednosti drugog
       elementa.
44    */

46    for(i=0,j=n-1;i<j;i++, j--)
      {
48          t = a[i];
            a[i] = a[j];
50          a[j] = t;
      }
52

}
54

/* c) Funkcija koja rotira niz ciklicno za jedno mesto u levo */
```

```
56 void rotiraj1(int a[], int n)
   {
58    int i;
      int tmp;
60
      /* Izdvaja se prvi element niza */
62    tmp=a[0];

64    /* Pomeraju se preostali elementi niza */
      for(i=0;i<n-1;i++){
66         a[i]=a[i+1];
      }
68
      /* Poslednjem elementu se dodeljuje sacuvana vrednost prvog
       elementa */
70    a[n-1] = tmp;
   }
72
   /* d) Funkcija koja rotira niz ciklicno za k mesta u levo */
74 void rotirajk(int a[], int n, int k)
   {
76    int i;

78    /* Odredjuje se vrednost broja k koja je u opsegu od 0 do n-1 kako
        bi se izbegla suvisna pomeranja */
      k=k%n;
80
      /* Niz se rotira za jednu poziciju ulevo k puta */
82    for(i=0;i<k;i++)
         rotiraj1(a,n);
84 }

86 int main()
   {
88   int a[MAX];
     int n;
90   int i;
     int k;
92   int m;


94
     /* Ucitava se dimenzija niza i proverava se njena ispravnost */
96   printf("Unesite dimenziju niza:");
     scanf("%d",&n);
98   if (n<1 || n>MAX)
     {
100         printf("Nedozvoljena vrednost!\n");
            return -1;
102  }

104  /* Ucitavaju se elementi niza */
     ucitaj(a,n);
```

```
106
    /* Testira se rad napisanih funkcija */
108
    /* a) */
110 printf("Unesite jedan ceo broj:");
    scanf("%d", &m);
112 printf("Elementi niza nakon uvecanja za %d:\n",m);
    uvecaj(a,n,m);
114 stampaj(a,n);

116 /* b) */
    printf("Elementi niza nakon obrtanja:\n");
118 obrni(a,n);
    stampaj(a,n);
120
    /* c) */
122 printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
    rotiraj1(a,n);
124 stampaj(a,n);

126 /* d) */
    printf("Unesite jedan pozitivan ceo broj:");
128 scanf("%d",&k);
    if (k<=0)
130 {
        printf("Nekorektan unos\n");
132     return -1;
    }
134 rotirajk(a,n,k);
    printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n",k);
136 stampaj(a,n);


138
    return 0;
140 }
```

Rešenje 3.1.15

```
1 #include <stdio.h>

3 #define MAX 100

5 int main()
  {
7   float brojevi[MAX];
    int n, i;
9
    /* Ucitava se dimenzija niza i proverava se njena ispravnost */
11  printf("Unesite broj elemenata niza: ");
    scanf("%d", &n);
13  if(n<1 || n>MAX)
```

```
  {
    printf("Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza:\n");
  for(i=0;i<n;i++){
    scanf("%f", &brojevi[i]);
  }

  /* Ukoliko je i-ti element niza brojevi[i] negativan broj,
  kvadriramo ga tako sto ga pomnozimo sa samim sobom. */
  for(i=0;i<n;i++){
    if(brojevi[i]<0)
      brojevi[i] *= brojevi[i];
  }

  /* Ispisuje se novodobijeni niz */
  for(i=0;i<n;i++){
    printf("%g ", brojevi[i]);
  }
  printf("\n");

  return 0;
}
```

Rešenje 3.1.16

```
#include <stdio.h>

#define MAX 100

int main()
{
  int brojevi[MAX];
  int n, i, k, indikator;

  /* Ucitava se dimenzija niza i proverava se njena ispravnost */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if(n<1 || n>MAX)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza: ");
  for(i=0;i<n;i++)
    scanf("%d", &brojevi[i]);
```

```
23
25    /* Ucitava se broj k i proverava se njegova ispravnost */
      printf("Unesite broj k: ");
27    scanf("%d", &k);
      if(k == 0)
29    {
        printf("Greska: Pogresan unos!\n");
31      return -1;
      }
33
      /*
35       Promenljiva koja cuva informaciju o tome da li je u nizu
        postojao element koji je deljiv brojem k.
         Inicijalna vrednost je 0.
37    */

39    indikator = 0;

41    /*
        Ukoliko je element niza deljiv brojem k, indikator se postavlja
        na 1
43      i ispisuje se indeks tog elementa.
      */
45
      for(i=0;i<n;i++){
47      if(brojevi[i]%k == 0)
          {
49          indikator = 1;
            printf("%d ",i);
51        }
      }
53
      /* Ukoliko je indikator jednak nuli to znaci da ne postoji element
        u nizu koji je deljiv brojem k. */
55
      if(indikator == 0){
57      printf("U nizu nema elemenata koji su deljivi brojem %d!\n",k);
      }
59

61    return 0;
  }
```

Rešenje 3.1.17

```
1   #include <stdio.h>

3   #define MAX 100

5   int main()
```

```
{
  int brojevi[MAX];
  int n, i, poz_max, poz_min, max, min, tmp;

  /* Ucitava se dimenzija niza i proverava se njena ispravnost */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if(n<1 || n>MAX)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza:\n");
  for(i=0;i<n;i++)
    scanf("%d", &brojevi[i]);

  /*
    Maksimalnim tj. minimalnim elementom niza proglasava se nulti
    element niza.
    Pozicije maksimalnog tj. minimalnog elementa se postavljaju na 0.
  */
  max = brojevi[0];
  min = brojevi[0];
  poz_max = 0;
  poz_min = 0;

  /* U prolazu kroz niz trazi se maksimalni i minimalni element i
    pamte se njihove pozicije */
  for(i=1;i<n;i++)
  {
    if(brojevi[i] > max)
    {
      max = brojevi[i];
      poz_max = i;
    }

    if(brojevi[i] < min)
    {
      min = brojevi[i];
      poz_min = i;
    }
  }

  /* Zamenjuju se elementi na pozicijama poz_min i poz_max */
  tmp = max;
  brojevi[poz_max] = min;
  brojevi[poz_min] = tmp;

  /* Ispisuje se rezultujuci niz */
  for(i=0;i<n;i++)
```

```
    printf("%d ", brojevi[i]);
  printf("\n");


  return 0;
}
```

Rešenje 3.1.18

Rešenje 3.1.19

```
#include <stdio.h>

#define MAX 100

int main()
{
  /* niz karaktera */
  char karakteri[MAX];
  char c;
  int i, n;


  for(i=0;i<MAX;i++)
  {
    /* Ucitava se karakter po karakter sa standardnog ulaza sve dok
    se ne unese * ili
     se ne prekoraci maksimalni broj karaktera */
    printf("Unesite karakter: ");
    scanf("%c", &c);

    /* Cita se znak za novi red nakon unesenog karaktera */
    getchar();

    /* Ukoliko je unet karakter * prekida se dalje citanje i izlazi
    se iz petlje */
    if(c == '*')
      break;

    /* Inace, procitani karakter se smesta u niz */
    karakteri[i] = c;
  }

  /* Broj unetih karaktera je nakon izlaska iz petlje i-1 */
  n = i-1;

  /* Ispisuju se karakteri u obrnutom redosledu */
  for(i=n;i>=0;i--)
  {
    printf("%c ", karakteri[i]);
```

```
  }
  printf("\n");


  return 0;
}
```

Rešenje 3.1.20

```
#include <stdio.h>

#define MAX 100

/* Funkcija koja vraca broj pojavljivanja broja x u nizu */
int broj_pojavljivanja(int niz[], int n, int x)
{
  int i;

  /* Broj pojavljivanja broja x */
  int brojac = 0;

  /* Obilazi se element po element niza */
  for(i=0;i<n;i++){
    /* Ukoliko je tekuci element jednak trazenom broju */
    if(niz[i] == x){
      /* Uvecava se broj pojavljivanja */
      brojac++;
    }
  }

  /* Vraca se izracunata vrednost */
  return rezultat;
}

int main()
{
  /* Niz elemenata koje zadaje korisnik */
  int a[MAX];

  /* Niz elemenata koji se pojavljuju tri puta */
  int b[MAX];

  int i, j, n, n_b;

  /* Ucitava se broj elemenata korisnickog niza i proverava se
    njegova ispravnost */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<1 || n>MAX)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
```

```
    return -1;
  }

  /* Ucitavaju se elementi korisnickog niza */
  printf("Unesite elemente niza a: ");
  for(i=0;i<n;i++)
    scanf("%d", &a[i]);

  /* j - brojac elemenata rezultujuceg niza b */
  j = 0;

  /* Obilazi se element po element niza a */
  for(i=0;i<n;i++)
  {
    /* Ukoliko se tekuci element pojavljuje tacno tri puta u nizu a i
     nije upisan u niz b
     koji trenutno ima j elemenata, dodaje se u niz b na poziciju j i
     uvecava se broj elemenata niza b */
    if(broj_pojavljivanja(a, n, a[i])==3 && broj_pojavljivanja(b, j,
    a[i])==0)
    {
      b[j] = a[i];
      j++;
    }
  }

  /* Ispisuje se rezultujuci niz b - broj elemenata u nizu b je j*/
  n_b = j;
  for(i=0;i<n_b;i++)
    printf("%d ", b[i]);
  printf("\n");

  return 0;
}
```

Rešenje 3.1.21

Rešenje 3.1.22

```
#include <stdio.h>

#define MAX 100

/*
  Funkcija koja vraca 1 ukoliko broj x postoji u nizu, 0 inace.
*/

int postoji(int niz[], int n, int x)
{
  int i;
```

```
12
   for(i=0;i<n;i++)
14   if(niz[i] == x)
       return 1;
16
   return 0;
18 }

20 int main()
   {
22   int a[MAX], b[MAX], unija[2*MAX], presek[MAX], razlika[MAX];
     int i, j, n_a, n_b, n_u, n_p, n_r, indikator;
24
     printf("Unesite broj elemenata niza a: ");
26   scanf("%d", &n_a);

28   if(n_a<1 || n_a>100)
     {
30     printf("Greska: pogresan unos!\n");
       return -1;
32   }

34   printf("Unesite elemente niza a: ");
     for(i=0;i<n_a;i++)
36     scanf("%d", &a[i]);

38   printf("Unesite broj elemenata niza b: ");
     scanf("%d", &n_b);
40
     if(n_b<1 || n_b>100)
42   {
       printf("Greska: pogresan unos!\n");
44     return -1;
     }
46
     printf("Unesite elemente niza b: ");
48   for(i=0;i<n_b;i++)
       scanf("%d", &b[i]);
50
     /*
52     Brojaci elemenata u nizovima unija, presek i razlika.
     */
54   n_u = 0;
     n_p = 0;
56   n_r = 0;

58   for(i=0;i<n_a;i++)
     {
60     /*
         Ukoliko se element a[i] ne nalazi u uniji, dodajemo ga u uniju
       i povecamo brojac elemenata u nizu unija.
62     */
```

```
    if(postoji(unija,n_u,a[i]) == 0)
64  {
      unija[n_u] = a[i];
66    n_u++;
    }
68
    /*
70    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  }

98printf("Unija: ");
  for(i=0;i<n_u;i++)
100 printf("%d ", unija[i]);

102printf("\nPresek: ");
  for(i=0;i<n_p;i++)
104 printf("%d ", presek[i]);

106printf("\nRazlika: ");
  for(i=0;i<n_r;i++)
108 printf("%d ", razlika[i]);

110return 0;
```

```
}
```

Rešenje 3.1.23

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX];
  int i, j, n_a;

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n_a);
  if(n_a<1 || n_a>100)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza: ");
  for(i=0;i<n_a;i++)
    scanf("%d", &a[i]);


  /*
    1. nacin

    int b[MAX], n_b;

    // Brojac j predstavlja poziciju u nizu b na koju treba smestiti
    element niza a. Njegova pocetna vrednost je 0.

  for(i=0, j=0;i<n_a;i++){
    // Ako je tekuci element niza a paran
    if(a[i]%2 == 0)
    {
      // Smesta se na poziciju j u nizu b
      b[j] = a[i];
      // Vrednost brojaca j se priprema za narednu iteraciju
      j++;
    }

    // Ako je element niza a neparan, sa njim nista ne treba raditi
  }

  // Broj elemenata novodobijenog niza b je j
```

```
46    n_b = j;
48    // Ispisuju se elementi niza b
      for(i=0;i<n_b;i++)
50      printf("%d ", b[i]);
52    */
54    /*
       2. nacin
56
       J predstavlja brojac prve slobodne pozicije na koju se moze
        upisati element niza koji treba da ostane u nizu.
58     Kada se naidje na element koji je paran, on se kopira na mesto a[j
        ] i poveca se vrednost brojaca j.
       Ukoliko se naidje na element koji je neparan, njega treba
        preskociti.
60    */
      for(i=0, j=0;i<n_a;i++)
62    {
        /* Ako je tekuci element niza a paran */
64      if(a[i]%2 == 0)
        {
66        /* Premesta se na poziciju j */
          a[j] = a[i];
68
          /* Vrednost brojaca j se priprema za narednu iteraciju */
70        j++;
        }
72
        /* Ako je tekuci element niza a neparan, sa njim nista ne treba
        raditi */
74    }
76    /* U nizu a se sada na pozicijama od 0,...,j-1 nalaze elementi koji
         su parni, te je njegova nova dimenzija j. */
      n_a=j;
78
      /* Ispisuju se elementi modifikovanog niza a */
80    for(i=0;i<n_a;i++){
        printf("%d ", a[i]);
82    }
      printf("\n");
84
      return 0;
86  }
```

Rešenje 3.1.24

```
#include <stdio.h>
2 #include <math.h>
```

```
4  #define MAX 100

6  /*
     Funkcija koja proverava da li je zadati broj prost broj.
8    Povratna vrednost funkcije je 1 ukoliko broj jeste prost, inace je
       0.
   */
10 int prost(int x)
   {
12   int i;

14   /* Posmatra se apsolutna vrednost broja kako bi se pokrio i slucaj
       negativnih brojeva */
     x=abs(x);
16
     /* Brojevi 1, 2 i 3 su prosti */
18   if(x == 1 || x == 2 || x == 3)
       return 1;
20
     /* Ako je broj paran nije prost */
22   if(x%2 == 0)
       return 0;
24
     /* Ako broj ima delioce u skupu [3, koren_broja(x)] takodje nije
       prost */
26   for(i=3;i<=sqrt(x);i+=2){
       if(x%i == 0)
28       return 0;
     }
30
     /* Ako su svi uslovi ispunjeni, broj je prost */
32   return 1;
   }
34
   int main()
36 {
     int a[MAX];
38   int i, j, n_a, n_b;

40   /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
        */
     printf("Unesite broj elemenata niza: ");
42   scanf("%d", &n_a);
     if(n_a<1 || n_a>MAX)
44   {
       printf("Greska: Nedozvoljena vrednost!\n");
46     return -1;
     }
48
     /* Ucitavaju se elementi niza a */
50   printf("Unesite elemente niza: ");
```

```
  for(i=0;i<n_a;i++)
52  scanf("%d", &a[i]);


54

  /*
56  1. nacin

58  int b[MAX];


60

  for(i=0, j=0;i<n_a;i++){
62  if(prost(a[i]) == 0)
    {
64    b[j] = a[i];
      j++;
66  }
  }
68

  // Broj elemenata novodobijenog niza b je j
70  n_b = j;

72  // Ispisuju se elementi niza b
  for(i=0;i<n_b;i++)
74  printf("%d ", b[i]);
  printf("\n");
76

  */
78

  /*
80  2. nacin
  */
82

  for(i=0, j=0; i<n_a; i++)
84  {
    if(prost(a[i]) == 0)
86  {
      a[j] = a[i];
88    j++;
    }
90  }

92  n_a = j;

94  /* Ispisuju se elementi modifikovanog niza a */
  for(i=0;i<n_a;i++)
96  printf("%d ", a[i]);
  printf("\n");
98

  return 0;
100 }
```

Rešenje 3.1.25

```c
#include <stdio.h>

#define MAX 100

/* Funkcija prebrojavanje vraca broj elemenata niza koji su manji od
    poslednjeg elementa*/
int prebrojavanje(int a[], int n)
{
  int i;

  /* Brojac elemenata koji su manji od poslednjeg */
  int broj_manjih=0;

  /* Obilazi se element po element niza */
  for(i=0;i<n-1;i++){
    /* Ako je tekuci element manji od poslednjeg (on se nalazi na
    poziciji n-1) */
    if(a[i]<a[n-1]){
      /* Uvecava se brojac */
      broj_manjih++;
  }
  }

  /* Vraca se izracunata vrednost */
  return broj_manjih;
}

int main()
{
  int a[MAX];
  int n;
  int i;

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     */
  printf("Unesite broj elemenata niza:");
  scanf("%d", &n);
  if(n<=0 || n>MAX)
  {
     printf("Greska: Nedozvoljena vrednost!\n");
     return 0;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza:");
  for(i=0;i<n;i++)
   scanf("%d",&a[i]);

  /* Ispisuje se rezultat poziva funkcije */
  printf("%d\n", prebrojavanje(a,n));
```

```
48
  return 0;
50 }
```

Rešenje 3.1.26

```
#include <stdio.h>
2
#define MAX 100
4
/* Funkcija vraca broj parnih elemenata niza koji prethode
    maksimalnom elementu niza */
6 int prebrojavanje(int a[], int n)
{
8   int i;

10  int maksimum;
  int pozicija_maksimuma;
12
  /* Brojac elemenata koji su parni i prethode maksimalnom */
14  int broj_parnih;

16  /* Pronalazi se maksimalni element niza i njegova pozicija */
  maksimum = a[0];
18  pozicija_maksimuma = 0;

20  for(i=1;i<n-1;i++)
    if(a[i]> maksimum)
22    {
      maksimum = a[i];
24    pozicija_maksimuma = i;
    }
26
  /* Prebrojavaju se parni elementi koji prethode maksimalnom */
28  broj_parnih = 0;
  for(i=0; i < pozicija_maksimuma; i++){
30    if(a[i]%2==0){
      broj_parnih++;
32    }
  }
34
  /* Vraca se izracunata vrednost */
36  return broj_parnih;
}
38
int main()
40 {
  int a[MAX];
42  int n;
  int i;
44
```

```
  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     */
  printf("Unesite broj elemenata niza:");
  scanf("%d", &n);
  if(n<=0 || n>MAX)
  {
     printf("Greska: Nedozvoljena vrednost!\n");
     return 0;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza:");
  for(i=0;i<n;i++){
   scanf("%d",&a[i]);
  }

  /* Ispisuje se rezultat poziva funkcije */
  printf("%d\n", prebrojavanje(a,n));

  return 0;
}
```

Rešenje 3.1.27

```
#include <stdio.h>
#include <ctype.h>

#define MAX 100

/* Funkcija prebrojava cifre u datom nizu karaktera */
int cifre(char a[], int n)
{
  int i;

  /* Brojac cifara */
  int broj_cifara = 0;

  /* Obilazi se element po element niza */
  for(i=0;i<n;i++){
    /* Ako je tekuci element cifra */
    if(isdigit(a[i])){
      /* Uvecava se broj cifara */
      broj_cifara++;
    }
  }

  /* Vraca se izracunata vrednost */
  return broj_cifara;
}

int main()
```

```
{
  char a[MAX];
  int n;
  int i;

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     */
  printf("Unesite broj elemenata niza:");
  scanf("%d", &n);
  if(n<=0 || n>MAX)
  {
     printf("Greska: Nedozvoljena vrednost!\n");
     return 0;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza:");
  for(i=0;i<n;i++) {
    /* Preskace se prethodno uneti znak za novi red */
   getchar();

   /* A zatim se ucitava sam karakter i smesta u niz */
   scanf("%c",&a[i]);
  }

  /* Ispisuje se rezultat poziva funkcije */
  printf("Broj cifara je: %d\n", cifre(a,n));

  return 0;
}
```

Rešenje 3.1.28

```
#include<stdio.h>

#define MAX 100

/* Funkcija racuna zbir elemenata niza od pozicije i do pozicije j */
int zbir(int a[], int n, int i, int j){
  int k;

  /* Zbir elemenata niza iz zadatog opsega */
  int z = 0;

  /* Obilaze se elementi niza */
  for(k=i; k<=j; k++){
  z+=a[k];
  }

  /* Vraca se izracunata vrednost */
  return z;
```

```
}

int main(){

  int n, i, j;
  int a[MAX];

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);
  if(n <= 0 || n > MAX)
  {
  printf("Greska: Nedozvoljena vrednost!\n");
  return 0;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza:");
  for(i=0; i<n; i++)
  scanf("%d", &a[i]);

  /* Ucitavaju se vrednosti granica */
  printf("Unesite vrednosti za i i j: ");
  scanf("%d%d", &i, &j);

  /* Proverava se korektnost zadatog intervala */
  if(i < 0 || j < 0 || i > n-1 || j > n-1 || i > j){
  printf("Greska: Nekorektne vrednosti granica!\n");
  return 0;
  }

  /* Ispisuje se rezultat poziva funkcije */
  printf("Zbir je: %d", zbir(a,n,i,j));

  return 0;
}
```

Rešenje 3.1.29

```
#include<stdio.h>

#define MAX 100

/* Funckija racuna zbir prvih k pozitivnih elemenata niza */
float zbir_pozitivnih(float a[], int n, int k){

  int i;

  /* Zbir pozitivnih elemenata */
  float zbir=0;
```

```
12
   /* Obilazi se element po element niza - postupk se zavrsava ukoliko
      se dodje do kraja niza
14    ili ukoliko se sabere k pozitivnih elemenata */
   for(i=0; i<n && k>0; i++){
16   /* Ako je tekuci element pozitivan broj */
   if(a[i] >= 0){
18   /* Dodaje se zbiru */
     zbir+=a[i];
20     /* I umanjuje se brojac pozitivnih elemenata */
     k--;
22 }
   }
24
   /* Vraca se izracunata vrednost */
26 return zbir;
}
28
int main(){
30 int n, i, k;
   float a[MAX];
32
   /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
      */
34 printf("Unesite broj elemenata niza: ");
   scanf("%d", &n);
36 if(n<=0 || n> MAX){
   printf("Greska: Nedozvoljena vrednost!\n");
38 return 0;
   }
40
   /* Ucitavaju se elementi niza */
42 printf("Unesite elemente niza: ");
   for(i=0; i<n; i++)
44 scanf("%f", &a[i]);

46 /* Ucitava se broj k i proverava se njegova ispravnost */
   printf("Unesite vrednost za k: ");
48 scanf("%d", &k);
   if(k<0 || k>n){
50 printf("Greska: Nedozvoljena vrednost!");
   return 0;
52 }

54 /* Ispisuje se rezultat poziva funkcije */
   printf("Zbir je: %.2f\n", zbir_pozitivnih(a,n,k));
56
   return 0;
58 }
```

Rešenje 3.1.30

```c
#include<stdio.h>

#define MAX 100

/* Funkcija koja kvadrira elemente niza koji se nalaze na parnim
    pozicijama */
void kvadriranje(float a[], int n){

  int i;

  /* Obilaze se elementi na parnim pozicijama */
  for(i=0; i<n; i+=2){
      /* I kvadriraju se: a[i] = a[i]*a[i] */
    a[i]*=a[i];
  }

}

int main(){

  int n, i, j;
  float a[MAX];

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);
  if(n <=0 || n>MAX)
  {
  printf("Greska: Nedozvoljena vrednost!\n");
  return 0;
  }

  /* Ucitavaju se elementi niza */
  printf("Unesite elemente niza:");
  for(i=0; i<n; i++)
  scanf("%f", &a[i]);

  /* Poziva se funkcije */
  kvadriranje(a,n);

  /* Ispisuje se elementi novodobijenog niza */
  /* Koriscenje specifikatora %g za stampanje realnih brojeva
    omogucava ispis broja
    na onoliko decimalnih mesta koliko ima i sam broj */
  for(i=0; i<n; i++){
  printf("%g ", a[i]);
  }
  printf("\n");

  return 0;
```

```
49 }
```

Rešenje 3.1.31

Rešenje 3.1.32

Rešenje 3.1.33

Rešenje 3.1.34

Rešenje 3.1.35

3.3 Pokazivači

Zadatak 3.3.1 Tekst

[Rešenje 3.3.19]

Zadatak 3.3.2 Tekst

[Rešenje 3.3.2]

Zadatak 3.3.3 Tekst

[Rešenje 3.3.3]

Zadatak 3.3.4 Tekst

[Rešenje 3.3.4]

Zadatak 3.3.5 Tekst

[Rešenje 3.3.5]

Zadatak 3.3.6 Tekst

[Rešenje 3.3.6]

Zadatak 3.3.7 Napisati program koji ispisuje zbir numeričkih argumenata komandne linije. Napomena: može se koristi 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 3.3.7]

Zadatak 3.3.8 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 3.3.8]

Zadatak 3.3.9 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 3.3.9]

Zadatak 3.3.10 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 3.3.10]

Zadatak 3.3.11 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 3.3.11]

Zadatak 3.3.12 Napisati funkciju $void\ modifikacija(char*\ s,\ char*\ t, int*\ br\_modifikacija)$ koja na osnovu niske s formira nisku t tako što svako malo slovo zamanjuje 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 3.3.12]

Zadatak 3.3.13 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 3.3.13]

Zadatak 3.3.14 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 3.3.14]

Zadatak 3.3.15 Napisati funckiju *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

50 i ispisuje vrednosti minimuma i maksimuma na tri decimale.

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 3.3.15]

Zadatak 3.3.16 Tekst

[Rešenje 3.3.19]

Zadatak 3.3.17 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 3.3.19]

Zadatak 3.3.18 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 naččće 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 3.3.19]

Zadatak 3.3.19 Parametri komandne linije su `n`, `a`, `b` ($a < b$). Treba popuniti prvih `n` elemenata niza `A` celim slučajnim brojevima koji su izmeụ `a` i `b`. Ištampati niz `A` na standarni 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 3.3.19]

3.4 Rešenja

Rešenje 3.3.19

```
/*
  Napisati funkciju uredi koja uredjuje svoja dva
  celobrojna argumenta tako da se u prvom nalazi manji
  a u drugom veci. Napisati potom glavni program koji
  ucitava dva cela broja i uredjuje njihove vrednosti
  primenom napisane funkcije. Na primer, ako su ucitane
  promenljive x=5 i y=2, njihove vrednosti nakon
  primene funkcije uredi treba da budu x=2 i y=5.
*/
```

```c

#include <stdio.h>

/*
   Argumenti funkcije uredi_pogresno, promenljive a i b,
   predstavljaju lokalne promenljive za ovu funkciju
   i prestaju da postoje po zavrsetku funkcije. Zbog toga
   se efekti razmene vrednosti promenljivih a i b u slucaju
   da je a>b vide u funkciji, ali se ne vide u glavnom programu.
*/
void uredi_pogresno(int a, int b)
{
  int t;

  if (a>b)
  {
     t = a;
     a = b;
     b = t;
  }
  printf("uredi_pogresno :: a=%d, b=%d\n", a, b);
  printf("uredi_pogresno :: &a=%p, &b=%p\n", &a, &b);
}

/*
   Argumenti funkcije uredi_tacno, promenljive pa i pb,
   takodje su lokalne promenljive za ovu funkciju i
   prestaju da postoje kada se funkcija zavrsi.
   Njima prosledjujemo adrese promenljivih a i b koje zelimo
   da razmenimo u slucaju da je a>b.

   Promenljivoj a pristupamo preko pokazivacke promenljive
   pa sa *pa i slicno, promenljivoj pb pristupamo sa *pb.

   Vrednosti promenljivih *pa i *pb razmenjujemo kao
   i vrednosti bilo koje dve celobrojne promenljive.

*/
void uredi_tacno(int * pa, int * pb)
{
  int t;
  if (*pa>*pb)
  {
     t = *pa;
     *pa = *pb;
     *pb = t;
  }
  printf("uredi_tacno :: *pa=%d, *pb=%d\n ", *pa, *pb);
  printf("uredi_tacno :: pa=%p, pb=%p\n ", pa, pb);

}
int main()
```

```
{
63   int a,b;

65   printf("Unesi dve celobrojne promenljive:");
     scanf("%d%d",&a,&b);
67
     printf("main :: a=%d, b=%d\n", a,b);
69   printf("main :: &a=%p, &b=%p\n", &a, &b);
     uredi_pogresno(a,b);
71   printf("main :: nakon uredi_pogresno, a=%d, b=%d\n", a, b);

73   /*
        Funkcija uredi_tacno kao argument ima dve pokazivacke
       promenljive
75      (int*,int*). Zbog toga joj je u pozivu funkcije neophodno
       proslediti
77      adrese promenljivih koje zelimo da uredimo rastuce, &a i &b.
     */

79   uredi_tacno(&a, &b);
     printf("main :: nakon uredi_tacno, a=%d, b=%d\n", a, b);
81
     return 0;
83 }
```

Rešenje 3.3.2

```
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 ucitava 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 aritmeticke operacije
19      imaju veci prioritet od operatora dereferenciranja (*).
     */
21   *a=1-(*a)/255;
     *b=1-(*b)/255;
23   *c=1-(*c)/255;
```

```
  /*
  Pomocu return ne mozemo vratiti vise od jedne vrednosti.

  Ceste greske:
  return a,b,c;          return vraca samo jednu vrednost
  return a; return b; return c; return ce vratiti samo a

  Zato je neophodno da promenljive ciju vrednost
  zelimo da promenimo prenesemo preko pokazivaca.
  */
}


int rgb_korektno(float a)
{
   if(a<0 || a>255)
      return 0;
   return 1;
}


int main()
{
  float a,b,c;

  /*
      Argumenti funkcije rgb_to_cmy su
      pokazivaci na float. Njima prosledjujemo
      adrese promenljivih a, b i c.
  */

  printf("Unesi boju u rgb formatu (vrednosti izmedju 0 i 255:");
  scanf("%f%f%f",&a,&b,&c);

  if(rgb_korektno(a) && rgb_korektno(b) && rgb_korektno(c))
     rgb_to_cmy(&a,&b,&c);
  else
  {
     printf("Nekorektan unos\n");
     return -1;
  }

  printf("Nakon konverzije: %.2f,%.2f,%.2f\n", a,b,c);

  return 0;
}
```

Rešenje 3.3.3

```
/*
   Napisati funkciju koja za dve prave date svojim koeficijentima
```

```
   pravca i slobodnim clanovima odredjuje njihovu tacku preseka.
   Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju
   tacku preseka (ako su paralelne). Napisati glavni program
   koji ucitava podatke o pravama, poziva napisanu funkciju i
   ispisuje odgovarajucu poruku.
*/

#include<stdio.h>

/*
   Funkcija presek treba da izracuna tri vrednosti:
   1. indikator da li su koeficijenti pravca jednaki ili ne
   2. prvu koordinatu presecne tacke (ukoliko prave nisu paralelne)
   3. drugu koordinatu presecne tacke (ukoliko prave nisu paralelne)

   Indikator funkcija vraca kao povratnu vrednost, preko kljucne reci
   return.

   Koordinate presecne tacke (ako postoji) funkcija vraca preko
   liste argumenata, zbog cega promenljive kojima ce koordinate
   biti dodeljene prenosimo preko pokazivaca (promenljive px i py)

   Promenljive koje sadrze podatke o pravama (k1,n1,k2,n2) se ne
   menjaju u funkciji i zbog toga ih ne moramo prenositi preko
   pokazivaca.
*/

int presek(float k1, float n1, float k2, float n2, float* px, float*
    py)
{
   if (k1==k2)
     return 0;

   *px = -(n1-n2)/(k1-k2);
   *py = k1*(*px)+n1;
   return 1;
}

int main()
{
   float k1,k2,n1,n2;
   float x,y;

   printf("Unesi k i n za prvu pravu:");
   scanf("%f%f",&k1,&n1);

   printf("Unesi k i n za drugu pravu:");
   scanf("%f%f",&k2,&n2);

   if(presek(k1,n1,k2,n2,&x,&y))
      printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
   else
```

```
54        printf("Prave su paralelne\n");

56    return 0;
}
```

Rešenje 3.3.4

```
/*
     Napisati program koji ispisuje broj navedenih argumenata komandne
      linije,
     a zatim i same argumenate i njihove redne brojeve.
*/

#include <stdio.h>

/*
   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.
*/

int main(int argc, char *argv[])
{
   int i;

   printf("Broj argumenata je: %d\n",argc);

   for(i=0; i<argc; i++)
      printf("%d: %s\n",i,argv[i]);

   return 0;
}
```

Rešenje 3.3.5

```
/*
   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:
13 ./a.out aladin lad
   4
15
   3:
17 ./a.out Aladin ala
   0
19
*/
21
#include <stdio.h>
23 #include <string.h>

25 /*
   Funkcija strspn(str,accept) je ugradjena funkcija koja vraca broj
    karaktera
27 stringa str koji se nalaze u stringu accept, pocev od pocetka
    stringa str.

29 Funkcija strspn se nalazi u zaglavlju string.h.

31 Funkcija strspn_klon je jedna implementacija funkcije strspn.

33 U zadacima cemo uvek koristiti ugradjenu funkciju strspn osim ako
    u tekstu zadatka
   nije naglaseno da se ona ne sme koristiti. Funkcija strspn_klon
    sluzi da pokaze na koji
35 nacin radi ugradjena funkcija strspn.

37 Ugradjena funkcija strspn poziva se na isti nacin kao funkcija
    strspn_klon:
   strspn(s1,s2)
39
*/
41
int strspn_klon(char str[], char accept[])
43 {
   int br=0;
45 int i;

47 for(i=0; str[i];i++)
      if(strchr(accept, str[i])!=NULL)
49       br++;
      else      /* ako pronadjemo karakter u stringu str koji nije */
51       break; /* u stringu accept, prekidamo petlju */

53 return br;
}
55
int main(int argc, char* argv[])
57 {
```

```
59    int br;

61    if(argc<3)
      {
63       printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
        arg2\n");
         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 3.3.6

```
/*
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:
   ./a.out abcd 3 -p X
36 Nekorektan unos

38 6:
   ./a.out ab12cd 2 -p D
40 cd12ef

42 */

44 #include <stdio.h>
   #include <string.h>
46 #include <stdlib.h>
   #define MAX 31
48
   void sifruj(char s[], char c, int k)
50 {
      int i;
52    int znak;
      char t;
54
      /*
56       S obzirom da ce korektnost unosa podataka
         biti ispitana pre poziva funkcije, promenljiva
58       c ce imati vrednost 'L' ili 'D'.

60       Promenljiva znak ima vrednost 1 ili -1
         i sluzi kao pomocna promenljiva u slucaju
62       da prilikom sifriranja konvertovani
         karakter izadje iz opsega malih ili velikih slova.
64
      */
66    znak=1;
      if (c=='L')
68       znak = -1;


70
      for(i=0; s[i];i++)
72       if(isalpha(s[i]))
         {
74          /*
               Promenljiva t predstavlja sifrirani karakter s[i].
76             Ako je promenljiva t izvan opsega malih ili velikih slova
     ,
               dodajemo joj ili oduzimamo ukupan broj slova u abecedi
```

```
    (26),
78              u zavisnosti od pravca sifriranja, kako bismo omogucili
                kruzno sifriranje.
80          */
            t = s[i]+znak*k;
82          if((islower(s[i]) && (t<'a' || t>'z')) || (isupper(s[i]) &&
    (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;
    char pravac;
94    char rec[MAX];

96    /*
        Program mozemo pozivati na dva nacina:
98      ./a.out abcd 2
        ili
100     ./a.out abcd 2 -p D

102     Zbog toga, broj argumenata moze biti 3 ili 5.
    */
104
    if (argc!=3 && argc!=5)
106 {
        printf("Nekorektan unos: broj argumenata moze biti 3 ili 5\n");
108     return -1;
    }
110
    /*
112     Argumenti komandne linije su stringovi. Ako program pokrecemo
        na sledeci nacin:
114     ./a.out abcd 2 -p D
        to znaci da je argument koji odgovara dvojci u stvari
116     string "2". Da bismo string konvertovali u ceo broj,
        koristimo ugradjenu funkciju atoi iz biblioteke stdlib.h.
118 */

120 k = atoi(argv[2]);

122 /*
        Ispitujemo korektnost datih podataka:
124 */
    if (k<=0)
126 {
        printf("Nekorektan unos: broj pozicija mora biti pozitivan ceo
```

```
    broj\n");
128       return -1;
    }
130
    /* Korektnost unosa je ispitana, sto znaci da
132 argc moze biti 3 ili 5 */

134 if (argc==3) /* Ako je argc 3: */
       pravac='D';
136 else         /* Ako argc nije 3, tada je sigurno 5, jer je */
    {            /* korektnost unosa ispitana, a unos je korektan
     jedino za argc==3 ili argc==5 */
138   /*
         Ispitujemo korektnost pretposlednjeg argumenta koji mora da
     bude u formatu "-p".
140      Ovaj argument je string argv[3]. Njegovom prvom karakteru (
     koji treba
         da bude '-') pristupamo sa argv[3][0] a drugom sa argv
     [3][1].
142   */
      if (argv[3][0] != '-')
144   {
         printf("Nekorektan unos: pri zadavanju opcija prvi karakter
     mora biti '-' \n");
146      return -1;
      }
148
      if (argv[3][1]!='p')
150   {
         printf("Nekorektan unos: nedozvoljena opcija\n");
152      return -1;
      }
154
      /*
156      Nakon argumenta -p sledi argument koji zadaje vrednost ove
     opcije. To je
         poslednji argument kome pristupamo sa argv[4]. Ovaj argument
      treba
158      da sadrzi samo jedan karakter - 'L' ili 'D' i njemu
     pristupamo sa
         argv[4][0].
160   */
      if(argv[4][0]=='L' || argv[4][0]=='D')
162      pravac=argv[4][0];
      else
164   {
         printf("Nekorektan unos: pravac moze biti L ili D\n");
166      return -1;
      }
168 }

170 strcpy(rec, argv[1]);
```

```
    sifruj(rec,pravac,k);

    printf("Sifrovana rec: %s\n", rec);

    return 0;
}
```

Rešenje 3.3.7

```
#include <stdio.h>

int main(int argc, char* argv[]) {

  int i;
  int s = 0;

  /* char *argv[] <--- niz niski koje predstavljaju argumente
     navedene iza poziva programa
   int argc      <--- ukupan broj niski (sa sve nazivom programa)
     navedenih prilikom pozivanja

   Ukoliko je program pozvan sa ./a.out 12 abc 6 5 3ab

   argv[0] = "./a.out".
   argv[1] = "12"
   argv[2] = "abc"
   argv[3] = "6"
   argv[4] = "5"
   argv[5] = "3ab"

   argc iznosi 6


   Kako je argv[] po prirodi niz,
   koristimo tzv. brojacku odnosno
   for petlju
   i obradjujemo svaki od argumenata.
  */

  /* Funkcija atoi() prihvata nisku,
   i racuna dekadnu vrednost prosledjene niske,
   dokle god se ona moze racunati.
   Na primer, ukoliko je niska "-123",
   atoi() vraca broj -123.
   Ako je, pak, niska "123abc",
   atoi() ce vratiti 123
   (prilikom prve pojave karaktera koji nije cifra, funkcija prekida
    izracunavanje).

   To za posledicu ima da, ukoliko je funkciji
   prosledjeno nesto
```

```
   sto se ne moze pretvoriti u broj,
   na primer niska "abcd",
   funkcija atoi() vraca 0.
*/

for(i = 1; i < argc; i++)
  s += atoi(argv[i]); /* Zbog nacina rada funkcije atoi(), mozemo
  je pozvati nad svim argumentima
            komandne linije, i sabrati odgovarajuce dekadne
  vrednosti.
            Ukoliko neki argument i nije broj, to ne predstavlja
  problem
            jer ce u tom slucaju odgovarajuci sabirak biti 0
          */


  printf("Zbir numerickih argumenata: %d\n", s);

return 0;
}
```

Rešenje 3.3.8

```
#include <stdio.h>

int main(int argc, char* argv[]) {

  int i;

  /* Prolazimo for petljom kroz niz argumenata,
   i trazimo one niske ciji je prvi karakter bas 'z'.
   Ukoliko je trenutni argument koji se ispituje
   argv[i],
   kako je on sam po sebi niska,
   do prvog karaktera dolazimo kao i pri dosadasnjem
   radu sa niskama --> argv[i][0]
                       ^
                       |
                     index prvog karaktera u niski argv[i]
  */

  for(i = 1; i < argc; i++)
    if(argv[i][0] == 'z')
      printf("%s ", argv[i]);

  putchar('\n');

  return 0;
}
```

Rešenje 3.3.9

```c
#include <stdio.h>
#include <string.h>

int main(int argc, char* argv[]) {

  int i;
  int br = 0;

  /* Da bismo proverili da li se karakter 'z' (tj. 'Z')
   nalazi u niski argv[i],
   to mozemo uciniti koriscenjem funkcije
   strchr() koja se nalazi u string.h.

   Ukoliko je karakter sadrzan u okviru niske,
   strchr() vraca pokazivac na taj karakter
   unutar same niske.
   Inace, ukoliko se karakter ne nalazi u niski,
   funkcija vraca NULL.
  */

  for(i = 1; i < argc; i++)
    if(strchr(argv[i], 'z') != NULL || strchr(argv[i], 'Z') != NULL)
      br++;

  printf("%d\n", br);

  return 0;
}
```

Rešenje 3.3.10

```c
#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
  int n,i;

  /*
    Ispisujemo gresku ukoliko nema dovoljno argumenata komandne
    linije.
  */
  if(argc != 2)
  {
    printf("Greska: nedostaje argument komandne linije!\n");
    return -1;
  }
```

```
  /*
    Pretvaramo argument komandne linije koji je string u ceo broj
    koriscenjem funkcije atoi
  */
  n = atoi(argv[1]);
  n = abs(n);

  for(i=(-1)*n;i<=n;i++)
    printf("%d ",i);

  return 0;
}
```

Rešenje 3.3.11

```
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
  int indikator = 0;
  int i,j;
  /*
    Ukoliko imamo samo jedan argument komandne linije,
    ispisujemo da nema istih i zavrsavamo program.
  */
  if(argc < 2)
  {
    printf("Medju argumentima nema istih.\n");
    return -1;
  }

  /*
    Prolazimo kroz niz argumenata i za svaki posebno proverimo
    da li medju ostalima postoji neki koji mu je jednak i ako postoji
    ispisujemo poruku i zavrsavamo program.
    Ako smo izasli iz prve petlje to znaci da nismo pronasli dva ista
     elementa
    i ispisujemo odgovarajucu poruku.
  */
  for(i=0;i<argc;i++)
  {
    for(j=0;j != i && j<argc; j++)
      if(strcmp(argv[i], argv[j]) == 0)
      {
        printf("Medju argumentima ima istih.\n");
        return 0;
      }
  }

  printf("Medju argumentima nema istih.\n");
```

```
  return 0;
}
```

Rešenje 3.3.12

```
#include <stdio.h>

#define MAX 21

void modifikacija(char *s, char *t, int *br_modifikacija)
{
  int i;
  for(i=0;s[i];i++)
    if(s[i]>='a' && s[i]<='z')
    {
      t[i] = toupper(s[i]);
      (*br_modifikacija)++;
    }
    else
      t[i] = s[i];
}

int main()
{
  char s[MAX], t[MAX];
  int br_modifikacija = 0;

  printf("Unesite nisku: ");
  scanf("%s", s);

  modifikacija(s, t, &br_modifikacija);

  printf("Modifikovana niska je: %s\nBroj modifikacija je: %d\n", t,
    br_modifikacija);

  return 0;
}
```

Rešenje 3.3.13

```
/*
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 tacaka i zareza. Napisati zatim program koji testira napisanu
    funkciju.
*/

```

```c
#include <stdio.h>

void interpunkcija(int* br_tacaka, int* br_zareza){

  int tacke=0, zarezi=0;
  char c;

  while((c=getchar())!=EOF){
    if(c=='.')
      tacke++;

    if(c==',')
      zarezi++;
  }

  *br_tacaka=tacke;
  *br_zareza=zarezi;

}

int main(){
  int br_tacaka, br_zareza;

  printf("Unesite tekst: \n");

  interpunkcija(&br_tacaka, &br_zareza);

  printf("Broj tacaka: %d\n", br_tacaka);
  printf("Broj zareza: %d\n", br_zareza);

  return 0;
}
```

Rešenje 3.3.14

```c
/*
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. Pokazivaci
    pn i nn
redom treba da sadrze broj elemenata niza parnih tj. niza neparnih
    elemenata.
Pretpostaviti da duzina niza a nece biti veca od 50. Napisati program
     koji
testira napisanu funkciju.
*/

#include <stdio.h>
#define MAX 50
```

```
12
   void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
       int* nn){
14
     int i, j, k;
16
     /* i - brojac niza a */
18   /* 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++;
         }
28       else{
             /* Inace, smestamo ga u niz neparnih brojeva i uvecavamo
       indeks niza k */
30           neparni[k]=a[i];
             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 */
54   printf("Unesite elemente niza: ");
     for(i=0; i<n; i++){
56     scanf("%d", &a[i]);
     }
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++)
66      printf("%d ", parni[i]);
      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 3.3.15

```
  /*
2   Napisati funckiju
      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 decimale
      .

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
        */
        if(a[i]>*max){
26        *max=a[i];
        }
28
```

```
    /* Ako je tekuca vrednost manja od minimalne, azuriramo minimum
    */
    if(a[i]<*min){
      *min=a[i];
    }
  }
}

int main(){
  int i, n;
  float a[MAX], min, max;

  /* Ucitavamo dimenziju niza */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);

  if(n<0 || n>MAX){
    printf("Greska: pogresna dimenzija niza!\n");
    return 0;
  }

  /* Ucitavamo elemente niza */
  printf("Unesite elemente niza:\n");
  for(i=0; i<n; i++){
    scanf("%f", &a[i]);
  }

  /* Pozivamo funkciju za racunanje maksimuma i minimuma */
  min_max(a, n, &min, &max);

  /* Ispisujemo rezultat */
  printf("Minimum: %.3f\n", min);
  printf("Maksimum: %.3f\n", max);

  return 0;

}
```

Rešenje 3.3.19

```
#include <stdio.h>

void suma(int a, int b, int *s);


int main()
{
    int a,b,s;

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

```

```
    suma(a,b,&s);

    printf("suma: %d\n",s);

    return 0;
}

void suma(int a, int b, int *s)
{
    *s = a + b;
}
```

Rešenje 3.3.19

```
#include <stdio.h>

void suma(int a, int b, int *s);


int main()
{
    int a,b,s;

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

    suma(a,b,&s);

    printf("suma: %d\n",s);

    return 0;
}

void suma(int a, int b, int *s)
{
    *s = a + b;
}
```

Rešenje 3.3.19

```
#include <stdio.h>

void suma(int a, int b, int *s);


int main()
{
    int a,b,s;

    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 3.3.19

```
   #include <stdio.h>
2
   void suma(int a, int b, int *s);
4

6  int main()
   {
8      int a,b,s;
10     scanf("%d%d",&a,&b);
12     suma(a,b,&s);
14     printf("suma: %d\n",s);
16     return 0;
   }
18
   void suma(int a, int b, int *s)
20 {
       *s = a + b;
22 }
```

3.5 Niske

Zadatak 3.5.1 Napisati funkciju koja konvertuje dati string tako sto mala slova menja u velika a velika u mala. Napisati potom glavni program koji ucitava string, poziva napisanu funkciju i ispisuje konvertovani string. Mozemo pretpostaviti da string ne sadrzi vise od 10 karaktera.

[Rešenje 3.5.1]

Zadatak 3.5.2 Napisati funkciju skrati koja uklanja beline sa kraja datog stringa. Napisati program koji testira napisanu funkciju na stringu `"rep belina  "`.

[Rešenje 3.5.2]

Zadatak 3.5.3 Napisati program koji ucitava string src i formira string dst trostrukim nadovezivanjem stringa src. Program treba da ispise string dst. Na primer, za uneti string ”dan”, string dst treba da bude ”dandandan”. Pretpostaviti da string src nije duzi od 30 karaktera.

[Rešenje 3.5.3]

Zadatak 3.5.4 Napisati funkciju int ucitaj_liniju(char s[], int n) koja ucitava liniju maksimalne duzine n u string s i vraca duzinu ucitane linije. Linija moze da sadrzi blanko znakove ali ne moze da sadrzi `\n` ili EOF.

Napisati potom glavni program koji ucitava linije do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko ima vise linija maksimalne duzine, ispisati prvu. Mozemo pretpostviti da svaka linija sadrzi najvise 80 karaktera, zajedno sa `\n`.

[Rešenje 3.5.4]

Zadatak 3.5.5 Napisati program koji pretvara nisku u ceo broj. Npr. za ulaz ”-1238” se generise rezultat -1238 Pogledati funkcije atoi i atof koje postoje u biblioteci stdlib.h

[Rešenje 3.5.5]

Zadatak 3.5.6 Napisati program koji pretvara zadatu broj u nisku. Npr. za broj -453 treba generisati nisku ”-453”

[Rešenje 3.5.6]

Zadatak 3.5.7 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 prvog. Ukoliko jeste, program treba da ispise pocev od kog indeksa prvog stringa pocinje drugi string. U suprotnom, ispisati odgovarajucu poruku. Mozemo pretpostaviti da stringovi ne sadrze vise od 20 karaktera.

[Rešenje 3.5.7]

Zadatak 3.5.8 Napisati program koji za uneti string s i karakter c utvrdjuje da li se c pojavljuje u stringu s i ukoliko se pojavljuje, ispisuje indeks prvog pojavljivanja a u suprotnom ispisuje odgovarajucu poruku. Mozemo pretpostaviti da string ima najvise 20 karaktera.

[Rešenje 3.5.8]

Zadatak 3.5.9

a) 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.

b) Napisati funkciju *int samoglasnik_na_kraju*(*char s*[]) koja proverava da li se niska *s* završava samoglasnikom (koristiti funkciju iz tačke a)).

c) 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 zavrsava samoglasnikom!
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: AaBb+cCdD
 Niska se ne zavrsava samoglasnikom!
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: pRograMiranjE
 Niska se zavrsava samoglasnikom!
```

[Rešenje 3.5.9]

Zadatak 3.5.10 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 3.5.10]

Zadatak 3.5.11 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 3.5.11]

Zadatak 3.5.12 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 3.5.12]

Zadatak 3.5.13 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: 0xE1A9
 57769
```

[Rešenje 3.5.13]

Zadatak 3.5.14 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 3.5.14]

Zadatak 3.5.15 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 3.5.15]

Zadatak 3.5.16 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 3.5.16]

Zadatak 3.5.17 Napisati implementaciju funkcije *char* strchr$_k$lon(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 3.5.17]

Zadatak 3.5.18

- 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 mnB RGa
```

Primer 3

```
Interakcija sa programom:
 -3
 -1
```

Primer 4

```
Interakcija sa programom:
 4 2abc AVF$ abc AV4
 abc
```

[Rešenje 3.5.35]

Zadatak 3.5.19 Napisati funkciju `void min_razlika(char s[], char s1[], char s2[])` koja u datotoj nisci s pronalazi dve reči koje imaju minimalnu razliku izmeų svojih samoglasnika. (Reč je niz karaktera izmeų dve praznine; razmak izmeų samoglasnika reči `danas` i `jutro` je 2, a razmak izmedju `sutrk` i `mnozenje` je 5). Testirati pozivom u main-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.5.35]

Zadatak 3.5.20 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 3.5.35]

Zadatak 3.5.21 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 3.5.35]

Zadatak 3.5.22 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 3.5.35]

Zadatak 3.5.23 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 111111111
```

Primer 4

```
Interakcija sa programom:
 981
 1111010101
```

[Rešenje 3.5.35]

Zadatak 3.5.24 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 najvise 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 3.5.35]

Zadatak 3.5.25 Napisati funkciju `void sifrat(char* rec, char* kljuc)` koja šifruje `rec` na sledeći način: za svako slovo reči `rec` i odgovarajuće slovo `kljuca` određuje koliki je (alfabetski) razmak između njih i označimo taj broj sa `k`. Potom to slovo `reci` zamenjuje `k-tim` slovom alfabeta. Podrazumeva se da je `kljuc` duži od reci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
bac
dfge
bed
```

[Rešenje 3.5.35]

Zadatak 3.5.26 Napisati funkciju `void obrni(char rec[], int k)` koja rotira reč za k mesta ulevo.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
sveska
2
eskasv
```

[Rešenje 3.5.35]

Zadatak 3.5.27 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 najvišse 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 3.5.35]

Zadatak 3.5.28 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 3.5.35]

Zadatak 3.5.29 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 "uvwbcdnoppqr" a reč "zec" sa "abcfghdef". Napisati program koji šifruje unetu reč sa standardnog ulaza i štampa dobijeni rezultat na standardni izlaz. Za reč pretpostaviti da nije duža od 20 karaktera. Unos reči ostvariti koristeci specifikator "

[Rešenje 3.5.35]

Zadatak 3.5.30 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 izmeu 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 3.5.35]

Zadatak 3.5.31 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 po- javljivanje 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 takoe 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. Dopušteno je koristiti pomoćan niz.

[Rešenje 3.5.35]

Zadatak 3.5.32 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 3.5.35]

Zadatak 3.5.33

a) Napisati C funkciju `int procitaj_recenicu(char *s, int max_len)`, koja sa standarnog 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 3.5.35]

Zadatak 3.5.34

a) Uvesti tip podataka `Sifra` kojim se opisuje način šifrovanja alfanumeričkih karaktera. Svaka šifra se opisuje celobrojnom 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 standarnog 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 standarnog ulaza (maksimalna dužina reči je 20 karaktera). Ispisati reči na standarni izlaz nakon primenjenih svih zadatih načina šifrovanja.

[Rešenje 3.5.35]

Zadatak 3.5.35 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.

[Rešenje 3.5.35]

3.6 Rešenja

Rešenje 3.5.1

```
/*
   Napisati funkciju koja konvertuje dati string tako sto
   mala slova menja u velika a velika u mala. Napisati
   potom glavni program koji ucitava string, poziva napisanu
   funkciju i ispisuje konvertovani string. Mozemo pretpostaviti
   da string ne sadrzi vise od 10 karaktera.
*/

#include <stdio.h>
#include <ctype.h>

/*
   Kada je niz argument funkcije, dodatni argument je obavezno
   njegova dimenzija. Kod stringova to nije slucaj jer svaki string
   ima isti poslednji element - terminirajucu nulu - i to je oznaka
   kraja stringa.
*/
void konvertuj(char s[])
{
   int i;

   for(i=0; s[i]!='\0'; i++)
      if (s[i]>='a' && s[i]<='z')
         s[i] = toupper(s[i]);  /* toupper - konvertuje malo slovo u
    odgovarajuce veliko */
      else if (s[i]>='A' && s[i]<='Z')
         s[i] = tolower(s[i]);  /* tolower - konvertuje veliko slovo
    u odgovarajuce malo */
   /*
      Funkcije toupper i tolower se nalaze u zaglavlju ctype.h.

      Konverzija malog slova u veliko bez upotrebe funkcije toupper:
          s[i] = s[i]-'a'+'A';
      Konverzija velikog slova u malo bez upotrebe funkcije tolower:
          s[i] = s[i]+'a'-'A';


   */
}

int main()
{
   /*
      Poslednji karakter svakog stringa je terminirajuca
      nula '\0', specijalni karakter ciji je ASCII kod 0.

      Ukoliko pretpostavljamo da string sadrzi najvise 30
```

```
      karaktera, neophodno je deklarisati niz od 31 karaktera,
      pri cemu se dodatni izdvaja za terminirajucu nulu.

   */
   char s[31];
   printf("Unesi string:");

   /*
      Za razliku od nizova koji se ucitavaju i stampaju
      element po element, stringovi se mogu ucitati i
      odstampati pomocu jedne scanf/printf naredbe koriscenjem
      specifikatora %s.

      Funkcija scanf ucitava string do prvog pojavljivanja razmaka.
   */
   scanf("%s", s);

   konvertuj(s);

   printf("Konvertovani string: %s\n", s);

   return 0;

}
```

Rešenje 3.5.2

```
/*
   Napisati funkciju skrati koja uklanja beline sa
   kraja datog stringa.

   Napisati glavni program koji testira napisanu
   funkciju na stringu "rep belina                    ".

*/

#include <stdio.h>
#include <ctype.h>

/*
    Funkcija koja racuna duzinu niza
    ne racunajuci '\0'.

    U biblioteci string.h definisan je veliki
    broj funkcija za rad sa stringovima,
    ukljucujuci i funkciju strlen koja racunana
    duzinu stringa.

    Funkcija strlen_klon predstavlja jednu
    implementaciju funkcije strlen.

```

```
    U zadacima cemo uvek koristiti ugradjenu
    funkciju strlen osim ako u tekstu zadatka
    nije naglaseno da se ona ne sme koristiti.
    Funkcija strlen_klon sluzi da pokaze na koji
    nacin radi ugradjena funkcija strlen.

    Ugradjena funkcija strlen poziva se na
    isti nacin kao funkcija strlen_klon:
    strlen(s1)

*/
int strlen_klon(char s[])
{
  int i=0;
  while(s[i]) /* ASCII kod karaktera '\0' je jednak nuli. */
     i++;

  return i;
}

void skrati(char s[])
{
  /*
     Poslednji karakter stringa s(ne racunajuci '\0') ima
     indeks strlen_klon(s)-1; ideja je da pocnemo od poslednjeg
     karaktera stringa i da smanjujemo indeks dokle god
     je karakter na poziciji i blanko znak.

  */
  int i;
  for(i=strlen_klon(s)-1;i>=0;i--)
      if (!isspace (s[i])) /* Ako s[i] nije blanko znak, prekidamo
    petlju. */
          break;

  s[i+1]='\0';  /* DOdajemo terminirajucu nulu iza indeksa i (prvi
    neblanko karakter gledano sdesna nalevo).*/

  /*
     Ugradjena funkcija isspace nalazi se u biblioteci ctype.h i
    vraca 1 ako
     je dati karakter blanko znak a 0 u suprotnom.

     Unarni logicki operator ! oznacava negaciju.

  */
}

int main()
{

  /*
```

```
74      Ukoliko string ne zelimo da ucitavamo 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    skrati(s);
      printf("Posle skracivanja: *%s*\n", s);
86
      return 0;
88  }
```

Rešenje 3.5.3

```
/*
2     Napisati program koji ucitava 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
        funkciju strcpy osim ako u tekstu zadatka
32      nije naglaseno da se ona ne sme koristiti.
```

```
    Funkcija strcpy_klon sluzi da pokaze na koji
    nacin radi ugradjena funkcija strcpy.

    Ugradjena funkcija strcpy poziva se na
    isti nacin kao funkcija strcpy_klon:
    strcpy(dst,src)
    gde karaktere stringa src kopiramo
    u string dst.


*/
void strcpy_klon(char kopija[], char original[])
{
  int i;
  for(i=0; original[i]; i++)
    kopija[i]=original[i];

  kopija[i] = '\0';
}

int main()
{
  char src[MAX+1];   /* src, skraceno od source (izvor, odnosno sta
    kopiramo) */
  char dst[3*MAX+1];   /* dst, skraceno od destination (odrediste,
    odnosno gde kopiramo) */

  /*
     Vazno je izdvojiti dovoljno memorijskog prostora
     za string dst: on treba da bude tri puta veci od
     maksimalne duzine stringa src + jedan karakter za
     terminirajucu nulu.
  */

  printf("Unesi jedan string:");
  scanf("%s", src);

  strcpy_klon(dst,src);

  /*
    Funkcija strcat(s,t) nadovezuje karaktere stringa
    t na kraj stringa s i novi string terminira
    karakterom '\0' .

    Funkcija strcat se nalazi u biblioteci string.h.
  */
  strcat(dst,src);
  strcat(dst,src);

  printf("Kada nadovezemo string %s triput: %s\n",src,dst);

  return 0;
```

```
}
```

Rešenje 3.5.4

```
/*
   Napisati funkciju int ucitaj_liniju(char s[], int n)
   koja ucitava liniju maksimalne duzine n u string s
   i vraca duzinu ucitane linije. Linija moze da sadrzi
   blanko znakove ali ne moze da sadrzi \n ili EOF.

   Napisati potom glavni program koji ucitava linije
   do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko
   ima vise linija maksimalne duzine, ispisati prvu. Mozemo
   pretpostviti da svaka linija sadrzi najvise 80 karaktera,
   zajedno sa \n.

*/

#include<stdio.h>
#include<string.h>
#define MAX 81

/*
   Ukoliko zelimo da ucitamo string koji sadrzi beline
   (npr liniju teksta), ne mozemo koristiti funkciju
   scanf jer ona ucitava string do prvog blanko znaka.

   Zbog toga je neophodno napisati funkciju koja ucitava
   string karakter po karakter.

   Ova funkcija ne dopusta unosenje vise karaktera od
   unapred odredjene granice (argument n).

   U standardnoj biblioteci stdio.h postoji definisana
   funkcija char *gets(char *s) koja ucitava karaktere
   dok se ne pojavi novi red ili EOF. Ova funkcija
   dopusta unosenje vise karaktera nego sto string
   s sadrzi, sto moze dovesti do neocekivanog ponasanja
   programa.

   Pored funkcije gets, koja vrsi ucitavanje sa standardnog
   ulaza, u standardnoj biblioteci stdio.h postoji
   i ugradjena funkcija fgets koja vrsi ucitavanje iz
   datoteke. Nju cemo koristiti za nekoliko casova
   kada budemo radili datoteke. Prototim funkcije fgets je
   ovakav:

   char *fgets(char *s, int size, FILE *stream);

   Argumenti funkcije fgets su:
   s - string u koji vrsimo ucitavanje
```

```
    size - maksimalna duzina unetog stringa
    stream - datoteka iz koje vrsimo ucitavanje

    Funkcija fgets, za razliku od funkcije gets, ne dopusta
    unos vise karaktera od date vrednosti size. Zbog toga
    je ona sigurnija nego funkcija gets. Funkciju fgets
    mozemo koristiti i za unos sa standardnog ulaza
    ukoliko kao treci argument navedemo stdin.

*/
int ucitaj_liniju(char s[], int n)
{
  int i=0;
  int c;

  while((c=getchar())!='\n' && i<n-2 && c!=EOF)
  {
    s[i] = c;
    i++;
  }

  /* Ucitavamo najvise n-2 karaktera jer na kraju dodajemo jos
     dva: '\n' i '\0' */

  s[i]='\n';
  s[i+1]='\0';

  return i;

}

int main()
{
  char linija[MAX];
  char najduza_linija[MAX];
  int max_duzina=0;
  int duzina;

  /*
     Petlja se zavrsava ukoliko je promenljiva duzina
     jednaka nuli, sto cemo postici zadavanjem linije koja ne sadrzi
     nijedan karakter osim EOF.
  */

  while ((duzina=ucitaj_liniju(linija, MAX))>0)
  {
     /*
        Proveravamo da li je uneta linija duza od trenutnog
        maksimuma i azuriramo promenljive max_duzina i najduza_linija
     .
     */
     if (max_duzina<duzina)
```

```
99      {
            max_duzina = duzina;
101         strcpy(najduza_linija,linija);
        }
103   }

105   printf("Najduza linija: %s duzine: %d\n", najduza_linija,
        max_duzina);

107   return 0;
}
```

Rešenje 3.5.5

```
/*
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
#include <stdio.h>
8 #include <ctype.h>
#define MAX 10
10 /*
    String b se sastoji od karaktera koji
12  cine jedan ceo broj, onim redom kojim
    se karakteri pojavljuju u zapisu broja.
14
    Ako je prvi karakter stringa b '-',
16  to znaci da je broj negativan i
    funkcija znak_broja vraca -1
18
    U suprotnom, broj je pozitivan i
20  funkcija znak_broja vraca 1

22 */

24 int znak_broja(char b[])
{
26     if(b[0]=='-')
           return -1;
28     return 1;
}
30


32 /*
    Funkcija formiraj_broj na osnovu
34  karaktera koji cine broj iz stringa
    b vraca ceo broj koji odgovara
36  zapisu datom u stringu b.
```

```
  Ako su cifre broja a,b,c i d, tada
  broj mozemo kreirati kao:
  a*10^3 + b*10^2 + c*10^1 + d*10^0

  Medjutim, efikasnije je koristiti
  Hornerovu semu:

  10*(10*(10*(10*0 + a)+b)+c)+d

*/

int formiraj_broj(char b[])
{
   int i;
   int n=0;
   int znak = znak_broja(b);

   /*
     Ako je broj negativan, cifre u nizu b
     pocinju od indeksa 1
   */

   i=0;
   if(znak==-1)
      i=1;

   /*
     Funkcija isdigit proverava da li je broj
     cifra. Nalazi se u biblioteci ctype.h

     Proveravamo da li je karakter u zapisu
     broja cifra kako bismo se osigurali
     od nekorektnog unosa, npr ako korisnik
     unese -123abc. Ovaj unos je moguc jer
     se vrsi sa scanf("%s",broj), gde unosimo
     karaktere do prvog blanko znaka

     Ako naidjemo na karakter koji nije cifra,
     prekidamo petlju

   */
   for(; b[i]!='\0'; i++)
      if(isdigit(b[i]))
         n = n*10 + b[i] - '0';
      else
         break;

   /* Formirani broj mnozimo znakom: */

   n*=znak;
   return n;
```

```
}

int main()
{
   char broj[MAX];
   int n;

   /* Ucitavamo broj: */
   scanf("%s", broj);

   /* Ispisujemo rezultat: */
   printf("Broj zapisan kao int: %d\n", formiraj_broj(broj));

   return 0;
}
```

Rešenje 3.5.6

```
/*
  Napisati program koji pretvara zadatu broj u nisku.
  Npr. za broj -453 treba generisati nisku "-453"
*/

#include <stdio.h>
#include <string.h>
#define MAX 10
/*

   Funkcija transformisi_negativan vraca
   1 ako je broj negativan i 0 u suprotnom, a
   uz to, ako broj jeste negativan, funkcija
   treba da ga konvertuje u  njegovu apsolutnu
   vrednost. S obzirom da funkcija treba da vrati dve
   vrednosti, to realizujemo na sledeci nacin:
   1. indikator da li je broj negativan
   ce vratiti kao povratnu vrednost
   2. apsolutnu vrednost broja ce vratiti
   preko liste argumenata, zbog cega broj
   prenosimo preko pokazivaca

*/
int transformisi_negativan(int* pn)
{
   if(*pn<0)
   {
      *pn = -(*pn);
      return 1;
   }
   return 0;
}
```

```
34 int formiraj_niz_cifara(int n, char b[], int neg)
   {
36    int i=0;
      char cifra;
38
      do
40    {
         cifra = n%10;
42
         /* Promenljiva b predstavlja string.
44          Da bismo na neku poziciju u stringu
            upisali karakter koji odgovara nekoj
46          cifri, npr '2', neophodno je da
            odgovarajucoj poziciji dodelimo vrednost
48          ASCII koda te cifre, konkretno za '2'
            ASCII kod je '0'+2.
50
            Greska bi bila navesti b[i]=2
52          jer 2 nije ASCII kod koji odgovara karakteru
            '2'.
54       */
         b[i]=cifra+'0';
56
         n/=10;
58       i++;
      } while(n);
60
      /* Ako je broj negativan, dodajemo znak minus: */
62    if(neg)
      {
64       b[i]='-';
         i++;
66    }

68    /* Svaki string se zavrsava terminirajucom nulom: */
      b[i]='\0';
70 }

72 void obrni(char s[])
   {
74
      char t;
76    int i,j;
      /*
78     Karaktere stringa obrcemo tako sto razmenimo karaktere na
       pozicijama 0 i n-1,
       zatom 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
        druge
80    */

82    for(i=0,j=strlen(s)-1;i<j;i++, j--)
      {
```

```
84          t = s[i];
            s[i] = s[j];
86          s[j] = t;
        }
88
    }
90
    void broj_u_niz_cifara(int n, char broj[])
92  {
        int negativan;
94
        /* Odredjujemo znak broja: */
96      negativan=transformisi_negativan(&n);

98      /* Izdvajamo cifre broja i smestamo ih u niz: */
        formiraj_niz_cifara(n, broj, negativan);
100
        /* S obzirom da cifre izdvajamo sa kraja broja, u nizu ce biti u
         obrnutom redosledu.
102        Na primer, za broj 234 niz ce sadrzati cifre 4 3 2. */
        obrni(broj);
104 }

106 int main()
    {
108     int n;
        char broj[MAX];
110     int negativan;

112     /* Ucitavamo broj: */
        scanf("%d", &n);
114
        /* Kreiramo broj na osnovu niza cifara: */
116     broj_u_niz_cifara(n,broj);

118     /* Ispisujemo rezultat: */
        printf("Broj zapisan kao string: %s\n", broj);
120
        return 0;
122 }
```

Rešenje 3.5.7

```
    /*
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
   pretpostaviti da stringovi ne sadrze vise od 20 karaktera.
*/

#include <stdio.h>
#include <string.h>

/*
       Funkcija strcmp(s,t) je ugradjena funkcija koja utvrdjuje da
    li su strinovi
       s i t jednaki. Ukoliko jesu, vraca 0, a u suprotnom vraca
    razliku
       ASCII kodova prva dva razlicita karaktera na istim pozicijama
       (npr strcmp("aa","ab") ce vratiti -1 a strcmp("ab","aa") 1).

       Funkcija strcmp se nalazi u zaglavlju string.h.

       Funkcija strcmp_klon je jedna implementacija funkcije strcmp.

       U zadacima cemo uvek koristiti ugradjenu funkciju strcmp osim
    ako u tekstu zadatka
       nije naglaseno da se ona ne sme koristiti. Funkcija
    strcmp_klon sluzi da pokaze na koji
       nacin radi ugradjena funkcija strcmp.

       Ugradjena funkcija strcmp poziva se na isti nacin kao funkcija
     strcmp_klon:
       strcmp(s1,s2)
       gde poredimo stringove s1 i s2.

*/

int strcmp_klon(char s1[], char s2[])
{
  int i;
  for(i=0; s1[i]==s2[i];i++)
    if (s1[i]=='\0')
      return 0;

  return s1[i] - s2[i];
}

int main()
{
   char s1[21];
   char s2[21];
   char* p;

   printf("Unesi dva stringa:");
   scanf("%s%s",s1,s2);

```

```
    /*
52      Funkcija strstr(s,t) je ugradjena funkcija koja utvrdjuje 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:
60      ako je string deklarisan sa s1[21], tada je njegov naziv s1
        ekvivalentan adresi prvog karaktera stringa:
62      s1 <=> &s1[0]
        i nadalje redom:
64      s1+1 <=> &s1[1]
        ...
66      u opstem slucaju:
        s1+i <=> &s1[i]
68
        To znaci da se indeks elementa na koji pokazuje s1+i moze
70      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
    return 0;
86  }
```

Rešenje 3.5.8

```
/*
2   Napisati program koji za uneti string s i karakter c utvrdjuje
    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.
*/
```

```c

#include <stdio.h>
#include <string.h>

int main()
{
   char s[21];
   char c;
   char* p;

   printf("Unesi karakter:");
   c=getchar();
   printf("Unesi string:");
   scanf("%s", s);

   /*
     Da smo ucitavali obrnutim redom (prvo string pa karakter)
     to bismo realizovali na sledeci nacin:
     printf("Unesi string:");
     scanf("%s",s);
     getchar();
     printf("Unesi karakter:");
     c=getchar();

     Dodatni getchar() bi sluzio da "pokupi" karakter kojim
     razdvajamo unos stringa i karaktera (razmak, novi red ili
     slicno).

   */

   /*
      Funkcija strchr(s,c) je ugradjena funkcija koja vraca pokazivac
      na prvi karakter u stringu s koji je jednak karakteru c, ako
    takav
      postoji, a NULL u suprotnom.

      Indeks od pokazivaca dobijamo na isti nacin kao u prethodnom
    zadatku
      sa strstr.
   */

   p = strchr(s,c);
   if(p!=NULL)
      printf("%c se pojavljuje u %s na poziciji %d\n", c, s, p-s);
   else
      printf("%c se NE pojavljuje u %s\n",c, s);

   return 0;
}
```

Rešenje 3.5.9

```c
/*
a) 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.
b) Napisati funkciju int samoglasnik_na_kraju(char s[]) koja
    proverava da li se niska s zavrsava samoglasnikom
(koristiti funkciju iz tacke a)).
c) Napisati program koji ucitava nisku maksimalne duzine 20 karaktera
    i ispisuje da li zavrsava samoglasnikom ili ne.
*/

#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define MAX_DUZINA 20

/* Funkcija proverava da li je karakter c samoglasnik */
int samoglasnik(char c){

  char C;

  /* Konvertujemo karakter u veliko slovo kako bismo smanjili broj
    provera */
  C=toupper(c);

  /* Samoglasnici su slova a, e, i, o i u */
  if(C=='A' || C=='E' || C=='I' || C=='O' || C=='U')
    return 1;

  return 0;
}

/* Funkcija koja proverava da li se niska s zavrsava samoglasnikom */
int samoglasnik_na_kraju(char s[]){
  int duzina;

  /* Odredjujemo duzinu niske */
  duzina=strlen(s);

  /* Proveravamo da li je niska prazna */
  if(duzina==0)
    return 0;

  /* Ako niska nije prazna, proveravamo da li se samoglasnik nalazi
    na kraju */
  /* Numeracija karaktera u niski pocinje nulom pa zato proveravamo
    poziciju duzina -1 */
  return samoglasnik(s[duzina-1]);
}
```

```
45
int main(){
47   char s[MAX_DUZINA+1];

49   /* Ucitavamo nisku */
     printf("Unesite nisku: ");
51   scanf("%s", s);

53   /* Proveravamo da li se zavrsava samoglasnikom i ispisujemo
       odgovarajucu poruku */
     if(samoglasnik_na_kraju(s))
55     printf("Niska se zavrsava samoglasnikom!\n");
     else
57     printf("Niska se ne zavrsava samoglasnikom!\n");

59   return 0;
}
```

Rešenje 3.5.10

```
/*
2 Napisati funkciju void kopiraj_n(char t[], char s[], int n) koja
      kopira najvise n karaktera niske s u nisku t.
  Napisati i program koji ucitava nisku maksimalne duzine 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 najvise 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];

28   /* Ucitavamo nisku */
```

```
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Ucitavamo broj n i proveravamo korektnost unosa */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<0 || n>MAX){
    printf("Greska: pogresan unos!\n");
    return 0;
  }

  /* Pozivamo funkciju za kopiranje */
  kopiraj_n(t, s, n);

  /* Ispisujemo dobijenu nisku */
  printf("Rezultujuca niska: %s\n", t);

  return 0;
}
```

Rešenje 3.5.11

```
/*
Napisati funkciju void dupliranje(char t[], char s[]) koja na osnovu
    niske s formira nisku t tako sto duplira svaki
karakter niske s. Napisati i program koji ucitava nisku maksimalne
    duzine 20 karaktera i testira rad napisane funkcije.
*/

#include <stdio.h>
#define MAX_DUZINA 20

void dupliranje(char t[], char s[]){
  int i, j;

  /* 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){
    t[j]=s[i];
    t[j+1]=s[i];
  }

  /* Upisujemo terminirajucu nulu u novodobijenu nisku */
  t[j]='\0';
}


int main(){
  int n;
  char s[MAX_DUZINA+1], t[2*MAX_DUZINA+1];

```

```
  /* Ucitavamo nisku */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Pozivamo funkciju za dupliranje */
  dupliranje(t, s);

  /* Ispisujemo dobijenu nisku */
  printf("%s\n", t);

  return 0;
}
```

Rešenje 3.5.12

```
/*
 Napisati funkciju int heksa_broj(char s[]) koja proverava da li je
    niskom s zadat korektan heksadekadni broj.
 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.
 Napisati i program koji ucitava nisku maksimalne duzine 7 karaktera
    i ispisuje rezultat rada funkcije.
 */

#include<stdio.h>
#define MAX 8


/*
 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.
 */
int heksa_cifra(char c){
  /*Pretvaramo karakter c u veliko slovo*/
  c = toupper(c);

  /*Proveravamo da li je u pitanju cifra ili slovo A,B,C,D,E,F i
    ukoliko jeste, vracamo 1*/
  if(isdigit(c) || (c >= 'A' && c <= 'F'))
  return 1;

  /*Ukoliko nije, vracamo 0*/
  return 0;
}

/*Funkcija koja proverava da li prosledjena niska s predstavlja
    ispravan heksadekadni broj */
```

```c
int heksa_broj(char s[]){
  int i;

  /*Kako heksadekadni brojevi pocinju sa 0x ili 0X, prvo proveravamo
    da li je taj uslov ispunjen,
  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 */
   if(s[0] != '0' || toupper(s[1]) != 'X')
   return 0;

   /*Prolazimo kroz nisku, pocev od pozicije 2 (jer su prve dve
    pozicije 0x) i za svaki karakter do kraja
   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
   i vracamo 0. */
   for(i=2; s[i] != '\0'; i++)
   if(!heksa_cifra(s[i]))
     return 0;

   /*Ako smo stigli do kraja, znaci da su svi karakteri date niske
    ispravne heksadekadne cifre
     i zato vracamo 1 */
   return 1;
}

int main(){
  char s[MAX];

  /*Ucitavamo nisku*/
  printf("Unesite nisku:");
  scanf("%s", s);

  /*Pozivamo funckiju i stampamo odgovarajucu poruku*/
  if(heksa_broj(s))
  printf("Korektan heksadekadni broj!\n");
  else
  printf("Nekorektan heksadekadni broj!\n");

  return 0;
}
```

Rešenje 3.5.13

```c
/* 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.
 * Pretpostaviti da je uneta niska korektan heksadekadni broj. */
```

```c
#include<stdio.h>
#include<string.h>

#define MAX 8

/*Funkcija koja racuna dekadnu vrednost heksadekadnog broja*/
int heksa_broj(char s[]){
  int i,k;
  char c;

  /*Racunamo duzinu niske koja predstavlja heksadekadni broj*/
  int n = strlen(s);

  /*Inicijalizujemo vrednost v na 0*/
  int v = 0;

  /*Prolazimo petljom kroz nisku, krenuvsi sa desne strane
  npr: 1a8e = e*1 + 8*16 + a*256 + 1*4096
  Promenljiva k ce nam biti mnozilac i ona uzima vrednosti 1, 16,
    256, 4096, ...
  Promenljiva c ce nam cuvati trenutnu heksadekadnu cifru (u nasem
    primeru e, 8, a, 1)
  U svakom koraku treba na ispravan nacin da pomnozimo c i k
  */
  for(i=n-1, k=1; i>=2; i--, k*=16)
  {
  /*U c smestamo trenutnu heksadekadnu cifru.
   Pozivamo funkciju toupper da bi obezbedili da radimo samo sa
     velikim slovima.
   Ako je s[i] cifra, funkcija toupper je nece promeniti.
   */
  c = toupper(s[i]);

  if(isdigit(c)){
    /*Ako je c cifra, onda samu vrednost te cifre dobijamo sa c-'0'
     NAPOMENA: Nije ispravno napisati c*k jer je c karakter!
     */
    v += (c-'0')*k;
  }else{
    /*Ako je c slovo izmedju A i F, mi treba da dobijemo odgovarajucu
     vrednost izmedju 10 i 15.
     Ova vrednost se dobija sa 10 + c - 'A'. npr. za A ce biti 10 + '
    A' - 'A' = 10, za B: 10 + 'B' - 'A' = 11, ...*/
    v += (c - 'A' + 10)*k;
  }
  }

  /*Na kraju vracamo izracunatu vrednost */
   return v;
}

int main(){
```

```
  char s[MAX];

  /*Ucitavamo nisku*/
  printf("Unesite nisku:");
  scanf("%s", s);

  /*Ispisujemo rezultat*/
  printf("%d\n", heksa_broj(s));

  return 0;
}
```

Rešenje 3.5.14

```
/*
 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.

 Napomena: u biblioteci string.h postoji funkcija strstr
 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
 u niski s.
 */

#include<stdio.h>
#define MAX 11

/*Funkcija koja proverava da li je t podniska od s*/
int podniska(char s[], char t[]){
  int i, j, k;

  /*Spoljna petlja ide redom po niski s*/
  for(i=0; s[i] != '\0'; i++){

  /*Unutrasnja petlja ide redom po niski t*/
  /*Promenljiva k pamti vrednost i, sluzi za poredjenje s[k] i t[j] i
   *zatim se vrsi pomeranje i po niski s i po niski t (k++, j++) */
  /*Cim naidjemo na situaciju da se karakteri ne poklapaju, izlazimo
    iz  unutrasnje petlje*/
  for(j=0, k = i; t[j] != '\0'; j++, k++){
    if(s[k] != t[j])
    break;
  }

  /*Ako smo prosli celu unutrasnju petlju (do kraja niske t), to
    znaci da su se svi karakteri iz t poklopili
   sa karakterima iz s i onda vracamo 1*/
  if(t[j] == '\0')
```

```
    return 1;
  }

  /*Na kraju vracamo 0*/
  return 0;
}


int main(){
  char s[MAX], t[MAX];

  /*Ucitavamo prvu nisku*/
  printf("Unesite nisku s: ");
  scanf("%s", s);

  /*Ucitavamo drugu nisku*/
  printf("Unesite nisku t: ");
  scanf("%s", t);

  /*Pozivamo funkciju i ispisujemo odgovarajucu poruku*/
  if(podniska(s,t))
  printf("t je podniska niske s!\n");
  else
  printf("t nije podniska niske s!\n");

  return 0;
}
```

Rešenje 3.5.15

Rešenje 3.5.16

Rešenje 3.5.17

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

Rešenje 3.5.35

3.7 Višedimenzioni nizovi

Zadatak 3.7.1

a) Napisati funkciju
`int refleksivna(int a[][MAX], int n)`
kojom se za relaciju zadatom matricom `a` (matruca je kvadratna) ispitije da li je refleksivna.

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` => `j@i=1`
Relacija je tranzitivna ako za svako `i`, `j` i `k` važi: `i@j=1` i `j@k=1` => `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 3.7.17]

Zadatak 3.7.2 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 3.7.17]

Zadatak 3.7.3 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 3.7.17]

Zadatak 3.7.4 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 3.7.17]

Zadatak 3.7.5 Napisati funkciju u kojoj se razmenjuju elemeti k-te i t-te vrste matrice(k i t su argumenti funkcije). Funkciju testirati pozivom u main-u i ispisom novodobijene matrice na standarni izlaz. Sa standarnog ulaza učitavaju se dimenzije matrice, a potom i elementi matrice i brojevi k i t. Maksimalna dimenzija matrice je 50x50. Funkciju testirati pozivom u main-u.

[Rešenje 3.7.17]

Zadatak 3.7.6 Sa standarnog 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 standari 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 3.7.17]

Zadatak 3.7.7 Sa standarnog 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 3.7.17]

Zadatak 3.7.8 Sa standarnog 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 3.7.17]

Zadatak 3.7.9 Sa standarnog 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 standarni 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 $a = (a_1, a_2, \ldots, a_n)$ i $b = (b_1, b_2, \ldots, b_n)$ je $a_1 \cdot b_1 + a_2 \cdot b_2 + \ldots + a_n \cdot b_n$.
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 3.7.17]

Zadatak 3.7.10 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 3.7.17]

Zadatak 3.7.11 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 3.7.17]

Zadatak 3.7.12 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 3.7.17]

Zadatak 3.7.13 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 3.7.17]

Zadatak 3.7.14 Sa standarnog 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 3.7.17]

Zadatak 3.7.15 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 100×100.

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 3.7.17]

Zadatak 3.7.16 Sa standarnog 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) izmedju 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 dato-

teku 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 3.7.17]

Zadatak 3.7.17 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 50×50. Ukoliko nisu zadati svi potrebni argumenti komadne 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 3.7.17]

3.8 Rešenja

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

Rešenje 3.7.17

3.9 Strukture

Zadatak 3.9.1 Data je struktura koja opisuje koordinate tacke u ravni:

```
typedef struct point
{
float x;
float y;
} POINT;
```

U glavnom programu date su dve tacke: tacka A sa fiksiranim koordinatama (1,2) i tacka B cije koordinate zadaje korisnik. Napisati funkcije:
a) za racunanje rastojanja izmedju dve date tacke
b) za odredjivanje tacke koja se nalazi na sredini duzi odredjene dvema datim tackama.

Testirati napisane funkcije u glavnom programu.

[Rešenje 3.9.1]

Zadatak 3.9.2 Data je struktura

```
typedef struct Student
{
  char ime[MAX];
  char prezime[MAX];
  char smer;
  float prosek;
} STUDENT;
```

(a) Napisati funkciju koja ucitava sa standardnog ulaza podatke o studentu. Mozemo pretpostaviti da ime i prezime studenta ne sadrze vise od 30 karaktera.

(b) Napisati funkciju koja ispisuje podatke o studentu na standardni izlaz.

(c) Ucitati niz od n studenata i :

  (a) ispisati imena i prezimena onih koji su na smeru R

  (b) ispisati podatke za studenta sa najvecim prosekom; ako ima vise takvih studenata, ispisati

    (a) sve njih

    (b) prvog

    (c) poslednjeg

[Rešenje 3.9.2]

Zadatak 3.9.3

Napisati program koji ucitava reci sa standardnog ulaza dok korisnik ne zada EOF i ispisuje ih na standardni izlaz svaku u posebnom redu, poravnatu udesno u odnosu na poslednji karakter najduze reci. Koristiti

```
strukturu typedef struct rec
 {
    char s[21];
    int duzina;
        }REC;
```

Na primer, ako su unesene sledece reci:

```
Danas imamo ispit iz programiranja1.
Nadam se da nece biti tesko!
```

onda ispis izgleda ovako:

```
        Danas
        imamo
        ispit
          iz
programiranja1.
        Nadam
          se
          da
         nece
         biti
       tesko!
```

Program realizovati kroz sledece funkcije:

(a) Funkciju za ucitavanje jedne reci u strukturu REC.

(b) Funkciju za ucitavanje niza struktura koja vraca dimenziju niza

(c) Funkciju koja odredjuje maksimalnu duzinu reci u datom nizu

(d) Funkciju koja ispisuje reci u trazenom formatu

Mozemo pretpostaviti da nijedna rec ne sadrzi vise od 30 karaktera i da nece biti uneto vise od 1000 reci.

[Rešenje 3.9.3]

Zadatak 3.9.4 Napisati program koji izracunava prosecnu cenu jedne potrosacke korpe. Potrosacka korpa se sastoji od broja kupljenih artikala i niza kupljenih artikala. Svaki artikal odredjen je svojim nazivom, kolicinom i cenom. Program treba da ucita broj potrosaca n (najvise 100), zatim podatke za n potrosackih korpi i da na osnovu ucitanih podataka izracuna prosecnu cenu potrosacke korpe. Ucitavanje se vrsi sa standarnog ulaza pri cemu se prvo zada broj artikala, a zatim za svaki artikal naziv, kolicina i cena. Mozemo pretpostaviti da nijedan potrosac nece kupiti vise od 20 artikala, kao i da naziv svakog artikla sadrzi maksimalno 30 karaktera.

[Rešenje 3.9.4]

Zadatak 3.9.5 Struktura IZRAZ opisuje numericki izraz nad celim brojevima koji se sastoji od dva celobrojna operanda, numericke operacije nad celim brojevima i vrednosti izraza:

```
typedef struct izraz  char o; int x; int y;  IZRAZ;
```

(a) Napisati funkciju koja ispituje da li je dati izraz korektno zadat i vraca 1 ako jeste a 0 u suprotnom. Podrazumevamo da je izraz korektno zadat ako operacija odgovara +,-,* ili / i u slucaju deljenja drugi operand je razlicit od 0.

(b) Napisati funkciju koja za dati izraz odredjuje vrednost izraza.

(c) Napisati funkciju koja ucitava dati izraz. Funkcija treba da ucita sa standardnog ulaza operaciju i dva operanda u polja o, x i y strukture IZRAZ. Funkcija vraca 1 ako je ucitavanje bilo uspesno, tj. ako je izraz bio korektno zadat ili 0 u suprotnom.

(d) Napisati funkciju koja stampa dati izraz infiksno, u obliku x o y = vr. Na primer, za izraz + 4 17 ispis treba da bude 4+17=21

Napisati glavni program koji ucitava prirodan broj $n < 1000$ a zatim n izraza u notaciji
+ 4 17
- 8 -16
Program treba da ispise maksimalnu vrednost medju unetim izrazima i da ispise one izraze cija je vrednost manja od polovine maksimalne vrednosti.

[Rešenje 3.9.5]

Zadatak 3.9.6 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 3.9.6]

Zadatak 3.9.7 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 3.9.7]

Zadatak 3.9.8 Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vocka* sadrži ime voćke (nisku maksimalne dužine 20 karaktera) i količinu vitamina C u miligramima (realan broj). Napisati program koji sa standardnog ulaza učitava podatke o voćkama sve do unosa reči KRAJ i ispisuje ime voćke sa 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:   pomorandza 53.2
 Unesite ime voćke i njenu količinu vitamina C:   KRAJ
 Voce sa najvise C vitamina je: kivi
```

[Rešenje 3.9.8]

Zadatak 3.9.9 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 3.9.9]

Zadatak 3.9.10 Definisati 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 3.9.10]

Zadatak 3.9.11 Definisati 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 3.9.11]

Zadatak 3.9.12 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 standarnog ulaza (najviše 100). Ispisati njihov zbir, raliku i proizvod na standardni izlaz.

[Rešenje 3.9.17]

Zadatak 3.9.13 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 3.9.17]

Zadatak 3.9.14 Sa standarnog 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 3.9.17]

Zadatak 3.9.15 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 3.9.17]

Zadatak 3.9.16 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 standarnog 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 standarni izlaz statistiku: koliko ima dečaka, a koliko devojčica i prosečnu ocenu. Potom ispisuje imena dece brojnijeg pola.

[Rešenje 3.9.17]

Zadatak 3.9.17

- 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 izmedju 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 3.9.17]

3.10 Rešenja

Rešenje 3.9.1

```
/*

  Data je struktura koja opisuje koordinate
  tacke u ravni:

  typedef struct point
  {
    float x;
    float y;
  } POINT;

  U glavnom programu date su dve tacke: tacka
  A sa fiksiranim koordinatama (1,2) i tacka B
  cije koordinate zadaje korisnik. Napisati
  funkcije:
  a) za racunanje rastojanja izmedju dve date tacke
  b) za odredjivanje tacke koja se nalazi na
     sredini duzi odredjene dvema datim tackama

  Testirati napisane funkcije u glavnom programu.

*/

#include <stdio.h>
```

```c
#include <math.h>

typedef struct point
{
  float x;
  float y;
} POINT;

/*
  Poljima strukture pristupamo pomocu
  operatora .

  Ako je promenljiva a tipa POINT,
  njenim koordinatama pristupamo
  pomocu a.x i a.y
*/

float rastojanje (POINT a, POINT b)
{
  return sqrt(pow(a.x-b.x,2)+pow(a.y-b.y,2));
}

POINT sredina (POINT a, POINT b)
{
  POINT s;
  s.x = (a.x+b.x)/2;
  s.y = (a.y+b.y)/2;
  return s;
}


int main()
{

  POINT a = {1,2};
  POINT b;
  POINT sredina_a_b;

  /* Ispisujemo koordinate tacke a. */
  printf("Tacka a ima koordinate %.2f,%.2f\n", a.x, a.y);

  /* Ucitavamo koordinate tacke b. */
  printf("Unesi prvu koordinatu tacke: ");
  scanf("%f", &b.x);
  printf("Unesi drugu koordinatu tacke: ");
  scanf("%f", &b.y);
  printf("Tacka b ima koordinate %.2f,%.2f\n", b.x, b.y);

  /* Strukture kao argumenti funkcije - prenos po vrednosti. */
  printf("Rastojanje izmedju tacaka a i b je %.2f\n", rastojanje(a,b)
    );

```

```
  /* Struktura kao povratna vrednost funkcije. */
77  sredina_a_b=sredina(a,b);
  printf("Tacka na sredini izmedju tacaka a i b je %.2f,%.2f\n",
    sredina_a_b.x, sredina_a_b.y);
79
  return 0;
81 }
```

Rešenje 3.9.2

```
1  /*
     Data je struktura
3      typedef struct Student
       {
5        char ime[MAX];
         char prezime[MAX];
7        char smer;
         float prosek;
9      } STUDENT;

11 I   Napisati funkciju koja ucitava sa standardnog ulaza podatke o
       studentu. Mozemo pretpostaviti da
       ime i prezime studenta ne sadrze vise od 30 karaktera.
13 II  Napisati funkciju koja ispisuje podatke o studentu na standardni
       izlaz.
   III Ucitati niz od n studenata i :
15       a) ispisati imena i prezimena onih koji su na smeru R
         b) ispisati podatke za studenta sa najvecim prosekom; ako ima
       vise takvih studenata, ispisati
17           1) sve njih
     2) prvog
19   3) poslednjeg
   */
21
   #include <stdio.h>
23 #define MAXST 100
   #define MAX 31
25
   typedef struct Student
27 {
     char ime[MAX];
29   char prezime[MAX];
     char smer;
31   float prosek;
   } STUDENT;
33
   /*
35    I

37    Ako je dat pokazivac na strukturnu promenljivu s,
      poljima ove strukture pristupamo sa
```

```
39    (*s).ime,(*s).prezime, itd.

41    Zagrade su neophodne zbog prioriteta operatora:
      operator * ima veci prioritet nego opetator . .
43
      Operator -> pruza skraceni zapis za prethodno
45    navedeni pristup poljima:
      s->ime je skraceno za (*s).ime
47    s->prezime je skraceno za (*s).prezime
      itd.
49
   */
51 void ucitaj(STUDENT* s)
   {
53   /* printf("Ime:"); */
     scanf("%s",s->ime);
55   /* printf("Prezime:"); */
     scanf("%s",s->prezime);
57   getchar();
     /* printf("Smer:"); */
59   scanf("%c",&s->smer);
     /* printf("Prosek:");*/
61   scanf("%f", &s->prosek);
   }
63
   /* II */
65 /*
     Kada neku promenljivu prenosimo u funkciju kao argument, obicno
67   je prenosimo po vrednosti (bez pokazivaca), ako se ona nece menjati
        u funkciji
     ili po adresi (preko pokazivaca), ako ce se njena vrednost
       promeniti u funkciji.
69
     Prilikom poziva funkcije, za svaki argument funkcije kreira se
       promenljiva
71   koja predstavlja lokalnu kopiju argumenta i koja prestaje da
       postoji po zavrsetku
     funkcije. S obzirom da se strukuture sastoje od vise polja,
       zauzimaju
73   vise memorije nego nestrukturne promenljive. Zbog toga je za
       njihovo kopiranje
     potrebno vise vremena i vise memorijskih resursa nego za kopiranje
       nestrukturnih
75   promenljivih.

77   Da bismo ucinili program efikasnijim, korisno je da strukturu uvek
       kao
     argument funkcije prenosimo po adresi (preko pokazivaca), bez
       obzira
79   da li ce se struktura u toj funkciji menjati ili ne. Pokazivac na
       strukturu
     zauzima manje memorije nego sama struktura pa je izrada njegove
```

```
    kopije
  brza a kopija pokazivaca uzima manji memorijski prostor nego kopija
  strukture.

  Kada prenosimo strukturnu promenljivu u funkciju po adresi (preko
    pokazivaca), tada
  imamo mogucnost da je u funkciji menjamo. Ukoliko zelimo da
    onemogucimo promenu,
  uz argument dodajemo kljucnu rec const. Ako pokusamo da promenimo
    argument
  funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
     s->smer='X';),
  kompajler ce prijaviti gresku. Na ovaj nacin obezbedjujemo da
    promenljiva
  koju smo preneli po adresi ne da bismo je promenili vec radi
    povecanja efikasnosti programa,
  ne bude, cak ni slucajno, izmenjena u funkciji.

*/

void ispisi(const STUDENT* s)
{
  printf("%s %s, %c, %.2f\n",s->ime, s->prezime, s->smer, s->prosek);
}


float najveci_prosek(STUDENT studenti[], int n)
{
  float m;
  int i;
  /* Pretpostavimo da student sa indeksom 0 ima
     maksimalni prosek. */
  m = studenti[0].prosek;
  for(i=1;i<n;i++)
    if(m<studenti[i].prosek) /* Ako student sa indeksom i ima veci
    prosek od maksimalnog, */
       m=studenti[i].prosek; /* menjamo maksimalni prosek */
  return m;
}

/*
   Struktura moze da bude povratna vrednost funkcije.
*/
STUDENT prvi_student_sa_najvecim_prosekom(STUDENT studenti[], int n,
    float m)
{
  STUDENT s;
  int i;
  for(i=0;i<n;i++)
    if(m==studenti[i].prosek) /* Ako naidjemo na studenta sa
    maksimalnim prosekom, prekidamo petlju. */
    {
```

```
123         /*
              Na strukture se moze primenjivati
125           naredba dodele.
            */
127         s = studenti[i];
            break;
129       }
    return s;
131 }

133 /*
      Strukturu mozemo preneti u funkciju preko pokazivaca. Strukture se
        obavezno
135     prenose preko pokazivaca ukoliko je neophodno promeniti vrednosti
       njihovih
      polja u funkciji.
137 */
    void poslednji_student_sa_najvecim_prosekom(STUDENT studenti[], int n
        , float m, STUDENT* s)
139 {
    int i;
141   for(i=0;i<n;i++)
        if(m==studenti[i].prosek)
143         *s = studenti[i];
    }
145

    /*
147     Napomena: funkcije
          1)prvi_student_sa_najvecim_prosekom
149       2)poslednji_student_sa_najvecim_prosekom
      odredjuju studenta sa najvecim prosekom po odredjenom kriterijumu.
151   Funkcija su realizovane na razlicite nacine kako bi ilustrovale:
      - strukturu kao povratnu vrednost
153   - prenos strukture preko pokazivaca u funkciju, s obzirom da ce se
        promeniti u funkciji

155   Prilikom izrade zadataka moze biti izabran bilo koji od opisanih
       nacina rada, osim
      ako neki nacin nije posebno naglasen u tekstu zadatka.
157

    */
159 int main()
    {
161   STUDENT studenti[MAXST];
    int n;
163   int i;
    float max_prosek;
165   STUDENT student_sa_max_prosekom;
    int indeks;
167


169 /*  printf("Unesi broj studenata:"); */
```

```
   scanf("%d", &n);

   if (n<0 || n>MAXST)
   {
      printf("Nekorektan unos\n");
      return -1;
   }

/*  printf("Unesi podatke o studentima:"); */
   for(i=0;i<n;i++)
   {
/*    printf("%d. student:\n", i); */
     ucitaj(&studenti[i]);
   }

   printf("Studenti sa R smera:\n");
   for(i=0;i<n;i++)
      if(studenti[i].smer == 'R')
         ispisi(&studenti[i]);
   printf("---------------------\n");


   /* b)1)

      Stampamo podatke o svim studentima sa
      maksimalnim prosekom.
   */

   max_prosek = najveci_prosek(studenti, n);
   printf("Svi studenti koji imaju maksimalni prosek:");
   for(i=0;i<n;i++)
      if(studenti[i].prosek==max_prosek)
         ispisi(&studenti[i]);

   /* b)2) */
   student_sa_max_prosekom = prvi_student_sa_najvecim_prosekom(
     studenti,n,max_prosek);

   printf("Prvi student u nizu sa najvecim prosekom: ");
   ispisi(&student_sa_max_prosekom);

   /* b)3) */
   poslednji_student_sa_najvecim_prosekom(studenti,n,max_prosek,&
     student_sa_max_prosekom);

   printf("Poslednji student u nizu sa najvecim prosekom: ");
   ispisi(&student_sa_max_prosekom);

   return 0;
}
```

Rešenje 3.9.3

```
/*

Napisati program koji ucitava reci sa standardnog ulaza dok korisnik ne zada EOF i ispisuje
ih na standardni izlaz svaku u posebnom redu, poravnatu udesno u odnosu
na poslednji karakter najduze reci. Koristiti
strukturu typedef struct rec
    {
       char s[21];
       int duzina;
          }REC;
Na primer, ako su unesene sledece reci:
Danas imamo ispit iz programiranja1.
Nadam se da nece biti tesko!
onda ispis izgleda ovako:
          Danas
          imamo
          ispit
             iz
programiranja1.
          Nadam
             se
             da
           nece
           biti
         tesko!

Program realizovati kroz sledece funkcije:
a) Funkciju za ucitavanje jedne reci u strukturu REC.
b) Funkciju za ucitavanje niza struktura koja vraca dimenziju niza
c) Funkciju koja odredjuje maksimalnu duzinu reci u datom nizu
d) Funkciju koja ispisuje reci u trazenom formatu

Mozemo pretpostaviti da nijedna rec ne sadrzi vise od 30 karaktera i da nece biti
uneto vise od 1000 reci.
*/

#include<stdio.h>
#include<string.h>
#define MAXRECI 100
#define MAX 31

typedef struct rec
{
   char s[MAX];
   int duzina;
}REC;

```

```
49 void ucitaj_rec(REC* rec)
{
51    scanf("%s", rec->s);
   rec->duzina = strlen(rec->s);
53 }

55 /*
   U funkciji ucitaj_niz_reci argument n oznacava broj
57    elemenata niza reci, koji ce biti poznat tek po
   zavrsetku funkcije. Ova promenljiva ce dobiti svoju
59    vrednost u funkciji i zbog toga mora biti prenesena
   preko pokazivaca.
61
*/
63
void ucitaj_niz_reci(REC reci[], int* pn, int granica)
65 {
   int i=0;
67    do
   {
69       ucitaj_rec(&reci[i]);
      i++;
71    }
   while(reci[i-1].duzina>0 && (i-1)<granica);
73
   /*
75       S obzirom da se promenljiva i ucitava
      pre ispitivanja uslova, uslov ispitujemo
77       za rec sa indeksom i-1
   */
79
   *pn = i-1;
81
   /*
83       S obzirom da se vrednost promenljive i
      ucitava i kada je unesen EOF, dimenzija
85       niza odgovarace vrednosti i-1
   */
87 }

89 int max_duzina(REC reci[], int n)
{
91    int najveca_duzina;
   int i;
93
   /*
95       Najvecu duzinu inicijalizujemo na duzinu
      prve reci.
97    */
   najveca_duzina = reci[0].duzina;
99
```

```
   for(i=1;i<n;i++)
101      if(reci[i].duzina>najveca_duzina)   /* Ukoliko u nizu naidjemo
    na rec duzine vece od najvece duzine, */
         najveca_duzina = reci[i].duzina; /* menjamo vrednost
    promenljive najveca_duzina. */
103
   return najveca_duzina;
105 }

107 /*
   Da bismo realizovali ispis u trazenom formatu, pre
109    svake reci ispisujemo onoliko razmaka koliko iznosi
   razlika maksimalne duzine i duzine date reci.
111 */

113 void ispis(REC reci[], int n, int max_d)
{
115    int i,j;

117    for(i=0;i<n;i++)
   {
119       for(j=0;j<max_d-reci[i].duzina;j++)
          printf(" ");
121       printf("%s\n", reci[i].s);
   }
123
}
125
int main(int argc, char* argv[])
127 {
   REC reci[MAXRECI];
129    int najveca_duzina;
   int n;
131
   ucitaj_niz_reci(reci, &n, MAXRECI);
133    najveca_duzina = max_duzina(reci,n);
   ispis(reci, n, najveca_duzina);
135
   return 0;
137 }
```

Rešenje 3.9.4

```
1 /*
   Napisati program koji izracunava prosecnu cenu jedne potrosacke
3    korpe. Potrosacka korpa se sastoji od broja kupljenih artikala i
   niza kupljenih artikala. Svaki artikal odredjen je svojim nazivom,
5    kolicinom i cenom. Program treba da ucita broj potrosaca n (
    najvise 100),
   zatim podatke za n potrosackih korpi i da na osnovu ucitanih
    podataka
```

```
   izracuna prosecnu cenu potrosacke korpe. Ucitavanje se vrsi sa
    standarnog
   ulaza pri cemu se prvo zada broj artikala, a zatim za svaki
    artikal naziv,
   kolicina i cena.  Mozemo pretpostaviti da nijedan
   potrosac nece kupiti vise od 20 artikala, kao i da naziv svakog
    artikla
   sadrzi maksimalno 30 karaktera.
*/

#include <stdio.h>
#define MAXART 20
#define MAXPOT 100
#define MAXNAZIV 31

typedef struct artikal
{
   char naziv[MAXNAZIV];
   int kolicina;
   float cena;
} ARTIKAL;

typedef struct korpa
{
   int br_art;
   ARTIKAL artikli[MAXART];
} KORPA;

/*
   Funkcija ucitaj_artikal ucitava podatke za jedan
   artikal i vraca 1 ako je ucitavanje bilo uspesno
   a 0 u suprotnom. Ucitavanje je neuspesno ukoliko
   kolicina nekog artikla ili njegova cena nisu pozitivni
   brojevi.

   S obzirom da funkcija ucitaj artikal treba da vrati
   dve vrednosti (ucitanu strukturu i indikator uspesnosti),
   strukturu ARTIKAL prenosimo preko pokazivaca a
   indikator uspesnosti vracamo kao povratnu vrednost.

*/

int ucitaj_artikal(ARTIKAL* a)
{

   scanf("%s", a->naziv);
   scanf("%d", &a->kolicina);

   if (a->kolicina<=0)
   {
      printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina
    );
```

```
55          return 0;
    }
57
    scanf("%f",&a->cena);
59  if (a->cena<0)
    {
61      printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
        return 0;
63  }

65  return 1;
}
67
/*
69  Funkcija izracunaj_racun izracunava racun date
    potrosacke korpe u kojoj su inicijalizovani
71  podaci o broju artikala i o svakom pojedinacnom
    artiklu.
73 */
float izracunaj_racun(const KORPA* k)
75 {
    int i;
77  float racun=0;
    for(i=0;i<k->br_art;i++)
79      racun+=k->artikli[i].kolicina * k->artikli[i].cena;
    return racun;
81 }

83 /*
    Pri ucitavanju korpe, zadaje se broj artikala a zatim
85  podaci za svaki artikal.

87  Funkcija ucitaj_korpu vraca 1 ako je ucitavanje uspesno
    i 0 u suprotnom. Do neuspesnog ucitavanja moze doci
89  ako broj artikala u korpi nije pozitivan ili ako dodje
    do neuspesnog ucitavanja nekog artikla.
91 */

93 int ucitaj_korpu(KORPA* k)
95 {
    int i;
    scanf("%d", &k->br_art);
97  if (k->br_art<=0)
    {
99      printf("Nekorektan unos za broj artikala: %d\n", k->br_art);
        return 0;
101 }
    for(i=0; i<k->br_art;i++)
103     if (ucitaj_artikal(&k->artikli[i])==0)
            return 0;
105
    return 1;
```

```
107 }
109 /*
       Funkcija ucitaj_niz_korpi ucitava podatke
111    za niz od n potrosackih korpi. Funkcija
       vraca 1 ako je ucitavanje uspesno i 0 ako
113    nije. Ucitavanje je neuspesno ukoliko ne uspe
       ucitavanje jedne od korpi.
115 */
117 int ucitaj_niz_korpi(KORPA korpe[], int n)
    {
119    int i,j;
       for(i=0; i<n; i++)
121       if(ucitaj_korpu(&korpe[i])==0)
             return 0;
123
       return 1;
125
    }
127
    /*
129    Funkcija stampaj_racun ispisuje na
       standardni izlaz racun za datu korpu
131    tako sto za svaki artikal ispise
       naziv, cenu i kolicinu i na kraju
133    ukupnu cenu za kupljene artikle.
    */
135
    void stampaj_racun(const KORPA* k)
137 {
       int i,j;
139    for(i=0;i<k->br_art;i++)
          printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
        kolicina, k->artikli[i].cena);
141    printf("------------------------\n");
       printf("\tukupno: %.2f\n", izracunaj_racun(k));
143
    }
145
    /*
147    Funkcija stampaj_racune_za_korpe
       ispisuje na standardni izlaz racune
149    za svaku korpu u nizu potrosackih
       korpi
151 */
153 void stampaj_racune_za_korpe(KORPA korpe[], int n)
    {
155    int i;
       for (i=0;i<n;i++)
157    {
```

```
        printf("\nKorpa %d:\n",i);
159     stampaj_racun(&korpe[i]);
    }
161 }

163 /*
    Funkcija prosek racuna prosecnu cenu
165 potrosacke korpe za dati niz potrosackih
    korpi
167 */
    float prosek(KORPA korpe[], int n)
169 {
    int i;
171 float p;

173 for(i=0;i<n;i++)
        p+=izracunaj_racun(&korpe[i]);
175
    return p/n;
177 }

179 int main()
    {
181 int n;
    KORPA korpe[MAXPOT];
183
    printf("Unesi broj potrosackih korpi:");
185 scanf("%d", &n);

187 if(n<0 || n>MAXPOT)
    {
189     printf("Nekorektan unos broja potrosackih korpi: %d\n",n);
        return -1;
191 }

193 if (ucitaj_niz_korpi(korpe, n)==0)
        return -1;
195
    stampaj_racune_za_korpe(korpe,n);
197 printf("Prosecna cena potrosacke korpe: %.2f\n", prosek(korpe, n))
      ;

199 return 0;
    }
```

Rešenje 3.9.5

```
/*
2   Struktura IZRAZ opisuje numericki izraz nad celim brojevima koji
     se sastoji
```

```
   od dva celobrojna operanda, numericke operacije nad celim
    brojevima i
   vrednosti izraza:

   typedef struct izraz
   {
  char o;
  int x;
  int y;
   } IZRAZ;

   a) Napisati funkciju koja ispituje da li je dati izraz korektno
   zadat i vraca 1 ako jeste a 0 u suprotnom. Podrazumevamo da je
   izraz korektno zadat ako operacija odgovara +,-,* ili / i u
    slucaju
   deljenja drugi operand je razlicit od 0.

   b) Napisati funkciju koja za dati izraz odredjuje vrednost izraza.

   c) Napisati funkciju koja ucitava dati izraz. Funkcija
   treba da ucita sa standardnog ulaza operaciju i dva
   operanda u polja o, x i y strukture IZRAZ. Funkcija vraca
   1 ako je ucitavanje bilo uspesno, tj. ako je izraz bio
   korektno zadat ili 0 u suprotnom.

   d) Napisati funkciju koja stampa dati izraz infiksno, u obliku
   x o y = vr. Na primer, za izraz + 4 17 ispis treba
   da bude 4+17=21


   e) Napisati glavni program koji ucitava prirodan broj n<1000 a
    zatim n izraza
   u notaciji
   + 4 17
   - 8 -16
   Program treba da ispise maksimalnu vrednost medju unetim izrazima
    i da ispise one
   izraze cija je vrednost manja od polovine maksimalne vrednosti.

*/

#include <stdio.h>
#define MAX 1000

typedef struct izraz
{
  char o;
  int x;
  int y;
} IZRAZ;


```

```
/*
   Funkcija korektan_izraz vraca 1 ako je izraz korektan a 0
   u suprotnom. Izraz je korektan ukoliko se sastoji od
   aritmetickih operacija +,-,* ili /, i ukoliko je u slucaju
   operacije deljenja drugi operand razlicit od nule.
*/
int korektan_izraz(const IZRAZ* izraz)
{
   if(izraz->o!='+' && izraz->o!='-' && izraz->o!='*' && izraz->o!='/
    ')
   {
      printf("Nedozvoljena operacija!\n");
      return 0;
   }
   if(izraz->o=='/' && izraz->y==0)
   {
      printf("Deljenje nulom!\n");
      return 0;
   }
   return 1;
}

/*
   Promenljiva izraz ce se promeniti u funkciji
   vrednost tako sto ce njenom neinicijalizovanom
   polju vr biti dodeljena vrednost izraza. Zbog
   toga ovu promenljivu funkciji prosledjujemo
   po adresi, preko pokazivaca
*/

int vrednost(const IZRAZ* izraz)
{
   int v;

   switch (izraz->o)
   {
      case '+':
         v=izraz->x+izraz->y;
         break;
      case '-':
         v=izraz->x-izraz->y;
         break;
      case '*':
         v=izraz->x*izraz->y;
         break;
      case '/':
         v=izraz->x/izraz->y;
         break;
   }
   return v;
}
```

```
102
/*
104    Promenljiva izraz ce se promeniti u funkciji
    ucitaj_izraz tako sto ce njenim neinicijalizovanim
106    poljima o,x,y biti dodeljene vrednosti ucitane
    sa standardnog ulaza. Zbog toga ovu promenljivu
108    funkciji prosledjujemo po adresi, preko pokazivaca.

110    S obzirom da ucitavanje karaktera nije prvo
    ucitavanje koje se obavlja u programu, funkcijom
112    getchar() "pokupimo" karakter kojim razdvajamo
    unos karaktera od prethodnog unosa (najcesce blanko
114    znak)

116 */

118 int ucitaj_izraz(IZRAZ* izraz)
{
120    getchar();
    scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
122    if (!korektan_izraz(izraz))
       return 0;
124    return 1;
}
126

128 void stampaj_izraz(const IZRAZ* izraz)
{
130    printf("%d %c %d = %d\n", izraz->x, izraz->o, izraz->y, vrednost(
     izraz));
}
132
int max_vr(IZRAZ izrazi[], int n)
134 {
    int i;
136    int max;
    /* Trazimo maksimalnu vrednost izraza */
138    max=vrednost(&izrazi[0]);

140    /* U petlji... */
    for(i=1; i<n; i++)
142    /* Ako je ona veca od maksimalne: */
       if(vrednost(&izrazi[i])>max)
144       /* Azuriramo max: */
          max=vrednost(&izrazi[i]);
146    return max;
}
148
int main()
150 {
    int n;
152    IZRAZ izrazi[MAX];
```

```
    int max;
154 int i;

156 /* Ucitavamo broj izraza: */
    scanf("%d", &n);
158 if(n<0 || n>MAX)
    {
160    printf("Nekorektna vrednost broja n!\n");
       return -1;
162 }


164
     /* U petlji ucitavamo jedan po jedan izraz: */
166 for(i=0; i<n; i++)
       if(ucitaj_izraz(&izrazi[i])==0)
168    {
          printf("Nekorektan unos\n");
170       return -1;
       }
172

174 printf("Svi izrazi:\n");
    for(i=0; i<n; i++)
176       stampaj_izraz(&izrazi[i]);

178 max = max_vr(izrazi, n);
    printf("Maksimalna vrednost izraza:%d\n", max);
180
    printf("Izrazi cija je vrednost manja od polovine maksimalne
     vrednosti:\n");
182
    for(i=0; i<n; i++)
184    if(vrednost(&izrazi[i])<max/2)/* Ako je vrednost tekuceg izraza
      manja od polovine maksimalne, ispisujemo ga. */
          stampaj_izraz(&izrazi[i]);
186
    return 0;
188 }
```

Rešenje 3.9.6

```
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;
7   float im;
  } Complex;
9
```

```
/* Funkcija kojom se izracunava zbir kompleksnih brojeva */
Complex saberi(Complex *a, Complex *b) {

  Complex c;
  c.re = a->re + b->re;
  c.im = a->im + b->im;
  return c;
}

/* Funkcija kojom se izracunava razlika kompleksnih brojeva */
Complex oduzmi(Complex *a, Complex *b) {

  Complex c;
  c.re = a->re - b->re;
  c.im = a->im - b->im;
  return c;
}

/* Funkcija kojom se izracunava proizvod kompleksnih brojeva */
Complex pomnozi(Complex *a, Complex *b) {

  Complex c;
  c.re = a->re * b->re - a->im * b->im;
  c.im = b->re * a->im + a->re * b->im;
  return c;
}

/* Funkcija kojom se izracunava kolicnik kompleksnih brojeva */
Complex podeli(Complex *a, Complex *b) {

  Complex c;
  c.re = (a->re * b->re + a->im * b->im) / (b->re*b->re + b->im*b->im
    );
  c.im = (b->re * a->im - a->re * b->im) / (b->re*b->re + b->im*b->im
    );
  return c;
}

int main() {

  Complex a, b;
  Complex c;

  /* Ucitavamo kompleksne brojeve */
  printf("Unesite realni i imaginarni deo prvog broja: ");
  scanf("%f%f", &a.re, &a.im);

  printf("Unesite realni i imaginarni deo drugog broja: ");
  scanf("%f%f", &b.re, &b.im);

  c = saberi(&a, &b);
  printf("Zbir: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im);
```

```
    /* Ukoliko je imaginarni deo negativan,
                                    njegov zapis vec ukljucuje znak,
                                    te to treba proveriti.
                                    Inace, broj je oblika a+b*i
                                  */

  c = oduzmi(&a, &b);
  printf("Razlika: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im)
    ;

  c = pomnozi(&a, &b);
  printf("Proizvod: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im
    );

  if(b.re != 0 || b.im != 0) {
    c = podeli(&a, &b);
    printf("Kolicnik: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.
    im);
  }
  else
    printf("Kolicnik ne postoji.\n"); /* Ni u polju kompleksnih
    brojeva
                    nije dozvoljeno deljenje nulom
                     */


  return 0;
}
```

Rešenje 3.9.7

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

#define MAX 50

typedef struct lopta {
  int poluprecnik;
  enum {plava, zuta, crvena, zelena} boja; /* tip "boja" je
    nabrajajuci tip,
                        a efekat nabrajanja mogucih vrednosti
                        {plava, zuta, zelena, crvena}
                        ekvivalentan je definisanju
                        4 celobrojne konstante direktivom #define
                      */
} LOPTA;

/*  Funkcija koja odredjuje zapreminu lopte */
float zapremina(LOPTA* l) {
 return pow(l->poluprecnik, 3)*4/3*M_PI;
}
```

```
21 /* Pomocna funkcija koja racuna zapreminu svih lopti */
   float ukupna_zapremina(LOPTA lopte[], int n) {
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;
     int i;
41   for(i = 0; i < n; i++)
       if(lopte[i].boja == boja)
43       br++;
     return br;
45 }

47 int main() {

49   LOPTA lopte[MAX];
     int n;
51   int i;
     int boja;
53
     printf("Unesite broj lopti: ");
55   scanf("%d", &n);

57   if(n < 1 || n > MAX) {

59     printf("Nekorektan unos.\n");
       return 0;
61   }

63   printf("Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta,
       3-crvena, 4-zelena):\n");
     for(i = 0; i < n; i++) {
65
       printf("%d. lopta: ", i+1);
67     scanf("%d%d", &lopte[i].poluprecnik, &boja);
```

```
69    /* U zavisnosti od unetog
         celog broja,
71       bira se boja lopte.
      */
73      switch(boja) {

75        case 1: lopte[i].boja = plava; break;
          case 2: lopte[i].boja = zuta; break;
77        case 3: lopte[i].boja = crvena; break;
          case 4: lopte[i].boja = zelena; break;
79        default:
          printf("Nekorektan unos.\n");
81        return 0;
        }
83    }

85    printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));

87    printf("Ukupno crvenih lopti: %d\n", broj_lopti_u_boji(lopte, n,
        crvena));

89    return 0;
}
```

Rešenje 3.9.8

```
#include <stdio.h>
2 #include <string.h>

4 #define MAX_DUZINA 21
  #define MAX_BR_VOCKI 50
6
  typedef struct vocka
8 {
    char ime[MAX_DUZINA];
10  float vitamin;
  } VOCKA;
12


14 int main()
  {
16  VOCKA vocke[MAX_BR_VOCKI];
    int i = 0, n, max_vocka;
18  char ime[MAX_DUZINA];

20  /*
      Ucitavamo podatke o vockama i smestamo ih u niz
22    sve dok ne unesemo rec KRAJ ili ucitamo MAX_BR_VOCKI vocki.
    */
24  do
    {
```

```
    printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
    scanf("%s",ime);
    /*
      Kada unesemo rec KRAJ prekidamo petlju.
    */
    if(strcmp(ime, "KRAJ") == 0)
      break;

    /*
      Inace ucitavamo i kolicinu vitamina
      i tu vrednost smestamo u vocku na poziciji i
    */
    strcpy(vocke[i].ime,ime);
    scanf("%f",&vocke[i].vitamin);
    i++;
  }
  while(i<MAX_BR_VOCKI);

  n = i;

  /*
    Pretpostavicemo da prva vocka ima najvise vitamina.
    Procicemo kroz niz vocki i ukoliko naidjemo na vocku koja ima
    vise vitamina
    od one koja trenutno ima najvise, azuriracemo vrednosti
    maksimalne vocke.

    Sve vreme cuvamo indeks vocke sa najvise vitamina C.
  */

  max_vocka = 0;
  for(i=1;i<n;i++)
    if(vocke[i].vitamin > vocke[max_vocka].vitamin)
    {
      max_vocka = i;
    }

  printf("Voce sa najvise C vitamina je: %s\n", vocke[max_vocka].ime)
    ;

  return 0;
}
```

Rešenje 3.9.9

```
#include <stdio.h>
#include <string.h>

#define MAX_IME_PREZIME 51
#define MAX_ZELJA 101
#define MAX_BR_STUDENATA 100
```

```
8  typedef struct student
   {
10   char imeipre[MAX_IME_PREZIME];
     char zelja[MAX_ZELJA];
12 } STUDENT;

14
   int main()
16 {
     STUDENT studenti[MAX_BR_STUDENATA];
18   char odgovor[3];
     char imeiprezime[MAX_IME_PREZIME];
20   int i = 0, n;
     int broj_pronalazaka;
22

24   /*
       Ucitavamo studente i njihove zelje i upisujemo ih u niz
26      sve dok to zelimo ili dok ne unesemo MAX_BR_STUDENATA studenata.
     */
28   while(i < MAX_BR_STUDENATA)
     {
30     printf("Ime i prezime studenta: \n");
       /*
32       Funkcija fgets ucitava jednu liniju i smesta je promenljivu
     koju zadajemo kao njen prvi argument.
       Drugi argument je maksimalna duzina linije.
34     Treci argument je kod nas stdin sto predstavlja standardni ulaz
     .
       */
36
       if(fgets(studenti[i].imeipre, MAX_IME_PREZIME, stdin) == NULL)
38     {
         printf("Greska: nismo dobro ucitali ime i prezime studenta.");
40       return 0;
       }
42
       printf("Njegova zelja: \n");
44
       if(fgets(studenti[i].zelja, MAX_ZELJA, stdin) == NULL)
46     {
           printf("Greska: nismo dobro ucitali zelju studenta.");
48         return 0;
       }
50
       i++;
52
       printf("Jos vrednih studenta (da/ne)?\n");
54
       scanf("%s", odgovor);
56
```

```
    /*
      Moramo da pokupimo karakter koji unesemo nakon odgovora
      kako ga ne bismo ucitali u sledecoj iteraciji petlje.
    */
    getchar();
    /*
      Ukoliko je nas odgovor "ne" prekidamo petlju.
      A ukoliko nije ni "da" ni "ne" onda nismo dobro uneli odgovor.
    */
    if(strcmp(odgovor,"ne") == 0)
      break;
    else if(strcmp(odgovor,"da") != 0)
    {
      printf("Greska: odgovor mora biti u obliku (da/ne)!");
      return 0;
    }

  }

  /*
    Postavljamo dimenziju niza.
  */
  n = i;
  printf("Za podsecanje uneti ime i prezime: \n");
  if(fgets(imeiprezime, MAX_IME_PREZIME, stdin) == NULL)
  {
    printf("Greska: nismo dobro ucitali ime i prezime studenta.");
    return -1;
  }

  /* Prolazimo kroz listu studenta i ispisujemo zelje studenta cije
     ime i prezime smo uneli. */
  broj_pronalazaka=0;
  for(i=0;i<n;i++){
    if(strcmp(imeiprezime, studenti[i].imeipre) == 0){
      broj_pronalazaka++;
      printf("Novogodisnja zelja: %s\n",studenti[i].zelja);
    }
  }

  /* Za slucaj da nismo pronasli studenta sa trazenim imenom */
  if(broj_pronalazaka==0){
    printf("Trazeni student ne postoji - mozda ce mu poklon odneti
    drugi Deda Mraz\n");
  }

  return 0;
}
```

Rešenje 3.9.10

```c
/*
  Definisati strukturu Grad u kojoj se nalazi ime grada (niska duzine
     20 karaktera) i prosecna temperatura u
  toku decembra (realan broj). Napisati program koji ucitava imena n
    (0<n<50) gradova i njihove prosecne
  temperature, a zatim ispisuje one gradove koji imaju idealnu
    temperaturu za klizanje: od 3 do 8 stepeni.
  Napomena: probati sa testiranjem zadataka pomocu preusmeravanja.
*/

#include <stdio.h>
#define MAX_DUZINA 20
#define MAX_BR_GRADOVA 50

typedef struct Grad{
  char ime_grada[MAX_DUZINA+1];
  float temperatura;
}Grad;


int main(){
  int n, i;
  Grad grad[MAX_BR_GRADOVA];

  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<0 || n>MAX_BR_GRADOVA){
    printf("Greska: pogresan unos!\n");
    return 0;
  }

  for(i=0; i<n; i++){
    printf("Unesite grad i temperaturu: ");
    scanf("%s %f", grad[i].ime_grada, &grad[i].temperatura);
  }

  printf("Gradovi sa idealnom temperaturom za klizanje u decembru:\n"
    );
  for(i=0; i<n; i++){
    if(grad[i].temperatura>=3 && grad[i].temperatura<=8){
      printf("%s\n", grad[i].ime_grada);
    }
  }

  return 0;
}
```

Rešenje 3.9.11

```c
/*
```

```c
  Definisati strukturu ParReci koja sadrzi rec na srpskom jeziku i
    odgovarajuci prevod na engleski jezik. Zatim
  sa standardnog ulaza sve do kraja ulaza ucitavati parove reci i,
    posebno, za recenicu koja se zadaje sa ulaza
  ispisati prevod - ako je rec u recenici nepoznata umesto nje
    ispisati odgovarajuci broj zvezdica. Reci nece biti
  duze od 50 karaktera, a ukupan broj reci nece biti veci od 100.
    Napomena: probati sa testiranjem zadataka
  pomocu preusmeravanja.
*/

#include <stdio.h>
#include <string.h>
#define MAX_DUZINA 20
#define MAX_BR_RECI 100
#define MAX_DUZINA_RECENICE 100

typedef struct ParReci{
  char sr[MAX_DUZINA+1];
  char en[MAX_DUZINA+1];
}ParReci;


/*
  Funkcija koja u recniku koji sadrzi n reci trazi prevod reci rec i
    upisuje ga u prevod.
  Ukoliko se rec ne nalazi u recniku, prevod se sastoji od zvezdica
    pri cemu broj zvezdica odgovara
  duzini nepoznate reci.
*/

void pronadji_prevod(ParReci recnik[], int n, char rec[], char prevod
    []){
  int i;

  /* Pretrazujemo recnik i trazimo zadatu rec */
  for(i=0; i<n; i++){

    /* Ukoliko se rec nalazi u recniku */
    if(strcmp(recnik[i].sr, rec)==0){
      /* Ocitavamo njen prevod */
      strcpy(prevod, recnik[i].en);
      /* I obustavljamo pretragu */
      return;
    }
  }

  /* Ukoliko rec nije pronadjena, formiramo prevod reci koji se
    sastoji od zvezdica */
  for(i=0; rec[i]; i++){
    prevod[i]='*';
  }
```

```
  prevod[i]='\0';
}


int main(){
  ParReci recnik[MAX_BR_RECI];
  int n;
  char sr[MAX_DUZINA+1];
  char en[MAX_DUZINA+1];
  int i, j, k;
  char recenica[MAX_DUZINA_RECENICE+1];
  char rec[MAX_DUZINA+1];
  char prevod[MAX_DUZINA+1];
  int citamo_rec;
  char* novi_red;

  /* Ucitavamo parove reci sa standardnog ulaza sve do kraja ulaza*/
  i=0;
  while(scanf("%s %s", sr, en)!=EOF){
    if(i==MAX_BR_RECI)
      break;

    strcpy(recnik[i].sr, sr);
    strcpy(recnik[i].en, en);

    i++;
  }
  /* Broj parova reci cuvamo u promenljivoj n */
  n=i;

  /* Ucitavamo recenicu - nisku karaktera sve do pojave znaka za novi
     red */
  printf("Unesite recenicu za prevod:\n");
  fgets(recenica, MAX_DUZINA_RECENICE, stdin);

  /* Ako postoji, zamenjujemo znak za novi red terminirajucom nulom
    */
  novi_red=strchr(recenica, '\n');
  if(novi_red!=NULL)
    *novi_red='\0';


  /* Izdvajamo rec po rec unesene recenice */
  /* j oznacava tekuci karakter recenice koji se obradjuje */
  j=0;
  /* citamo_rec sa mogucim vrednostima 1 i 0 ce biti indikator koji
    pokazuje da li citamo rec ili ne */
  citamo_rec=0;

  while(1){
    /* 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) */
        if(recenica[j]!=' ' && recenica[j]!='\n' && recenica[j]!='\t'){
99          /* Smestamo ga u rec */
            if(citamo_rec==0){
101           citamo_rec=1;
              k=0;
103         }
            rec[k]=recenica[j];
105         k++;
        }
107     else{
          /* Inace, procitali smo karakter koji ne treba ukljuciti u rec
        */
109       /* Ako smo pre toga citali rec */
          if(citamo_rec==1){
111         /* Prekidamo citanje */
            citamo_rec=0;
113         rec[k]='\0';

115         /* I trazimo i ispisujemo odgovarajuci prevod reci */
            pronadji_prevod(recnik, n, rec, prevod);
117         printf("%s ", prevod);
          }
119     }

121     /* Prelazimo na sledeci karakter recenice */
        j++;
123   }

125   /* Za slucaj da nije obradjena, obradjujemo poslednju rec i
        ispisujemo njen prevod */
      if(citamo_rec){
127     rec[k]='\0';
        pronadji_prevod(recnik, n, rec, prevod);
129     printf("%s\n", prevod);
      }
131

133   return 0;

135 }
```

Rešenje 3.9.17

Rešenje 3.9.17

Rešenje 3.9.17

Rešenje 3.9.17

Rešenje 3.9.17

Rešenje 3.9.17

4

Ulaz i izlaz programa

4.1 Datoteke

Zadatak 4.1.1 Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u datoteku izlaz.txt karakter po karakter.

[Rešenje 4.1.1]

Zadatak 4.1.2 Napisati program koji u datoteci cije se ime navodi kao prvi argument komandne linije odredjuje liniju maksimalne duzine i ispisuje je na standarni izlaz. Ukoliko ima vise takvih linija , ispisati onu koja je leksikografski prva. Mozemo pretpostaviti da datoteka ne sadrzi linije duze od 80 karaktera.

[Rešenje 4.1.2]

Zadatak 4.1.3 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 koliko k-tocifrenih brojeva postoji u datoteci , pri cemu se prirodan broj k zadaje kao drugi argument komandne linije.

[Rešenje 4.1.3]

Zadatak 4.1.4 U datoteci cije se ime navodi kao prvi argument komandne linije navedena je rec r i niz linija. Napisati program koji u datoteku cije se ime navodi kao drugi argument komandne linije upisuje sve linije u kojima se rec 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.

[Rešenje 4.1.4]

Zadatak 4.1.5 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: 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 iz datoteke , a potom:
a) ukoliko je navedena opcija x, upisati u datoteku cije je ime drugi argument komandne linije reciprocni 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 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 najvise 100 razlomaka.

[Rešenje 4.1.5]

Zadatak 4.1.6 Za svaki automobil poznati su marka , model i cena. Iz datoteke cije se ime zadaje sa standardnog ulaza ucitava se broj automobila a potom i podaci za svaki automobil. Program treba da:
a) izracuna prosecnu cenu po marki kola
b) za maksimalnu cenu koju je kupac spreman da plati , a koja se zadaje kao argument komandne linije, da ispise automobile u tom cenovnom rangu zajednu sa prosecnom cenom odgovarajuce marke

Mozemo pretpostaviti da se model i marka sastoje od jedne reci i da svaka od njih sadrzi najvise 30 karaktera kao i da se u datoteci nalaze podaci za najvise 100 automobila.

[Rešenje 4.1.6]

Zadatak 4.1.7 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 4.1.7]

Zadatak 4.1.8 Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*.

Primer 1

```
ULAZ.TXT
 Volim programiranje.
IZLAZ.TXT
 Vipgmae
```

[Rešenje 4.1.8]

Zadatak 4.1.9 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 obradjivane:
 Petlje
 Funkcije
 Nizovi
 Strukture

IZLAZ:
 Teme koje su obradjivane:
 Funkcije
 Strukture
```

Primer 2

```
POKRETANJE: ./a.out test.txt

IZLAZ:
 Greska: Pogresan broj argumenata!
```

[Rešenje 4.1.9]

Zadatak 4.1.10 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 4.1.10]

Zadatak 4.1.11 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 4.1.11]

Zadatak 4.1.12 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
  mi14005 8 8 9 8 10
  ml15112 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 4.1.12]

Zadatak 4.1.13 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 4.1.13]

Zadatak 4.1.14 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 4.1.14]

Zadatak 4.1.15 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 karaktera.

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 4.1.15]

Zadatak 4.1.16 Sa standarnog 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 ucitan karakter `u` program prilikom prepisivanja treba da zamenjuje sva mala slova velikim, a ukoliko je učitan karakter `l` sva velika slova se zamenjuju malim. U slučaju greske ispisati -1. Greška može biti neuspešno otvaranje datoteke ili pogrešno zadat karakter. Maksimalna dužina naziva datoteka 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 4.1.33]

Zadatak 4.1.17 Sastaviti program koji sa standardnog ulaza prima ime datoteke koju treba otvoriti. Ispisati (na standardnom izlazu) koja cifra (među svim 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 4.1.33]

Zadatak 4.1.18 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 4.1.33]

Zadatak 4.1.19 Napisati program koji za dve datoteke čija su imena data kao prvi i drugo na standarnom 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

     11112111121120021111211112112000211112111111112112000211112111111112112000
```

[Rešenje 4.1.33]

Zadatak 4.1.20 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 4.1.33]

Zadatak 4.1.21 Sa standarnog ulaza se učitava ime tekstualne datoteke i prirodan broj k. Podrazumeva se da zadata datoteka sadrži samo slova i beline i 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 4.1.33]

Zadatak 4.1.22 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 neoznacen
 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-primera sa
 i dugackim recima kao prestolonaslednik
 brojevima1234567890
IZLAZ.TXT
 konstruisanje test-primera
 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 4.1.33]

Zadatak 4.1.23 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 izmeų 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 4.1.33]

Zadatak 4.1.24 U datoteci čije se ime navodi na standarnom 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 4.1.33]

Zadatak 4.1.25 U datoteci čije se ime navodi na standarnom 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 podvlaku. [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 4.1.33]

Zadatak 4.1.26 Imena dve datoteke se zadaje na standarnom 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 4.1.33]

Zadatak 4.1.27 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 pov cevši 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 4.1.33]

Zadatak 4.1.28 Definisati 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 medju pravougaonicima koji nisu kvadrati. U slučaju unosa nekorektnih dužina stranica pravougaonika ili nekorektne 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 4.1.33]

Zadatak 4.1.29 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 4.1.33]

Zadatak 4.1.30 Napraviti strukturu `STUDENT` koja sadrži:

- `ime` (u polju se čuva ime i prezime studenta, npr. ”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 koristiti `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 4.1.33]

Zadatak 4.1.31

a) 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 naiẹ na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspesno učitani.

b) Napisati funkciju `void prebroj(char *s, int *br_slova,int *br_cifara)` kojom se odreụje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.

c) Napisati glavni program gde se unose podaci o skupu elemenata. Ime datoteke se zadaje kao argument komandne linije. Na stadardni 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 4.1.33]

Zadatak 4.1.32 Definisati 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 4.1.33]

Zadatak 4.1.33 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 4.1.33]

4.2 Rešenja

Rešenje 4.1.1

```c
/*
    Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
    datoteku izlaz.txt karakter po karakter.
*/

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

int main()
{
   int c;
   FILE *ulaz, *izlaz;

   /*
      Promenljive ulaz i izlaz predstavljaju
      pokazivace na ugradjenu strukturu FILE.
      Unutar ove strukture nalaze se polja neophodna
      za rad sa datotekama.

      Kada zelimo da radimo sa nekom datotekom,
```

```
21      moramo je prvo otvoriti. Ugradjena funkcija
        fopen(dat, mode) otvara datoteku sa nazivom
23      dat. Datoteka moze biti otvorena za citanje,
        pisanje ili nadovezivanje, sto odredjuje
25      argument mode koji moze imati vrednost "r" (read),
        "w"(write) ili "a"(append).
27  */

29  ulaz=fopen("ulaz.txt","r");

31  /*
        Do neuspesnog otvaranja datoteke moze doci
33      ukoliko ne postoji datoteka sa datim nazivom
        ili je putanja do datoteke pogresna. U tom
35      slucaju, funkcija fopen vraca pokazivac na NULL
        i tada treba prijaviti gresku. Datoteka stderr
37      predstavlja standardnu datoteku u koju se upisuju
        greske. Stderr je podrazumevano postavljen
39      na standardni izlaz.

41      Ugradjena funkcija exit prouzrokuje zavrsetak programa.
        Argument ove funkcije je jedna od konstanti definisanih
43      u biblioteci stdlib.h koje pokazuju da li se program
        zavrsio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).
45
47  */
    if(ulaz==NULL)
    {
49      fprintf(stderr,"error fopen(): Neuspelo otvoranje datoteke ulaz
     .txt za citanje.\n");
        exit(EXIT_FAILURE);
51  }

53  izlaz= fopen("izlaz.txt", "w");
    if(izlaz==NULL)
55  {
        fprintf(stderr,"error fopen():  Neuspelo otvoranje datoteke
     izlaz.txt za citanje.\n");
57      exit(EXIT_FAILURE);
    }
59

    /*
61      Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
        Povratna vrednost ove funkcije je ascii kod unetog
63      karaktera.

65      Funkcija fputc ispisuje karakter c u datoteku izlaz.
    */
67

    while((c=fgetc(ulaz))!=EOF)
69      fputc(c,izlaz);
```

```
71    /*
          Nakon zavrsetka rada sa datotekama, neophodno ih je
73        zatvoriti pomocu ugradjene funkcije fclose.
      */
75    fclose(ulaz);
      fclose(izlaz);
77    return 0;
}
```

Rešenje 4.1.2

```
/*
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];
16    char max_linija[MAX_LEN];
      int duzina;
18    int max_duzina;

20    FILE *ulaz, *izlaz;

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]);
28       exit(EXIT_FAILURE);
      }
30
      ulaz=fopen(argv[1],"r");
32    if(ulaz==NULL)
      {
34       fprintf(stderr,"error fopen(): Neuspelo otvoranje 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 ucitavanje 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 ucitana linije
52          veca od max_duzina ili kada su jednake, ali je ucitana 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      */
62      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 ucitavanje 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);

78      fclose(ulaz);
        return 0;
80  }
```

Rešenje 4.1.3

```
/*
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>
    #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  */
    int ucitaj_i_prebroj(FILE* f, int k)
18  {
       int n;
20     int x;
       int i;
22     int br;

24     /* U datoteci je prvo naveden ukupan broj brojeva. */
       fscanf(f,"%d",&n);
26
       /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
        */
28     if(n<=0)
       {
30        fprintf(stderr, "Greska: broj n mora biti prirodan\n");
          exit(EXIT_FAILURE);
32     }

34     br=0;
       for(i=0;i<n;i++)
36     {
          fscanf(f,"%d",&x);
38        if(broj_cifara(x)==k)
             br++;
```

```
40      }
42      return br;
}
44
int broj_cifara(int x)
46 {
      int br_c;
48
      br_c=0;
50
      /*
52       Do while petlja je pogodnija od petlji sa preduslovom
         jer tacno racuna broj cifara i za broj 0.
54    */
      do
56    {
        br_c++;
58      x/=10;
      } while(x);
60
      return br_c;
62 }
64 int main(int argc, char* argv[])
{
66    int n;
      int k;
68    FILE* f;
      int br;
70
      if(argc!=3)
72    {
         fprintf(stderr,"Greska: program se pokrece sa: %s
       naziv_datoteke k \n", argv[0]);
74       exit(EXIT_FAILURE);
      }
76
      f=fopen(argv[1], "r");
78
      if(f==NULL)
80    {
         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
       .\n", argv[1]);
82       exit(EXIT_FAILURE);
      }
84
      /* Argumenti komandne linije su stringovi. Da bismo konvertovali
       string
86       u ceo broj koristimo ugradjenu funkciju atoi. */
      k = atoi(argv[2]);
88
```

```
    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 4.1.4

```
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>
   #include <stdlib.h>
13 #define MAXL 81
   #define MAXR 31
15
   /*
17     Funkcija broj_pojavljivanja broji koliko
       se puta pojavio string t u stringu s
19 */
   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
            je prestao da vazi uslov petlje, odnosno karakter t[j] je
39          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];
     char linija[MAXL];
53   FILE* in, *out;
     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] );
67      exit(EXIT_FAILURE);
     }
69
     out= fopen(argv[2],"w");
71   if(out==NULL)
     {
73      fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
      .\n", argv[2] );
        exit(EXIT_FAILURE);
75   }

77   printf("Unesi n:");
     scanf("%d", &n);
79
     if(n<=0)
81   {
        fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
83      exit(EXIT_FAILURE);
     }
```

```
85
    fscanf(in,"%s",rec);
87
    while(fgets(linija,MAXL,in)!=NULL)
89  {
       br = broj_pojavljivanja(linija,rec);
91     if (br>=n)
          fprintf(out,"%d: %s\n", br, linija);
93  }
    fclose(in);
95  fclose(out);
    return 0;
97  }
```

Rešenje 4.1.5

```
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           reciprocni 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 najvise 100 razlomaka.
11 */

13 /*
     Prilikom pokretanja programa se, pored naziva ulazne i izlazne
15   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
   */
25
   #include <stdio.h>
```

```c
#include <stdlib.h>
#include <ctype.h>

#define MAX 100

typedef struct razlomak
{
  int br;
  int im;
} RAZLOMAK;

/*
   Funkcija ucitaj_razlomke ucitava razlomke iz datoteke
   na koju pokazuje f u niz. Dimenzija niza, na koju
   pokazuje pokazivac dim, nije poznata. Prva vrednost
   u datoteci je ukupan broj razlomaka i tu vrednost
   ucitavamo u promenljivu dim.

   Funkcija fscanf se koristi isto kao i funkcija scanf
   uz dodatni prvi argument koji predstavlja naziv
   datoteke iz koje se vrsi ucitavanje.

*/
int ucitaj_razlomke(RAZLOMAK niz[], int* dim, FILE* f)
{
   int i;

   fscanf(f,"%d", dim);
   for (i=0; i<*dim; i++)
   {
      fscanf(f,"%d %d", &niz[i].br, &niz[i].im);
      if (niz[i].im==0)
         return 0;
   }
   return 1;
}

RAZLOMAK reciprocni(RAZLOMAK* r)
{
   RAZLOMAK rec;
   rec.im = r->br;
   rec.br = r->im;
   return rec;
}

float vrednost(RAZLOMAK* r)
{
   return 1.0*r->br/r->im;
}

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
        for(j=1;argv[3][j]!='\0';j++)
135         switch(argv[3][j])
            {
137             case 'x': xoption=1;
                          break;
139             case 'y': yoption=1;
                          break;
141             default:
                          fprintf(stderr, "Greska: nedozvoljeni karakter\n"
       );
143                       exit(EXIT_FAILURE);
            }
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.

157         Funkcija fprintf se koristi na isti nacin kao
            funkcija printf uz dodatni prvi argument koji
159         oznacava naziv datoteke u koju se vrsi 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)
               fprintf(out, "%f ", vrednost(&r));
177
            fprintf(out,"\n");
179     }

181     fclose(in);
        fclose(out);
183
        return 0;
185 }
```

Rešenje 4.1.6

```
1  /*
     Za svaki automobil poznati su marka, model i cena. Iz datoteke cije
3    se ime zadaje sa standardnog ulaza ucitava se broj automobila a
       potom
     i podaci za svaki automobil. Program treba da:
5    a) izracuna prosecnu cenu po marki kola
     b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
       zadaje
7    kao argument komandne linije, da ispise automobile u tom cenovnom
     rangu zajednu sa prosecnom cenom odgovarajuce marke
9
     Mozemo pretpostaviti da se model i marka sastoje od jedne reci i
11   da svaka od njih sadrzi najvise 30 karaktera kao i da se u datoteci
     nalaze podaci za najvise 100 automobila.
13
   */
15
   #include <stdio.h>
17 #include <stdlib.h>
   #include <string.h>
19 #define MAX 31
   #define MAXA 100
21
   typedef struct automobil
23 {
      char marka[MAX];
25    char model[MAX];
      float cena;
27 } AUTOMOBIL;

29 /*
      Struktura INFO sadrzi naziv
31    marke automobila, prosek cena
      za tu marku i broj automobila
33    te marke
   */
35 typedef struct info
```

```
{
    char marka[MAX];
    float vrednost;
    int n;
} INFO;

int ucitaj_podatke(FILE* f, AUTOMOBIL a[], int* pn, int max)
{
    int i;

    fscanf(f,"%d", pn);
    if (*pn<=0 || *pn>max)
    {
        printf("Nekorektan unos dimenzije niza automobila\n");
        return 0;
    }
    for(i=0;i<*pn;i++)
        fscanf(f,"%s %s %f", a[i].marka, a[i].model, &a[i].cena);

    return 1;
}

/*
    Funkcija sadrzi ispituje da li se u nizu proseka po marki
    nalazi prosek za marku m. Posto podatak o marki automobila
    predstavlja string, poredjenje vrsimo pomocu funkcije strcmp.

    Povratna vrednost ove funkcije je indeks pojavljivanja, ukoliko
    se marka m pojavljuje u nizu proseka, ili -1 u suprotnom.
*/

int sadrzi(INFO p[], int n, char m[])
{
    int i;
    for(i=0;i<n;i++)
        if(strcmp(p[i].marka,m)==0)
            return i;

    return -1;
}

/*
    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.

    Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
    sumu cena automobila te marke (koju cemo smestiti u polje vrednost
      strukture
    INFO), i broj automobila te marke (koju cemo smestiti u polje
    n strukture INFO) i da na kraju podelimo ove dve promenljive
    i tako dobijemo prosecnu 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
   vredost cene automobila a[i] i uvecavamo broj automobila sa
91 tom markom. U suprotnom, dodajemo novi element u niz p. Posto
   ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
93 na koju pokazuje pokazivac *pn.
*/
95 void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
{
97    int i,j;
   int ind;
99    for(i=0;i<n;i++)
   {
101      /* Proveravamo da li se marka automobila a[i] vec nalazi u
            nizu p (niz proseka po markama) */
103      ind = sadrzi(p,*pn1,a[i].marka);
         if(ind==-1) /* Ako se ne nalazi, uvodimo novi element niza na
     kraj, na poziciju *pn. */
105      {
            strcpy(p[*pn1].marka, a[i].marka);
107         p[*pn1].vrednost = a[i].cena;
            p[*pn1].n = 1;
109         (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
       */
         }
111      else /* Ako se nalazi, azuriramo polja strukture. */
         {
113         p[ind].vrednost+=a[i].cena;
            p[ind].n++;
115      }
   }
117
   /* Na osnovu sume cena i broja automobila racunamo prosecnu
     vrednost. */
119 for(i=0;i<*pn1;i++)
      p[i].vrednost = p[i].vrednost/p[i].n;
121
}
123
void stampaj_informacije(INFO p[], int n)
125 {
   printf("Informacije o broju automobila i prosecnoj ceni po markama
     :\n");
127 int i;
   for(i=0;i<n;i++)
129    printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
}
131
/*
133 Funkcija stampa automobile cija je cena manja od maksimalne
   cene koju je korisnik naveo u komandnoj liniji da je spreman
```

```
135    da plati, zajedno sa prosecnom cenom za tu marku automobila
   */
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
       n1)
   {
139    /*
          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
          nije neophodno proveravati da li je povratna vrednost funkcije
143       sadrzi razlicita od -1.
       */
145    int i;
       printf("Kola u vasem cenovnom rangu:\n");
147    for(i=0;i<n;i++)
          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);
   }
151
   int main(int argc, char* argv[])
153 {
       AUTOMOBIL kola[MAXA];
155    FILE* f;
       char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
        ulaza. */
157    float granica; /* Maksimalna cena koju je korisnik spreman da
        plati.
                          Zadaje se kao argument komandne linije.
159                    */
       INFO infos[MAXA];
161    int dim_kola,dim_infos;
       int i;
163
       if(argc!=2)
165    {
          fprintf(stderr,"Greska: program se pokrece sa: %s
       gornja_granica_cene \n", argv[0]);
167       exit(EXIT_FAILURE);
       }
169
       /* Argumenti komandne linije su stringovi. Da bismo od stringa
        dobili
171       realan broj, koristimo ugradjenu funkciju atof. */
       granica = atof(argv[1]);
173
       printf("Unesi naziv datoteke:");
175    scanf("%s", dat);

177    f=fopen(dat, "r");

179    if(f==NULL)
       {
```

```
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 ucitavanju 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 4.1.7

```
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

     /*Ispisujemo rezultat*/
25   printf("Broj malih slova je: %d\n", broj_malih);
```

```
27    /*Zatvaramo datoteku*/
      fclose(in);
29
      return 0;
31  }
```

Rešenje 4.1.8

```
/* 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 4.1.9

```
/* 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 veca od k. Moze se
     pretpostaviti da duzina linije nece biti veca
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*/
30    k = atoi(argv[2]);

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*/
  fclose(in);
  return 0;
}
```

Rešenje 4.1.10

```
/* Napisati program koji prebrojava koliko se linija datoteke ulaz.
    txt zavrsava niskom s koja se ucitava sa stan-
dardnog ulaza. Moze se pretpostaviti da duzina linije nece biti veca
    od 80 karaktera, kao i da duzina niske s
ne ce biti veca od 20 karaktera */

#include<stdio.h>
#include <string.h>
#define MAXL 81
#define MAXS 21

/*Funkcija brojLinija proverava koliko linija u datoteci in se
    zavrsava niskom s.
 Funkcija radi tako sto cita jednu po jednu liniju iz datoteke,
 i zatim kraj te linije poredi sa niskom s.*/
int brojLinija(FILE* in, char* s){

  char linija[MAXL];
  int broj_linija = 0;
  int duzina_s = strlen(s);
  int duzina_linije;

  while(fgets(linija, MAXL, in) != NULL){
  duzina_linije = strlen(linija);

  /* Ukoliko je znak za novi red bio indikacija kraja linije,
    uklanjamo ga kako bi mogli da izvrsimo
   *ispravno poredjenje (jer niska s nema novi red na kraju) */
  if(linija[duzina_linije-1]=='\n'){
    linija[duzina_linije-1] = '\0';
    duzina_linije--;
  }

  /*linija + duzina_linije ce nas odvesti na kraj tog stringa, a kada
     oduzmemo duzinu stringa s,
    a kada od toga oduzmemo duzinu niske s, dobicemo bas onoliko
    poslednjih karaktera, koliko
    nam i treba. U primeru uspravna crta (|) oznacava pokazivac
       s              ab
       duzina_s           2
       Linija:            aaabbbdfsssab
                    |
       Linija + duzina linije   aaabbbdfsssab
                           |
```

```
       Linija + duzina linije - 2 aaabbbdfsssab
                              |
      kada kazemo strcmp(linija + duzina_linije - duzina_s, s), mi
    cemo u nasem primeru zaista porediti "ab" i "ab".
   */
  if(strcmp(linija + duzina_linije - duzina_s, s) == 0)
    broj_linija++;
  }
  return broj_linija;
}

int main(){

  FILE* in;
  char s[MAXS];

  /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
    uspesno otvorili*/
  in = fopen("ulaz.txt", "r");
  if(in == NULL){
  printf("Greska: neuspesno otvaranje datoteke!\n");
  return 0;
  }

  /*Ucitavamo nisku*/
  printf("Unesite nisku s: ");
  scanf("%s", s);

  /*Ispisujemo koliko linija iz datoteke se zavrsava sa niskom s*/
  printf("Broj linija: %d\n", brojLinija(in, s));

  /*Zatvaramo datoteku*/
  fclose(in);

  return 0;
}
```

Rešenje 4.1.11

```
/* Napisati program koji pronalazi maksimum brojeva zapisanih u
    datoteci brojevi.txt */

#include<stdio.h>

int main(){

  FILE* in;
  float broj, max_broj;

  /*Otvaramo datoteku brojevi.txt za citanje i proveravamo da li smo
    je uspesno otvorili*/
```

```
  in = fopen("brojevi.txt", "r");
  if(in == NULL){
  printf("Greska pri otvaranju datoteke!");
  return 0;
  }

  /*
   Kako bismo inicijalizovali promenljivu max_broj,
   citamo jedan broj iz datoteke i smestamo ga u
   ovu promenljivu. */
  fscanf(in, "%f", &max_broj);

  /*U petlji citamo sve ostale brojeve i poredimo ih sa trenutnim
    maksimumom.*/
  while(fscanf(in, "%f", &broj) != EOF){
  if(broj > max_broj)
    max_broj = broj;
  }

  /*Ispisujemo rezultat*/
  printf("Najveci broj je: %.2f\n", max_broj);

  /*Zatvaramo datoteku brojevi.txt*/
  fclose(in);

  return 0;
}
```

Rešenje 4.1.12

```
/* U datoteci studenti. txt se nalaze informacije o studentima: prvo
    broj studenata, a zatim u pojedinacnim linijama
korisnicko 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 nece biti veci od 100. */

#include<stdio.h>

#define MAXS 100

/*Definisemo strukturu za cuvanje studenata*/
typedef struct st{
  char korisnicko_ime[8];
  float prosek;
}STUDENT;

int main(){

  FILE *ulaz;
  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
    Ovde paralelno sa ucitavanjem pronalazimo i studenta sa najvecim
     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 najvecim prosekom.

39  */
   while(fscanf(ulaz, "%s%d%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 4.1.13

Rešenje 4.1.14

```c
/* 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 */

#include<stdio.h>
#include<string.h>

#define MAXS 21

/*Funkcija rotira nisku za jedno mesto u desno.
 Duzina niske n nije obavezan argument. Mogli smo
 i da je racunamo u okviru funkcije, ali kako ce sve niske
 sa kojima radimo biti iste duzine, efikasnije je da jednom
 izracunamo tu duzinu u glavnom programu,
 pa da je prosledjujemo kao argument.*/
void rotiraj_za_1(char* s, int n){
  int i;
  char c = s[0];
  for(i=0; i<n-1; i++){
  s[i] = s[i+1];
  }
  s[n-1] = c;
}

int main(){

  char s[MAXS];
  int n, i;
  FILE * izlaz;

  /*Otvaramo datoteku rotacije.txt za pisanje i proveravamo da li smo
     je uspesno otvorili*/
  izlaz = fopen("rotacije.txt", "w");
  if(izlaz == NULL){
  printf("Greska pri otvaranju fajla!");
  return 0;
  }

  /*Sa standardnog ulaza ucitavamo rec koju treba da rotiramo*/
  scanf("%s", s);

  /*Racunamo njenu duzinu*/
  n = strlen(s);

  /*U petlji, ispisujemo tu rec u datoteku, pa je rotiramo za jedno
    mesto u desno.*/
  for(i=0; i<n; i++){
  fprintf(izlaz, "%s\n", s);
  rotiraj_za_1(s,n);
  }
```

```
49    /*Zatvaramo datoteku rotacije.txt*/
      fclose(izlaz);
51
      return 0;
53
}
```

Rešenje 4.1.15

```
/* 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,
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.
*/

#include<stdio.h>
#include<string.h>
#include<ctype.h>

#define MAXL 81

int main(int argc, char* argv[]){

  char linija[MAXL];
  FILE* izlaz;

  /*Indikatori koji oznacavaju koja opcija je navedena kao argument
    komandne linije
  vind - ispisuju se recenice koje pocinju velikim slovom
  mind - ispisuju se recenice koje pocinju malim slovom
   */
  int vind=0, mind = 0;

  /*Proveravamo da li je program ispravno pozvan*/
  if(argc > 2){
  printf("Greska pri pozivanju programa!\n");
  return 0;
  }

  /*Ako opcije nisu zadate, onda treba da se ispisuju sve recenice,
    pa postavljamo oba indikatora na 1*/
  if(argc == 1){
  vind = mind = 1;
  }else{

  /*Proveravamo da li je postavljena neka od opcija -v,-V,-m, -M
   Ako jeste, postavljamo odgovarajuci indikator
   Ako nije, onda ispisujemo poruku o gresci*/
```

```
  if(strcmp(argv[1], "-v") == 0 || strcmp(argv[1], "-V") == 0)
39    vind = 1;
  else if(strcmp(argv[1], "-m") == 0 || strcmp(argv[1], "-M") == 0)
41    mind = 1;
  else{
43    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 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

5

Razni zadaci

5.1 Rešenja

Dodatak A

Ispitni zadaci

A.1 Testovi/Kolokvijumi

A.1.1 Programiranje 1, i-smer, kolokvijum

Grupa I

Zadatak A.1.1 Napisati URM program koji izračunava funkciju:

$$f(x,y) = \begin{cases} 2x - y & 2x \geq y \\ 3y & \text{inače} \end{cases}$$

[Rešenje A.3.19]

Zadatak A.1.2 Sa standardnog ulaza unose se jedan karakter (**p** ili **n**) i dva pozitivna trocifrena broja. Na osnovu vrednosti unetog karaktera izračunati i ispisati na standardni izlaz:
p - zbir cifara na parnim pozicijama unetih brojeva
n - zbir cifara na neparnim pozicijama unetih brojeva
Cifre se broje sa desne strane, tako da cifri jedinice odgovara pozicija 1.
U slučaju greške (ukoliko karakter nije p ili n ili nisu uneti pozitivni trocifreni brojevi)ispisati -1.

Primer 1

```
Interakcija sa programom:
p 235 645
8
```

Primer 2

```
Interakcija sa programom:
n 567 101
14
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
A 432 543
-1
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
p 102 1234
-1
```

[Rešenje A.3.19]

Zadatak A.1.3 Sa standardnog ulaza učitava se pozitivan ceo broj i ceo broj `i` ($1 \leq i$). Na standardni izlaz ispisati broj koji se dobija kada se ukloni `i`-ta cifra broja. Cifre se broje sa desne strane, tako da cifri jedinice odgovara pozicija 1. Neispravan ulaz je kada se unose negativan broj ili negativna vrednost ili nula za `i` i u tom slučaju na standardni izlaz ispisati -1. Ukoliko broj nema `i`-tu cifru broj ostaje nepromenjen.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
35243 2
3523
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
-14423 1
-1
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
1234 5
1234
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
523156 6
23156
```

[Rešenje A.3.19]

Grupa II

Zadatak A.1.4 Napisati URM program koji izračunava funkciju:

$$f(x, y, z) = 4x + 2y + 3z$$

[Rešenje A.3.19]

Zadatak A.1.5 Korisnik unosi 7 karaktera koji predstavljaju indeks studenta koji je oblika OOGGBBB. OO je oznaka smera i moze biti mi, ma, mr,

ms, mm, mv. GG je oznaka godine upisa. BBB je oznaka broja koji moze biti jednocifren, trocifren ili dvocifren sa vodećim nulama. Na osnovu ovih podataka na standarni izlaz ispisati ime smera kome student pripada i indeks u obliku broj/godina. U slučaju greške (ukoliko OO kao oznaka smera nije ispravna ili ostali karakteri nisu brojevi) ispisati -1. Nazivi smerova su: mi - informatika, ma - astronomija, mr - racunarstvo i informatika, ms - statistika, mm - teorijska matematika, mp - primenjena matematika

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
mi11275
informatika 275/2011
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
mm98005
teorijska matematika 5/1998
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
mo23112
-1
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
ms12001
statistika 1/2012
```

[Rešenje A.3.19]

Zadatak A.1.6 Državna lutrija došla je na ideju o novoj igri na sreću. Ova igra na sreću igra se tako što se izvuče jedan broj od 1000 do 9999, Nagrada koja se dobija ako ste pogodili izvučen broj je proizvod njegovih parnih cifara i samog broja. Vaš zadatak je da na osnovu izučenog broja izračunate nagradu koja se dobija. Kao ulaz sigurno ćete dobiti ispravan broj. Ako broj nema parnih cifara, nagrada je sam taj broj. Na standardni izlaz ispišite nagradu.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
1321
2642
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
3284
210176
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
1111
1111
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
2222
35552
```

Primer 5

```
INTERAKCIJA SA PROGRAMOM:
6031
0
```

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
4321
34568
```

[Rešenje A.3.19]

Grupa III

Zadatak A.1.7 Napisati URM program koji izračunava funkciju:

$$f(x,y,z) = \begin{cases} 2 \cdot x + 2 \cdot y & x \leq z \\ z + 3 & \text{inače} \end{cases}$$

[Rešenje A.3.19]

Zadatak A.1.8 Napisati C program koji sa standardnog ulaza učitava 4 velika slova abedece i nenegativan ceo broj k. Program na standardni izlaz ispisuje 4 karaktera koji se dobijaju cikličkim pomeranjem (u okviru karakterske tabele) unetih karaktera za k mesta unapred. Na primer, karakter A pomeren za 4 mesta unapred postaje E dok karakter Z pomeren za 3 mesta unapred postaje C. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ako neki od unetih karaktera ne predstavlja veliko slovo abecede ili ako je broj k negativan, pretpostaviti da se na ulazu uvek zadaje tačno četiri karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 BABA 3
 EDED
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 DEDA 26
 DEDA
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 ZABC 53
 ABCD
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 PERA -2
 -1
```

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 abcd
 -1
```

[Rešenje A.3.19]

Zadatak A.1.9 Napisati C program koji sa standardnog ulaza učitava dva četvorocifrena, pozitivna, cela broja i proverava da li je broj koji se dobija učešljavanjem unetih brojeva palindrom. Ako uneti brojevi imaju cifre a1 a2 a3 a4 i b1 b2 b3 b4 tada su cifre učešljanog broja a1 b1 a2 b2 a3 b3 a4 b4. Broj je palindrom ako se čita isto sa obe strane. Ukoliko je broj palindrom ispisati na standardni izlaz 1, ukoliko nije tada ispisati 0, a u slučaju neispravnog ulaza ispisati -1, neispravnim ulazom smatraju se negativni brojevi i brojevi sa brojem cifara manjim ili većim od 4.

```
Primer 1:       Primer 2:       Primer 3:       Primer 4:
1234 5678       1342 2431       1234 4321       -1234 1234
0               1               1               -1
```

[Rešenje A.3.19]

A.2 Kvalifikacioni zadaci

A.3 Ispitni rokovi

A.3.1 Programiranje 1, i–smer, Završni ispit, januar, 23.01.2016.

Zadatak A.3.1 *(5 poena)* Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} 2(x-1) & x \geq 1 \\ 0 & \text{inače} \end{cases}$$

[Rešenje A.3.19]

Grupa I

Zadatak A.3.2 *(4 poena)* Napisati C program koji sa standardnog ulaza učitava pozitivan ceo broj **n** i na standardni izlaz ispisuje n-ti član niza:

$$a_n = \begin{cases} 1 & n = 1 \\ 3 & n = 2 \\ 2a_{n-1} + 3a_{n-2} + 4 & n \geq 3 \end{cases}$$

Neispravnim ulazom se smatra broj manji ili jednak nuli i u tom slučaju na standardni izlaz ispisati -1. Dozvoljeno je korišćenje nizova. Maksimalna vrednost za **n** je **2000**.

```
Primer 1:       Primer 2:     Primer 3:     Primer 4:
-123            1             4             10
-1              1             39            29523
```

[Rešenje A.3.19]

Zadatak A.3.3 *(7 poena)* Napisati funkciju

```
void f3(char s[], char* c, int* br)
```

koja proverava koji karakter se najviše puta pojavio u niski *s*. Taj karakter smešta u promenljivu **c**, a broj pojavljivanja karaktera u promenljivu **br**. Sa standardnog ulaza unosi se linija teksta (može sadržati beline). Testirati rad funkcije *f3* programom koji sa standardnog ulaza učitava nisku i na standarni izlaz ispisati koji karakter se najviše puta pojavio u okviru nje, kao i broj pojavljivanja datog karaktera. Ukoliko postoji više karaktera čiji broj pojavljivanja odgovara maksimalnom broju, ispisati onaj sa najmanjim kodom u ASCII tabeli. Pretpostaviti da se na sistemu koristi `ASCII` tabela.

```
Primer 1:      Primer 2:    Primer 3:    Primer 4:
abrakadabra    cvrcak       jorgovan99   s@rm@ ponek@d v@zno
a 5          c 2          9 2          @ 4
```

[Rešenje A.3.19]

Zadatak A.3.4 *(7 poena)* Igra "Minesweeperšastoji se od pravougaone table izdeljene na polja koja mogu biti bezbedna ili su na njima rasporedjene mine. Sa standardnog ulaza učitavaju se brojevi **n** i **m** koji označavaju dimenzije table. Nako toga unosi se broj **k** kojim se navodi koliko mina se nalazi na tabli i k pozicija (**i**, **j**) koja označavaju pozicije na tabli na kojima se nalaze mine (i-ti red, j-ta kolona). Korisnik zatim unosi koordinate **l** i **m** za koje se na standardni izlaz ispisuje broj koliko se mina nalazi na poljima susednim tom polju. Proveravaju se susedna polja u svih 8 pravaca. Ukoliko je polje koje se proverava baš mina ispisati na standardni izlaz **MINA**. Maksimalna dimenzija table je 100x100. Ukoliko je neka od koordinata izvan dimenzija table ili su dimenzije table izvan dozvoljenih granica na standardni izlaz ispisati -1.

```
Primer 1:          Primer 2:          Primer 3:        Primer 4:
Ulaz:    Izlaz:    Ulaz:    Izlaz:    Ulaz    Izlaz:   Ulaz:    Izlaz:
4 4      2         4 4      MINA      2 3     -1       101 10   -1
3                  2                  1               1
0 1                0 1                -1 0             45 67
1 2                1 2                2 2              30 31
2 3                2 3
2 2                2 3
```

[Rešenje A.3.19]

Zadatak A.3.5 *(7 poena)* Služba gradskog prevoza želi da u svakom trenutku ima evidenciju o opterećenju svojih linija. Na linijama saobraćaju autobusi, trolejbusi i tramvaji. Maksimalni kapacitet autobusa je 25, trolejbusa 20 a tramvaja 30 putnika. Broj linije je pozitivan ceo broj manji od 1000.

a) *(1 poen)* Definisati strukturu kojim se opisuje vozilo. Svako vozilo zadato je svojim tipom (autobus, trolejbus, tramvaj), linijom na kojom saobraća i brojem putnika koji se u vozilu nalaze.

b) *(6 poena)* Sa standardnog ulaza se učitava broj **n** ($0 \leq n \leq 1000$)**, n vozila i broj linije**. Za zadati broj linije na standardni izlaz ispisati ukupan broj slobodnih mesta na toj liniji. Koristiti strukturu definisanu pod **a)**. Neispravnim ulazom smatraju se negativan broj putnika, broj putnika veći od dozvoljenog kapaciteta za navedeni tip vozila, tip vozila sa nazivom različitim od navedena tri ili negativan broj linije. U tim slučajevima na standardni izlaz ispisati -1.

```
Primer 1:                    Primer 2:                    Primer 3:
Ulaz:            Izlaz:      Ulaz:            Izlaz:      Ulaz             Izlaz:
4                8           3                -1          3                0
autobus   27 18              AutobuS 65 23                tramvaj 7  29
trolejbus 28 15              Kombi   1  10                tramvaj 3  15
tramvaj   7  29              minibus 6  21                tramvaj 12 12
autobus   27 24             6                            14
27
```

```
Primer 4:                    Primer 5:
Ulaz:            Izlaz:      Ulaz:        Izlaz:
2                -1          500          -1
autobus 26 20
tramvaj 9  32
```

[Rešenje A.3.19]

Grupa II

Zadatak A.3.6 *(4 poena)* Napisati C program koji za uneti niz celobrojnog tipa i neparne dužine n ispisuje po k elemenata levo i desno od sredine niza (ne uključujući sredinu). Prvo se unosi n, zatim niz od n elemenata, a na kraju i k.

Neispravnim ulazom se smatra niz parne ili negativne dužine, kao i k koje je negativno ili veće od polovine dužine niza. U slučaju neispravnog ulaza ispisati -1 na standardni izlaz.

Smatrati da je maksimalna veličina niza 100 elemenata.

```
Primer 1:      Primer 2:          Primer 3:    Primer 4:   Primer 5:
Ulaz:          Ulaz:              Ulaz:        Ulaz:       Ulaz:
5              9                  6            3           5
1 2 3 4 5      9 8 7 6 5 4 3 2 1  1 2 3 4 5 6  1 2 3       10 9 8 7 6
2              1                  5            10          -6
Izlaz:         Izlaz:             Izlaz:       Izlaz:      Izlaz:
1 2 3 4        6 4                -1           -1          -1
```

[Rešenje A.3.19]

Zadatak A.3.7 *(7 poena)* Barkod kodira broj proizvoda dodajući mu kontrolnu cifru. Kontrolna cifra izračunava se kao poslednja cifra zbira jedinica u zapisu svake cifre broja proizvoda. Npr. broj 86012 kodira se kao 1000 0110 0000 0001 0010 a kontrolna cifra je $(1 + 1 + 1 + 1 + 1) \bmod 10 = 5$.

Napisati funkciju

```
void kontrolna(char broj_proizvoda[], int *kont)
```

koja izračunava kontrolnu cifru broja proizvoda, koji se zadaje kao niska, i smešta ga u promenljivu kont. Niska može sadržati beline i druge karaktere, ali ih pri izračunavanju kontrolne cifre treba ignorisati, samo cifre uzeti u obzir.

Napisati program koji sa standardnog ulaza učitava liniju teksta kojom je predstavljen broj proizvoda i testira funkciju kontrolna. Na standardni izlaz ispisati izračunatu kontrolnu cifru. Maksimalna dužina niske je 100 karaktera.

Na sistemu se koristi `ASCII` tabela. Ukoliko ne postoji ni jedna cifra u barkodu, onda je kontrolna cifra `0`.

```
Primer 1:      Primer 2:     Primer 3:     Primer 4:
Ulaz:          Ulaz:         Ulaz:         Ulaz:
86012          001-223-4     555 555-555   AB-- 123 --BA
Izlaz:         Izlaz:        Izlaz:        Izlaz:
5              6             8             4
```

[Rešenje A.3.19]

Zadatak A.3.8 *(7 poena)* Napisati program koji ispisuje prosek zbirova svih kolona matrice čiji su elementi tipa `double`.

Prvo se unosi broj redova matrice n, zatim broj kolona matrice m, i onda n redova sa po m elemenata.

Maksimalna veličina matrice je 100×100. Ukoliko je ulaz neispravan (za vrednosti m i n) prekinuti rad programa i ispisati `-1`.

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:
Ulaz:              Ulaz:              Ulaz:              Ulaz:
4 4                3 2                2 4                3 3
0.2 0.4 0.7 1.3    1.23 4.56          0.1 0.2 0.3 0.4    1 0 0
1.5 1.7 2.2 2.5    0 1                10.98 7.65 4.32 1  0 1 0
6.3 -1.2 4.4 5.6   7.89 1             Izlaz:             0 0 1
1.6 2.3 2.8 3.5    Izlaz:             6.2375             Izlaz:
Izlaz:             7.8400                                1.000
8.9500
```

[Rešenje A.3.19]

Zadatak A.3.9 *(7 poena)* Profesor na jednom predmetu je uveo pravilo da njegov predmet položio svako ko na ispitu osvoji broj poena koji je veći ili jednak od proseka poena umanjenog za 10.

a) *(1 poen)* Definisati strukturu kojom se opisuje svaki student sa indeksom (`indeks-u-obliku-alas-naloga`) i brojem poena koji je osvojio (ceo broj od 0 do 100).

b) *(6 poena)* Na ulazu ćete dobiti n ($0 \leq n \leq 300$), broj studenata koji su polagali predmet, i onda n redova oblika

`indeks-u-obliku-alas-naloga broj-poena-na-ispitu`

Ispisati na standardni izlaz indekse svih studenata koji su položili ovaj predmet. Koristiti strukturu definisanu pod **a)**.

Smatrati da je indeks pravilno zapisan. U slučaju loše vrednosti za n ili loše vrednosti za broj poena ispisati -1.

```
Primer 1:          Primer 2:          Primer 3:          Primer 4:          Primer 5:
Ulaz:              Ulaz:              Ulaz:              Ulaz:              Ulaz:
4                  4                  6                  4                  3
mi12123 80         mr12345 91         mi00001 20         mi11110 100        mi05900 98
mi15512 70         ml54321 80         mi00002 32         mi11111 99         mi13034 120
mi15555 99         mv36925 29         mi00003 96         mi11112 98         mi11234 34
mi13333 40         mi14725 55         mi00004 52         mi11113 87         Izlaz:
Izlaz:             Izlaz:             mi00005 41         Izlaz:             -1
mi12123            mr12345            mi00006 15         mi11110
mi15512            ml54321            Izlaz:             mi11111
```

```
mi15555          mi14725          mi00003          mi11112
                                  mi00004          mi11113
                                  mi00005
```

[Rešenje A.3.19]

A.3.2 Programiranje 1, i–smer, Završni ispit, februar, 11.02.2016.

Zadatak A.3.10 Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} 2(x-y) & x \geq y \\ 0 & \text{inače} \end{cases}$$

[Rešenje A.3.19]

Zadatak A.3.11 Sa standardnog ulaza se unose celi, nenegativni brojevi sve dok se ne unese nula. Na standardni izlaz ispisati kvadrat razlike najvećeg i najmanjeg od unetih brojeva. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ukoliko je unet negativan broj ili ukoliko nije unet ni jedan broj osim nule.

```
Primer 1:      Primer 2:      Primer 3:    Primer 4:
1 2 3 4 5 0    1 2 3 -4 5 0   1 1 1 1 0    0
16            -1             0            -1
```

[Rešenje A.3.19]

Zadatak A.3.12 **a)** Napisati funkciju

```
void mutacije(char s1[], char s2[], int *br)
```

koja za navedene niske **s1** i **s2** iste dužine proverava na koliko mesta se karakteri niski razlikuju i rezultat upisuje u promenljivu **br**. Pri poređenju ignorisati beline.

b) Napisati program koji sa standardnog ulaza učitava dve DNK sekvence (niske karaktera A, T, C ili G) iste dužine i testira funkciju **mutacije** ispisujući vrednost promenljive **br** na standardni izlaz. Maksimalna dužina niski je 100 karaktera. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ukoliko neka od niski sadrži karakter koji ne pripada skupu {A, T, C, G} i nije belina ili je jedna niska duža od druge.

```
Primer 1:      Primer 2:       Primer 3:       Primer 4:

Ulaz:          Ulaz:           Ulaz:           Ulaz:
AGTC CGCT AGT   ATCG ATCG ATCG   AGTTGTTGT ATGX   AGGGATGGATGAG
AGTCC GC TAGT   ACCG ATGC ATCA   TTGTATGGA GGAT   TTGATGACGT

Izlaz:         Izlaz:          Izlaz:          Izlaz:
0              3               -1              -1
```

[Rešenje A.3.19]

Zadatak A.3.13 Krtice su organizovano napale baštu šargarepa. Farmer je napravio pravougaonu mapu bašte dimenzija n x m, gde je znakom **X** označio polje na kome se nalazi krtičnjak, dok je netaknuta polja označio znakom **-** . Kako je bašta velika, farmer želi da bez mnogo muke izračuna broj krtičnjaka u proizvoljnom pravougaonom delu svoje bašte. Sa standardnog ulaza unose se dimenzije mape **n** i **m**, zatim mapa bašte sa oznakama krtičnjaka i netaknutih polja. Nakon toga farmer zadaje koordinate (**i1**, **j1**) i (**i2**, **j2**) koje označavaju gornji levi i donji desni ugao pravouganika za koji farmer pita koliko krtičnjaka je obuhvaćeno na mapi tim pravouganikom. Na standardni izlaz ispisati broj krtičnjaka u zadatom pravouganiku. Maksimalna dimenzija mape je 100 x 100. U slučaju neispravnih koordinata uglova pravouganika, neispravnih dimenzija mape ili oznaka na tabli van skupa { X, - } na standardni izlaz ispisati -1.

```
Primer 1:       Primer 2:       Primer 3:       Primer 4:

Ulaz:           Ulaz:           Ulaz:           Ulaz:
4 4             4 4             4 4             4 4
- - X -         - - X K         - X - X         - X - X
X - - -         - X - -         X - X -         X - X -
- X - X         X - - -         - X - X         - X - X
X - X -         X X - X         X - X -         X - X -
0 1             1 2             3 4             0 0
2 2             3 4             1 2             3 3

Izlaz:          Izlaz:          Izlaz:          Izlaz:
2               -1              -1              8
```

[Rešenje A.3.19]

Zadatak A.3.14 Vlasnik pekare želi da utvrdi koliko je isplativa prodaja njegovog najskupljeg peciva.

a) Definisati strukturu **Pecivo** koja sadrži podatke o imenu peciva (najviše 50 karaktera) i ceni peciva (realan broj tipa double).

b) Sa standardnog ulaza se unosi broj **n** a zatim mesečni obračun sa n prodatih komada peciva, pri čemu je naziv peciva u jednom redu a cena u narednom. Na standardni izlaz ispisati ukupnu zaradu od prodaje najskupljeg peciva zaokruženu na dva decimalna mesta. U slučaju negativne cene peciva ili u slučaju da je n manje ili jednako nuli ispisati -1. Pretpostaviti da će samo jedna vrsta peciva imati maksimalnu cenu.

```
Primer 1:           Primer 2:       Primer 3:       Primer 4:

Ulaz:              Ulaz:           Ulaz:           Ulaz:
5                  3               -1              5
burek sa mesom       mafin                           kroasan sa dzemom
100.50             -50.03                          49.99
buhtla sa cokoladom   krofna                         kroasan sa dzemom
50.00              56.00                           49.99
burek sa mesom       krofna                         kroasan sa dzemom
100.50             56.00                           49.99
rol virsla                                         kroasan sa dzemom
75.00                                              49.99
kroasan sa kremom                                    kroasan sa dzemom
60.00                                              49.99

Izlaz:             Izlaz           Izlaz:          Izlaz:
201.00             -1              -1              249.95
```

[Rešenje A.3.19]

A.3.3 1. Grupa, I smer, Programiranje 1 2015/2016, ispit, jun

Zadatak A.3.15 Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} x - y + 2 & x + 2 \geq y \\ 0 & \text{inače} \end{cases}$$

[Rešenje A.3.19]

Zadatak A.3.16 Napisati program koji sa standardnog ulaza učitava prvo pozitivan ceo broj n ($0 < n \leq 99$), a zatim i n celih brojeva i izračunava zbir parnih. Izračunati zbir ispisati na standardni izlaz. U slučaju greške (za $n \leq 0$ ili $n \geq 100$) na standardni izlaz ispisati -1.

Ulaz	5 1 2 3 4 5	5 -1 -2 -3 -4 -5	3 10 -10 10	-3 1 2 3
Izlaz	6	-6	10	-1

[Rešenje A.3.19]

Zadatak A.3.17 Napisati funkciju *void f(char s[], char c, int *prva, int* poslednja)* koja u datoj nisci s pronalizi indekse prvog i poslednjeg pojavljivanja datog karaktera c i dobijene vrednosti redom smešta u promenljive *prva* i *poslednja*. Ukoliko se karakter ne pojavljuje u nisci, obe vrednosti postaviti na *-1*.

Potom napisati program koji sa standardnog ulaza učitava karaktersku nisku (dužine ne veće od 150 karaktera) i jedan karakter i nakon toga poziva funkciju *f*, a potom na standarni izlaz ispisuje indekse prvog i poslednjeg pojavljivanja datog karaktera u datoj nisci. Pretpostaviti da je ulaz u ispravnom formatu.

Ulaz	ucionica i	ucionica u	ucionica o	ucionica p
Izlaz	2 5	0 0	3 3	-1 -1

[Rešenje A.3.19]

Zadatak A.3.18 Sa standarnog ulaza se zadaje dimenzija kvadratne matrice n ($0 < n \leq 99$), a zatim elementi matrice koji su celi brojevi. Na standardni izlaz ispisati redni broj vrste koja ima najveći zbir elemenata. U slučaju greške (za $n \leq 0$ ili $n \geq 100$) na standardni izlaz ispisati -1.

[Rešenje A.3.19]

Zadatak A.3.19 Definisati strukturu *Tacka* za predstavljanje tačaka u ravni sa koordinatama tipa *double*. Sa standardnog ulaza se ucitava broj n ($1 < n \leq 99$), zatim niz od n tačaka tako sto se unosi prvo x, pa y koordinata za svaku tačku. Za zadate tačke ispisati na standardni izlaz dužinu najduže duži koja se može obrazovati od neke dve tačke iz učitanog niza. Rezultat ispisati na dve decimale. Dužina duži između tačaka $a(x1; y1)$ i $b(x2; y2)$ se računa po formuli

$$\sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

U slučaju greške (za $n \leq 1$ ili $n \geq 100$) na standardni izlaz ispisati -1.

[Rešenje A.3.19]

A.3.4 Praktični deo ispita, jun ...

A.4 Rešenja

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19

Rešenje A.3.19