PROGRAMIRANJE 1

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

PROGRAMIRANJE 1
Zbirka zadataka

Beograd
2017.

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
Aleksandra Kocić, 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 x i y ispisuje 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 ispravan.*

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite vrednost promenljive x: -3
 Unesite vrednost 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. Cene artikala su pozitivni celi brojevi. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite cenu prvog artikla: 384
 Unesite 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. Obe ulazne vrednosti su pozitivni celi brojevi. NAPOMENA: *Pretpostaviti da je unos ispravan.*

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. Sve ulazne vrednosti su pozitivni celi brojevi. NAPOMENA: *Pretpostaviti da je unos ispravan.*

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 x i y. Napisati program koji razmenjuje njihove vrednosti.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite vrednosti x i y: 5 7
 Pre zamene: x=5, y=7
 Posle zamene: x=7, y=5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite vrednosti x i y: 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:
 Unesite vrednosti a i b: 5 7
 Nove vrednosti su: a=12, b=-2
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite vrednosti a i b: 237 -592
 Nove vrednosti su: a=-355, b=829
```

[Rešenje 1.1.9]

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

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 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. Cena proizvoda je pozitivan ceo broj. NAPOMENA: *Pretpostaviti da je unos ispravan.*

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 i ispisuje broj dobijen obrtanjem njegovih cifara. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 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 pozitivnom celom broju. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.1.14]

Zadatak 1.1.15 Napisati program koji učitava pozitivan ceo broj n i pozitivan dvocifreni broj m i 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 ceo broj: 12345
 Unesite pozitivan dvocifreni broj: 67
 Novi broj je 1267345
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite pozitivan ceo 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:
 Unesite broj inca: 4.69
 4.69 in = 11.91 cm
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 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:
 Unesite broj milja: 50.42
 50.42 mi = 81.14 km
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 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:
 Unesite broj funti: 2.78
 2.78 lb = 1.26 kg
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 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. NAPOMENA: *Pretpostaviti da je unos ispravan.* UPUTSTVO: *Veza između farenhajta i celzijusa je zadata narednom formulom* $F = \frac{9 \cdot C}{5} + 32$

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite 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
 Determinanta: -2.0000
```

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 1.1.20]

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 poluprecnik: 4.2
 Obim: 26.39
 Povrsina: 55.42
```

Primer 2

```
Interakcija sa programom:
 Unesite poluprecnik: 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.* Uputstvo: *Za računanje korena broja koristiti funkciju* `sqrt` *čija se deklaracija nalazi u zaglavlju* `math.h`.

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. NAPOMENA: *Pretpostaviti da je unos ispravan.*

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
```

[Rešenje 1.1.25]

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: 6.67
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite tri cela broja: 3 -8 13
 Aritmeticka sredina: 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 celobrojne vrednosti 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 celobrojene cene usluge po kvadratnom metru program izračuna ukupnu cenu krečenja. Sve realne vrednosti ispisati zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

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 cele brojeve x, p i c ispisuje broj koji se dobija ubacivanjem cifre c u broj x na poziciju p. Pretpostaviti da numeracija cifara počinje od nule, odnosno da se cifra najmanje težine nalazi se na nultoj poziciji. NAPOMENA: *Pretpostaviti da je unos ispravan.* UPUTSTVO: *Koristiti funkciju* `pow` *čija se deklaracija nalazi u zaglavlju* `math.h`.

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 1.1.31]

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

```c
#include<stdio.h>

int main()
{
  /* Ispisuje se trazena poruka. Na kraju poruke se ispisuje i
     novi red. */
  printf("Zdravo svima!\n");

  /* Povratna vrednost 0 se obicno koristi da oznaci da je prilikom
     izvrsavanja programa sve proslo u redu. */
  return 0;
}
```

Rešenje 1.1.2

```c
#include <stdio.h>

int main()
{
  /* Deklaracija celobrojne promenljive. */
  int n;

  /* Ucitava se vrednost celog broja. */
  printf("Unesite ceo broj: ");
  scanf("%d", &n);

  /* Ispis kvadratne vrednosti unetog broja. */
  printf("Kvadrat: %d\n", n * n);

  /* Ispis kubne vrednosti unetog broja. */
  printf("Kub: %d\n", n * n * n);

  return 0;
}
```

Rešenje 1.1.3

```c
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int x, y, rezultat;
```

```
    /* Ucitava se vrednost broja x. */
    printf("Unesite vrednost promenljive x: ");
    scanf("%d", &x);

    /* Ucitava se vrednost broja y. */
    printf("Unesite vrednost promenljive y: ");
    scanf("%d", &y);

    /* I nacin ispisa: dodela zbira x+y promenljivoj rezultat i
       ispis vrednosti promenljive rezultat. */
    rezultat = x + y;
    printf("%d + %d = %d\n", x, y, rezultat);

    /* II nacin ispisa: direktan ispis vrednosti izraza, bez njegovog
       dodeljivanja posebnoj promenljivoj. */
    printf("%d - %d = %d\n", x, y, x - y);
    printf("%d * %d = %d\n", x, y, x * y);

    /* Kada se operator / primeni na dva celobrojna argumenta x i y,
       kao rezultat se dobije ceo deo pri deljenju broja x brojem y,
       a ne kolicnik. Na primer, rezultat primene operatora / na 7 i 2
       je 3, a ne 3.5. */
    printf("%d / %d = %d\n", x, y, x / y);

    /* Operator % izracunava ostatak pri celobrojnom deljenju dve
       celobrojne promenljive.
       Da bi se odstampao karakter %, u naredbi printf se pise %%. */
    printf("%d %% %d = %d\n", x, y, x % y);

    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

```
#include <stdio.h>

```

```c
int main()
{
  /* Deklaracija promenljivih cija je vrednost neoznacen ceo broj. */
  unsigned int cena, kolicina, iznos;
  unsigned int kusur;

  /* Ucitavaju se vrednosti cene, kolicine i iznosa. */
  printf("Unesite cenu, kolicinu i iznos:\n");
  scanf("%u%u%u", &cena, &kolicina, &iznos);

  /* Izracunava se kusur. */
  kusur = iznos - kolicina * cena;

  /* Ispis vrednosti kusura. */
  printf("Kusur je %u dinara.\n", kusur);

  return 0;
}
```

Rešenje 1.1.7

```c
#include <stdio.h>

int main()
{

  /* Deklaracija potrebnih promenljivih. */
  unsigned int poletanje, poletanje_sat, poletanje_minut;
  unsigned int sletanje, sletanje_sat, sletanje_minut;
  unsigned int duzina, duzina_sat, duzina_minut;

  /* Ucitavaju se sat i minut vremena poletanja. */
  printf("Unesite vreme poletanja: ");
  scanf("%u%u", &poletanje_sat, &poletanje_minut);

  /* Ucitavaju se sat i minut vremena sletanja. */
  printf("Unesite vreme sletanja: ");
  scanf("%u%u", &sletanje_sat, &sletanje_minut);

  /* Obe vrednosti se pretvaraju u sekunde,
     kako bi se lakse izracunala razlika. */
  poletanje = poletanje_sat * 3600 + poletanje_minut * 60;
  sletanje = sletanje_sat * 3600 + sletanje_minut * 60;

  /* Racunanje razlike u sekundama izmedju sletanja i poletanja. */
  duzina = sletanje - poletanje;

  /* Razlika u sekundama se pretvara u razliku u satima i minutima.
     Razlika u satima se dobija celobrojnim deljenjem broja sekundi
     sa 3600.
     Preostali broj minuta se dobija deljenjem preostalog broja
```

```
     sekundi sa 60. */
  duzina_sat = duzina / 3600;
  duzina_minut = (duzina - duzina_sat * 3600) / 60;

  /* II nacin: duzina_minut = (duzina % 3600) / 60; */

  /* Ispis rezultata. */
  printf("Duzina trajanja leta je %u h i %u min\n", duzina_sat,
         duzina_minut);

  return 0;
}
```

Rešenje 1.1.8

```
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int x, y;
  int p;

  /* Ucitavaju se vrednosti x i y. */
  printf("Unesite vrednosti x i y: ");
  scanf("%d%d", &x, &y);

  /* Ispis vrednosti promenljivih pre zamene. */
  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;

  /* Ispis vrednosti promenljivih nakon zamene. */
  printf("Posle zamene: x=%d, y=%d\n", x, y);

  return 0;
}
```

Rešenje 1.1.9

```
#include<stdio.h>

int main()
{
```

```c
  /* Deklaracija potrebnih promenljivih. */
  int a, b;

  /* Ucitavaju se vrednosti a i b. */
  printf("Unesite vrednosti a i b: ");
  scanf("%d%d", &a, &b);

  /* U promenljivu a se smesta suma a+b. */
  a = a + b;

  /* U promenljivu b se smesta izraz a - 2*b, cija je vrednost (nakon
     promene promenljive a) jednaka  a + b - 2*b = a - b. */
  b = a - 2*b;

  /* Ispis rezultata. */
  printf("Nove vrednosti su: a=%d, b=%d\n", a, b);

  return 0;
}
```

Rešenje 1.1.10

```c
#include <stdio.h>

int main()
{
  /* Deklaracija neoznacenog broja. */
  unsigned int x;

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

  /* Ucitava se trocifren broj. */
  printf("Unesite trocifreni broj: ");
  scanf("%u", &x);

  /* Izdvajaju se cifre jedinice, desetice i stotine. */
  cifra_jedinice = x % 10;
  cifra_desetice = (x / 10) % 10;
  cifra_stotine = x / 100;

  /* Ispis rezultata.
     NAPOMENA: Kada se stampa numericka vrednost promenljive tipa
     char koristi se %d. Kada se stampa karakter ciji je ASCII
     kod jednak vrednosti te promenljive, tada se koristi %c.
     U ovom slucaju je potrebno stampati numericku vrednost. */
  printf("jedinica %d, desetica %d, stotina %d\n", cifra_jedinice,
         cifra_desetice, cifra_stotine);

```

```
  /* II nacin: Ispis rezultata bez uvodjenja dodatnih promenljivih
31    cifra_jedinice, cifra_desetice i cifra_stotine:

33    printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10, x
     /100); */

35  return 0;
}
```

Rešenje 1.1.11

```
1 #include <stdio.h>

3 int main()
{
5   /* Deklaracija i ucitavanje cene proizvoda. */
  unsigned int x;
7   printf("Unesite cenu proizvoda: ");
  scanf("%u", &x);
9
  /* Vrednost x/5000 predstavlja maksimalan broj novcanica od 5000
11    dinara koje je moguce iskoristiti za placanje racuna.
    Na primer, neka je uneta cena 8367 dinara, vrednost izraza
13    8367/5000 je jednaka 1. */
  printf("%u = %u*5000 + ", x, x / 5000);
15
  /* Da bi se isti postupak primenio i na ostale novcanice, potrebno
17    je izracunati preostali iznos. Jedan nacin da se to uradi je
    racunanje ostatka pri deljenju unete vrednosti x
19    (u primeru 8367) sa 5000. On iznosi 3367. Ovu vrednost
    dodeljujemo promeljivoj x. */
21  x = x % 5000;

23  /* Postupak se ponavlja i za za ostale novcanice. */
  printf("%u*2000 + ", x / 2000);
25  x = x % 2000;
  printf("%u*1000 + ", x / 1000);
27  x = x % 1000;
  printf("%u*500 + ", x / 500);
29  x = x % 500;
  printf("%u*200 + ", x / 200);
31  x = x % 200;
  printf("%u*100 + ", x / 100);
33  x = x % 100;
  printf("%u*50 + ", x / 50);
35  x = x % 50;
  printf("%u*20 + ", x / 20);
37  x = x % 20;
  printf("%u*10 + ", x / 10);
39  x = x % 10;
  printf("%u*1\n", x);
```

```
41
  return 0;
43 }
```

Rešenje 1.1.12

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     unsigned int x;
7    unsigned int obrnuto_x;
     char cifra_jedinice;
9    char cifra_desetice;
     char cifra_stotine;
11
     /* Ucitava se neoznacen trocifreni broj. */
13   printf("Unesite trocifreni broj: ");
     scanf("%u", &x);
15
     /* Izdvajaju se pojedinacne cifre broja. */
17   cifra_jedinice = x % 10;
     cifra_desetice = (x / 10) % 10;
19   cifra_stotine = x / 100;

21   /* Formira se rezultujuci broj. */
     obrnuto_x = cifra_jedinice * 100 + cifra_desetice * 10 +
       cifra_stotine;
23
     /* Ispis rezultata. */
25   printf("Obrnuto: %u\n", obrnuto_x);

27   return 0;
   }
```

Rešenje 1.1.13

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     unsigned int n, broj_obrnuto, broj_zamena;
7    char jedinice, desetice, stotine, hiljade;
     int proizvod_cifara, razlika_cifara, suma_kvadrata;
9
     /* Ucitava se jedan neoznacen broj. */
11   printf("Unesite cetvorocifreni broj: ");
```

```
  scanf("%u", &n);

  /* Izdvajaju se cifre ucitanog broja. */
  jedinice = n % 10;
  desetice = (n / 10) % 10;
  stotine = (n / 100) % 10;
  hiljade = n / 1000;

  /* Izracunava se proizvod cifara. */
  proizvod_cifara = jedinice * desetice * stotine * hiljade;
  printf("Proizvod cifara: %d\n", proizvod_cifara);

  /* Izracunava se razlika sume krajnjih i srednjih cifara. */
  razlika_cifara = (hiljade + jedinice) - (stotine + desetice);
  printf("Razlika sume krajnjih i srednjih: %d\n", razlika_cifara);

  /* Izracunava se suma kvadrata cifara. */
  suma_kvadrata = jedinice * jedinice + desetice * desetice +
      stotine * stotine + hiljade * hiljade;
  printf("Suma kvadrata cifara: %d\n", suma_kvadrata);

  /* Izracunava se broj zapisan istim ciframa ali u obrnutom
     redosledu. */
  broj_obrnuto = jedinice * 1000 + desetice * 100 + stotine * 10 +
    hiljade;
  printf("Broj u obrnutom poretku: %u\n", broj_obrnuto);

  /* Izracunava se broj u kojem su cifra jedinica i cifra stotina
     zamenile mesta. */
  broj_zamena = hiljade * 1000 + jedinice * 100 + desetice * 10 +
    stotine;
  printf("Broj sa zamenjenom cifrom jedinica i stotina: %u\n",
    broj_zamena);

  return 0;
}
```

Rešenje 1.1.14

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int broj, novibroj;
  unsigned int levo, desno;

  /* Ucitava se neoznacen ceo broj. */
  printf("Unesite broj: ");
  scanf("%u", &broj);
```

```
13    /* Desni deo rezultata je cifra jedinice unetog broja.
         Na primer, za broj 1234, desni deo je cifra 4. */
15    desno = broj%10;

17    /* Levi deo rezultata su sve cifre levo od cifre desetice.
         Na primer, za broj 1234, levi deo je broj 12 i dobija se
19       deljenjem unetog broja sa 100. */
      levo = broj/100;
21
      /* Rezultat se dobija spajanjem levog i desnog dela.
23       U datom primeru: 12*10 + 4 = 124. */
      novibroj = levo*10 + desno;
25
      /* Ispis rezultata. */
27    printf("Rezultat je: %u\n", novibroj);

29    return 0;
  }
```

Rešenje 1.1.15

```
  #include <stdio.h>
2
  int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6   unsigned int n, novibroj;
    unsigned int levi, desni, m;
8
    /* Ucitavaju se brojevi n i m. */
10  printf("Unesite pozitivan ceo broj: ");
    scanf("%u", &n);
12  printf("Unesite pozitivan dvocifreni broj: ");
    scanf("%u", &m);
14
    /* Levi deo rezultata su sve cifre levo od cifre stotina.
16     Na primer, ako je n=12345, levi deo rezultata je 12.
       On se dobija deljenjem unetog broja sa 1000. */
18  levi = n / 1000;

20  /* Desni deo rezultata su sve cifre desno od cifre hiljada.
       Za n=12345, desni deo rezultata je 345. */
22  desni = n % 1000;

24  /* Srednji deo rezultata je broj m.
       U navedenom primeru, rezultat se dobija nadovezivanjem
26     brojeva 12, 67 i 345. Ovo se radi mnozenjem delova sa
       odgovarajucim stepenom broja 10 i njihovim sabiranjem. */
28  novibroj = levi * 100000 + m * 1000 + desni;

30  /* Ispis rezultata. */
```

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

  return 0;
}
```

Rešenje 1.1.16

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float in, cm;

  /* Ucitava se realna vrednost koja predstavlja broj inca. */
  printf("Unesite broj inca: ");
  scanf("%f", &in);

  /* Izracunava se rezultat (1 in = 2.54 cm) */
  cm = in * 2.54;

  /* Ispis rezultata (na dve 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 1.1.20

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float a11, a12, a21, a22;
  float determinanta;
```

```
  /* Ucitavaju se elementi matrice. */
  printf("Unesite brojeve: ");
  scanf("%f%f%f%f", &a11, &a12, &a21, &a22);

  /* Izracunava se determinanta matrice. */
  determinanta = a11*a22 - a12*a21;

  /* Ispis rezultata na cetiri decimale. */
  printf("Determinanta: %.4f\n", determinanta);

  return 0;
}
```

Rešenje 1.1.21

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float a, b;
  float obim, povrsina;

  /* Ucitavaju se duzine stranica pravougaonika. */
  printf("Unesite duzine stranica pravougaonika: ");
  scanf("%f%f", &a, &b);

  /* Izracunava se obim pravougaonika. */
  obim = 2 * (a + b);

  /* Izracunava se povrsina pravougaonika. */
  povrsina = a * b;

  /* Ispis rezultata na dve decimale. */
  printf("Obim: %.2f\n", obim);
  printf("Povrsina: %.2f\n", povrsina);

  return 0;
}
```

Rešenje 1.1.22

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float r, obim, povrsina;
```

```
8
   /* Ucitava se poluprecnik kruga. */
10 printf("Unesite poluprecnik: ");
   scanf("%f", &r);
12
   /* Racunaju se obim i povrsina.
14    M_PI je konstanta koja se nalazi u zaglavlju math.h
      i njena vrednost odgovara priblizno vrednosti broja pi. */
16 obim = 2 * r * M_PI;
   povrsina = r * r * M_PI;
18
   /* Ispis rezultata na dve decimale. */
20 printf("Obim: %.2f\nPovrsina: %.2f\n", obim, povrsina);

22 return 0;
}
```

Rešenje 1.1.23

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

4 int main()
{
6   /* Deklaracija potrebnih promenljivih. */
   float a, povrsina, obim;
8
   /* Ucitava se duzina stranice. */
10 printf("Unesite duzinu stranice trougla: ");
   scanf("%f", &a);
12
   /* Racunaju se obim i povrsina. */
14 obim = 3 * a;
   povrsina = (a * a * sqrt(3)) / 4;
16
   /* Ispis rezultata na dve decimale. */
18 printf("Obim: %.2f\n", obim);
   printf("Povrsina: %.2f\n", povrsina);
20
   return 0;
22 }
```

Rešenje 1.1.24

```
1 #include <stdio.h>
  #include <math.h>
3
  int main()
5 {
```

```
  /* Deklaracija potrebnih promenljivih. */
  float a, b, c;
  float obim, s, povrsina;

  /* Ucitavaju se duzine stranica. */
  printf("Unesite duzine stranica trougla:\n");
  scanf("%f%f%f", &a, &b, &c);

  /* Racuna se obim. */
  obim = a + b + c;

  /* Racuna se povrsina koriscenjem Heronovog obrasca. */
  s = obim / 2;
  povrsina = sqrt(s * (s - a) * (s - b) * (s - c));

  /* Ispis rezultata. */
  printf("Obim: %.2f\n", obim);
  printf("Povrsina: %.2f\n", povrsina);

  return 0;
}
```

Rešenje 1.1.25

Nakon ispravnog izračunavanja dužina stranica, zadatak se rešava analogno zadatku 1.1.21.

Rešenje 1.1.26

```
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b, c;
  float as;

  /* Ucitavaju se tri cela broja. */
  printf("Unesite tri cela broja:");
  scanf("%d%d%d", &a, &b, &c);

  /* Pogresan nacin: as = (a+b+c)/3;
     Kada se operacija / koristi nad celim brojevima,
     deljenje je celobrojno.
     Na primer, (1+1+3)/3 ima vrednost 1.*/

  /* Ispravan nacin je da se bar jedan operand
     pretvori u realan broj. */
  as = (a + b + c) / 3.0;

```

```
  /* Drugi ispravni nacini:
     as=1.0*(a+b+c)/3;
     as=(0.0+a+b+c)/3;
     as=((float)(a+b+c))/3; */

  /* Ispis rezultata. */
  printf("Aritmeticka sredina:  %.2f\n", as);

  return 0;
}
```

Rešenje 1.1.27

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int duzina, sirina, visina;
  unsigned int cena;
  float povrsina_za_krecenje;
  float ukupna_cena;

  /* Ucitavaju se vrednosti duzine, sirine i visine sobe. */
  printf("Unesite dimenzije sobe: ");
  scanf("%u%u%u", &duzina, &sirina, &visina);

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

  /* Povrsina za krecenje odgovara povrsini kvadra
     umanjena za povrsinu poda jer se on ne kreci. */
  povrsina_za_krecenje = 0.8 * (duzina * sirina +
                                2 * duzina * visina +
                                2 * sirina * visina);

  /* Racuna se ukupna cena. */
  ukupna_cena = povrsina_za_krecenje * cena;

  /* Ispis rezultata. */
  printf("Moler treba da okreci %.2f m2\n", povrsina_za_krecenje);
  printf("Cena krecenja je %.2f\n", ukupna_cena);

  return 0;
}
```

Rešenje 1.1.28

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int x, p, c;
  unsigned int levo, desno;
  unsigned int novo_x;

  /* Ucitavaju se broj, pozicija i cifra. */
  printf("Unesite redom x, p i c: ");
  scanf("%u%u%u", &x, &p, &c);

  /* Racuna se deo broja koji se nalazi desno od pozicije p.
     Funkcija pow kao povratnu vrednost vraca realan broj dvostruke
     tacnosti, a operacija % ocekuje celobrojne operande. Iz tog
     razloga je neophodno izvrsiti pretvaranje povratne vrednosti
     u tip unsigned int. */
  desno = x % (unsigned int) pow(10, p);

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

  /* Rezultat se racuna nadovezivanjem levog dela, cifre c
     i desnog dela. */
  novo_x =levo * (unsigned int) pow(10, p + 1) +
         c * (unsigned int) pow(10, p) + desno;

  /* Ispis rezultata. */
  printf("Rezultat je: %u\n", novo_x);

  return 0;

}
```

Rešenje 1.1.29

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b;
  int rezultata, rezultatb, rezultatc;

  /* Ucitavaju se dva cela broja. */
  printf("Unesite dva cela broja: ");
  scanf("%d%d", &a, &b);
```

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

17    /* Izraz a%2==0 && b%2==0 je konjunkcija koja se sastoji od dve
         relacije poredjenja jednakosti. Izraz a%2==0 ima vrednost 1 ako
19       je ova relacija tacna, a 0 u suprotnom. */
      rezultatb = (a % 2 == 0 && b % 2 == 0);
21
      /* Izraz a>0 && a<=100 && b>0 && b<=100 je konjunkcija koja se
23       sastoji od cetiri konjunkata. Svaki od konjunkata je izraz
         koji sadrzi relacioni operator i ima vrednost 1 ako relacija
25       vazi, a 0 ako ne vazi. */
      rezultatc = (a > 0 && a <= 100 && b > 0 && b <= 100);
27
      /* Ispis rezultata. */
29    printf("a) rezultat=%d\n", rezultata);
      printf("b) rezultat=%d\n", rezultatb);
31    printf("c) rezultat=%d\n", rezultatc);

33    return 0;
  }
```

Rešenje 1.1.30

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     int a, b, max;
7
     /* Ucitavaju se dve celobrojne vrednosti. */
9    printf("Unesite dva cela broja: ");
     scanf("%d%d", &a, &b);
11
     /* Racuna se maksimum koriscenjem ternarnog operatora uslova. */
13   max = (a > b) ? a : b;

15   /* Ispis rezultata. */
     printf("Maksimum je %d\n", max);
17
     return 0;
19 }
```

Rešenje 1.1.31

Zadatak se rešava analogno zadatku 1.1.30

Rešenje 1.1.32

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float a, b, rez;
  float min, max;

  /* Ucitavaju se dva realna broja. */
  printf("Unesite dva realna broja: ");
  scanf("%f%f", &a, &b);

  /* Racunaju se minimalna i maksimalna vrednost unetih brojeva. */
  min = (a < b) ? a : b;
  max = (a > b) ? a : b;

  /* Racuna se vrednost rezultata. */
  rez = (min + 0.5) / (1 + max * max);

  /* Ispis rezultata. */
  printf("Rezultat je %.2f\n", rez);

  return 0;
}
```

2

Kontrola toka

2.1 Naredbe grananja

Zadatak 2.1.1 Napisati program koji ispisuje najmanji od tri uneta cela broja.

Primer 1

```
Interakcija sa programom:
 Unesite tri cela broja: 5 18 -1
 Najmanji: -1
```

Primer 2

```
Interakcija sa programom:
 Unesite tri cela broja: 0 43 16
 Najmanji: 0
```

Primer 3

```
Interakcija sa programom:
 Unesite tri cela broja: 3 3 3
 Najmanji: 3
```

Primer 4

```
Interakcija sa programom:
 Unesite tri cela broja: -5 -5 -5
 Najmanji: -5
```

[Rešenje 2.1.1]

Zadatak 2.1.2 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
 Apsolutna vrednost: 7.42
```

Primer 2

```
Interakcija sa programom:
 Unesite jedan realan broj: -562.428
 Apsolutna vrednost: 562.43
```

Primer 3

```
Interakcija sa programom:
 Unesite jedan realan broj: 0
 Apsolutna vrednost: 0.00
```

Primer 4

```
Interakcija sa programom:
 Unesite jedan realan broj: 52
 Apsolutna vrednost: 52.00
```

[Rešenje 2.1.2]

Zadatak 2.1.3 Napisati program koji za uneti ceo broj ispisuje njegovu recipročnu vrednost zaokruženu na četiri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

```
Interakcija sa programom:
 Unesite jedan ceo broj: 0
 Greska: nedozvoljeno je deljenje nulom.
```

Primer 4

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

[Rešenje 2.1.3]

Zadatak 2.1.4 Napisati program koji učitava tri cela broja i ispisuje zbir pozitivnih.

Primer 1

```
Interakcija sa programom:
 Unesite tri cela broja: 1 3 -6
 Zbir pozitivnih: 4
```

Primer 2

```
Interakcija sa programom:
 Unesite tri cela broja: -15 81 0
 Zbir pozitivnih: 81
```

Primer 3

```
Interakcija sa programom:
 Unesite tri cela broja: -719 -48 -123
 Zbir pozitivnih: 0
```

Primer 4

```
Interakcija sa programom:
 Unesite tri cela broja: 16 2 576
 Zbir pozitivnih: 594
```

[Rešenje 2.1.4]

Zadatak 2.1.5 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. Cene artikala su pozitivni celi brojevi. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite tri cene: 35 125 97
 Cena sa popustom: 223 din
 Usteda: 34 din
```

Primer 2

```
Interakcija sa programom:
 Unesite tri cene: 1034 15 25
 Cena sa popustom: 1060 din
 Usteda: 14 din
```

Primer 3

```
Interakcija sa programom:
 Unesite tri cene: 500 500 500
 Cena sa popustom: 1001 din
 Usteda: 499 din
```

Primer 4

```
Interakcija sa programom:
 Unesite tri cene: 247 -133 126
 Greska: neispravan unos cene.
```

[Rešenje 2.1.5]

Zadatak 2.1.6 Napisati program koji za uneto vreme u formatu *sat:minut* ispisuje koliko je sati i minuta ostalo do ponoći. Broj sati treba da bude iz intervala $[0, 24)$, a broj minuta iz intervala $[0, 60)$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite vreme:  18:19
 Do ponoci: 5 sati i 41 minuta
```

Primer 2

```
Interakcija sa programom:
 Unesite vreme:  23:7
 Do ponoci: 0 sati i 53 minuta
```

Primer 3

```
Interakcija sa programom:
 Unesite vreme:  24:20
 Greska: neispravan unos vremena.
```

Primer 4

```
Interakcija sa programom:
 Unesite vreme:  14:0
 Do ponoci: 10 sati i 0 minuta
```

[Rešenje 2.1.6]

Zadatak 2.1.7 Napisati program koji za unetu godinu ispisuje da li je prestupna. Godina je neoznačen ceo broj.

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 2.1.7]

Zadatak 2.1.8 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
 ASCII kod: 48
```

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 2.1.8]

Zadatak 2.1.9 Napisati program koji učitava tri karaktera i ispisuje proizvod svih karaktera koji su cifre. Ukoliko među unetim karakterima nema cifara, program treba da ispiše odgovarajuću poruku. Napomena: *Karakteri koji se unose su razmaknuti blanko znacima.*

Primer 1

```
Interakcija sa programom:
 Unesite karaktere: A 5 3
 Proizvod cifara: 15
```

Primer 2

```
Interakcija sa programom:
 Unesite karaktere: k ! m
 Medju unetim karakterima nema cifara.
```

Primer 3

```
Interakcija sa programom:
 Unesite karaktere: 9 9 9
 Proizvod cifara: 729
```

Primer 4

```
Interakcija sa programom:
 Unesite karaktere: a 8 0
 Proizvod cifara: 0
```

[Rešenje 2.1.9]

Zadatak 2.1.10 Kasirka unosi šifru artikla koja se zadaje kao tri spojena karaktera koji mogu biti mala slova, velika slova ili cifre. U kasi, sve šifre su zapisane malim slovima i ciframa. Napisati program koji kasirkin unos konvertuje u unos koji je odgovarajući za kasu, tj. koji sva velika slova pretvara u odgovarajuća mala, a ostale karaktere ne menja. U slučaju neispravnog unosa šifre, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite sifru: aBc
 abc
```

Primer 2

```
Interakcija sa programom:
 Unesite sifru: a?!
 Greška: ? je neispravan karakter.
```

Primer 3

```
Interakcija sa programom:
 Unesite karaktere: 5A5
 5a5
```

Primer 4

```
Interakcija sa programom:
 Unesite karaktere: 123
 123
```

[Rešenje 2.1.10]

Zadatak 2.1.11 Napisati program koji za uneti četvorocifreni broj ispisuje njegovu najveću cifru. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 6835
 Najveca cifra je: 8
```

Primer 2

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 7777
 Najveca cifra je: 7
```

Primer 3

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 238
 Greska: niste uneli cetvorocifreni broj.
```

Primer 4

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: -2002
 Najveca cifra je: 2
```

[Rešenje 2.1.11]

Zadatak 2.1.12 Broj je Armstrongov ako je jednak zbiru kubova svojih cifara. Napisati program koji za dati pozitivan trocifreni broj proverava da li je Armstrongov. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
Unesite pozitivan trocifreni broj:
153
Broj je Armstrongov.
```

Primer 2

```
Interakcija sa programom:
Unesite pozitivan trocifreni broj:
111
Broj nije Armstrongov.
```

Primer 3

```
Interakcija sa programom:
Unesite pozitivan trocifreni broj:
84
Greska: niste uneli pozitivan trocifreni broj.
```

Primer 4

```
Interakcija sa programom:
Unesite pozitivan trocifreni broj:
371
Broj je Armstrongov.
```

[Rešenje 2.1.12]

Zadatak 2.1.13 Napisati program koji ispisuje proizvod parnih cifara unetog četvorocifrenog broja. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: 288
Greska: niste uneli cetvorocifreni broj.
```

Primer 4

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

[Rešenje 2.1.13]

Zadatak 2.1.14 Napisati program koji učitava četvorocifreni broj i ispisuje broj koji se dobija kada se unetom broju razmene najmanja i najveća cifra. U slučaju da se najmanja ili najveća cifra pojavljuju na više pozicija, uzeti prvo pojavljivanje, gledajući sa desna na levo. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
Interakcija sa programom:
Unesite cetvorocifreni broj: 1192
Rezultat: 1912
```

Primer 3

```
Interakcija sa programom:
 Unesite cetvorocifreni broj: 247
 Greska: niste uneli cetvorocifreni broj.
```

Primer 4

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

[Rešenje 2.1.14]

Zadatak 2.1.15 Napisati program koji za uneti četvorocifreni broj proverava da li su njegove cifre uređene neopadajuće, nerastuće ili nisu uređene i štampa odgovarajuću poruku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: 88
 Greska: niste uneli cetvorocifreni broj.
```

Primer 4

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

[Rešenje 2.1.15]

Zadatak 2.1.16 Napisati program koji ispituje da li se tačke $A(x_1, y_1)$ i $B(x_2, y_2)$ nalaze u istom kvadrantu. Koordinate tačaka su realni brojevi jednostruke tačnosti.

Primer 1

```
Interakcija sa programom:
 Unesite koordinate tacke A: 1.5 6
 Unesite koordinate tacke B: 2.33 9.8
 Tacke se nalaze u istom kvadrantu.
```

Primer 2

```
Interakcija sa programom:
 Unesite koordinate tacke A: -3 6
 Unesite koordinate tacke B: 0.33 -5
 Tacke se ne nalaze u istom kvadrantu.
```

Primer 3

```
Interakcija sa programom:
 Unesite koordinate tacke A: 0 -6
 Unesite koordinate tacke B: -1 -99.66
 Tacke se nalaze u istom kvadrantu.
```

Primer 4

```
Interakcija sa programom:
 Unesite koordinate tacke A: 3 -6
 Unesite koordinate tacke B: -0.33 0
 Tacke se ne nalaze u istom kvadrantu.
```

[Rešenje 2.1.16]

Zadatak 2.1.17 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.

Primer 1

```
Interakcija sa programom:
 Unesite koordinate tacke A: 1.5 6
 Unesite koordinate tacke B: -2.5 -10
 Unesite koordinate tacke C: 3 12
 Tacke se nalaze na istoj pravoj.
```

Primer 2

```
Interakcija sa programom:
 Unesite koordinate tacke A: -1.5 3
 Unesite koordinate tacke B: -0.4 9.8
 Unesite koordinate tacke C: 2 3
 Tacke se ne nalaze na istoj pravoj.
```

Primer 3

```
Interakcija sa programom:
 Unesite koordinate tacke A: 1.55 6
 Unesite koordinate tacke B: -8.4 9.8
 Unesite koordinate tacke C: 5 4.682412
 Tacke se nalaze na istoj pravoj.
```

Primer 4

```
Interakcija sa programom:
 Unesite koordinate tacke A: 5.5 3.5
 Unesite koordinate tacke B: 5.5 3.5
 Unesite koordinate tacke C: 5.5 3.5
 Tacke se nalaze na istoj pravoj.
```

Primer 5

```
Interakcija sa programom:
 Unesite koordinate tacke A: 1 2
 Unesite koordinate tacke B: 1 2
 Unesite koordinate tacke C: -56 1.3
 Tacke se nalaze na istoj pravoj.
```

Primer 6

```
Interakcija sa programom:
 Unesite koordinate tacke A: 3.4 3.5
 Unesite koordinate tacke B: -10 -1
 Unesite koordinate tacke C: -10 -1
 Tacke se nalaze na istoj pravoj.
```

[Rešenje 2.1.17]

Zadatak 2.1.18 Napisati program za rad sa intervalima. Za dva celobrojna intervala $[a_1, b_1]$ i $[a_2, b_2]$, program treba da odredi:

a) dužinu preseka datih intervala

b) presečni interval datih intervala

c) dužinu prave koju pokrivaju dati intervali

d) najmanji interval koji sadrži date intervale.

Primer 1

```
Interakcija sa programom:
 Unesite a1, b1, a2 i b2: 2 9 4 11
 Duzina preseka: 5
 Presecni interval: [4,9]
 Duzina koju pokrivaju: 9
 Najmanji interval: [2, 11]
```

Primer 2

```
Interakcija sa programom:
 Unesite a1, b1, a2 i b2: 1 2 10 13
 Duzina preseka: 0
 Presecni interval: prazan
 Duzina koju pokrivaju: 4
 Najmanji interval: [1, 13]
```

[Rešenje 2.1.18]

Zadatak 2.1.19 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 2.1.19]

Zadatak 2.1.20 U nizu 12345678910111213....9899 ispisani su redom brojevi od 1 do 99. Napisati program koji za uneti ceo broj k ($1 \leq k \leq 189$) ispisuje cifru koja se nalazi na k-toj poziciji datog niza. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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.
```

Primer 3

```
Interakcija sa programom:
 Unesite k: 200
 Greska: neispravan unos pozicije.
```

Primer 4

```
Interakcija sa programom:
 Unesite k: 10
 Na 10-toj poziciji je broj 1.
```

[Rešenje 2.1.20]

Zadatak 2.1.21 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 $F(x, k)$ koja se dobija tako što se funkcija f primeni k-puta ($F(x,1) = f(x), F(x,2) = f(f(x)), F(x,3) = f(f(f(x)))$ i ispisuje je zaokruženu na dve decimale. U slučaju neispravnog unosa, ispisati 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 2.1.21]

Zadatak 2.1.22 Napisati program koji za uneti redni broj dana u nedelji ispisuje ime odgovarajućeg dana. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: neispravan unos dana.
```

Primer 4

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

[Rešenje 2.1.22]

Zadatak 2.1.23 Napisati program koji za uneti karakter ispituje da li je samoglasnik ili ne.

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 2.1.23]

Zadatak 2.1.24 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 neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite izraz: 8 - 11
 Rezultat je: -3
```

Primer 2

```
Interakcija sa programom:
 Unesite izraz: 14 / 0
 Greska: deljenje nulom.
```

Primer 3

```
Interakcija sa programom:
 Unesite izraz: 5 ? 7
 Greska: nepoznat operator.
```

Primer 4

```
Interakcija sa programom:
 Unesite izraz: 19 / 5
 Rezultat je: 3
```

[Rešenje 2.1.24]

Zadatak 2.1.25 Napisati program koji za uneti datum u formatu *dan.mesec.* ispisuje godišnje doba kojem pripadaju. Napomena: *Pretpostaviti da je unos ispravan.*

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 2.1.25]

Zadatak 2.1.26 Napisati program koji za unetu godinu i mesec ispisuje naziv meseca kao i koliko dana ima u tom mesecu te godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite godinu: 2018
 Unesite mesec: 1
 Januar, 31 dan
```

Primer 2

```
Interakcija sa programom:
 Unesite godinu: 2000
 Unesite mesec: 2
 Februar, 29 dana
```

Primer 3

```
Interakcija sa programom:
 Unesite godinu: 2018
 Unesite mesec: 13
 Greska: neispravan unos meseca.
```

Primer 4

```
Interakcija sa programom:
 Unesite godinu: 1998
 Unesite mesec: 2
 Februar, 28 dana
```

[Rešenje 2.1.26]

Zadatak 2.1.27 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 2.1.27]

Zadatak 2.1.28 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.
```

Primer 3

```
Interakcija sa programom:
 Unesite datum: 1.1.2019.
 Prethodni datum: 31.12.2018.
```

Primer 4

```
Interakcija sa programom:
 Unesite datum: 10.12.2015.
 Prethodni datum: 9.11.2015.
```

[Rešenje 2.1.28]

Zadatak 2.1.29 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.
```

Primer 3

```
Interakcija sa programom:
 Unesite datum: 31.12.2008.
 Naredni datum: 1.1.2009.
```

Primer 4

```
Interakcija sa programom:
 Unesite datum: 5.5.2005.
 Naredni datum: 6.5.2005.
```

[Rešenje 2.1.29]

*** Zadatak 2.1.30** Polje šahovske table se definiše parom celih brojeva (x, y), $1 \leq x, y \leq 8$, gde je x redni broj reda, a y redni broj kolone. Napisati program koji za unete parove (k, l) i (m, n) proverava

a) da li su polja (k, l) i (m, 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)

Pretpostaviti da je polje $(1, 1)$ crno i da predstavlja donji levi ugao šahovske table. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite (k,l): 1 1
 Unesite (m,n): 2 2
 Polja su iste boje.
 Kraljica sa (1,1) ugrozava (2,2).
 Konj sa (1,1) ne ugrozava (2,2).
```

Primer 2

```
Interakcija sa programom:
 Unesite (k,l): 1 1
 Unesite (m,n): 3 2
 Polja su razlicite boje.
 Kraljica sa (1,1) ne ugrozava (3,2).
 Konj sa (1,1) ugrozava (3,2).
```

Primer 3

```
Interakcija sa programom:
 Unesite (k,l): 5 4
 Unesite (m,n): 3 3
 Polja su razlicite boje.
 Kraljica sa (5,4) ne ugrozava (3,3).
 Konj sa (5,4) ugrozava (3,3).
```

Primer 4

```
Interakcija sa programom:
 Unesite (k,l): 0 1
 Unesite (m,n): 3 9
 Greska: neispravna pozicija.
```

[Rešenje 2.1.30]

2.2 Rešenja

Rešenje 2.1.1

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b, c, najmanji;

  /* Ucitavaju se ulazne vrednosti. */
  printf("Unesite tri cela broja: ");
  scanf("%d%d%d", &a, &b, &c);

  /* Najmanji broj se inicijalizuje na vrednost prvog broja. */
  najmanji = a;

  /* Ako je vrednost drugog broja manji od vrednosti tekuceg
     minimuma, vrednost minimuma se azurira. */
  if (b < najmanji)
    najmanji = b;

  /* Postupak se ponavlja za treci broj. */
  if (c < najmanji)
    najmanji = c;

  /* Ispis rezultata. */
  printf("Najmanji: %d\n", najmanji);

  return 0;
}
```

Rešenje 2.1.2

```c
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float x, apsolutno_x;

  /* Ucitava se vrednost broja. */
  printf("Unesite jedan realan broj:");
  scanf("%f", &x);

  /* Racuna se apsolutna vrednost unetog broja. */
  apsolutno_x = x;
  if (x < 0)
    apsolutno_x = -x;

  /* Ispis rezultata. */
  printf("Apsolutna vrednost: %.2f\n", apsolutno_x);

```

```
  /* II nacin: koriscenjem funkcije fabs cija se deklaracija nalazi
     u zaglavlju math.h: apsolutno_x=fabs(x); */

  return 0;
}
```

Rešenje 2.1.3

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int x;
  float reciprocno_x;

  /* Ucitavanje vrednosti broja x. */
  printf("Unesite jedan ceo broj:");
  scanf("%d", &x);

  /* Vrsi se provera ispravnosti ulaznih podataka. Napomena: za
     razliku od izlaza iz programa sa kodom 0 (return 0;) koji
     sluzi kao indikator da se program zavrsio uspesno, izlaz iz
     programa sa izlaznim kodom koji se razlikuje od nule sluzi
     kao indikator da je pri izvrsavanju programa doslo do neke
     greske. */
  if (x == 0) {
    printf("Greska: nedozvoljeno je deljenje nulom.\n");
    return -1;
  }

  /* Racuna se reciprocna vrednost. */
  reciprocno_x = 1.0 / x;

  /* Ispis rezultata. */
  printf("Reciprocna vrednost: %.4f\n", reciprocno_x);

  return 0;
}
```

Rešenje 2.1.4

```
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b, c, suma;

```

```
  /* Ucitavaju se ulazne vrednosti. */
  printf("Unesite tri cela broja:");
  scanf("%d%d%d", &a, &b, &c);

  /* Pocetna vrednost sume se postavlja na 0. */
  suma = 0;

  /* Na sumu se dodaju vrednosti onih brojeva cija je vrednost
     pozitivna. Uvecavanje je moguce uraditi na dva nacina:
     I nacin: suma = suma + vrednost;
     II nacin: suma += vrednost; */
  if (a > 0)
    suma = suma + a;

  if (b > 0)
    suma += b;

  if (c > 0)
    suma += c;

  /* Ispis rezultata. */
  printf("Zbir pozitivnih: %d\n", suma);

  return 0;
}
```

Rešenje 2.1.5

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b, c;
  int najjeftiniji;
  int cena_bez_popusta, cena_sa_popustom;

  /* Ucitavaju se vrednosti cena. */
  printf("Unesite tri cene: ");
  scanf("%d%d%d", &a, &b, &c);

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (a <= 0 || b <= 0 || c <= 0) {
    printf("Greska: neispravan unos cene.");
    return -1;
  }

  /* Racuna se vrednost najjeftinijeg artikla. */
  najjeftiniji = a;

  if (b < najjeftiniji)
```

```
    najjeftiniji = b;

  if (c < najjeftiniji)
    najjeftiniji = c;

  /* Racunaju se cene sa i bez popusta. */
  cena_bez_popusta = a + b + c;
  cena_sa_popustom = cena_bez_popusta - najjeftiniji + 1;

  /* Ispis rezultata. */
  printf("Cena sa popustom: %d din\n", cena_sa_popustom);
  printf("Usteda: %d din\n", cena_bez_popusta - cena_sa_popustom);

  return 0;
}
```

Rešenje 2.1.6

```
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int sati, minuti;
  int preostali_sati, preostali_minuti;

  /* Ucitavaju se podaci o vremenu. Napomena: Vreme se zadaje u
     formatu sat:minut. Iz tog razloga je i odgovarajuci format u
     funkciji scanf %d:%d. */
  printf("Unesite vreme: ");
  scanf("%d:%d", &sati, &minuti);

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (sati > 24 || sati < 0 || minuti > 59 || minuti < 0) {
    printf("Greska: neispravan unos vremena.\n");
    return -1;
  }

  /* Racuna se preostalo vreme. */
  preostali_sati = 24 - sati - 1;
  preostali_minuti = 60 - minuti;

  if (preostali_minuti == 60) {
    /* Uvecavanje vrednosti broja za 1 se moze uraditi na vise
       nacina. Neki od njih su:
       broj = broj + 1;
       broj += 1;
       broj++; */
    preostali_sati++;
    preostali_minuti = 0;
  }
```

```
35    /* Ispis rezultata. */
      printf("Do ponoci: %d sati i %d minuta\n",
37           preostali_sati, preostali_minuti);

39    return 0;
}
```

Rešenje 2.1.7

```
1  #include <stdio.h>

3  int main()
{
5    /* Deklaracija potrebne promenljive. */
     unsigned int x;
7
     /* Ucitava se vrednost godine. */
9    printf("Unesite godinu:");
     scanf("%u", &x);
11
     /* Proverava se da li je godina prestupna ili ne i ispisuje se
13      odgovarajuca poruka. Godina je prestupna ukoliko vazi jedan od
        narednih uslova:
15      1. da je deljiva sa 4, a nije sa 100
        2. da je deljiva sa 400. */
17   if ((x % 4 == 0 && x % 100 != 0) || x % 400 == 0)
       printf("Godina je prestupna.\n");
19   else
       printf("Godina nije prestupna.\n");
21
     return 0;
23 }
```

Rešenje 2.1.8

```
   #include <stdio.h>
2
   int main()
4  {
     /* Deklaracija karakterske promenljive. */
6    char c;

8    /* Ucitava se jedan karakter. */
     printf("Unesite karakter: ");
10   scanf("%c", &c);

12   /* Ispis karaktera i vrednosti njegovog ASCII koda. */
     printf("Uneti karakter: %c\n", c);
```

```
14    printf("ASCII kod: %d\n", c);

16    /* Karakteri koji odgovaraju velikim slovima su u ASCII tablici
         smesteni sekvencijalno. Na primer, ASCII kod karaktera 'A' je
18       65, 'B' je 66, ..., 'Z' je 90. Isto vazi i za mala slova: 'a'
         je 97, 'b' je 98, ..., 'z' je 122.
20
         Odavde, ako se vrsi provera da li je neki karakter veliko
22       slovo, dovoljno je proveriti da li se njegov ASCII kod nalazi
         izmedju ASCII kodova slova 'A' i slova 'Z'.
24
         Dodatno, moze se primetiti da je razlika izmedju ASCII koda
26       svakog malog i odgovarajuceg velikog slova konstanta koja ima
         vrednost 'a'-'A', sto je isto sto i 'b'-'B', itd. Zbog toga,
28       ako je potrebno od velikog slova dobiti malo, onda je
         dovoljno ASCII kodu velikog slova dodati pomenutu konstantu.
30       Za mala slova, vazi obrnuto - da bi se dobilo veliko slovo,
         ova konstanta se oduzima. */
32
    if (c >= 'A' && c <= 'Z') {
34    printf("Odgovarajuce malo slovo: %c\n", c + ('a' - 'A'));
      printf("ASCII kod: %d\n", c + ('a' - 'A'));
36  }

38  if (c >= 'a' && c <= 'z') {
      printf("Odgovarajuce veliko slovo: %c\n", c - ('a' - 'A'));
40    printf("ASCII kod: %d\n", c - ('a' - 'A'));
    }
42
    return 0;
44 }
```

Rešenje 2.1.9

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     unsigned int broj_cifara = 0;
7    unsigned int proizvod_cifara = 1;

9    /* I nacin ucitavanja ulaza: koriscenjem funkcije getchar()
        Funkcija getchar cita jedan karakter sa ulaza i vraca njegov
11      ASCII kod. Napomena: razmaci su takodje karakteri i nece
        automatski biti preskoceni. Iz tog razloga se getchar poziva 5
13      puta u ovom primeru. Posto je poznato da su drugi i cetvrti
        karakter blanko znaci, nema potrebe da se cuva povratna
15      vrednost tih poziva. */
     int c1, c2, c3;
17   printf("Unesite karaktere: ");
```

```
    c1 = getchar();
    getchar();
    c2 = getchar();
    getchar();
    c3 = getchar();

    /* II nacin ucitavanja ulaza: koriscenjem funkcije scanf()
       Blanko znaci se navode kao deo ocekivanog formata ulaza.
       char c1, c2, c3;
       scanf("%c %c %c", &c1, &c2, &c3); */

    /* Pogresan nacin ucitavanja ulaza:
       scanf("%c%c%c", &c1, &c2, &c3);
       U ovom slucaju ce u c1 biti upisan prvi karakter, u c2
       blanko i u c3 drugi karakter. */

    /* Karakteri koji predstavljaju cifre su u ASCII tablici takodje
       smesteni sekvencijalno. Na primer, '0' ima ASCII kod 48, '1'
       49, ..., '9' ima ASCII kod 57.

       Odavde, ako se proverava da li je karakter cifra, dovoljno je
       proveriti da li se njegov ASCII kod nalazi izmedju '0' i '9'.

       Dodatno, ako je potrebno izracunati dekadnu vrednost karaktera
       koji je cifra, dovoljno je od ASCII koda tog karaktera,
       oduzeti ASCII kod karaktera '0'. Na primer, '4'-'0' = 52 - 48
       = 4. */

    /* Racuna se proizvod onih karaktera koji su cifre. */
    if (c1 >= '0' && c1 <= '9') {
      proizvod_cifara *= (c1 - '0');
      broj_cifara++;
    }

    if (c2 >= '0' && c2 <= '9') {
      proizvod_cifara *= (c2 - '0');
      broj_cifara++;
    }

    if (c3 >= '0' && c3 <= '9') {
      proizvod_cifara *= (c3 - '0');
      broj_cifara++;
    }

    /* Ispis rezultata. */
    if (broj_cifara == 0)
      printf("Medju unetim karakterima nema cifara.\n");
    else
      printf("Proizvod cifara: %u\n", proizvod_cifara);

    return 0;
}
```

Rešenje 2.1.10

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int c1, c2, c3;

  /* Ucitava se sifra artikla. */
  printf("Unesite sifru: ");
  c1 = getchar();
  c2 = getchar();
  c3 = getchar();

  /* Funkcije islower, isupper i isdigit proveravaju da li je
     prosledjeni karakter malo slovo, veliko slovo ili cifra.
     Deklaracije ovih funkcija se nalaze u zaglavlju ctype.h.

     Ukoliko prvi karakter nije ni malo slovo ni veliko slovo, ni
     cifra, ispisuje se odgovarajuca poruka o gresci i izlazi se
     iz programa. */
  if (!islower(c1) && !isupper(c1) && !isdigit(c1)) {
    printf("Greska: %c je neispravan karakter.\n", c1);
    return -1;
  }

  /* Postupak se ponavlja za druga dva karaktera. */
  if (!islower(c2) && !isupper(c2) && !isdigit(c2)) {
    printf("Greska: %c je neispravan karakter.\n", c2);
    return -1;
  }

  if (!islower(c3) && !isupper(c3) && !isdigit(c3)) {
    printf("Greska: %c je neispravan karakter.\n", c3);
    return -1;
  }

  /* Funkcija tolower(c) radi sledece: ako je c veliko slovo, kao
     povratnu vrednost vraca odgovarajuce malo slovo, u suprotnom
     vraca c. Dakle, tolower('A') je 'a', a tolower('6') = '6',...

     Slicno, samo obrnuto, radi i funkcija toupper(c). Deklaracije
     ovih funkcija se takodje nalaze u zaglavlju ctype.h. */
  c1 = tolower(c1);
  c2 = tolower(c2);
  c3 = tolower(c3);

  printf("%c%c%c\n", c1, c2, c3);

  return 0;
```

```
}
```

Rešenje 2.1.11

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n;
  char jedinica, desetica, stotina, hiljada, najveca_cifra;

  /* Ucitava se vrednost broja n. */
  printf("Unesite cetvorocifreni broj: ");
  scanf("%d", &n);

  /* Da bi program radio ispravno i za negativne brojeve, uzima se
     apsolutna vrednost broja n. */
  n = abs(n);

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (n < 1000 || n > 9999) {
    printf("Greska: niste uneli cetvorocifreni broj.\n");
    return -1;
  }

  /* Izdvajaju se cifre broja n. */
  jedinica = n % 10;
  desetica = (n / 10) % 10;
  stotina = (n / 100) % 10;
  hiljada = n / 1000;

  /* Racuna se najveca cifra broja n. */
  najveca_cifra = jedinica;

  if (desetica > najveca_cifra)
    najveca_cifra = desetica;

  if (stotina > najveca_cifra)
    najveca_cifra = stotina;

  if (hiljada > najveca_cifra)
    najveca_cifra = hiljada;

  /* Ispis rezultata */
  printf("Najveca cifra je: %d\n", najveca_cifra);

  return 0;
}
```

Rešenje 2.1.12

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n;
  char jedinica, desetica, stotina;

  /* Ucitava se vrednost broja n. */
  printf("Unesite pozitivan trocifreni broj: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (n < 100 || n > 999) {
    printf("Greska: niste uneli pozitivan trocifreni broj.\n");
    return -1;
  }

  /* Izdvajaju se cifre broja n. */
  jedinica = n % 10;
  desetica = (n / 10) % 10;
  stotina = n / 100;

  /* Ispis rezultata. */
  if (n == jedinica * jedinica * jedinica +
      desetica * desetica * desetica + stotina * stotina * stotina)
    printf("Broj je Armstrongov.\n");
  else
    printf("Broj nije Armstrongov.\n");

  return 0;
}
```

Rešenje 2.1.13

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n;
  char jedinica, desetica, stotina, hiljada;
  int broj_parnih, proizvod_parnih;

  /* Ucitava se vrednost broja n. */
  printf("Unesite cetvorocifreni broj: ");
```

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

15  /* Da bi program radio ispravno i za negativne vrednosti, uzima
       se apsolutna vrednost broja n. */
17  n = abs(n);

19  /* Vrsi se provera ispravnosti ulaznih podataka. */
    if (n < 1000 || n > 9999) {
21    printf("Greska: niste uneli cetvorocifreni broj.\n");
      return -1;
23  }

25  /* Izdvajaju se cifre broja n. */
    jedinica = n % 10;
27  desetica = (n / 10) % 10;
    stotina = (n / 100) % 10;
29  hiljada = n / 1000;

31  /* Inicijalizacija brojaca i rezultata. */
    broj_parnih = 0;
33  proizvod_parnih = 1;

35  /* Za svaku cifru se vrsi provera da li je parna i ukoliko jeste
       tekuci rezultat se mnozi sa tekucom cifrom. */
37  if (jedinica % 2 == 0) {
      proizvod_parnih = proizvod_parnih * jedinica;
39    broj_parnih++;
    }
41
    if (desetica % 2 == 0) {
43    proizvod_parnih = proizvod_parnih * desetica;
      broj_parnih++;
45  }

47  if (stotina % 2 == 0) {
      proizvod_parnih = proizvod_parnih * stotina;
49    broj_parnih++;
    }
51
    if (hiljada % 2 == 0) {
53    proizvod_parnih = proizvod_parnih * hiljada;
      broj_parnih++;
55  }

57  /* Ispis rezultata. */
    if (broj_parnih == 0) {
59    printf("Nema parnih cifara.\n");
    } else {
61    printf("Proizvod parnih cifara: %d\n", proizvod_parnih);
    }
63
    return 0;
```

```
65 }
```

Rešenje 2.1.14

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   int main()
5  {
     /* Deklaracija potrebnih promenljivih. */
7    int n, n_abs;
     char jedinica, desetica, stotina, hiljada;
9    int najveca, najmanja, stepen_najvece, stepen_najmanje;
     int rezultat;
11
     /* Ucitava se broj vrednost broja n. */
13   printf("Unesite cetvorocifreni broj: ");
     scanf("%d", &n);
15
     /* Da bi program radio ispravno i za negativne vrednosti, uzima
17      se apsolutna vrednost broja n. */
     n_abs = abs(n);
19
     /* Vrsi se provera ispravnosti ulaznih podataka. */
21   if (n_abs < 1000 || n_abs > 9999) {
       printf("Greska: niste uneli cetvorocifreni broj.\n");
23     return -1;
     }
25
     /* Izdvajaju se cifre broja n. */
27   jedinica = n_abs % 10;
     desetica = (n_abs / 10) % 10;
29   stotina = (n_abs / 100) % 10;
     hiljada = n_abs / 1000;
31
     /* Po algoritmu za trazenje najvece/najmanje cifre (koji je
33      prikazan u zadatku 2.1.11) racunaju se najveca i najmanja
        cifra broja n, kao i pozicija na kojoj se one nalaze.
35      Radi lakseg izracunavanja, pozicija se pamti kao stepen broja
        10. Na primer, pozicija cifre jedinica je 1, cifre desetica
37      10, itd... */
     najveca = jedinica;
39   stepen_najvece = 1;

41   if (desetica > najveca) {
       najveca = desetica;
43     stepen_najvece = 10;
     }
45
     if (stotina > najveca) {
47     najveca = stotina;
```

```
    stepen_najvece = 100;
  }

  if (hiljada > najveca) {
    najveca = hiljada;
    stepen_najvece = 1000;
  }

  /* Racunanje najmanje cifre. */
  najmanja = jedinica;
  stepen_najmanje = 1;

  if (desetica < najmanja) {
    najmanja = desetica;
    stepen_najmanje = 10;
  }

  if (stotina < najmanja) {
    najmanja = stotina;
    stepen_najmanje = 100;
  }

  if (hiljada < najmanja) {
    najmanja = hiljada;
    stepen_najmanje = 1000;
  }

  /* Ideja: U broju 4179, najmanja cifra je 1 i njen stepen je 100,
     a najveca cifra je 9 i njen stepen je 1. Zamena mesta se vrsi
     tako sto se oduzme 9 i doda 1, a zatim oduzme 100 i doda 900. */
  rezultat = n_abs - najveca * stepen_najvece
                   + najmanja * stepen_najvece
                   - najmanja * stepen_najmanje
                   + najveca * stepen_najmanje;

  /* Ako je pocetni broj bio negativan i rezultat treba da bude
     negativan. */
  if(n < 0)
    rezultat = -rezultat;

  /* Ispis rezultata. */
  printf("Rezultat: %d\n", rezultat);

  return 0;
}
```

Rešenje 2.1.15

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

```

```c
int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n;
  char jedinica, desetica, stotina, hiljada;

  /* Ucitava se vrednost broja n. */
  printf("Unesite cetvorocifreni broj: ");
  scanf("%d", &n);

  /* Da bi program radio ispravno i za negativne vrednosti, uzima
     se apsolutna vrednost broja n. */
  n = abs(n);

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (n < 1000 || n > 9999) {
    printf("Greska: niste uneli cetvorocifreni broj.\n");
    return -1;
  }

  /* Izdvajaju se cifre broja n. */
  jedinica = n % 10;
  desetica = (n / 10) % 10;
  stotina = (n / 100) % 10;
  hiljada = n / 1000;

  /* Ispis rezultata. */
  if (hiljada <= stotina && stotina <= desetica
      && desetica <= jedinica)
    printf("Cifre su uredjene neopadajuce. \n");
  else if (hiljada >= stotina && stotina >= desetica
           && desetica >= jedinica)
    printf("Cifre su uredjene nerastuce. \n");
  else
    printf("Cifre nisu uredjene.\n");

  return 0;
}
```

Rešenje 2.1.16

```c
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float xa, ya, xb, yb;

  /* Ucitavaju se koordinate tacaka A i B. */
  printf("Unesite koordinate tacke A: ");
  scanf("%f%f", &xa, &ya);
```

```
  printf("Unesite koordinate tacke B: ");
  scanf("%f%f", &xb, &yb);

  /* Proverava se da li su obe tacke u istom kvadrantu. */
  if ((xa >= 0 && ya >= 0 && xb >= 0 && yb >= 0) ||
      (xa <= 0 && ya >= 0 && xb <= 0 && yb >= 0) ||
      (xa >= 0 && ya <= 0 && xb >= 0 && yb <= 0) ||
      (xa <= 0 && ya <= 0 && xb <= 0 && yb <= 0)) {
    printf("Tacke se nalaze u istom kvadrantu.\n");
  } else {
    printf("Tacke se ne nalaze u istom kvadrantu.\n");
  }

  return 0;
}
```

Rešenje 2.1.17

```
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float xa, ya, xb, yb, xc, yc;
  float k, n;

  /* Ucitavaju se koordinate tacaka A, B i C. */
  printf("Unesite koordinate tacke A: ");
  scanf("%f%f", &xa, &ya);

  printf("Unesite koordinate tacke B: ");
  scanf("%f%f", &xb, &yb);

  printf("Unesite koordinate tacke C: ");
  scanf("%f%f", &xc, &yc);

  /* Ako su bilo koje dve tacke jednake, onda se sigurno sve tri
     nalaze na jednoj pravoj. */
  if ((xa == xb && ya == yb) ||
      (xa == xc && ya == yc) || (xb == xc && yb == yc)) {
    printf("Tacke se nalaze na istoj pravoj.\n");
    return 0;
  }

  /* Odredjuju se koeficijent pravca k i odsecak na y osi n, prave
     y = k*x + n koja prolazi kroz tacke A i B. Napomena: u
     slucaju kada je xb jednako xa, ova prava je paralelna sa y
     osom i k ima vrednost beskonacno, a n ima vrednost 0, tj.
     jednacina prave je x = xa (sto je isto sto i x = xb). Da bi se
     izbeglo deljenje nulom (xb-xa), ovaj slucaj se posebno
```

```
33      obradjuje. */
  if (xb != xa) {
35      k = (yb - ya) / (xb - xa);
      n = ya - k * xa;
37      /* Proverava se da li tacka C pripada pravoj y=k*x + n na
           kojoj se vec nalaze tacke A i B. */
39      if (yc == k * xc + n)
          printf("Tacke se nalaze na istoj pravoj.\n");
41      else
          printf("Tacke se ne nalaze na istoj pravoj.\n");
43    } else {
      /* Proverava se da li se i tacka C nalazi na pravoj x = xb. */
45      if (xc == xb)
          printf("Tacke se nalaze na istoj pravoj.\n");
47      else
          printf("Tacke se ne nalaze na istoj pravoj.\n");
49    }

51    /* II nacin: Tacke su kolinearne ako je:
         |xa ya 1 |
53       |xb yb 1 | = 0
         |xc yc 1 |
55       odnosno, ako je:
         xa*yb + ya*xc + xb*yc - ya*xb - xa*yc - yb*xc = 0
57
         if(xa*yb + ya*xc + xb*yc - ya*xb - xa*yc - yb*xc == 0)
59         printf("Tacke se nalaze na istoj pravoj. \n");
         else
61         printf("Tacke se ne nalaze na istoj pravoj. \n"); */

63    return 0;
  }
```

Rešenje 2.1.18

```
  #include<stdio.h>
2
  int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6   int a1, a2, b1, b2;

8   /* Ucitavaju se granice intervala. */
    printf("Unesite a1, b1, a2 i b2: ");
10  scanf("%d%d%d%d", &a1, &b1, &a2, &b2);

12  /* U zavisnosti od razlicitih polozaja dva intervala, racunaju se
       i ispisuju trazene vrednosti. */
14  if (a1 <= a2 && b1 >= a2) {
      /* I slucaj: intervali se seku i [a1,b1] je pre [a2,b2]. */
16    printf("Duzina preseka:: %d\n", b1 - a2);
```

```
    printf("Presecni interval: [%d, %d]\n", a2, b1);
    printf("Duzina koju pokrivaju: %d\n", b2 - a1);
    printf("Najmanji interval: [%d, %d]\n", a1, b2);
  } else if (a2 <= a1 && b2 >= a1) {
    /* II slucaj: intervali se seku i [a2,b2] je pre [a1,b1]. */
    printf("Duzina preseka:: %d\n", b2 - a1);
    printf("Presecni interval: [%d, %d]\n", a1, b2);
    printf("Duzina koju pokrivaju: %d\n", b1 - a2);
    printf("Najmanji interval: [%d, %d]\n", a2, b1);
  } else if (a1 >= a2 && b1 <= b2) {
    /* III slucaj: interval [a1,b1] se nalazi unutar [a2,b2]. */
    printf("Duzina preseka:: %d\n", b1 - a1);
    printf("Presecni interval: [%d, %d]\n", a1, b1);
    printf("Duzina koju pokrivaju: %d\n", b2 - a2);
    printf("Najmanji interval: [%d, %d]\n", a2, b2);
  } else if (a2 >= a1 && b2 <= b1) {
    /* IV slucaj: interval [a2,b2] se nalazi unutar [a1,b1]. */
    printf("Duzina preseka:: %d\n", b2 - a2);
    printf("Presecni interval: [%d, %d]\n", a2, b2);
    printf("Duzina koju pokrivaju: %d\n", b1 - a1);
    printf("Najmanji interval: [%d, %d]\n", a1, b1);
  } else {
    /* V slucaj: intervali su disjunktni. */
    printf("Duzina preseka:: 0\n");
    printf("Presecni interval: prazan\n");
    printf("Duzina koju pokrivaju: %d\n", b1 - a1 + b2 - a2);
    if (a1 < a2)
      printf("Najmanji interval: [%d, %d]\n", a1, b2);
    else
      printf("Najmanji interval: [%d, %d]\n", a2, b1);
  }

  return 0;
}
```

Rešenje 2.1.19

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float a, b, c;
  float D;

  /* Ucitavaju se koeficijenti kvadratne jednacine. */
  printf("Unesite koeficijente A, B i C:");
  scanf("%f%f%f", &a, &b, &c);

  /* Racunaju se resenja jednacine u zavisnosti od vrednosti
```

```
15      koeficijenata a, b i c i ispisuje se odgovarajuci rezultat. */
    if (a == 0) {
17    if (b == 0) {
        if (c == 0) {
19        /* Slucaj a==0 && b==0 && c==0: beskonacno mnogo resenja. */
          printf("Jednacina ima beskonacno mnogo resenja\n");
21      } else {
          /* Slucaj a==0 && b==0 && c!=0: nema resenja. */
23        printf("Jednacina nema resenja\n");
        }
25    } else {
        /* Slucaj a=0 && b!=0: jedinstveno resenje. */
27      printf("Jednacina ima jedinstveno realno resenje %.2f\n",
              -c / b);
29    }
    } else {
31    /* Slucaj a != 0: racuna se diskriminanta. */
      D = b * b - 4 * a * c;
33
      /* U zavisnosti od vrednosti diskriminante, ispisuje se
35       rezultat. */
      if (D < 0) {
37      printf("Jednacina nema realnih resenja\n");
      } else if (D > 0) {
39      printf("Jednacina ima dva realna resenja %.2f i %.2f\n",
              (-b + sqrt(D)) / (2 * a), (-b - sqrt(D)) / (2 * a));
41    } else {
        printf("Jednacina ima jedinstveno realno resenje %.2f\n",
43            -b / (2 * a));
      }
45  }

47  return 0;
}
```

Rešenje 2.1.20

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     int k, broj;
7
     /* Ucitava se trazena pozicija. */
9    printf("Unesite k: ");
     scanf("%d", &k);
11
     /* Vrsi se provera ispravnosti ulaznih podataka. */
13   if (k < 1 || k > 189) {
       printf("Greska: neispravan unos pozicije.\n");
```

```
    return -1;
  }

  /* Racuna se rezultat. */
  if (k < 10) {
    /* I slucaj: trazi se jednocifreni broj. */
    printf("Na %d-toj poziciji je broj %d.\n", k, k);
  } else {
    /* II slucaj: trazi se dvocifreni broj. */

    /* Ideja: izracunati broj na koji pokazuje pozicija k. Zatim,
       ako je k parno, uzeti cifru desetica tog broja, a ako je k
       neparno, uzeti cifru jedinica tog broja.

       Na primer, za k=14 i k=15, broj koji se nalazi na ovim
       pozicijama je 12, pa u slucaju da je k=14, treba ispisati 1,
       a u slucaju da je k=15, treba ispisati 2. */

    /* Odredjivanje odgovarajuceg broja: Kada bi niz izgledao
       10111213...9899, za dato k, broj bi se dobio kao 9 + k/2 + 1
       za neparne vrednosti k, odnosno 9 + k/2 za parne (dodaje se
       vrednost detet jer je prvi broj u nizu desetka.) Na primer:
       k=1, broj = 9 + 1/2 + 1 = 9 + 0 + 1 = 10 k=2, broj = 9 + 2/2
       = 10 k=3, broj = 9 + 3/2 + 1 = 9 + 1 + 1 = 11 k=4, broj = 9
       + 4/2 = 11 ... Posto ovde postoji i 9 pozicija ispred,
       potrebno je i njih uzeti u obzir - odatle: broj = 9 +
       (k-9)/2 + 1 za neparne vrednosti k, odnosno broj = 9 +
       (k-9)/2 za parne vrednosti k. */
    if (k % 2 != 0) {
      broj = 9 + (k - 9) / 2;
      printf("Na %d-toj poziciji je broj %d.\n", k, broj % 10);
    } else {
      broj = 9 + (k - 9) / 2 + 1;
      printf("Na %d-toj poziciji je broj %d.\n", k, broj / 10);
    }
  }

  return 0;
}
```

Rešenje 2.1.21

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float x, Fx;
  int k;

```

```
  /* Ucitavaju se vrednosti x i k. */
  printf("Unesite redom x i k: ");
  scanf("%f %d", &x, &k);

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (k < 1 || k > 3) {
    printf("Greska: nedozvoljena vrednost za k.\n");
    return 0;
  }

  /* U zavisnosti od vrednosti k, data funkcija ce se izracunati
     jednom, dva puta ili tri puta. */
  Fx = 2 * cos(x) - x * x * x;
  if (k > 1)
    Fx = 2 * cos(Fx) - Fx * Fx * Fx;
  if (k > 2)
    Fx = 2 * cos(Fx) - Fx * Fx * Fx;

  /* Ispis rezultata. Napomena: ispis realnih brojeva sa %g
     rezultuje ispisom na onaj broj decimala koliko sam broj ima.
     Dakle, broj 1 ce se ispisati kao 1, broj 2.33 kao 2.33, broj
     0.9999 kao 0.9999. */
  printf("F(%g,%d)=%.2f\n", x, k, Fx);

  return 0;
}
```

Rešenje 2.1.22

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int dan;

  /* Ucitava se redni broj dana u nedelji. */
  printf("Unesite broj: ");
  scanf("%d", &dan);

  /*I nacin: koriscenjem if-else naredbe.
  if(dan == 1)
    printf("ponedeljak\n");
  else if(dan == 2)
    printf("utorak\n");
  else if(dan == 3)
    printf("sreda\n");
  else if(dan == 4)
    printf("cetvrtak\n");
  else if(dan == 5)
    printf("petak\n");
```

```
  else if(dan == 6)
    printf("subota\n");
  else if(dan == 7)
    printf("nedelja\n");
  else
    printf("Greska: neispravan unos dana.\n"); */

  /* II nacin: koriscenjem switch naredbe.*/
  switch (dan) {
  case 1:
    /* Ako dan ima vrednost 1, ispisuje se ponedeljak. */
    printf("ponedeljak\n");

    /* Ako se naredba break ne navede, izvrsice se i sledeca
       naredba, tj. ispis ce biti "ponedeljak utorak". */
    break;
  case 2:
    /* Postupak se ponavlja i za ostale dane. */
    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:
    /* Ako vrednost promenljive dan nije ni jedna od vrednosti
       izmedju 1 i 7, onda je uneta vrednost neispravna. */
    printf("Greska: neispravan unos dana.\n");
  }

  return 0;
}
```

Rešenje 2.1.23

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
```

```
  char c;

  /* Ucitava se jedan karakter. */
  printf("Unesite jedan karakter:");
  scanf("%c", &c);

  /* Proverava se da li je karakter c samoglasnik, tj. da li
     odgovara nekom od sledecih karaktera: A,E,I,O,U,a,e,i,o,u. */
  switch (c) {
  case 'A':
  case 'E':
  case 'I':
  case 'O':
  case 'U':
  case 'a':
  case 'e':
  case 'i':
  case 'o':
  case 'u':
    printf("Uneti karakter je samoglasnik.\n");
    break;
  default:
    printf("Uneti karakter nije samoglasnik.\n");
    break;
  }

  return 0;
}
```

Rešenje 2.1.24

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  char op;
  int x, y;

  /* Ucitava se izraz. */
  printf("Unesite izraz: ");
  scanf("%d %c %d", &x, &op, &y);

  /* U zavisnosti od unete operacije, racuna se vrednost izraza. */
  switch (op) {
  case '+':
    printf("Rezultat je: %d\n", x + y);
    break;
  case '-':
    printf("Rezultat je: %d\n", x - y);
    break;
```

```
21    case '*':
        printf("Rezultat je: %d\n", x * y);
23      break;
      case '/':
25      if (y == 0)
          printf("Greska: deljenje nulom.\n");
27      else
          printf("Rezultat je: %d\n", x / y);
29      break;
      case '%':
31      printf("Rezultat je: %d\n", x % y);
        break;
33    default:
        printf("Greska: nepoznat operator.\n");
35    }

37    return 0;
}
```

Rešenje 2.1.25

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     int dan, mesec;
7
     /* Ucitava se vrednost datuma koji je zadat u formatu:
9       dan.mesec. */
     printf("Unesite dan i mesec");
11   scanf("%d.%d.", &dan, &mesec);

13   /* Odredjuje se godisnje doba. */
     switch (mesec) {
15   case 1:
     case 2:
17     /* Ako je mesec januar ili februar, onda je sigurno u pitanju
          zima. */
19     printf("zima\n");
       break;
21   case 3:
       /* Ako je mesec mart, onda se godisnje doba odredjuje u
23        zavisnosti od dana u mesecu. */
       if (dan < 21)
25       printf("zima\n");
       else
27       printf("prolece\n");
       break;
29   case 4:
     case 5:
```

```
    /* Ako je mesec april ili maj, onda je sigurno u pitanju
       prolece. */
    printf("prolece\n");
    break;
  case 6:
    /* Ako je mesec jun, onda se godisnje doba odredjuje u
       zavisnosti od dana u mesecu. */
    if (dan < 21)
      printf("prolece\n");
    else
      printf("leto\n");
    break;
  case 7:
  case 8:
    /* Ako je mesec jul ili avgust, onda je sigurno u pitanju
       leto. */
    printf("leto\n");
    break;
  case 9:
    /* Ako je mesec septembar, onda se godisnje doba odredjuje u
       zavisnosti od dana u mesecu. */
    if (dan < 23)
      printf("leto\n");
    else
      printf("jesen\n");
    break;
  case 10:
  case 11:
    /* Ako je mesec oktobar ili novembar, onda je sigurno u pitanju
       jesen. */
    printf("jesen\n");
    break;
  case 12:
    /* Ako je mesec decembar, onda se godisnje doba odredjuje u
       zavisnosti od dana u mesecu. */
    if (dan < 22)
      printf("jesen\n");
    else
      printf("zima\n");
  }

  return 0;
}
```

Rešenje 2.1.26

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
```

```
6   int godina;
    int mesec;
8   int prestupna;

10  /* Ucitava se vrednost godine. */
    printf("Unesite godinu: ");
12  scanf("%d", &godina);

14  /* Vrsi se provera ispravnosti ulaznih podataka. */
    if (godina < 0) {
16    printf("Greska: neispravan unos godine.\n");
      return -1;
18  }

20  /* Vrsi se provera da li je godina prestupna, zbog februara */
    if ((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
22    prestupna = 1;
    else
24    prestupna = 0;

26  /* Ucitava se redni broj meseca. */
    printf("Unesite redni broj meseca: ");
28  scanf("%d", &mesec);

30  /* U zavisnosti od vrednosti meseca, ispisuje se odgovarajuci
       rezultat. */
32  switch (mesec) {
    case 1:
34    printf("Januar, 31 dan\n");
      break;
36  case 2:
      if (prestupna)
38      printf("Februar, 29 dana\n");
      else
40      printf("Februar, 28 dana\n");
      break;
42  case 3:
      printf("Mart, 31 dan\n");
44    break;
    case 4:
46    printf("April, 30 dana\n");
      break;
48  case 5:
      printf("Maj, 31 dan\n");
50    break;
    case 6:
52    printf("Jun, 30 dana\n");
      break;
54  case 7:
      printf("Jul, 31 dan\n");
56    break;
    case 8:
```

```
58      printf("Avgust, 31 dan\n");
        break;
60    case 9:
        printf("Septembar, 30 dana\n");
62      break;
      case 10:
64      printf("Oktobar, 31 dan\n");
        break;
66    case 11:
        printf("Novembar, 30 dana\n");
68      break;
      case 12:
70      printf("Decembar, 31 dan\n");
        break;
72    default:
        printf("Greska: neispravan unos meseca.\n");
74      return -1;
      }
76
      return 0;
78  }
```

Rešenje 2.1.27

```
1   #include <stdio.h>

3   int main()
    {
5     /* Deklaracija potrebnih promenljivih. */
      int dan, mesec, godina, dozvoljeni_broj_dana;
7
      /* Ucitava se datum. */
9     printf("Unesite datum: ");
      scanf("%d.%d.%d", &dan, &mesec, &godina);
11
      /* Vrsi se provera korektnosti vrednosti unete godine. */
13    if (godina < 0) {
        printf("Datum nije korektan.\n");
15      return 0;
      }
17
      /* Vrsi se provera korektnosti vrednosti unetog meseca. */
19    if (mesec < 1 || mesec > 12) {
        printf("Datum nije korektan.\n");
21      return 0;
      }
23
      /* Vrsi se provera korektnosti vrednosti unetog dana. */
25    switch (mesec) {
      case 1:
27    case 3:
```

```
  case 5:
29  case 7:
  case 8:
31  case 10:
  case 12:
33    /* Dozvoljeni broj dana za januar, mart, maj, jul, avgust,
         oktobar i decembar je 31 */
35    dozvoljeni_broj_dana = 31;
      break;
37  case 2:
      /* Dozvoljeni broj dana za februar je 28 ili 29 u zavisnosti od
39       toga da li je godina prestupna ili ne. */
      if ((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
41      dozvoljeni_broj_dana = 29;
      else
43      dozvoljeni_broj_dana = 28;
      break;
45  case 4:
  case 6:
47  case 9:
  case 11:
49    /* Dozvoljeni broj dana za april, jun, septembar i novembar je
         30. */
51    dozvoljeni_broj_dana = 30;
      break;
53  }

55  if (dan < 0 || dan > dozvoljeni_broj_dana) {
      printf("Datum nije korektan.\n");
57    return 0;
  }
59

  /* Kako su sve provere korektnosti prosle, datum se smatra
61     korektnim. */
  printf("Datum je korektan.\n");
63

  return 0;
65}
```

Rešenje 2.1.28

```
1 #include <stdio.h>

3 int main()
{
5   /* Deklaracija potrebnih promenljivih. */
  int dan, mesec, godina;
7   int prethodni_dan, prethodni_mesec, prethodni_godina;

9   /* Ucitava se datum. */
  printf("Unesite datum: ");
```

```
  scanf("%d.%d.%d.", &dan, &mesec, &godina);

  /* Racunaju se dan, mesec i godina prethodnog dana. */
  prethodni_dan = dan - 1;
  prethodni_mesec = mesec;
  prethodni_godina = godina;

  /* Ako je potrebno, vrse se korekcije. */
  if (prethodni_dan == 0) {
    prethodni_mesec = mesec - 1;
    if (prethodni_mesec == 0) {
      prethodni_mesec = 12;
      prethodni_godina = godina - 1;
    }

    switch (prethodni_mesec) {
    case 1:
    case 3:
    case 5:
    case 7:
    case 8:
    case 10:
    case 12:
      prethodni_dan = 31;
      break;
    case 2:
      if ((prethodni_godina % 4 == 0 && prethodni_godina % 100 != 0)
          || prethodni_godina % 400 == 0)
        prethodni_dan = 29;
      else
        prethodni_dan = 28;
      break;
    case 4:
    case 6:
    case 9:
    case 11:
      prethodni_dan = 30;
    }
  }

  /* Ispis rezultata. */
  printf("Prethodni datum: %d.%d.%d.\n",
         prethodni_dan, prethodni_mesec, prethodni_godina);

  return 0;
}
```

Rešenje 2.1.29

Rešenje je analogno rešenju zadatka 2.1.28.

Rešenje 2.1.30

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int k, l, m, n;

  /* Ucitavaju se vrednosti pozicija na tabli. */
  printf("Unesite (k,l): ");
  scanf("%d%d", &k, &l);

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

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (k < 1 || k > 8 || l < 1 || l > 8 ||
      m < 1 || m > 8 || n < 1 || n > 8) {
    printf("Greska: neispravna pozicija.\n");
    return -1;
  }

  if(k == m && l == n){
    printf("Greska: pozicije moraju biti razlicite.\n");
    return -1;
  }

  /* Proverava se da li su (k,l) i (m,n) iste boje. Polja su iste
     boje ako su: 1) oba reda parna i obe kolone parne ILI 2) oba
     reda neparna i obe kolone neparne. */
  if (((k % 2 == m % 2) && (l % 2 == n % 2))
      || ((k % 2 != m % 2) && (l % 2 != n % 2)))
    printf("Polja su iste boje.\n");
  else
    printf("Polja su razlicite boje.\n");

  /* Proverava se da li kraljica sa (k,l) napada polje (m,n).
     Kraljica napada polje u sledecim situacijama:
     1) Ako se nalaze u istom redu (k==m)
     2) Ako se nalaze u istoj koloni (l==n)
     3) Ako se nalaze na istoj dijagonali. Dijagonala moze biti:
        a) paralelna glavnoj dijagonali (abs(k-l) == abs(m-n))
        b) paralelna sporednoj dijagonali (k+l == m+n) */
  if ((k == m) || (l == n) || (abs(k - l) == abs(m - n))
      || (k + l == m + n)){
    printf("Kraljica sa (%d, %d) ugrozava (%d, %d).\n",
           k, l, m, n);
  }
  else {
    printf("Kraljica sa (%d, %d) ne ugrozava (%d, %d).\n",
```

```
            k, l, m, n);
52  }

54  /* Proverava se da li konj sa (k, l) napada polje (m, n). Postoji
       8 mogucih vrednosti za polja koja konj napada. Vrsi se
56     provera da li je (m,n) jednako nekom od tih polja. */
    if ((abs(k-m) == 2 &&  abs(n-l) == 1) || (abs(n-l) == 2 && abs(m-k)
       == 1))
58    printf("Konj sa (%d, %d) ugrozava (%d, %d).\n",
            k, l, m, n);
60  else
      printf("Konj sa (%d, %d) ne ugrozava (%d, %d).\n",
62          k, l, m, n);

64  return 0;
}
```

2.3 Petlje

Zadatak 2.3.1 Napisati program koji pet 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 pozitivan ceo broj n i n puta ispisuje tekst `Mi volimo da programiramo`. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 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 broj n: 0
 Greska: pogresan unos broja n.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: -5
 Greska: pogresan unos broja n.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 1
 Mi volimo da programiramo.
```

[Rešenje 2.3.2]

Zadatak 2.3.3 Napisati program koji učitava nenegativan ceo broj n a potom ispisuje sve cele brojeve od 0 do n. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite broj n: -10
 Greska: pogresan unos broja n.
```

[Rešenje 2.3.3]

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

(a) Koristiti `while` petlju.

(b) Koristiti `for` petlju.

(c) Koristiti `do-while` petlju.

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite granice intervala: -2 4
 -2 -1 0 1 2 3 4
```

Primer 2

```
Interakcija sa programom:
 Unesite granice intervala: 10 6
 Greska: pogresan unos granica.
```

[Rešenje 2.3.4]

Zadatak 2.3.5 Napisati program koji učitava nenegativan ceo broj n i izračunava njegov faktorijel. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 18
 18! = 6402373705728000
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 8
 8! = 40320
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 40
 Pri racunanju 40! ce doci do prekoracenja.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: -5
 Greska: neispravan unos.
```

[Rešenje 2.3.5]

Zadatak 2.3.6 Napisati program koji učitava realan broj x i ceo nenegativan broj n i izračunava n-ti stepen broja x, tj. x^n. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 4 3
 Rezultat: 64.00000
```

Primer 2

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 5.8 5
 Rezultat: 6563.56768
```

Primer 3

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 11.43 -6
 Greska: neispravan unos broja n.
```

Primer 4

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 11.43 0
 Rezultat: 1.00000
```

[Rešenje 2.3.6]

Zadatak 2.3.7 Napisati program koji učitava realan broj x i ceo broj n i izračunava n-ti stepen broja x.

Primer 1

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 2 -3
 Rezultat: 0.125
```

Primer 2

```
Interakcija sa programom:
 Unesite redom brojeve x i n: -3 2
 Rezultat: 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 pozitivan ceo broj n ispisuje sve njegove prave delioce. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 100
 2 4 5 10 20 25 50
```

Primer 2

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

[Rešenje 2.3.8]

Zadatak 2.3.9 Napisati program koji za uneti ceo broj ispisuje broj dobijen uklanjanjem svih nula sa desne strane unetog broja.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 12000
 12
```

Primer 2

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

Primer 3

```
Interakcija sa programom:
 Unesite broj: -1400
 -14
```

Primer 4

```
Interakcija sa programom:
 Unesite broj: 147
 147
```

[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 pozitivan ceo broj ispisuje da li je on deljiv sumom svojih cifara. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 12
 Broj 12 je deljiv sa 3.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 2564
 Broj 2564 nije deljiv sa 17.
```

Primer 3

```
Interakcija sa programom:
Unesite broj: -4
Greska: neispravan ulaz.
```

Primer 4

```
Interakcija sa programom:
Unesite broj: 0
Greska: neispravan ulaz.
```

[Rešenje 2.3.11]

Zadatak 2.3.12 Knjigovođa vodi evidenciju o transakcijama jedne firme i treba da napiše izveštaj o godišnjem poslovanju te firme. Firma je tokom godine imala t transakcija. Transakcije su predstavljene celim brojevima i u slučaju da je vrednost transakcije pozitivna, ta transakcija označava prihod firme, a u slučaju da je negativna rashod. Napisati program koji učitava nenegativan ceo broj t i podatke o t transakcija i zatim izračunava i ispisuje ukupan prihod, ukupan rashod i zaradu, odnosno gubitak koji je firma ostvarila tokom godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
Unesite broj t: 7
Unesite transakcije:
8 -50 45 2007 -67 -123 14
Prihod: 2074
Rashod: -240
Zarada: 1834
```

Primer 2

```
Interakcija sa programom:
Unesite broj t: 5
Unesite transakcije:
-5 -20 -4 -200 -8
Prihod: 0
Rashod: -237
Gubitak: 237
```

Primer 3

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

Primer 4

```
Interakcija sa programom:
Unesite broj n: 0
Nema evidentiranih transakcija.
```

Primer 5

```
Interakcija sa programom:
Unesite broj t: 2
Unesite transakcije:
120 -120
Prihod: 120
Rashod: -120
Zarada: 0
```

[Rešenje 2.3.12]

Zadatak 2.3.13 Napisati program koji učitava pozitivan ceo broj n, a potom i n celih brojeva. Izračunati i ispisati zbir onih brojeva koji su istovremeno neparni i negativni. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
Unesite broj n: 5
Unesite n brojeva:
1 -5 -6 3 -11
Zbir neparnih i negativnih: -16
```

Primer 2

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

Primer 3

```
Interakcija sa programom:
Unesite broj n: 4
Unesite n brojeva:
5 8 13 17
Zbir neparnih i negativnih: 0
```

Primer 4

```
Interakcija sa programom:
Unesite broj n: 0
Greska: neispravan unos.
```

[Rešenje 2.3.13]

Zadatak 2.3.14 Napisati program koji učitava pozitivan ceo broj n, a potom n celih brojeva i računa i ispisuje sumu brojeva koji su deljivi sa 5, a nisu deljivi sa 7. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

```
Interakcija sa programom:
Unesite broj n: 10
Unesite n brojeva:
-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 Napisati program koji učitava cele brojeve sve dok se ne unese nula i ispisuje proizvod onih unetih brojeva koji su pozitivni.

Primer 1

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

Primer 2

```
Interakcija sa programom:
Unesite brojeve: 0
Nije unet nijedan broj.
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve:
 -5 -200 -43 0
 Medju unetim brojevima nema pozitivnih.
```

Primer 4

```
Interakcija sa programom:
 Unesite brojeve: 1 0
 Proizvod pozitivnih brojeva je 1.
```

[Rešenje 2.3.15]

Zadatak 2.3.16 Napisati program koji za uneti ceo broj proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 1857
 Broj 1857 sadrzi cifru 5.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 84
 Broj 84 ne sadrzi cifru 5.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -2515
 Broj -2515 sadrzi cifru 5.
```

[Rešenje 2.3.16]

Zadatak 2.3.17 Napisati program koji učitava cele brojeve sve do unosa broja nula, a zatim 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: 0
 Nisu uneti brojevi.
```

Primer 3

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

[Rešenje 2.3.17]

Zadatak 2.3.18 U prodavnici se nalaze artikali čije su cene pozitivni realni brojevi. Napisati program koji učitava cene artikala sve do unosa broja nula i izračunava i ispisuje prosečnu vrednost cena u radnji. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite cene: 6.32 -9
 Greska: neispravan unos cene.
```

Primer 3

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

[Rešenje 2.3.18]

Zadatak 2.3.19 Napisati program koji učitava pozitivan ceo broj n, a potom n realnih brojeva, a zatim određuje i ispisuje koliko puta je prilikom unosa došlo do promene znaka. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 9
 Unesite brojeve:
7.82 4.3 -1.2 56.8 -3.4 -72.1 8.9 11.2 -11.2
 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
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 0
 Greska: neispravan unos.
```

[Rešenje 2.3.19]

Zadatak 2.3.20 U prodavnici se nalazi n artikala čije su cene pozitivni realni brojevi. Napisati program koji učitava n, a potom i cenu svakog od n artikala i određuje i ispisuje najmanju cenu. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj artikla: 6
 Unesite cene artikala:
12 3.4 90 100.53 53.2 12.8
 Najmanja cena: 3.400000
```

Primer 2

```
Interakcija sa programom:
 Unesite broj artikla: 3
 Unesite cene artikala:
4 -8 92
 Greska: neispravan unos cene.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj artikla: -9
 Greska: neispravan unos.
```

[Rešenje 2.3.20]

Zadatak 2.3.21 Nikola želi da obraduje baku i da joj kupi jedan poklon u radnji. On na raspolaganju ima m dinara. 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 odredi broj atikala. Program učitava realan nenegativan broj m, ceo nenegativan broj n i n pozitivnih realnih brojeva. Ispisati koliko artikala ima cenu čija je vrednost manja ili jednaka m. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Nikolin budzet: 12.37
 Unesite broj artikala: 5
 Unesite cene artikala: 11 54.13 6 13 8
 Ukupno artikala: 3
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Nikolin budzet: 2
 Unesite broj artikala: 4
 Unesite cene artikala: 1 11 4.32 3
 Ukupno artikala: 1
```

[Rešenje 2.3.21]

Zadatak 2.3.22 Napisati program koji učitava ceo nenegativan broj n, n celih brojeva i zatim izračunava i ispisuje tražene vrednosti. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

(a) 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: 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: -89.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: 4
 Unesite brojeve:
 100 200 300 400
 Broj sa najvecom cifrom desetica: 100.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: -12
 Greska: neispravan unos.
```

(b) Broj sa najvećim brojem cifara. Ukoliko ima više takvih, ispisati prvi.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite n brojeva: 18 -365 251 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.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 0
 Nisu uneti brojevi.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: -7
 Greska: neispravan unos.
```

Primer 5

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite n brojeva: 0 1 2 -3 4
 Najvise cifara ima broj 0.
```

Primer 6

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite n brojeva: -5 4 -3 2 1
 Najvise cifara ima broj -5.
```

(c) 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 2

```
Interakcija sa programom:
 Unesite broj n: 3
 Unesite n brojeva: 0 0 0
 Broj sa najvecom vodecom cifrom je 0.
```

Primer 3

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

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 0
 Nisu uneti brojevi.
```

[Rešenje 2.3.22]

Zadatak 2.3.23 Vršena su merenja nadmorskih visina na određenom delu teritorije i naučnike zanima razlika između najveće i najmanje nadmorske visine. Napisati program koji učitava cele 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
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve: 0
 Nisu unete nadmorske visine.
```

Primer 4

```
Interakcija sa programom:
 Unesite brojeve: -500 0
 Razlika: 0
```

Primer 5

```
Interakcija sa programom:
 Unesite brojeve: -500 -300 -5000 0
 Razlika: 4700
```

[Rešenje 2.3.23]

Zadatak 2.3.24 Napisati program koji učitava ceo broj n ($n > 1$), nenegativan ceo broj d, a zatim i n celih brojeva i izračunava i ispisuje koliko ima parova uzastopnih brojeva među unetim brojevima koji se nalaze na rastojanju d. Rastojanje između brojeva je definisano sa $d(x, y) = |y - x|$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve n i d: 5 0
 Unesite n brojeva: 1 1 1 1 1
 Broj parova: 4
```

Primer 4

```
Interakcija sa programom:
 Unesite brojeve n i d: 1 3
 Greska: neispravan unos.
```

[Rešenje 2.3.24]

Zadatak 2.3.25 Napisati program koji uneti pozitivan ceo broj transformiše tako što svaku parnu cifru u zapisu broja uveća za jedan. Ispisati dobijeni broj. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 2417
 Rezultat: 3517
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 138
 Rezultat: 139
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 59
 Rezultat: 59
```

[Rešenje 2.3.25]

Zadatak 2.3.26 Napisati program koji učitava jedan ceo broj i zatim formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre polaznog broja, idući sa desna na levo.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 21854
 Rezultat: 284
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 18
 Rezultat: 8
```

Primer 3

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

Primer 4

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

[Rešenje 2.3.26]

*** Zadatak 2.3.27** Napisati program koji na osnovu unetog pozitivnog celog broja formira i ispisuje broj koji se dobija izbacivanjem cifara koje su u polaznom broju jednake zbiru svojih suseda. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
 Greska: neispravan unos.
```

[Rešenje 2.3.27]

*** Zadatak 2.3.28** Broj je *palindrom* ukoliko se isto čita i sa leve i sa desne strane. Napisati program koji učitava pozitivan ceo broj i proverava da li je učitani broj palindrom. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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 2.3.28]

Zadatak 2.3.29 Fibonačijev niz počinje ciframa 0 i 1, a svaki član se dobija kao zbir prethodna dva. Napisati program koji učitava nenegativan ceo broj n i određuje i ispisuje n-ti član Fibonačijevog niza. Niz se indeksira počevši od nule. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 10
 F[10] = 55
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: -100
 Greska: neispravan unos.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 40
 F[40] = 102334155
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 20
 F[20] = 6765
```

[Rešenje 2.3.29]

Zadatak 2.3.30 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 (pozitivan ceo broj) štampa niz brojeva sve do onog člana niza koji je jednak 1. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite prvi clan: -48
 Greska: neispravan unos.
```

Primer 3

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

[Rešenje 2.3.30]

*** Zadatak 2.3.31** 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 nenegativan broj k ispisuje dimenzije papira A_k u milimetrima. Rezultat ispisati kao celobrojne vrednosti. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite format papira: 0
 840 1189
```

Primer 3

```
Interakcija sa programom:
 Unesite format papira: -7
 Greska: neispravan unos.
```

Primer 4

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

[Rešenje 2.3.31]

Zadatak 2.3.32 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 2.3.32]

Zadatak 2.3.33 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 2.3.33]

Zadatak 2.3.34 Program učitava pozitivan ceo 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. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 3

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

Primer 4

```
Interakcija sa programom:
 Unesite broj n: -7
 Greska: neispravan unos.
```

[Rešenje 2.3.34]

Zadatak 2.3.35 Program učitava pozitivan ceo broj n, a zatim i n karaktera. Napisati program koji proverava da li se od unetih karaktera može napisati reč *Zima*. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 0
 Greska: neispravan unos.
```

[Rešenje 2.3.35]

Zadatak 2.3.36 Napisati program koji učitava pozitivan ceo broj n i ispisuje vrednost sume kubova brojeva od 1 do n, odnosno $s = 1 + 2^3 + 3^3 + \ldots + n^3$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 14
 Suma kubova: 11025
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 25
 Suma kubova: 105625
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 0
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: -5
 Greska: neispravan unos.
```

[Rešenje 2.3.36]

Zadatak 2.3.37 Napisati program koji učitava pozitivan ceo broj n i ispisuje sumu kubova, $s = 1+2^3+3^3+\ldots+k^3$, za svaku vrednost $k = 1, \ldots, n$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 k=1, suma=1
 k=2, suma=9
 k=3, suma=36
 k=4, suma=100
 k=5, suma=225
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 8
 k=1, suma=1
 k=2, suma=9
 k=3, suma=36
 k=4, suma=100
 k=5, suma=225
 k=6, suma=441
 k=7, suma=784
 k=8, suma=1296
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 0
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: -5
 Greska: neispravan unos.
```

[Rešenje 2.3.37]

Zadatak 2.3.38 Napisati program koji učitava realan broj x i pozitivan ceo broj n i izračunava i ispisuje sumu $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \ldots + n \cdot x^n$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 3

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 5.5 0
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite redom brojeve x i n: -0.5 -5
 Greska: neispravan unos.
```

[Rešenje 2.3.38]

Zadatak 2.3.39 Napisati program koji učitava realan broj x i pozitivan ceo broj n i izračunava i ispisuje sumu $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \ldots \frac{1}{x^n}$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 3

```
Interakcija sa programom:
 Unesite redom brojeve x i n: 5.5 0
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite redom brojeve x i n: -0.5 -5
 Greska: neispravan unos.
```

[Rešenje 2.3.39]

*** Zadatak 2.3.40** Napisati program koji učitava realne 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 2.3.40]

*** Zadatak 2.3.41** Napisati program koji učitava realne 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 2.3.41]

Zadatak 2.3.42 Napisati program koji učitava realan broj x i pozitivan ceo broj n i izračunava proizvod $P = (1 + \cos(x)) \cdot (1 + \cos(x^2)) \cdot \ldots \cdot (1 + \cos(x^n))$. U

slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Voditi računa o efikasnosti rešenja.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite redom brojeve x i n: 3.4 5
 P = 0.026817
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite redom brojeve x i n: 12 8
 P = 2.640565
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite redom brojeve x i n: 12 0
 Greska: neispravan unos.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite redom brojeve x i n: 12 -6
 Greska: neispravan unos.
```

[Rešenje 2.3.42]

*** Zadatak 2.3.43** Napisati program koji učitava pozitivan ceo broj n i ispisuje vrednost razlomka

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: 4
 R = 0.697674
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: 20
 R = 0.697775
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: 0
 Greska: neispravan unos.
```

[Rešenje 2.3.43]

*** Zadatak 2.3.44** Napisati program koji učitava realan broj x i pozitivan ceo broj n i računa sumu

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.*

Primer 1

```
Interakcija sa programom:
 Unesite x i n: 5.6 8
 S=0.779792
```

Primer 2

```
Interakcija sa programom:
 Unesite x i n: 14.32 11
 S=-6714.066406
```

Primer 3

```
Interakcija sa programom:
 Unesite x i n: 2 -6
 Greska: neispravan unos.
```

[Rešenje 2.3.44]

*** Zadatak 2.3.45** Napisati program koji učitava pozitivan ceo broj n i koji računa proizvod

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. Napomena: *Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.*

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 P = 1.838108
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 7
 P = 1.841026
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 0
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 10
 P = 1.841077
```

[Rešenje 2.3.45]

*** Zadatak 2.3.46** Napisati program koji učitava neparan ceo broj n ($n \geq 5$) i izračunava i ispisuje sumu

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

U slučaju greške 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
 Greska: neispravan unos.
```

Primer 4

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

[Rešenje 2.3.46]

Zadatak 2.3.47 Napisati program koji učitava realne brojeve x i a i pozitivan ceo broj n i zatim izračunava i ispisuje vrednost izraza

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve x i a: 3.2 0.2
 Unesite broj n: 5
 Izraz = 135380494030332048.000000
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve x i a: 2 1
 Unesite broj n: 3
 Izraz = 10201.000000
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve x i a: 2.6 0.3
 Unesite broj n: 3
 Izraz = 5800.970129
```

Primer 4

```
Interakcija sa programom:
 Unesite brojeve x i a: 5.4 7
 Unesite broj n: -2
 Greska: neispravan unos.
```

[Rešenje 2.3.47]

Zadatak 2.3.48 Napisati programe koji za unetu pozitivnu celobrojnu vrednost n ispisuju tražene tablice. Napomena: *Pretpostaviti da je unos ispravan.*

(a) Napisati program koji za unetu 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 2.3.48]

Zadatak 2.3.49 Napisati program koji za uneti pozitivan ceo broj n zvezdicama iscrtava odgovarajuću sliku. NAPOMENA: *Pretpostaviti da je unos ispravan.*

(a) Slika predstavlja 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 predstavlja rub kvadrata dimenzije n.

Primer 1

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

Primer 2

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

(c) Slika predstavlja 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 2.3.49]

*** Zadatak 2.3.50** Napisati program koji za uneti pozitivan ceo broj n zvezdicama iscrtava slovo X dimenzije n. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 2.3.50]

*** Zadatak 2.3.51** Napisati program koji za uneti neparan pozitivan broj n korišćenjem znaka + iscrtava veliko + dimenzije n. Napomena: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 2.3.51]

Zadatak 2.3.52 Napisati program koji učitava pozitivan ceo broj n, a potom iscrtava odgovarajuću sliku. Napomena: *Pretpostaviti da je unos ispravan.*

(a) Slika predstavlja 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 predstavlja 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 predstavlja 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 predstavlja 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 predstavlja 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 predstavlja 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 2.3.52]

Zadatak 2.3.53 Napisati program koji učitava pozitivan ceo broj n, a potom iscrtava odgovarajuću sliku. Napomena: *Pretpostaviti da je unos ispravan.*

(a) Slika predstavlja 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 predstavlja 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 predstavlja 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 predstavlja 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 2.3.54** Napisati program koji za uneti pozitivan ceo broj n iscrtava strelice dimenzije n. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 2.3.54]

*** Zadatak 2.3.55** Napisati program koji učitava pozitivan 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. Napomena: *Pretpostaviti da je unos ispravan.*

Primer 1

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

[Rešenje 2.3.55]

*** Zadatak 2.3.56** Napisati program koji učitava pozitivne cele brojeve m i n i iscrtava jedan do drugog n kvadrata čija je svaka strana sastavljena od m zvezdica razdvojenih prazninom. Napomena: *Pretpostaviti da je unos ispravan.*

Primer 1

```
Interakcija sa programom:
 Unesite brojeve n i m: 5 3
 * * * * * * * * * * * * * *
 *       *       *       *
 *       *       *       *
 *       *       *       *
 * * * * * * * * * * * * * *
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve n i m: 4 4
 * * * * * * * * * * * * * * *
 *     *     *     *     *
 *     *     *     *     *
 * * * * * * * * * * * * * * *
```

[Rešenje 2.3.56]

*** Zadatak 2.3.57** Napisati program koji učitava pozitivan ceo broj n i štampa romb sastavljen od minusa u pravougaoniku sastavljenom od zvezdica. Napomena: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 2.3.57]

Zadatak 2.3.58 Napisati program koji učitava ceo broj n ($n \geq 2$) i koji iscrtava sliku kuće sa krovom: kuća je kvadrat 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 2.3.58]

*** Zadatak 2.3.59** Napisati program koji učitava pozitivan ceo broj n i 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. NAPOMENA: *Pretpostaviti da je unos ispravan.*

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: 6
 1 2 3 4 5 6 2 3 4 5 3 4
```

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 2.3.59]

*** Zadatak 2.3.60** Napisati program koji učitava pozitivan ceo 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). NAPOMENA: *Pretpostaviti da je unos ispravan.*

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
```

[Rešenje 2.3.60]

2.4 Rešenja

Rešenje 2.3.1

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int i;

  /* Promenljiva i kontrolise koliko puta ce se petlja izvrsiti i
     naziva se brojac petlje. Njena pocetna vrednost se postavlja na
     0 jer se u pocetku petlja nije ni jednom izvrsila. */
  i = 0;

  /* Petlja ce se izvrsiti za i=0,1,2,3,4. Kada i dostigne vrednost
     5 uslov i < 5 nece biti ispunjen i prelazi se na prvu sledecu
     naredbu nakon tela petlje. */
  while (i < 5) {

    /* Ispis poruke. */
    printf("Mi volimo da programiramo.\n");

    /* Uvecavanje brojaca za 1. */
    i++;
  }

  return 0;
}
```

Rešenje 2.3.2

```c
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int i, n;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: pogresan unos broja n.\n");
    return -1;
  }
```

```
17
   /* Inicijalizacija brojaca. */
19 i = 0;

21 /* Trazena poruka se ispisuje n puta. */
   while (i < n) {
23   printf("Mi volimo da programiramo.\n");
     i++;
25 }

27 return 0;
}
```

Rešenje 2.3.3

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     int i, n;
7
     /* Ucitava se vrednost broja n. */
9    printf("Unesite broj n: ");
     scanf("%d", &n);
11
     /* Vrsi se provera ispravnosti ulaza. */
13   if (n < 0) {
       printf("Greska: pogresan unos broja n.\n");
15     return -1;
     }
17
     /* Inicijalizacija brojaca. */
19   i = 0;

21   /* Posto je potrebno ispisati sve brojeve [0,n], telo petlje
        se izvrsava za svako i <= n. */
23   while (i <= n) {

25     /* Ispisuje se trenutna vrednost brojaca. */
       printf("%d\n", i);
27
       /* Prelazi se na sledeci broj.  */
29     i++;
     }
31
     return 0;
33 }
```

Rešenje 2.3.4

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, m, i;

  /* Ucitavaju se vrednosti granica intervala. */
  printf("Unesite granice intervala: ");
  scanf("%d%d", &n, &m);

  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (m < n) {
    printf("Greska: pogresan unos granica.\n");
    return -1;
  }

  /* a) I nacin: koriscenjem while petlje. */
  /* Inicijalizacija brojaca na levu granicu intervala. */
  i = n;

  /* Ispisuju se sve vrednosti brojaca izmedju leve i desne
     granice intervala, ukljucujuci i same granice. */
  while (i <= m) {
    printf("%d ", i);
    i++;
  }

  /* b) II nacin: koriscenjem for petlje.

     Naredba i=n se izvrsava jednom, pre prve iteracije.
     Uslov petlje i<=m se proverava pre svake iteracije.
     Naredba i++ se izvrsava nakon svake iteracije.

  for (i = n; i <= m; i++){
    printf("%d ", i);
  }  */

  /* c) III nacin: koriscenjem do while petlje.

     Uslov petlje se proverava na kraju svake iteracije.
     Zbog toga se do while petlja izvrsava bar jednom, cak i u
     slucaju da uslov petlje nikada nije ispunjen. U ovom slucaju
     je to ispravno jer je poznato da ce interval imati bar
     jedan element. U opstem slucaju to ne mora da vazi.

  i = n;
  do {
    printf("%d ", i);
    i++;
```

```
  }
  while (i <= m);  */

  printf("\n");

  return 0;
}
```

Rešenje 2.3.5

```
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, tekuca_vrednost;

  /* Za cuvanje vrednosti faktorijela se koristi tip unsigned long
     jer izracunata vrednost moze da bude jako veliki broj. */
  unsigned long faktorijel;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n < 0) {
    printf("Greska: neispravan unos..\n");
    return -1;
  }

  if (n >= 22) {
    printf("Pri racunanju %d! ce doci do prekoracenja.\n", n);
    return -1;
  }

  /* Tekuca vrednost uzima vrednosti n, n-1, n-2, ..., 2.
     Na pocetku se inicijalizuje na n, a zatim se u svakoj
     iteraciji umanjuje za 1. */
  tekuca_vrednost = n;

  /* Inicijalizacija vrednosti faktorijela. */
  faktorijel = 1;

  /* Racuna se vrednost faktorijela tako sto se trenutni rezultat
     u svakoj iteraciji mnozi sa promenljivom cija vrednost krece
     od n, a zatim se u svakoj iteraciji umanjuje za 1. */
  while (tekuca_vrednost > 1) {
    faktorijel = faktorijel * tekuca_vrednost;
    tekuca_vrednost--;
  }
```

```
43    /* Ispis rezultata. */
      printf("%d! = %lu\n", n, faktorijel);
45
      return 0;
47  }
```

Rešenje 2.3.6

```
1   #include <stdio.h>

3   int main()
    {
5     /* Deklaracije potrebnih promenljivih. */
      int n, i;
7     float x, rezultat;

9     /* Ucitavaju se vrednosti brojeva x i n. */
      printf("Unesite redom brojeve x i n: ");
11    scanf("%f %d", &x, &n);

13    /* Vrsi se provera ispravnosti ulaza. */
      if (n < 0) {
15      printf("Greska: neispravan unos broja n.\n");
        return -1;
17    }

19    /* Inicijalizacija rezultata. */
      rezultat = 1;
21
      /* Vrednost n-tog stepena broja x se dobija tako sto se tekuca
23       vrednost rezultata n puta pomnozi sa brojem x.
         (rezultat = x * x * ... * x) = x^n */
25    for (i=0; i<n; i++)
        rezultat = rezultat * x;
27
      /* Ispis rezultata. */
29    printf("Rezultat: : %.5f\n", rezultat);

31    return 0;
    }
```

Rešenje 2.3.7

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    int main()
5   {
```

```
  /* Deklaracije potrebnih promenljivih. */
  int n, i, znak;
  float x, rezultat;

  /* Ucitavaju se vrednosti brojeva x i n. */
  printf("Unesite redom brojeve x i n: ");
  scanf("%f %d", &x, &n);

  /* Pamti se znak stepena i uzima se apsolutna vrednost stepena. */
  znak = 1;
  if(n < 0){
    znak = -1;
    n = abs(n);
  }
  /* Inicijalizacija rezultata. */
  rezultat = 1;

  /* Racuna se vrednost x^n. */
  for (i=0; i<n; i++)
    rezultat = rezultat * x;

  /* Ako je stepen bio negativan, rezultat je 1/x^n. */
  if (znak == -1)
    printf("Rezultat: %.3f\n", 1 / rezultat);
  else
    printf("Rezultat: %.3f\n", rezultat);

  return 0;
}
```

Rešenje 2.3.8

```
#include<stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, i;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* I nacin: Za svaki broj iz intervala [2, n-1] se proverava da
     li deli broj n (tj. da li je ostatak pri deljenju sa n jednak
```

```
     nuli). Ako je uslov ispunjen, taj broj se ispisuje.
    for (i = 2; i < n; i++) {
      if (n % i == 0)
        printf("%d ", i);
    }
    printf("\n");
    */

  /* II nacin (brzi): Provera se ne vrsi za sve brojeve iz
     intervala [2, n-1], vec za brojeve iz intervala
     [2, sqrt(n)], tj. za sve brojeve k za koje vazi da je
     k*k <= n. */
  for (i = 2; i*i <= n; i++) {
    /* Ako i deli n, treba razlikovati dva slucaja. */
    if (n % i == 0){
      if (i == n / i) {
        /* I slucaj: kada je i koren broja, npr. 4 za 16,
         ispisuje se samo broj i. */
        printf("%d ", i);
      }
      else {
        /* II slucaj: u suprotnom, ispisuje se taj broj i
           broj n / i, npr. 2 za 16, ispisuju se i 2 i 8. */
        printf("%d %d ", i, n / i);
      }
    }
  }
  printf("\n");

  return 0;
}
```

Rešenje 2.3.9

```
#include <stdio.h>

int main()
{
  /* Deklaracija broja n. */
  int n;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj: ");
  scanf("%d", &n);

  /* Slucaj kada broj n ima vrednost nula se posebno obradjuje.
     Kada ovo ne bi bilo navedeno, petlja u nastavku bi se
     u ovom slucaju izvrsavala beskonacno. */
  if (n == 0) {
    printf("0\n");
    return 0;
```

```
  }

  /* Dok god je poslednja cifra broja n nula, broj n se deli sa
      10 i na taj nacin se iz broja uklanja poslednja cifra. */
  while (n % 10 == 0)
    n = n / 10;

  /* Ispis rezultata. */
  printf("%d\n", n);

  return 0;
}
```

Rešenje 2.3.10

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int x;

  /* Ucitava se vrednost broja x. */
  printf("Unesi ceo broj:");
  scanf("%d", &x);

  /* Uzima se apsolutna vrednost broja da bi izdvojene cifre bile
     pozitivni brojevi. Na primer, 123%10 je 3, a -123%10 je -3. */
  x = abs(x);

  /* Slucaj kada je uneti broj 0 se posebno obradjuje. */
  if(x == 0)
  {
    printf("0\n");
    return 0;
  }

  /* U petlji se obradjuje cifra po cifra broja, dok god ima
     neobradjenih cifara u broju. */
  while (x != 0) {
    /* Ispisuje se poslednja cifra broja x. */
    printf("%d ", x % 10);

    /* Uklanja se poslednja cifra broja x. */
    x /= 10;
  }
  printf("\n");

  return 0;
}
```

Rešenje 2.3.11

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, suma, pom_n;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Pravi se kopija originalnog broja, da bi originalna vrednost
     n ostala nepromenjena. */
  pom_n = n;

  /*Inicijalizacija sume cifara. */
  suma = 0;

  /* Racuna se suma cifara. */
  while (pom_n != 0) {
    /* Na sumu se dodaje poslednja cifra broja. */
    suma += pom_n % 10;
    /* Sa broja se skida poslednja cifra. */
    pom_n /= 10;
  }

  /* Ispis rezultata. */
  if (n % suma == 0)
    printf("Broj %d je deljiv sa %d.\n", n, suma);
  else
    printf("Broj %d nije deljiv sa %d.\n", n, suma);

  return 0;
}
```

Rešenje 2.3.12

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

```
3
int main()
5 {
  /* Deklaracija potrebnih promenljivih. */
7   int t, x, i;
  int ukupan_prihod, ukupan_rashod, ukupan_rashod_abs;
9
  /* Ucitava se vrednost broja t. */
11  printf("Unesite broj t:");
  scanf("%d", &t);
13
  /* Vrsi se provera ispravnosti ulaza. */
15  if (t < 0) {
    printf("Greska: neispravan unos.\n");
17    return -1;
  }
19  else if( t == 0) {
    printf("Nema evidentiranih transakcija.");
21    return 0;
  }
23
  /* Inicijalizacija suma. */
25  ukupan_prihod = 0;
  ukupan_rashod = 0;
27
  /* Ucitavanje transakcija i izracunavanje suma. */
29  printf("Unesite transakcije: ");
  i = 0;
31  while (i < t) {
    /* Ucitava se jedna transakcija. */
33    scanf("%d", &x);

35    /* Dodaje se na odgovarajucu sumu. */
    if (x < 0)
37      ukupan_rashod += x;
    else
39      ukupan_prihod += x;

41    /* Uvecava se brojac. */
    i++;
43  }

45  /* Ispis rezultata. */
  printf("Prihod: %d\n", ukupan_prihod);
47  printf("Rashod: %d\n", ukupan_rashod);

49  ukupan_rashod_abs = abs(ukupan_rashod);
  if(ukupan_prihod >= ukupan_rashod_abs)
51    printf("Zarada: %d\n", ukupan_prihod - ukupan_rashod_abs);
  else
53    printf("Gubitak: %d\n", ukupan_rashod_abs - ukupan_prihod);
```

```
55    return 0;
}
```

Rešenje 2.3.13

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, x, i;
  int zbir = 0;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Ucitava se n brojeva i izracunava se trazeni zbir. */
  printf("Unesite n brojeva: ");
  i = 0;
  while (i < n) {
    /* Ucitava se jedan broj. */
    scanf("%d", &x);

    /* Ako je ucitani broj negativan i neparan,
       dodaje se na zbir. */
    if (x < 0 && x % 2 != 0)
      zbir = zbir + x;

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

  /* Ispis rezultata. */
  printf("Zbir neparnih i negativnih: %d\n", zbir);

  return 0;
}
```

Rešenje 2.3.14

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

```c
int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, broj;
  int suma = 0;
  int i;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Ucitava se n brojeva i izracunava se trazena suma. */
  printf("Unesite brojeve: ");
  for (i = 0; i < n; i++) {
    scanf("%d", &broj);

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

  /* Ispis rezultata. */
  printf("Suma je %d.\n", suma);

  return 0;
}
```

Rešenje 2.3.15

```c
#include <stdio.h>
int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int x, proizvod;

  /* Indikator koji oznacava da li je korisnik uneo bar jedan
     broj. */
  int unet_bar_jedan = 0;

  /* Indikator koji oznacava da li je korisnik uneo bar jedan
     pozitivan broj. */
  int unet_pozitivan = 0;

  /* Inicijalizacija proizvoda. */
  proizvod = 1;

```

```
  printf("Unesite brojeve:");

  /* Petlja ciji je uslov uvek ispunjen. */
  while (1) {

    /* Ucitava se jedan broj. */
    scanf("%d", &x);

    /* Ako je uneta nula, petlja se prekida naredbom break. */
    if (x == 0)
      break;

    /* Ako petlja nije prekinuta, znaci da je unet bar jedan broj.
       Iz tog razloga se vrednost indikatora za unete brojeve
       postavlja na 1. */
    unet_bar_jedan = 1;

    /* Proverava se da li je broj x pozitivan. */
    if(x > 0){
      /* Ako jeste, znaci da je unet bar jedan pozitivan broj i iz
         tog razloga se vrednost odgovarajuceg indikatora postavlja
         na 1. */
      unet_pozitivan = 1;

      /* Azurira se vrednost proizvoda pozitivnih brojeva. */
      proizvod = proizvod * x;
    }
  }

  /* Ispis rezultata. */
  if (unet_bar_jedan == 0)
    printf("Nije unet nijedan broj.\n");
  else if (unet_pozitivan == 0)
    printf("Medju unetim brojevima nema pozitivnih.\n");
  else
    printf("Proizvod pozitivnih brojeva je %d.\n", proizvod);

  return 0;
}
```

Rešenje 2.3.16

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

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n, cifra, n_original;
  int pronadjena_petica = 0;

```

```
  /* Ucitava se vrednost broja n. */
  printf("Unesite broj: ");
  scanf("%d", &n);

  /* Pamti se originalna vrednost unetog broja. */
  n_original = n;

  /* Uzima se apsolutna vrednost unetog broja. */
  if (n < 0)
    n = abs(n);

  /* Petlja se izvrsava dok god ima cifara u broju. */
  while (n > 0) {
    /* Izdvaja se poslednja cifra broja. */
    cifra = n % 10;

    /* Proverava se da li je ona jednaka broju 5 */
    if (cifra == 5) {
      /* Ako jeste, vrednost odgovarajuceg indikatora se postavlja
         na 1 i petlja se prekida. */
      pronadjena_petica = 1;
      break;
    }

    /* Ako petlja nije prekinuta, iz broja se uklanja poslednja
       cifra i postupak se ponavlja dok god ima neobradjenih
       cifara. */
    n = n / 10;
  }

  /* Ispis rezultata.
     Napomena: Koristi se unapred zapamcena promenljiva n_original
     jer je promenljiva n izmenjena u petlji. */
  if (pronadjena_petica == 0)
    printf("Broj %d sadrzi cifru 5.\n", n_original);
  else
    printf("Broj %d ne sadrzi cifru 5.\n", n_original);

  return 0;
}
```

Rešenje 2.3.17

```
#include <stdio.h>

int main()
{
  /* Deklaracije i inicijalizacije potrebnih promenljivih. */
  int x;
  int broj_brojeva = 0;
  int suma = 0;
```

```
9
   /* Brojevi se ucitavaju u petlji sve do unosa broja 0. */
11 printf("Unesite brojeve: ");
   while (1) {
13   scanf("%d", &x);

15   if (x == 0)
       break;
17
     /* Procitani broj se dodaje na sumu. */
19   suma += x;

21   /* Uvecava se broj ucitanih brojeva. */
     broj_brojeva++;
23 }

25 /* Ispis rezultata.
      Napomena: primetiti da su i suma i broj_brojeva celi brojevi
27    i da je neophodno bar jednu od te dve vrednosti pretvoriti
      u realan broj kako deljenje ne bi bilo celobrojno. */
29 if (broj_brojeva == 0)
     printf("Nisu uneti brojevi.\n");
31 else
     printf("Aritmeticka sredina: %.4f\n",
33          (double) suma / broj_brojeva);

35 return 0;
}
```

Rešenje 2.3.18

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracije potrebnih promenljivih. */
     float cena, suma = 0;
7    int broj_artikla = 0;

9    /* Cene se ucitavaju sve do unosa broja 0. */
     printf("Unesite cene: ");
11   while (1) {
       scanf("%f", &cena);
13
       if (cena == 0)
15       break;

17     /* Vrsi se provera ispravnosti ulaza. */
       if (cena < 0) {
19       printf("Greska: neispravan unos cene.\n");
         return -1;
```

```
21      }
23      /* Suma se uvecava za vrednost unete cene. */
        suma += cena;
25
        /* Broj unetih artikala se uvecava za 1. */
27      broj_artikla++;
    }
29
    /* Ispis rezultata. */
31  if (broj_artikla == 0)
      printf("Nisu unete cene.\n");
33  else
      printf("Prosecna cena: %.4f\n", suma / broj_artikla);
35
    return 0;
37  }
```

Rešenje 2.3.19

```
1   #include <stdio.h>

3   int main()
    {
5     /* Deklaracija potrebnih promenljivih. */
      int n, i, broj_promena = 0;
7     double prethodni, trenutni;

9     /* Ucitava se vrednost broja n. */
      printf("Unesite broj n ");
11    scanf("%d", &n);

13    /* Vrsi se provera ispravnosti ulaza. */
      if (n <= 0) {
15      printf("Greska: neispravan unos.\n");
        return -1;
17    }

19    printf("Unesite brojeve: ");
      /* Provera promene znaka se vrsi za svaka dva susedna uneta
21       broja. Prvi broj se ucitava pre petlje i smesta se u
         promenljivu prethodni. Zatim se u petlji ucitava drugi i
23       njihov znak se poredi. Postupak se ponavlja za sve parove,
         tako sto se uvek na kraju petlje poslednji ucitani broj
25       postavi da bude prethodni za sledecu iteraciju. */
      scanf("%lf", &prethodni);
27
      /* Kako je vec jedan broj unet, brojac se postavlja na 1, a ne
29       na 0. */
      for (i = 1; i < n; i++) {
31
```

```
    /* Ucitava se broj. */
    scanf("%lf", &trenutni);

    /* Proverava se da li je doslo do promene znaka izmedju
       prethodnog i trenutnog. Oni su razlicitog znaka ako vazi:
       1. da im je proizvod negativan ILI
       2. da im je proizvod nula, a jedan od njih je negativan. */
    if (prethodni * trenutni < 0)
      broj_promena++;
    else if (prethodni * trenutni == 0 &&
      (prethodni < 0 || trenutni < 0))
      broj_promena++;

    /* Trenutni broj postaje prethodni za sledecu iteraciju. */
    prethodni = trenutni;
  }

  /* Ispis rezultata. */
  printf("Broj promena je %d.\n", broj_promena);

  return 0;
}
```

Rešenje 2.3.20

```
#include <stdio.h>

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n, i;
  float cena, min_cena;

  /* Ucitava se broj artikala. */
  printf("Unesite broj artikala:");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  printf("Unesite cene artikala:");

  /* Minimalna cena se inicijalizuje na cenu prvog artikla. Zbog
     toga se cena prvog artikla ucitava pre petlje. */
  scanf("%f", &cena);
  if (cena <= 0) {
    printf("Greska: neispravan unos cene.\n");
    return -1;
```

```
  }
  min_cena = cena;

  /* Ucitava se i preostalih n-1 cena i racuna se najmanja. */
  for(i=1; i<n; i++){
    scanf("%f", &cena);

    if (cena <= 0) {
      printf("Greska: neispravan unos cene.\n");
      return -1;
    }

    if (cena < min_cena)
      min_cena = cena;
    i++;
  }

  /* Ispis rezultata. */
  printf("Najmanja cena: %f\n", min_cena);

  return 0;
}
```

Rešenje 2.3.21

```
#include <stdio.h>

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  float cena, m;
  unsigned int n, i, broj_artikala = 0;

  /* Ucitava se vrednost broja m. */
  printf("Nikolin budzet: ");
  scanf("%f", &m);

  /* Ucitava se broj artikala. */
  printf("Unesite broj artikala: ");
  scanf("%u", &n);

  /* Unose se cene artikala i racuna se rezultat. */
  printf("Unesite cene artikala: ");

  for(i=0; i<n; i++){
    /* Ucitava se cena artikla. */
    scanf("%f", &cena);

    /* Provera se da li Nikola moze da kupi trenutni artikal. */
    if (cena <= m)
      broj_artikala++;
```

```
27    }
29    /* Ispis rezultata. */
      printf("Ukupno artikala: %d\n", broj_artikala);
31
      return 0;
33  }
```

Rešenje 2.3.22

Rešenje (a)

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    int main()
5   {
      /* Deklaracije potrebnih promenljivih. */
7     int n, i, x, rezultat;
      int x_desetica, najveca_desetica;
9
      /* Ucitava se vrednost broja n. */
11    printf("Unesite broj n: ");
      scanf("%d", &n);
13
      /* Vrsi se provera ispravnosti ulaza. */
15    if (n < 0) {
        printf("Greska: neispravan unos.\n");
17      return -1;
      }
19
      /* Ako nema unetih brojeva, ispisuje se odgovarajuca poruka. */
21    if (n == 0) {
        printf("Nisu uneti brojevi.\n");
23      return 0;
      }
25
      printf("Unesite brojeve: ");
27
      /* Prvi broj se ucitava pre petlje, zbog ispravne
29       inicijalizacije. */
      scanf("%d", &x);
31    /* Promenljiva najveca_desetica se postavlja na cifru desetica
         ucitanog broja. Napomena: pri racunanju se uzima apsolutna
33       vrednost broja jer je npr. (-123/10)= -12 i -12 % 10 = -2,
         a cifra desetica treba da bude 2. */
35    najveca_desetica = (abs(x) / 10) % 10;
      /* Kako je na kraju potrebno ispisati broj cija je cifra desetica
37       najveca, trenutna vrednost rezultata se postavlja na vrednost
         ucitanog broja. */
```

```
39    rezultat = x;

41    /* Ucitava se i preostalih n-1 brojeva i ako se naidje na broj
         cija je cifra desetica veca od trenutno najvece, azuriraju
43       se vrednosti najvece desetice i rezultata. */
      for (i = 1; i < n; i++) {
45      scanf("%d", &x);

47      x_desetica = (abs(x) / 10) % 10;

49      if (x_desetica > najveca_desetica) {
          najveca_desetica = x_desetica;
51        rezultat = x;
        }
53    }

55    /*II nacin: Inicijalizacija najvece desetice na neku vrednost
       koja je sigurno manja od svih vrednosti koje cifra desetica
57     moze da uzme (dakle, bilo sta sto je manje od 0 jer cifra
       desetica moze imati vrednosti izmedju 0 i 9).
59     Zatim se u petlji izracunava rezultat, analogno prvom nacinu.

61    najveca_desetica = -1;
      for(i=0; i<n; i++)
63    {
        scanf("%d", &x);
65
        x_desetica = (abs(x) / 10) % 10;
67
        if (x_desetica > najveca_desetica) {
69        najveca_desetica = x_desetica;
          rezultat = x;
71      }
      }
73    */

75    /* Ispis rezultata. */
      printf("Broj sa najvecom cifrom desetica: %d\n", rezultat);
77
      return 0;
79  }
```

Rešenje (b)

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    int main()
5   {
      /* Deklaracije potrebnih promenljivih. */
7     int n, i;
```

```
  int x, x_kopija, broj_cifara;
  int najveci_broj_cifara, rezultat;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n < 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Ako nema unetih brojeva, ispisuje se odgovarajuca poruka. */
  if (n == 0) {
    printf("Nisu uneti brojevi.\n");
    return 0;
  }

  /* Maksimalan broj cifara se postavlja na 0 jer svaki broj ima
     vise od 0 cifara. */
  najveci_broj_cifara = 0;

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

    /* Racuna se broj cifara unetog broja x. */
    x_kopija = abs(x);
    broj_cifara = 0;
    do {
      broj_cifara++;
      x_kopija = x_kopija / 10;
    } while (x_kopija != 0);

    /* Ako je broj cifara unetog broja veci od najveceg broja
       cifara, azuriraju se vrednosti najveceg broja cifara i
       tekuceg rezultata. */
    if (broj_cifara > najveci_broj_cifara) {
      najveci_broj_cifara = broj_cifara;
      rezultat = x;
    }
  }

  /* Ispis rezultata. */
  printf("Najvise cifara ima broj %d.\n", rezultat);

  return 0;
}
```

Rešenje (c)

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

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n, i;
  int x, x_kopija, vodeca_cifra;
  int najveca_vodeca_cifra, rezultat;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n < 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Ako nema unetih brojeva, ispisuje se odgovarajuca poruka. */
  if (n == 0) {
    printf("Nisu uneti brojevi.\n");
    return 0;
  }

  /* Inicijalizacija najvece vodece cifre na -1. */
  najveca_vodeca_cifra = -1;

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

    /* Racuna se vodeca cifra ucitanog broja x. */
    x_kopija = abs(x);
    while (x_kopija > 10) {
      x_kopija = x_kopija / 10;
    }
    vodeca_cifra = x_kopija;

    /* Ako je izdvojena cifra veca od najvece vodece cifre,
       azuriraju se vrednosti najvece vodece cifre i rezultata. */
    if (vodeca_cifra > najveca_vodeca_cifra) {
      najveca_vodeca_cifra = vodeca_cifra;
      rezultat = x;
    }
  }

  /* Ispis rezultata. */
  printf("%d\n", rezultat);
```

```
52    return 0;
}
```

Rešenje 2.3.23

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int x;
  int najmanja, najveca;

  printf("Unesite brojeve: ");
  /* Prvi broj se ucitava izvan petlje zbog inicijalizacije
     najvece i najmanje vrednosti nadmorske visine.
     Napomena: Ovde bi inicijalizacija najveca=-1 bila pogresna
     jer moze da se desi da su svi uneti brojevi negativni i manji
     od -1 i onda bi najveca i nakon izvrsavanja tela petlje ostala
     -1. */
  scanf("%d", &x);
  najveca = x;
  najmanja = x;

  /* Ako nema unetih nadmorskih visina, ispisuje se odgovarajuca
     poruka. */
  if(x == 0)
  {
    printf("Nisu unete nadmorske visine.");
    return 0;
  }

  /* Za svaki ucitani broj se proverava da li je manji od najmanje
     ili veci od najvecee i vrsi se azuriranje odgovarajucih
     vrednosti. Petlja se prekida kada se unese broj 0.*/
  while (1) {
    scanf("%d", &x);

    if(x == 0)
        break;

    if (x > najveca)
      najveca = x;

    if (x < najmanja)
      najmanja = x;
  }

  /* Ispis rezultata. */
  printf("Razlika: %d\n", najveca - najmanja);
```

```
47    return 0;
}
```

Rešenje 2.3.24

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, d, i;
  int x, y;
  int broj_parova = 0;

  /* Ucitavaju se vrednosti n i d. */
  printf("Unesite brojeve n i d: ");
  scanf("%d %d", &n, &d);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 1 || d < 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  printf("Unesite n brojeva: ");

  /* Prvi broj se ucitava pre petlje. */
  scanf("%d", &x);

  for (i = 1; i < n; i++) {
    scanf("%d", &y);

    /* Provera se 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;
  }

  /* Ispis rezultata. */
  printf("Broj parova: %d\n", broj_parova);

  return 0;
}
```

Rešenje 2.3.25

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, cifra;
  unsigned int rezultat;
  int pozicija;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Inicijalizacija pozicije i rezultata.
     Pozicija oznacava tezinu trenutne cifre i moze imati vrednosti
     1, 10, 100, 1000, ... */
  pozicija = 1;
  rezultat = 0;

  /* U petlji se izdvaja cifra po cifra, dok god ima neobradjenih
     cifara. */
  while (n > 0) {
    /* Izdvaja se poslednja cifra iz zapisa i ako je njena vrednost
       paran broj, uvecava se za 1. */
    cifra = n % 10;
    if (cifra % 2 == 0)
      cifra++;

    /* Novi broj se formira tako sto se izdvojena cifra pomnozi
       odgovarajucom tezinom (stepenom) pozicije i doda na tekuci
       rezultat. */
    rezultat += cifra * pozicija;

    /* Uklanja se poslednja cifra broja. */
    n /= 10;

    /* Pozicija se mnozi sa 10. */
    pozicija *= 10;
  }

  /* Ispisuje se izracunatu vrednost. */
  printf("Rezultat: %d\n", rezultat);

  return 0;
}
```

Rešenje 2.3.26

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

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int x, pozicija, rezultat, cifra;
  int znak = 1;

  /* Ucitava se vrednost polaznog broja. */
  printf("Unesite broj: ");
  scanf("%d", &x);

  /* Ako je broj negativan, uzima se njegova apsolutna vrednost
     i azurira se vrednost znaka broja. */
  if (x < 0) {
    x = abs(x);
    znak = -1;
  }

  /* Pozicija oznacava tezinu trenutne cifre rezultata i moze imati
     vrednosti 1, 10, 100, ... */
  pozicija = 1;
  rezultat = 0;

  /* Ideja: u rezultatu se zadrzavaju cifre jedinice, stotine,..
     Na primer, x=12345
     Pre petlje: pozicija = 1, rezultat = 0
     1. iteracija:
     cifra = 5, rezultat = 0+5*1=5, x = 123, pozicija = 10
     2. iteracija:
     cifra = 3, rezultat = 5+3*10 = 35, x = 1, pozicija = 100
     3. iteracija:
     cifra = 1, rezultat = 35+1*100, x = 0, pozicija = 1000
     Petlja se zavrsava jer x ima vrednost 0. */
  while (x > 0) {
    /* Izdvajanje poslednje cifre. */
    cifra = x % 10;

    /* Rezultat se uvecava za vrednost cifre pomnozene sa vrednoscu
       tezine njene pozicije u broju. */
    rezultat += cifra * pozicija;

    /* Iz polaznog broja se uklanjaju poslednje dve cifre jer u
       rezultatu treba da ostane svaka druga cifra polaznog
       broja.*/
    x /= 100;

    /* Pozicija se mnozi sa 10, kako bi imala ispravnu vrednost u
       sledecoj iteraciji. */
```

```
    pozicija *= 10;
  }

  /* Ispis rezultata */
  printf("Rezultat: %d\n", znak * rezultat);

  return 0;
}
```

Rešenje 2.3.27

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n;
  int c1, c2, c3;
  int pozicija, rezultat;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Ako broj nema bar tri cifre, rezultat ima vrednost unetog
     broja. */
  if(n <= 99)
  {
    printf("Rezultat: %d\n", n);
    return 0;
  }

  /* Izdvajaju se poslednje tri cifre polaznog broja. */
  c1 = n%10;
  c2 = (n/10)%10;
  c3 = (n/100)%10;

  /* Poslednja cifra se uvek nalazi u rezultatu jer ona nema
    oba suseda. Zato rezultat inicijalizujemo na poslednju cifru,
    a poziciju na 10. */
  rezultat = c1;
  pozicija = 10;

  /* Petlja se izvrsava dok god broj ima bar tri cifre. */
  while(n>99)
```

```
41    {
        /* Proverava se da li c2 treba da se nadje u rezultatu. Ako
43         treba, rezultat se uvecava za vrednost cifre pomnozenu sa
           vrednoscu tezine njene pozicije u rezultatu i tezina
45         pozicije se mnozi sa 10. */
      if(c2 != c1 + c3)
47    {
        rezultat += c2*pozicija;
49      pozicija *= 10;
      }
51
      /* Vrsi se pomeranje na sledece tri cifre polaznog broja.
53       Iz polaznog broja brisemo poslednju cifru. Prva i druga
         cifra su vec izracunate, samo se vrsi njihovo premestanje
55       iz c2 i c3 u c1 i c2. Cifra c3 se racuna. */
      n = n/10;
57    c1 = c2;
      c2 = c3;
59    c3 = (n/100)%10;
    }
61
    /* Po zavrsetku petlje, broj n je dvocifren i njegova cifra
63     desetica odgovara vodecoj cifri polaznog broja. Vodeca cifra
       polaznog broja uvek treba da se nadje u rezultatu jer nema
65     oba suseda i iz tog razloga je dodajemo na tekuci rezultat. */
    rezultat += (n/10)*pozicija;
67
    /* Ispis rezultata. */
69  printf("%d\n", rezultat);

71  return 0;
}
```

Rešenje 2.3.28

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     int x, x_kopija, x_obrnuto;
7
     /* Ucitava se vrednost pocetnog broja. */
9    printf("Unesite broj: ");
     scanf("%d", &x);
11
     /* Uzima se apsolutna vrednost unetog broja.  */
13   if (x < 0)
       x = -x;
15
     /* Racuna se broj koji se dobije kada se broju x obrnu cifre.
```

```
17      Na primer, od 12345 treba da se dobije 54321.
        Broj se obrce tako sto se u svakoj iteraciji njegova vrednost
19      pomnozi sa 10 i doda mu se sledeca cifra polaznog broja.
        Za x_kopija=12345, x_obrnuto = 0
21      1. iteracija: x_obrnuto = 0*10 + 5 = 5, x_kopija = 1234
        2. iteracija: x_obrnuto = 5*10 + 4 = 54, x_kopija = 123,
23      3. iteracija: x_obrnuto = 54*10 + 3 = 543, itd.*/
    x_kopija = x;
25  x_obrnuto = 0;
    while (x_kopija != 0) {
27    x_obrnuto = x_obrnuto * 10 + x_kopija % 10;
      x_kopija /= 10;
29  }

31  /* Broj je palindrom ako je jednak broju koji se dobije
       obrtanjem njegovih cifara.
33     Npr. x = 12321, x_obrnuto je takodje 12321.*/
    if (x == x_obrnuto)
35    printf("Broj je palindrom.\n");
    else
37    printf("Broj nije palindrom.\n");

39  return 0;
}
```

Rešenje 2.3.29

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracija potrebnih promenljivih. */
     int n, i;
7    int fib1 = 0, fib2 = 1, fib3;

9    /* Ucitava se vrednost broja n. */
     printf("Unesite broj n: ");
11   scanf("%d", &n);

13   /* Vrsi se provera ispravnosti ulaza. */
     if (n < 0) {
15     printf("Greska: neispravan unos.\n");
       return -1;
17   }

19   /* Ako je n=0, F[0] = 0, slicno ako je n=1 F[1] = 1. */
     if(n < 2){
21     printf("F[%d] = %d\n",n, n);
       return 0;
23   }
```

```
25    fib3 = fib1 + fib2;
      for(i=2; i<n; i++) {
27      /* Vrsi se pomeranje na sledecu trojku. */
        fib1 = fib2;
29      fib2 = fib3;
        fib3 = fib1 + fib2;
31    }

33    /* Ispis rezultata. */
      printf("F[%d] = %d\n", n, fib3);
35
      return 0;
37  }
```

Rešenje 2.3.30

```
    #include<stdio.h>
2
    int main()
4   {
      /* Deklaracija potrebnih promenljivih. */
6     int an;

8     /* Ucitava se vrednost prvog clana. */
      printf("Unesite prvi clan:");
10    scanf("%d", &an);

12    /* Vrsi se provera ispravnosti ulaza. */
      if (an <= 0) {
14      printf("Greska: neispravan unos.\n");
        return -1;
16    }

18    /* Dok se ne dodje do clana koji je 1, stampa se vrednost
         trenutnog clana i vrsi se izracunavanje narednog, po
20       zadatoj formuli. */
      while (an != 1) {
22      printf("%d ", an);

24      if (an % 2 != 0)
          an = (3 * an + 1) / 2;
26      else
          an = an / 2;
28    }

30    /* Na kraju se stampa i jedinica. */
      printf("1\n");
32
      return 0;
34  }
```

Rešenje 2.3.31

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

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int format, i;
  double sirina, duzina, nova_duzina;

  /* Ucitava se format papira. */
  printf("Unesite format papira: ");
  scanf("%d", &format);

  /* Vrsi se provera ispravnosti ulaza. */
  if (format < 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* duzina/sirina = 1/sqrt(2)
     duzina*sirina = 1000mm x 1000mm
     =>
     duzina = sirina/sqrt(2)
     duzina*sirina = 1000mm x 1000mm
     =>
     sirina*sirina/sqrt(2) = 1000*1000
     sirina*sirina = sqrt(2) * 1000 * 1000
     sirina = sqrt(sqrt(2) * 1000 * 1000)
     =>
     duzina = sirina/sqrt(2) */
  sirina = sqrt(1000 * 1000 * sqrt(2));
  duzina = sirina / sqrt(2);

  /* U petlji se racunaju duzina i sirina za uneti format. */
  for (i = 1; i <= format; i++) {
    nova_duzina = sirina / 2;
    sirina = duzina;
    duzina = nova_duzina;
  }

  /* Ispis rezultata. Napomena: duzina i sirina celi brojevi. */
  printf("%d %d\n", (int) duzina, (int) sirina);

  return 0;
}
```

Rešenje 2.3.32

```c
#include <stdio.h>

int main()
{
  char c;

  /* I nacin ucitavanja:
     U samom uslovu petlje se vrsi ucitavanje jednog karaktera,
     njegovo smestanje u promenljivu c i provera da li je ucitani
     karakter tacka. Zagrade oko (c=getchar()) su obavezne jer
     relacioni operator != ima veci prioritet od dodele i kada ne
     bi postojale zagrade, redosled operacija bi bio:
     (c = (getchar() != '.')), sto znaci da bi se u c smestio
     rezultat poredjenja, odnosno 0 ili 1. */
  while ((c = getchar()) != '.') {
    /* Proveravaju se uslovi i vrsi se ispis odgovarajuceg
       karaktera.*/
    if (c >= 'A' && c <= 'Z')
      putchar(c + 'a' - 'A');
    else if (c >= 'a' && c <= 'z')
      putchar(c - 'a' + 'A');
    else
      putchar(c);
  }

  /*II nacin:
  while(1) {
    c = getchar();
    if(c == '.')
      break;

    if (c >= 'A' && c <= 'Z')
      putchar(c + 'a' - 'A');
    else if (c >= 'a' && c <= 'z')
      putchar(c - 'a' + 'A');
    else
      putchar(c);
  } */

  return 0;
}
```

Rešenje 2.3.33

```c
#include <stdio.h>

int main()
{
  /* Deklaracije i inicijalizacije. */
  char c;
  int broj_velikih = 0, broj_malih = 0;
```

```
  int broj_cifara = 0, suma_cifara = 0, broj_belina = 0;

  /* Petlja se zavrsava kada korisnik zada konstantu oznaku za kraj
     ulaza (konstanta EOF cija je vrednost -1). Ova konstanta se
     zadaje kombinacijom tastera CTRL+D. */
  while ((c = getchar()) != EOF) {
    if (c >= 'A' && c <= 'Z')
      broj_velikih++;
    else if (c >= 'a' && c <= 'z')
      broj_malih++;
    else if (c >= '0' && c <= '9') {
      broj_cifara++;
      suma_cifara = suma_cifara + c - '0';
    }
    else if (c == '\t' || c == '\n' || c == ' ')
      broj_belina++;
  }

  /* Ispis rezultata. */
  printf("velika: %d, mala: %d\n", broj_velikih, broj_malih);
  printf("cifre: %d, beline: %d\n", broj_cifara, broj_belina);
  printf("suma cifara: %d\n", suma_cifara);

  return 0;
}
```

Rešenje 2.3.34

```
#include <stdio.h>

int main()
{
  /* Deklaracija i inicijalizacija potrebnih promenljivih. */
  int n, i;
  int broj_a = 0, broj_e = 0, broj_i = 0, broj_o = 0, broj_u = 0;
  char c;

  /* Ucitava se broj karaktera. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n < 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Kako je korisnik nakon unosa broja n uneo oznaku za novi red,
     potrebno je preskociti taj novi red jer bi u suprotnom on bio
     ucitan kao prvi od n karaktera (oznaka za novi red je
     regularan karakter kao sto je to 'a' ili ' ').*/
```

```
  getchar();

  /* Ucitavaju se karakteri i broje se samoglasnici. */
  for (i = 0; i < n; i++) {
    scanf("%c", &c);

    switch (c) {
    case 'a':
    case 'A':
      broj_a++;
      break;
    case 'e':
    case 'E':
      broj_e++;
      break;
    case 'i':
    case 'I':
      broj_i++;
      break;
    case 'o':
    case 'O':
      broj_o++;
      break;
    case 'u':
    case 'U':
      broj_u++;
      break;
    }
  }

  /* Ispis rezultata. */
  printf("Samoglasnik a: %d\n", broj_a);
  printf("Samoglasnik e: %d\n", broj_e);
  printf("Samoglasnik i: %d\n", broj_i);
  printf("Samoglasnik o: %d\n", broj_o);
  printf("Samoglasnik u: %d\n", broj_u);

  return 0;
}
```

Rešenje 2.3.35

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

int main()
{
  /* Deklaracija i inicijalizacija potrebnih promenljivih. */
  int n, i;
  int broj_Z = 0, broj_i = 0, broj_m = 0, broj_a = 0;
  char novi_red, c;
```

```
  /* Ucitava se broj karaktera. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Ucitavaju se karakteri. */
  for (i = 1; i <= n; i++) {
    printf("Unestite %d. karakter: ", i);

    /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
       pa tek posle procitane beline se cita uneti karakter. */
    scanf("%c%c", &novi_red, &c);

    /* Obradjuje se ucitani karakter. */
    switch (c) {
    case 'Z':
      broj_Z++;
      break;
    case 'i':
      broj_i++;
      break;
    case 'm':
      broj_m++;
      break;
    case 'a':
      broj_a++;
      break;
    }
  }

  /* Ako su svi brojaci razliciti od nule, rec "Zima" se moze
     napisati pomocu unetih karaktera. */
  if (broj_Z && broj_i && broj_m && broj_a)
    printf("Moze se napisati rec Zima.\n");
  else
    printf("Ne moze se napisati rec Zima.\n");

  return 0;
}
```

Rešenje 2.3.36

```
#include <stdio.h>

int main()
```

```
{
  /* Deklaracije potrebnih promenljivih. */
  int n, i;
  int suma_kubova;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n:");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Racuna se suma kubova svih brojeva iz intervala [1,n]. */
  suma_kubova = 0;
  for (i = 1; i <= n; i++)
    suma_kubova += i * i * i;

  /* Ispis rezultata. */
  printf("Suma kubova: %d\n", suma_kubova);

  return 0;
}
```

Rešenje 2.3.37 Rešenje je analogno rešenju zadatka 2.3.36.

Rešenje 2.3.38

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, i;
  float x, suma, x_i;

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

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Vrednost sume se inicijalizuje na nulu, a vrednost x^i
     na x. */
```

```
21    suma = 0;
      x_i = x;
23
      /* Promenljiva x^i ima vrednosti [x, x^2, ..., x^n].
25       Vrednost sume se u svakoj iteraciji uvecava za i*x^i. */
      for (i = 1; i <= n; i++) {
27      suma += i * x_i;
        x_i *= x;
29    }

31    /* Ispis rezultata. */
      printf("S=%f\n", suma);
33
      return 0;
35  }
```

Rešenje 2.3.39

```
1   #include <stdio.h>

3   int main()
    {
5     /* Deklaracija potrebnih promenljivih. */
      int n, i;
7     float x, suma, x_i;

9     /* Ucitavaju se vrednosti x i n. */
      printf("Unesite redom brojeve x i n: ");
11    scanf("%f %d", &x, &n);

13    /* Vrsi se provera ispravnosti ulaza. */
      if (n <= 0) {
15      printf("Greska: neispravan unos.\n");
        return -1;
17    }

19    /* Racuna se trazena suma. */
      suma = 1;
21    x_i = x;
      for (i = 1; i <= n; i++) {
23      suma += 1 / x_i;
        x_i *= x;
25    }

27    /* Ispis rezultata. */
      printf("S=%f\n", suma);
29
      return 0;
31  }
```

Rešenje 2.3.40

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

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int i;
  float suma, clan;
  float x, eps;

  /* Ucitavaju se vrednosti x i eps. */
  printf("Unesite x: ");
  scanf("%f", &x);

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

  /* Inicijalizacija sume, prvog clana i brojaca. */
  suma = 0;
  clan = 1;
  i = 1;

  /* U svakoj iteraciji na sumu se dodaje prethodno izracunati
     clan sume i zatim se racuna sledeci clan. Petlja se prekida
     kada vrednost sledeceg clana postane manja ili jednaka eps. */
  while (clan > eps) {
    suma += clan;
    clan = clan * x / i;
    i++;
  }

  /* Ispis rezultata. */
  printf("S=%f\n", suma);

  return 0;
}
```

Rešenje 2.3.41

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

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int i;
  float suma;
  float x, eps, clan;
```

```
10
   /* Ucitavaju se vrednosti x i eps. */
12 printf("Unesite x: ");
   scanf("%f", &x);
14
   printf("Unesite tacnost eps: ");
16 scanf("%f", &eps);

18 /* Inicijalizacije. */
   suma = 0;
20 clan = 1;
   i = 1;
22
   /* Kako clanovi sume mogu biti negativni, potrebno je posmatrati
24    apsolutnu vrednost clana. */
   while (fabs(clan) > eps) {
26   suma += clan;

28   /* U svakoj iteraciji se racuna novi clan i mnozi se sa -1.
        Na taj nacin se postize da je vrednost clana naizmenicno
30      pozitivna i negativna. */
     clan = clan * x / i;
32   clan *= -1;

34   i++;
   }
36
   /* Ispis rezultata. */
38 printf("S=%f\n", suma);

40 return 0;
}
```

Rešenje 2.3.42

```
1  #include <stdio.h>
   #include <math.h>
3
   int main()
5  {
     /* Deklaracije potrebnih promenljivih. */
7    int n, i;
     double x, x_i, proizvod;
9
     /* Ucitavaju se vrednosti x i n. */
11   printf("Unesite redom brojeve x i n: ");
     scanf("%lf %d", &x, &n);
13
     /* Vrsi se provera ispravnosti ulaza. */
15   if (n <= 0) {
       printf("Greska: neispravan unos.\n");
```

```
17      return -1;
    }
19
    /* Racuna se trazeni proizvod. */
21  x_i = 1;
    proizvod = 1;
23  for (i = 0; i < n; i++) {
      x_i *= x;
25    proizvod *= 1 + cos(x_i);
    }
27
    /* Ispis rezultata. */
29  printf("P = %lf\n", proizvod);

31  return 0;
}
```

Rešenje 2.3.43

```
1  #include <stdio.h>

3  int main()
   {
5    /* Deklaracije potrebnih promenljivih. */
     int n, i;
7    double razlomak;

9    /* Ucitava se vrednost broja n.  */
     printf("Unesite broj n: ");
11   scanf("%d", &n);

13   /* Vrsi se provera ispravnosti ulaza. */
     if (n <= 0) {
15     printf("Greska: neispravan unos.\n");
       return -1;
17   }

19   /* Razlomak se izracunava "od nazad", odnosno, krece se od
        najnizeg razlomka 1/n i od njega se nadalje formira sledeci,
21      "visi" razlomak itd. Zavrsava se kada se stigne do koraka 0 +
        1/R. */
23   razlomak = n;
     for (i = n - 1; i >= 0; i--)
25     razlomak = i + 1 / razlomak;

27   /* Ispis rezultata. */
     printf("R = %lf\n", razlomak);
29
     return 0;
31 }
```

Rešenje 2.3.44

```c
#include <stdio.h>

int main () {
  /* Deklaracije potrebnih promenljivih. */
  int i , n;
  float suma, x, clan;

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

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
      printf("Greska: neispravan unos.\n");
      return -1;
  }

  /* Inicijalizacije. */
  suma = 1;
  clan = 1;
  i = 2;

  /* Racuna se trazena suma. */
  while (i <= 2 * n) {
      /* Svaki clan suma se od prethodnog clana razlikuje za
         x^2/(i*(i-1)). */
      clan = clan * x * x / (i * (i - 1));
      clan *= -1;
      suma += clan;
      i += 2;
  }

  /* Ispis rezultata. */
  printf("S=%f\n", suma);

  return 0;
  }
```

Rešenje 2.3.45

```c
#include <stdio.h>

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n, i;
  double clan, proizvod = 1;
```

```
  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Racuna se trazeni proizvod. */
  clan = 1;
  for (i = 2; i <= n; i++) {
    clan = clan / i;
    proizvod *= 1 + clan;
  }

  /* Ispis rezultata. */
  printf("P = %lf\n", proizvod);

  return 0;
}
```

Rešenje 2.3.46

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, i;
  long int clan, suma = 0;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n < 5 || n % 2 == 0) {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Izracunava se trazena suma. */
  clan = -1 * 3;
  for (i = 5; i <= n; i += 2) {
    clan = -1 * clan * i;
    suma += clan;
  }

  /* Ispis rezultata. */
```

```
27    printf("S = %ld\n", suma);

29    return 0;
}
```

Rešenje 2.3.47

```
#include <stdio.h>
2
int main()
4 {
    /* Deklaracije potrebnih promenljivih. */
6   int n, i;
    double rezultat;
8   double x, a;

10  /* Ucitavaju se vrednosti ulaznih promenljivih. */
    printf("Unesite brojeve x i a: ");
12  scanf("%lf%lf", &x, &a);
    printf("Unesite broj n: ");
14  scanf("%d", &n);

16  /* Vrsi se provera ispravnosti ulaza. */
    if (n <= 0) {
18    printf("Greska: neispravan unos.\n");
      return -1;
20  }

22  /* Racuna se vrednost zadatog izraza. Krece se od
       rezultat = (x+a)^2 i ide se ka spolja.
24     Svaki put vrednost rezultata treba zameniti sa
       (rezultat + a)^2. */
26  rezultat = x;
    for (i = 0; i < n; i++)
28    rezultat = (rezultat + a) * (rezultat + a);

30  /* Ispis rezultata. */
    printf("Izraz = %lf\n", rezultat);
32
    return 0;
34 }
```

Rešenje 2.3.48

Rešenje (a)

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

```c
int main()
{
  /* Deklaracije potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Ispis tablice mnozenja dimenzije n*n. */
  for (i = 1; i <= n; i++) {
    for (j = 1; j <= n; j++){
      /* Vrednost svakog polja je proizvod vrste i kolone. */
      printf("%3d ", i * j);
    }
    /* Na kraju svake vrste se ispisuje novi red. */
    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```c
#include <stdio.h>

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  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);

    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;
}
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Ispis trazene tablice. */
  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()
{
  /* Deklaracije potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Ispis trazenog trougla. */
  for (i = 0; i < n; i++) {
    for (j = 0; j < n - i; j++)
      printf("(%d, %d)", i, j);

    printf("\n");
```

```
18    }

20    return 0;
}
```

Rešenje 2.3.49

Rešenje (a)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Kvadrat predstavlja tabelu sa n vrsta gde svaka vrsta sadrzi
     n polja, a svako polje je isto i predstavlja karakter *. */
  for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter *,
     a unutrasnjost kvadrata je karakter blanko. */
  for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++){
```

```
      /* Provera se da li je u pitanju ivica. */
      if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
        printf("*");
      else
        printf(" ");
    }
    printf("\n");
  }

  return 0;
}
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* 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 (i = 0; i < n; i++) {
    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 2.3.50

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
```

```
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Veliko slovo X se dobija tako sto se na dijagonalama kvadrata
   ispisuju karakteri *, a na ostalim mestima blanko. */
  for (i = 0; i < n; i++) {
    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 2.3.51

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Iscrtava se znak plus tako sto se na pozicijama koje
     odgovaraju sredisnjoj vrsti i sredisnjoj kolini ispisuje
     +, a na ostalim pozicijama se ispisuje blanko. */
  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 2.3.52

Rešenje (a)

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Iscrtava se trazeni trougao. */
  for (i = 0; i < n; i++) {
    for (j = 0; j < n - i; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Iscrtava se trazeni trougao. */
  for (i = 0; i < n; i++) {
    for (j = 0; j <= i; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (c)

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Iscrtava se trazeni trougao. */
  for (i = 0; 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 (d)

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Iscrtava se trazeni trougao. */
  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()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Iscrtava se gornji deo trazenog trougla. */
  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");
  }

  /* Iscrtava se donji deo trazenog trougla. */
  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()
{
  /* Deklaracije potrebnih promenljivih. */
  unsigned int n, i, j;
  char c, novi_red;
```

```
  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Ucitava se karakter koji ce se koristiti za iscrtavanje.
     Napomena: voditi racuna da treba preskociti novi red koji
     korisnik zadaje nakon unosa broja n. */
  printf("Unesite karakter c: ");
  scanf("%c%c", &novi_red, &c);

  /* Iscrtavanje trazenog trougla. Iscrtavaju se samo ivice
     trougla, ostalo se popunjava belinama. */
  for (i = 0; i < n; i++) {
    for (j = 0; j <= i; j++)
      if (i == n - 1 || j == 0 || j == i)
        printf("%c", c);
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.53

Rešenje (a)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, i, j;

  /* Ucitava se vrednost broja n. */
  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");
  }
```

```
23    return 0;
}
```

Rešenje (b)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  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 >= 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++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

```

```
  /* Slika se crta iz dva dela. */

  /* 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");
  }

  /* II deo: crtanje donjeg trougla. 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++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (d)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  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. */
    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");
  }

  return 0;
}
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  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. */
    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");
  }

  /* II deo: crtanje donjeg trougla. 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 2.3.54

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Strelica se moze posmatrati kao spojena dva pravougla trougla
     kojima se ispisuje hipotenuza i jedna kateta. */

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    for (j = 0; j <= i; j++)
      /* Proverava se da li se ispisuje karakter na hipotenuzi
         (j == i) ili da se ispisuje poslednji red (i == n-1). */
      if (j == i || i == n - 1)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  /* II deo: crtanje donjeg dela slike, odnosno donji trougao.
     Brojac i odredjuje koji red donjeg trougla se trentno iscrtava.
     Kako je prvi red donjeg trougla vec iscrtan (to je poslednji
     red gornjeg trougla), brojac se postavlja na 1. */
  for (i = 1; i < n; i++) {
    for (j = 0; j < n - i; j++)
      /* Provera da li se ispisuje hipotenuza. */
      if (j == n - i - 1)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

```

```
  return 0;
}
```

Rešenje 2.3.55

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j, k;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Brojac j odredjuje koliko ukupno karaktera (praznina i
     karaktera *) u svakom redu se ispisuje. U svakom drugom redu
     ovaj broj se povecava za 2. Na pocetku je 1 (jer se ispisuje
     samo jedna zvezda). */
  j = 1;

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 1; i <= n; i++) {
    /* U svakom drugom redu broj karaktera koji treba da se
       ispisu se uvecava za 2. */
    if (i % 2 == 0)
      j += 2;

    /* Ispisuje se j karaktera. */
    for (k = 0; k < j; k++)
      /* U svakom parnom redu se naiazmenicno
         ispisuju * i praznina. */
      if (i % 2 == 0) {
        if (k % 2 == 0)
          printf("*");
        else
          printf(" ");
      }
      else
      {
        /*U svakom neparnom redu se ispisuju samo *. */
        printf("*");
      }
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.56

```c
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, m;
  int i, j;

  /* Ucitavaju se dimenzije slike. */
  printf("Unesite brojeve n i m: ");
  scanf("%u%u", &n, &m);

  /* Brojac i odredjuje koji red slike se trenutno ispisuje.
     Ukupno ima m redova. */
  for (i = 1; i <= m; i++) {
    /* Brojac j oznacava koja kolona se trenutno ispisuje.
       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 plus jedna kolona
       za najdesniju ivicu, duzina je n*2*(m-1) + 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 ivicama kvardata se iscrtavaju * a na ostalim mestima
           beline. */
        if (j % (2 * (m - 1)) == 0)
          printf("*");
        else
          printf(" ");

    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.57

```c
#include <stdio.h>

int main()
```

```
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Romb se crta crtanjem dva spojena trougla koji se nezavisno
     iscrtavaju. */

  /* 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("*");
    /* Potom se ispisuju karakteri -. */
    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");
  }

  /* II deo: crtanje donjeg trougla. 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 * koje prethode karakterima -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    /* Potom se ispisuju karakteri -. */
    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 2.3.58

```
#include <stdio.h>

int main()
{
```

```
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Slika se sastoji iz dva dela, trougla i kvadrata i svaki deo
     se nezavisno iscrtava. */

  /* I deo: crtanje trougla (krova). */
  for (i = 0; i < n - 1; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");

    /* Posle belina se ispisuje sam trougao.*/
    for (j = 0; j < 2 * i + 1; j++)
      if (j == 0 || j == 2 * i)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  /* II deo: crtanje kvadrata. 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 2.3.59

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;
```

```
  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Prva petlja oznacava broj 'serija' koje ce se ispisati.
     Na primer, za n=5, prva serija je 1 2 3 4 5, druga serija je
     2 3 4 i treca serija je 3.
     Kako se u svakoj sledecoj seriji broj brojeva smanjuje za 2,
     do 0 karaktera u seriji se dolazi posle n/2 koraka, ali
     zaokruzeno navise (5/2 = 2.5 --> 3), a to je isto sto i
     celobrojno (n+1)/2. */
  for (i = 1; i <= (n + 1) / 2; i++) {
    /* U svakoj seriji se ispisuju brojevi izmedju i i n-i+1. */
    for (j = i; j <= n + 1 - i; j++)
      printf("%d ", j);
  }

  return 0;
}
```

Rešenje 2.3.60

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n;
  int i, j;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%u", &n);

  /* Brojac i je redni broj vrste koja se ispisuje. */
  for (i = 1; i <= n; i++) {
    /* U svakoj vrsti se ispisuju brojevi izmedju 1 i n,
       sa korakom i. */
    for (j = 1; j <= n; j+=i)
        printf("%d ", j);

    printf("\n");
  }

  return 0;
}
```

2.5 Funkcije

Zadatak 2.5.1 Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji učitava tri cela broja i ispisuje njihov minimum.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 19 8 14
 Minimum: 8
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: -6 11 -12
 Minimum: -12
```

[Rešenje 2.5.1]

Zadatak 2.5.2 Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja x. Napisati program koji učitava jedan realan broj i ispisuje njegov razlomljeni deo na šest decimala.

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 2.5.2]

Zadatak 2.5.3 Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja n. Napisati program koji učitava ceo pozitivan broj k i ispisuje zbir delilaca svakog broja od 1 do k. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: neispravan unos.
```

[Rešenje 2.5.3]

Zadatak 2.5.4 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 neoznačena broja i ispisuje da li vrednost prvog broja odgovara vrednosti nekog stepena drugog broja.

Primer 1

```
Interakcija sa programom:
 Unesite dva broja: 81 3
 Jeste: 81 = 3^4
```

Primer 2

```
Interakcija sa programom:
 Unesite dva broja: 162 11
 Broj 162 nije stepen broja 11.
```

[Rešenje 2.5.4]

Zadatak 2.5.5 Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najvećeg zajedničkog delioca primenom Euklidovog algoritma. Napisati program koji učitava dva cela broja i ispisuje vrednost njihovog najvećeg zajedničkog delioca.

Primer 1

```
Interakcija sa programom:
 Unesite dva cela broja: 1024 832
 Najveci zajednicki delilac: 64
```

Primer 2

```
Interakcija sa programom:
 Unesite dva cela broja: -900 112
 Najveci zajednicki delilac: 4
```

[Rešenje 2.5.5]

Zadatak 2.5.6 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 pozitivan broj n i ispisuje odgovarajući zbir zaokružen na dve decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 10
 Zbir reciprocnih: 2.93
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 100
 Zbir reciprocnih: 5.19
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: -100
 Greska: neispravan unos.
```

[Rešenje 2.5.6]

Zadatak 2.5.7 Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj x pojavljuje u nizu brojeva koji se unose sve do unosa broja nula. Napisati program koji učitava vrednost broja x i ispisuje koliko puta se njegova vrednost pojavila u unetom nizu.

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: 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: 0
```

[Rešenje 2.5.7]

Zadatak 2.5.8 Broj je prost ako je deljiv samo sa 1 i sa samim sobom.

a) Napisati funkciju `int prost(int x)` koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati jedinicu ako je broj prost i nulu u suprotnom.

b) Napisati funkciju `void prvih_n_prostih(int n)` koja ispisuje prvih n prostih brojeva.

c) Napisati funkciju `void prosti_brojevi_manji_od_n(int n)` koja ispisuje sve proste brojeve manje od broja n.

Napisati program koji učitava pozitivan ceo broj n i ispisuje prvih n prostih brojeva, kao i sve proste brojeve manje od n. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
Unesite broj n: 5
Prvih n prostih: 2 3 5 7 11
Prosti manji od n: 2 3
```

Primer 2

```
Interakcija sa programom:
Unesite broj n: 2
Prvih n prostih: 2 3
Prosti manji od n: ne postoje
```

Primer 3

```
Interakcija sa programom:
Unesite broj n: -11
Greska: neispravan unos.
```

[Rešenje 2.5.8]

Zadatak 2.5.9 Rešiti sledeće zadatke korišćenjem funkcija.

a) Zadatak 1.1.2 rešiti korišćenjem funkcija `int kvadrat(int x)` koja računa kvadrat datog broja i `int kub(int x)` koja računa kub datog broja.

b) Zadatak 2.1.2 rešiti korišćenjem funkcije `float apsolutna_vrednost(float x)` koja izračunava apsolutnu vrednost datog broja.

c) Zadatak 2.3.7 rešiti korišćenjem funkcije `float stepen(float x, int n)` koja računa vrednost n-tog stepena realnog broja x.

d) Zadatak 2.3.29 rešiti korišćenjem funkcije `int fibonaci(int n)` koja računa n-ti element Fibonačijevog niza.

Zadatak 2.5.10 Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji učitava ceo broj i ispisuje aritmetičku sredinju njegovih cifara zaokruženu 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 2.5.10]

Zadatak 2.5.11 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 jedinicu ako se cifra nalazi u broju, a nulu inače. Napisati program koji učitava jedan ceo broj i jednu cifru i u zavisnosti od toga da li se uneta cifra nalazi u zapisu unetog broja, ispisuje odgovarajuću poruku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj i cifru: 17890 7
 Cifra 7 se nalazi u zapisu broja 17890.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj i cifru: 19 6
 Cifra 6 se ne nalazi u zapisu broja 19.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj i cifru: 17890 26
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj i cifru: -1982 9
 Cifra 9 se nalazi u zapisu broja -1982.
```

[Rešenje 2.5.11]

Zadatak 2.5.12 Napisati funkciju `int broj_neparnih_cifara(int x)` koja određuje broj neparnih cifara u zapisu datog celog broja. Napisati program koji učitava cele brojeve sve do unosa broja nula i ispisuje broj neparnih cifara svakog unetog broja.

Primer 1

```
Interakcija sa programom:
Unesite cele brojeve:
2341
Broj neparnih cifara: 2
78
Broj neparnih cifara: 1
800
Broj neparnih cifara: 0
-99761
Broj neparnih cifara: 4
0
```

Primer 2

```
Interakcija sa programom:
Unesite cele brojeve:
987611
Broj neparnih cifara: 4
135
Broj neparnih cifara: 3
-701
Broj neparnih cifara: 2
602
Broj neparnih cifara: 0
-884
Broj neparnih cifara: 0
79901
Broj neparnih cifara: 4
0
```

[Rešenje 2.5.12]

Zadatak 2.5.13 Napisati program za ispitivanje svojstava cifara datog celog broja.

(a) Napisati funkciju `int sve_parne_cifre(int x)` koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati jedinicu ako su sve cifre broja parne, a nulu inače.

(b) Napisati funkciju `int sve_cifre_jednake(int x)` koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati jedinicu ako su sve cifre broja jednake, a nulu inače.

Program učitava ceo broj i u zavisnosti od toga da li su navedena svojstva ispunjena ili ne, ispisuje odgovarajuću poruku.

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
Broj sadrzi bar jednu neparnu cifru.
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
Broj sadrzi bar jednu neparnu cifru.
Cifre broja nisu jednake.
```

[Rešenje 2.5.13]

Zadatak 2.5.14 Napisati funkciju `int ukloni(int n, int p)` koja menja broj n tako što iz njegovog zapisa uklanja cifru na poziciji p. Pozicije se broje sa desna na levo. Cifra jedinica ima poziciju 1. Napisati program koji učitava redni broj pozicije i zatim za cele brojeve koji se unose sve do unosa broja nula, ispisuje brojeve kojima je uklonjena cifra na poziciji p. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite poziciju: 3
 Unesite broj: 1210
 110
 Unesite broj: 18
 18
 Unesite broj: 3856
 356
 Unesite broj: 0
```

Primer 2

```
Interakcija sa programom:
 Unesite poziciju: 1
 Unesite broj: -9632
 -963
 Unesite broj: -2
 0
 Unesite broj: 246
 24
 Unesite broj: -52
 -5
 Unesite broj: 0
```

Primer 3

```
Interakcija sa programom:
 Unesite poziciju: 0
 Greska: neispravan unos.
```

[Rešenje 2.5.14]

Zadatak 2.5.15 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 jedinicu ako je uslov ispunjen, a nulu inače. Napisati program koji učitava dva cela broja i ispisuje da li je za njih pomenuti uslov ispunjen ili ne.

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.
```

Primer 4

```
Interakcija sa programom:
 Unesite dva broja: -777 77
 Uslov nije ispunjen.
```

[Rešenje 2.5.15]

Zadatak 2.5.16 Napisati funkciju `int neopadajuce(int n)` koja ispituje da li su cifre datog celog broja u neopadajućem poretku. Funkcija treba da vrati jedinicu ako cifre ispunjavaju uslov, a nulu inače. Napisati program koji učitava ceo broj i ispisuje poruku da li su cifre unetog broja u neopadajućem poretku.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 2289
 Cifre su u neopadajucem poretku.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 5
 Cifre su u neopadajucem poretku.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 6628
 Cifre nisu u neopadajucem poretku.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj: -23
 Cifre su u neopadajucem poretku.
```

[Rešenje 2.5.16]

Zadatak 2.5.17 Napisati funkciju `int par_nepar(int n)` koja ispituje da li su cifre datog celog broja naizmenično parne i neparne. Funkcija treba da vrati jedinicu ako cifre ispunjavaju uslov, a nulu inače. Napisati program koji učitava ceo broj i ispisuje da li on ispunjava pomenuti uslov ili ne.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 2749
 Broj ispunjava uslov.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: -963
 Broj ispunjava uslov.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 27449
 Broj ne ispunjava uslov.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj n: 5
 Broj ispunjava uslov.
```

[Rešenje 2.5.17]

Zadatak 2.5.18 Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog celog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sve do unosa broja nula 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 2.5.18]

Zadatak 2.5.19 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 jedinicu ako je broj srećan, a nulu inače. Napisati program koji za uneti pozitivan ceo broj n ispisuje sve srećne brojeve od 1 do n. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: 100
 Srecni brojevi:
 1 10 19 28 37 46 55 64 73 82 91 100
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: 0
 Greska: neispravan unos.
```

[Rešenje 2.5.19]

Zadatak 2.5.20 Prirodan 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 jedinicu ako je broj Armstrongov, a nulu inače. Napisati program koji za učitani pozitivan ceo broj proverava da li je Armstrongov. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 0
 Greska: neispravan unos.
```

[Rešenje 2.5.20]

Zadatak 2.5.21 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 2.5.21]

Zadatak 2.5.22 Napisati funkciju `void ispis(float x, float y, int 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 granice intervala i broj tačaka i ispisuje odgovarajuće vrednosti sinusne funkcije, zaokružene na četiri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dva realna broja: 7 32
 Unesite broj n: 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 broj n: 5
 -0.8934 -0.8979 -0.1920 0.6658 0.9968
```

Primer 3

```
Interakcija sa programom:
 Unesite dva realna broja: 8 8
 Greska: neispravan unos.
```

Primer 1

```
Interakcija sa programom:
 Unesite dva realna broja: 7 32
 Unesite broj n: -10
 Greska: neispravan unos.
```

[Rešenje 2.5.22]

Zadatak 2.5.23 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 pre 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 nenegativan ceo broj k, a zatim i karaktere sve do kraja ulaza i nakon svakog učitanog karaktera ispisuje njegovu šifru. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj k: 2
Unesite tekst (CTRL+D za prekid):
c
a
8
8
+
+
Z
X
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj k: -2
Greska: neispravan unos.
```

[Rešenje 2.5.23]

Zadatak 2.5.24 Rešiti sledeće zadatke korišćenjem funkcija.

a) Zadatak 2.3.32 rešiti korišćenjem funkcije `char konverzija(char c)` koja malo slovo pretvara u odgovarajuće veliko i obrnuto.

b) Zadatak 2.3.33 rešiti korišćenjem funkcije `void prebrojavanje()` koja učitava karaktere sve do kraja ulaza i ispisuje broj malih slova, velikih slova, cifara, belina, kao i sumu svih unetih cifara.

Zadatak 2.5.25 Napisati program koji učitava tri cela broja i ispisuje datum sledećeg dana. Zadatak rešiti korišćenjem narednih funkcija.

a) `int prestupna(int godina)` koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati jedinicu ako je godina prestupna ili nulu ako nije.

b) `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu.

c) `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan.

d) `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum ispisuje datum sledećeg dana.

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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.
 Greska: neispravan unos.
```

[Rešenje 2.5.25]

Zadatak 2.5.26 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. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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.
 Greska: neispravan unos.
```

[Rešenje 2.5.26]

Zadatak 2.5.27 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. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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 do Nove godine je: 0
```

Primer 3

```
Interakcija sa programom:
 Unesite datum: 28.2.2004.
 Broj dana do Nove godine je: 307
```

Primer 4

```
Interakcija sa programom:
 Unesite datum: 31.4.2004.
 Greska: neispravan unos.
```

[Rešenje 2.5.27]

Zadatak 2.5.28 Napisati funkciju `int broj_dana_izmedju(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati program koji učitava dva datuma u formatu `dd.mm.gggg` i na standarni izlaz ispisuje broj dana između ta dva datuma. U slučaju neispravnog unosa, ispisati odgovarajuću 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: 104301
```

Primer 3

```
Interakcija sa programom:
 Unesite prvi datum: 24.8.1998.
 Unesite drugi datum: 12.10.2010.
 Broj dana izmedju dva datuma je: 4440
```

Primer 4

```
Interakcija sa programom:
 Unesite prvi datum: 24.8.1998.
 Unesite drugi datum: 31.4.2004.
 Greska: neispravan unos.
```

[Rešenje 2.5.28]

Zadatak 2.5.29 Napisati funkciju `void romb(int n)` koja iscrtava romb čija je stranica dužine n. Napisati program koji učitava ceo pozitivan broj i iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

```
Interakcija sa programom:
 Unesite broj n: -5
 Greska: neispravan unos.
```

[Rešenje 2.5.29]

Zadatak 2.5.30 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 iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 4 1 7 5
 ****
 *
 *******
 *****
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: 5 2 2 10
 *****
 **
 **
 **********
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve: 8 -2 5 4
 Greska: neispravan unos.
```

[Rešenje 2.5.30]

Zadatak 2.5.31 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 iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 4 1 7 5
   *
   *
   **
 * **
 * **
 * **
 ****
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: 5 2 2 4
 *
 *  *
 *  *
 ****
 ****
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve: 8 -2 5 4
 Greska: neispravan unos.
```

[Rešenje 2.5.31]

2.6 Rešenja

Rešenje 2.5.1

```
#include <stdio.h>

/* Funkcija racuna minimum tri cela broja.
   Promenljive u listi argumenata funkcije (x, y i z), kao i one
   deklarisane u samoj funkciji (minimum), lokalne su za tu
   funkciju, sto znaci da im se ne moze pristupiti nigde izvan
   funkcije min. */
int min(int x, int y, int z)
{
  /* Vrednost minimuma se postavlja na vrednost broja x. */
  int minimum = x;
```

```
  /* Vrsi se poredjenje sa druga dva broja i po potrebi
     azuriranje vrednosti minimuma. */
  if (minimum > y)
    minimum = y;
  if (minimum > z)
    minimum = z;

  /* Vrednost minimuma se vraca kao povratna vrednost funkcije. */
  return minimum;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int x, y, z;

  /* Ucitavaju se vrednosti tri broja. */
  printf("Unesite brojeve: ");
  scanf("%d%d%d", &x, &y, &z);

  /* Poziv funkcije i ispis rezultata. */
  printf("Minimum: %d\n", min(x, y, z));

  return 0;
}
```

Rešenje 2.5.2

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

/* Funkcija vraca razlomljeni deo prosledjenog broja. */
float razlomljeni_deo(float x)
{
  /* Napomena: Funkcija fabs racuna apsolutnu vrednost realnog
     broja i njena deklaracija se nalazi u zaglavlju math.h.
     Funkcija abs racuna apsolutnu vrednost celog broja i njena
     deklaracija se nalazi u zaglavlju stdlib.h. */
  x = fabs(x);

  /* Razlomljeni deo broja se dobija tako sto se od samog broja
     oduzme njegov ceo deo. */
  return x - (int) x;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float n;
```

```
23    /* Ucitava se ulazna vrednost. */
      printf("Unesite broj:");
25    scanf("%f", &n);

27    /* Ispis rezultata. */
      printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));
29
      return 0;
31  }
```

Rešenje 2.5.3

```
1   #include <stdio.h>

3   /* Funkcija racuna zbir delilaca broja x. */
    int zbir_delilaca(int x)
5   {
      int i;
7
      /* Inicijalizacija zbira na 0. */
9     int zbir = 0;

11    /* Svaki broj i izmedju 1 i sqrt(x) koji deli broj x se dodaje
         u zbir. Ako je u pitanju broj koji za koji vazi da je i*i
13       jednako x, onda se dodaje samo vrednost i, a ako nije, onda
         se pored vrednosti i dodaje i x/i.
15       Na primer, za x=6, kada je i=2, dodaju se i 2 i 6/2 = 3, a za
         x = 4, kada je i=2, dodaje se samo 2. */
17    for (i = 1; i*i <= x; i++)
      {
19      if (x % i == 0)
        {
21        zbir += i;
          if(i != x/i)
23          zbir += x/i;
        }
25    }

27    /* Povratna vrednost funkcije je dobijeni zbir. */
      return zbir;
29  }

31  int main()
    {
33    /* Deklaracija potrebnih promenljivih. */
      int k, i;
35
      /* Ucitava se broj k. */
37    printf("Unesite broj k:");
      scanf("%d", &k);
39
```

```
  /* Vrsi se provera ispravnosti ulaznih podataka. */
  if (k <= 0)
  {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Za svaki broj od 1 do k se ispisuje zbir delilaca. */
  for (i = 1; i <= k; i++)
    printf("%d ", zbir_delilaca(i));
  printf("\n");

  return 0;
}
```

Rešenje 2.5.4

```
#include <stdio.h>

/* Funkcija za dva 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 int x, unsigned int n)
{
  /* Na pocetku, s = n^i = n^1 = n. */
  int i = 1;
  unsigned int s = n;

  /* U svakoj iteraciji petlje, s se azurira tako da ima
     vrednost n^i. Postupak se ponavlja dok je s manji od x. */
  while (s < x)
  {
    s = s * n;
    i++;
  }

  /* Kako s ima vrednost n^i, ako vazi da je s jednako x, onda
     je bas brojac i trazeni izlozilac. */
  if (s == x)
    return i;

  /* Ako nije, onda se vraca -1. */
  return -1;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int x, n;
  int st;
```

```
35    /* Ucitavaju se vrednosti x i n. */
      printf("Unesite dva broja: ");
37    scanf("%u%u", &x, &n);

39    /* Poziva se napisana funkcija. */
      st = je_stepen(x, n);
41
      /* U zavisnosti od povratne vrednosti funkcije, vrsi se
43       ispis rezultata. */
      if (st != -1)
45      printf("Jeste: %u=%u^%d\n", x, n, st);
      else
47      printf("Broj %u nije stepen broja %u.\n", x, n);

49    return 0;
  }
```

Rešenje 2.5.5

```
1   #include <stdio.h>

3   /* Funkcija racuna nzd(x,y) primenom Euklidovog algoritma. */
    int euklid(int x, int y)
5   {
      int ostatak;
7     /* Euklidov algoritam: trazi se nzd(x,y).
         Na primer nzd(12,18). Postupak koji se primenjuje je sledeci:
9        1. ostatak = x % y = 12 % 18 = 12.
         2. x postaje y => x = 18
11       3. y postaje ostatak => y = 12
         =>
13       1. ostatak = x % y = 18 % 12 = 6
         2. x postaje y => x = 12
15       3. y postaje ostatak => y = 6
         =>
17       1. ostatak = x % y = 12 % 6 = 0
         2. x postaje y => x = 6
19       3. y postaje ostatak => y = 0
         => procedura se zavrsava jer je y jednako 0, a
21       rezultat je poslednji ne-nula ostatak, tj. x.*/
      while (y)
23    {
        ostatak = x % y;
25      x = y;
        y = ostatak;
27    }

29    /* Kao povratna vrednost funkcije se vraca x. */
      return x;
31  }
```

```
33 int main()
   {
35   /* Deklaracija potrebnih promenljivih. */
     int a, b;
37
     /* Ucitavaju se vrednosti a i b. */
39   printf("Unesite dva cela broja:");
     scanf("%d%d", &a, &b);
41
     /* Ispis rezultata. */
43   printf("Najveci zajednicki delilac: %d\n", euklid(a, b));

45   return 0;
   }
```

Rešenje 2.5.6

```
1  #include <stdio.h>

3  /* Funkcija racuna zbir reciprocnih vrednosti brojeva
       iz intervala [1,n]. */
5  float zbir_reciprocnih(int n)
   {
7    float zbir = 0;
     int i;
9
     /* Za svako i izmedju 1 i n na zbir se dodaje vrednost 1/i.
11      Napomena: zbog celobrojnog deljenja mora da stoji 1.0/i. */
     for (i = 1; i <= n; i++)
13     zbir += 1.0 / i;

15   /* Kao povratna vrednost funkcije se vraca izracunati zbir. */
     return zbir;
17 }

19 int main()
   {
21   /* Deklaracija potrebne promenljive. */
     int n;
23
     /* Ucitava se vrednost broja n. */
25   printf("Unesite broj n:\n");
     scanf("%d", &n);
27
     /* Vrsi se provera ispravnosti ulaza. */
29   if(n <= 0)
     {
31     printf("Greska: neispravan unos.\n");
       return -1;
33   }
```

```
35    /* Ispis rezultata. */
      printf("Zbir reciprocnih: %.2f\n", zbir_reciprocnih(n));
37
      return 0;
39  }
```

Rešenje 2.5.7

```
1   #include <stdio.h>

3   /* Funkcija broji koliko puta se realan broj x javlja u nizu unetih
       brojeva. */
5   int prebrojavanje(float x)
    {
7     float y;
      int broj_pojavljivanja = 0;
9
      /* Brojevi se ucitavaju sve do unosa broja nula.
11       Svaki put kada se unese broj koji je jednak broju x,
         brojac pojavljivanja se uveca za 1. */
13    printf("Unesite brojeve:\n");
      while(1)
15    {
        scanf("%f", &y);
17
        if(y == 0)
19        break;

21      if (x == y)
          broj_pojavljivanja++;
23    }

25    return broj_pojavljivanja;
    }
27
    int main()
29  {
      /* Deklaracija potrebnih promenljivih. */
31    float x;
      int rezultat;
33
      /* Ucitava se vrednost broja x. */
35    printf("Unesite broj x: ");
      scanf("%f", &x);
37
      /* Poziva se napisana funkcija i u promenljivoj rezultat se
39       cuva njena povratna vrednost. */
      rezultat = prebrojavanje(x);
41
      /* Ispis rezultata. */
43    printf("Broj pojavljivanja broja %.2f: %d\n", x, rezultat);
```

```
  return 0;
}
```

Rešenje 2.5.8

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

/* Funkcija vraca 1 ako je broj prost, a 0 u suprotnom.  */
int prost(int x)
{
  int i;

  /* Brojevi 2 i 3 su prosti. */
  if (x == 2 || x == 3)
    return 1;

  /* Parni brojevi nisu prosti. */
  if (x % 2 == 0)
    return 0;

  /* Ako se naidje na broj koji deli broj x, onda broj x nije
     prost. Provera se vrsi za sve neparne brojeve izmedju 3 i
     sqrt(x), jer kada bi x imao parnog delioca, onda bi i broj 2
     delio x, a taj uslov je vec proveren. */
  for (i = 3; i <= sqrt(x); i += 2)
    if (x % i == 0)
      return 0;

  /* Ako nijedan od prethodnih uslova nije bio ispunjen, to znaci
     da nijedan broj ne deli x, pa je on prost. */
  return 1;
}

/* Funkcija ispisuje prvih n prostih brojeva.
   Kljucna rec void oznacava da funkcija nema povratnu vrednost. */
void prvih_n_prostih(int n)
{
  int broj_prostih = 0;
  int k = 2;

  /* Petlja se izvrsava dok god se ne istampa n prostih brojeva. */
  while(broj_prostih < n)
  {
    /* Ako se naidje na broj koji je prost, ispisuje se njegova
       vrednost i uvecava se brojac. */
    if(prost(k))
    {
        printf("%d ", k);
        broj_prostih++;
```

```
    }

    /* Prelazi se na sledeci broj. */
    k++;
  }
  printf("\n");
}

/* Funkcija ispisuje sve proste brojeve cija je vrednost manja
   od n. */
void prosti_brojevi_manji_od_n(int n)
{
  /* Ukoliko je n manje ili jednako 2, onda nema prostih brojeva
     koji su manji od njega. U tom slucaju se ispisuje odgovarajuca
     poruka i naredbom return; se izlazi iz funkcije. */
  if(n<=2)
  {
    printf("ne postoje\n");
    return;
  }

  /* Za svaki broj k izmedju 2 i n-1 se vrsi provera da li je prost
     i ako jeste, ispisuje se njegova vrednost. */
  int k = 2;
  while(k < n)
  {
    if(prost(k))
      printf("%d ", k);
    k++;
  }
  printf("\n");
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n;

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

  /* Vrsi se provera ispravnosti ulaza. */
  if(n <= 0)
  {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Ispis rezultata. */
  printf("Prvih n prostih: ");
  prvih_n_prostih(n);
```

```
  printf("Prosti manji od n: ");
  prosti_brojevi_manji_od_n(n);

  return 0;
}
```

Rešenje 2.5.10

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

/* Funkcija racuna aritmeticku sredinu cifara datog celog broja. */
float aritmeticka_sredina(int x)
{
  /* Aritmeticka sredina broja 0 je 0. */
  if (x == 0)
    return 0;

  /* Deklaracija i inicijalizacija brojaca. */
  int zbir_cifara = 0;
  int broj_cifara = 0;

  /* Uzima se apsolutna vrednost broja x kako bi program ispravno
     radio i za negativne brojeve. */
  x = abs(x);

  /* Dok god ima neobradjenih cifara, na zbir se dodaje poslednja
     cifra, brojac cifara se uvecava za 1 i sa broja x se uklanja
     poslednja cifra. */
  while (x)
  {
    zbir_cifara += x % 10;
    broj_cifara++;
    x /= 10;
  }

  /* Kao povratna vrednost funkcije se vraca odgovarajuci
     kolicnik. */
  return (float) zbir_cifara / broj_cifara;
}

int main()
{
  /* Deklaracija potrebne promenljive. */
  int x;

  /* Ucitava se vrednost broja x. */
  printf("Unesite broj: ");
  scanf("%d", &x);

  /* Ispis rezultata. */
```

```
  printf("%.3f\n", aritmeticka_sredina(x));

  return 0;
}
```

Rešenje 2.5.11

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

/* Funkcija ispituje da li se cifra c nalazi u zapisu celog broja
   x. Vraca 1 ako je uslov ispunjen i 0 u suprotnom. */
int sadrzi(int x, int c)
{
  /* Uzima se apsolutna vrednost broja x. */
  x = abs(x);

  /* Izdvaja se cifra po cifra broja x.
     Ako se naidje na cifru cija je vrednost c, onda se kao
     rezultat funkcije vraca 1 (jer x sadrzi c). */
  while (x)
  {
    if (x % 10 == c)
      return 1;
    x /= 10;
  }

  /* Ako se petlja zavrsila, znaci da se nijednom nije naislo
     na cifru c, sto znaci da broj x ne sadrzi cifru c i kao
     povratna vrednost funkcije se vraca 0. */
  return 0;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int x, c;

  /* Ucitavaju se vrednosti x i c. */
  printf("Unesite broj i cifru:");
  scanf("%d%d", &x, &c);

  /* Vrsi se provera ispravnosti ulaza. */
  if(c < 0 || c > 9)
  {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* U zavisnosti od povratne vrednosti funkcije, vrsi se ispis
     odgovarajuce poruke. */
```

```
45   if (sadrzi(x, c))
       printf("Cifra %d se nalazi u zapisu broja %d\n", c, x);
47   else
       printf("Cifra %d se ne nalazi u zapisu broja %d\n", c, x);
49   return 0;
}
```

Rešenje 2.5.12

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

/* Funkcija odredjuje broj neparnih cifara u zapisu datog celog
   broja. */
int broj_neparnih_cifara(int x)
{
  int brojac_neparnih = 0;
  char cifra;
  x = abs(x);

  while (x)
  {
    /* Izdvaja se poslednja cifra broja. */
    cifra = x % 10;
    /* Moze se izbeci koriscenje naredbe if pomocu narednog izraza.
       Naime, vrednost izraza cifra%2 je 1 kada je cifra neparna,
       odnosno 0 kada je parna. Tako ce na broj neparnih cifara
       biti dodata jednica ako je cifra neparna, a ako je parna
       bice dodata 0, sto jeste zeljeno ponasanje. */
    brojac_neparnih += (cifra % 2);
    x /= 10;
  }

  return brojac_neparnih;
}

int main()
{
  /* Deklaracija potrebne promenljive. */
  int x;

  /* Ucitavaju se brojevi sve do unosa broja nula i vrsi se ispis
     broja neparnih cifara za svaki ucitani broj. */
  printf("Unesite cele brojeve:\n");
  while(1)
  {
    scanf("%d", &x);
    if(x == 0)
      break;

    printf("Broj neparnih cifara: %d\n", broj_neparnih_cifara(x));
```

```
  }

  return 0;
}
```

Rešenje 2.5.13

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

/* Funkcija proverava da li su sve cifre broja x parne i vraca
   1 ako je uslov ispunjen i 0 ako nije. */
int sve_parne_cifre(int x)
{
  char cifra;
  x = abs(x);

  /* Ako se naidje na cifru koja nije parna, onda se kao povratna
     vrednost funkcije vraca 0. */
  while (x > 0)
  {
    cifra = x % 10;
    if (cifra % 2 == 1)
      return 0;
    x /= 10;
  }

  /* Ako se doslo do kraja petlje, znaci da se nije naislo ni na
     jednu neparnu cifru, sto znaci da su sve cifre parne i da
     treba da se vrati 1. */
  return 1;
}

/* Funkcija proverava da li su sve cifre broja x jednake i vraca
   1 ako jesu, a 0 u suprotnom. */
int sve_cifre_jednake(int x)
{
  char poslednja_cifra;
  x = abs(x);

  /* Izdvaja se poslednja cifra broja. */
  poslednja_cifra = x % 10;
  x /= 10;

  /* Za sve ostale cifre se proverava da li su jednake poslednjoj.
     Ako se naidje na neku koja nije, onda nisu sve cifre broja
     x jednake i kao povratna vrednost se vraca 0. */
  while (x)
  {
    if (x % 10 != poslednja_cifra)
      return 0;
```

```
    x /= 10;
  }

  /* Ako se stiglo do kraja petlje, znaci da su sve cifre broja
     bile jednake poslednjoj cifri, pa se kao povratna vrednost
     vraca 1. */
  return 1;
}

int main()
{
  /* Deklaracija potrebne promenljive. */
  int x;

  /* Ucitava se broj x. */
  printf("Unesite broj:");
  scanf("%d", &x);

  /* U zavisnosti od povratne vrednosti napisanih funkcija
     vrsi se ispis odgovarajucih poruka. */
  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("Cifre broja su jednake.\n");
  else
    printf("Cifre broja nisu jednake.\n");

  return 0;
}
```

Rešenje 2.5.14

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

/* Funkcija uklanja cifru sa pozicije p iz broja n.
   Cifra jedinica ima poziciju 1, desetica 2, itd.*/
int ukloni(int n, int p)
{
  int znak, tezina_pozicije, levi_deo, desni_deo;

  /* Pamti se znak broja. */
  znak = n < 0 ? 1 : -1;

  /* Uzima se apsolutna vrednost. */
  n = abs(n);
```

```
17    /* Racuna se tezina prosledjene pozicije. */
      tezina_pozicije = pow(10, p-1);
19
      /* Broj se deli na dva dela - deo levo od cifre koja se izbacuje
21       i deo desno od cifre koja se izbacuje. */
      levi_deo = n/(10*tezina_pozicije);
23    desni_deo = n%tezina_pozicije;

25    /* Povratna vrednost funkcije se dobija spajanjem levog i desnog
         dela i mnozenjem sa znakom pocetnog broja. */
27    return znak * (levi_deo*10 + desni_deo);
  }
29
  int main()
31{
      /* Deklaracija potrebnih promenljivih. */
33    int broj, p;

35    /* Ucitava se vrednost pozicije. */
      printf("Unesite poziciju: ");
37    scanf("%d", &p);

39    /* Vrsi se provera ispravnosti ulaza. */
      if(p <= 0)
41    {
        printf("Greska: neispravan unos.\n");
43      return -1;
      }
45
      /* Ucitavaju se brojevi dok se ne unese nula i za svaki
47       ucitani broj se ispisuje broj koji se dobije uklanjanjem
         cifre koja se nalazi na poziciji p. */
49    while (1)
      {
51      printf("Unesite broj: ");
        scanf("%d", &broj);
53
        if (broj == 0)
55        break;

57      printf("%d\n", ukloni(broj, p));
      }
59
      return 0;
61}
```

Rešenje 2.5.15

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

```
3
/* Funkcija proverava da li se neka cifra nalazi u zapisu celog
5    broja i ako se nalazi vraca odgovarajucu poziciju (tj. njenu
     tezinu koja je neki stepen broja 10), a u suprotnom vraca -1.
7    Na primer, za broj = 1234 i cifra = 2, funckija vraca 100. */
int pozicija_cifre(int broj, int cifra)
9 {
  int tezina_pozicije = 1;
11
  while(broj)
13  {
    if(broj%10 == cifra)
15      return tezina_pozicije;

17    tezina_pozicije *= 10;
    broj /= 10;
19  }

21  return -1;
}
23
/* Funkcija iz zapisa broja izbacuje cifru koja se nalazi
25    na prosledjenoj poziciji. Pozicija je stepen broja 10.
     Na primer, za x=1234 i pozicija = 10, treba da se izbaci 3.
27    levi_deo = 1234/(10*10) = 12
     desni_deo = 1234%10 = 4
29    Povratna vrednost je 12*10 + 4 = 124. */
int izbaci_cifru(int broj, int pozicija)
31 {
  int levi_deo = broj/(pozicija*10);
33  int desni_deo = broj%pozicija;
  return levi_deo*10 + desni_deo;
35 }

37 /* Funkcija proverava da li su dva cela broja napisana pomocu istih
     cifara. Vraca 1 ako je uslov ispunjen, a 0 u suprotnom. */
39 int zapis(int x, int y)
{
41  int pozicija;
  x = abs(x);
43  y = abs(y);

45  while (x)
  {
47    /* Proverava se da li y sadrzi poslednju cifru broja x. */
    pozicija = pozicija_cifre(y, x % 10);
49
    /* Ako ne sadrzi, x i y se ne zapisuju pomocu istih cifara. */
51    if(pozicija == -1)
      return 0;
53
    /* Ako sadrzi, iz x se izbacuje poslednja cifra, a iz y se
```

```
55        izbacuje ista ta cifra (koja se nalazi na pronadjenoj
          poziciji. */
57      x /= 10;
        y = izbaci_cifru(y, pozicija);
59    }

61    /* Na kraju petlje iz x su izbacene sve cifre, a vazi da su
         brojevi zapisani pomocu istih cifara samo ukoliko ni u y
63       nema preostalih cifara. */
    return y == 0;
65  }

67  int main()
    {
69    /* Deklaracija potrebnih promenljivih. */
    int x, y;
71
    /* Ucitavaju se vrednosti x i y. */
73  printf("Unesite dva cela broja: ");
    scanf("%d%d", &x, &y);
75
    /* U zavisnosti od povratne vrednosti napisane funckija,
77     ispisuje se odgovarajuca poruka. */
    if (zapis(x, y))
79    printf("Uslov je ispunjen.\n");
    else
81    printf("Uslov nije ispunjen.\n");

83  return 0;
    }
```

Rešenje 2.5.16

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    /* Funkcija proverava da li se cifre u zapisu broja nalaze u
5      neopadajucem poretku. */
    int neopadajuce(int n)
7   {
      int tekuca_cifra, prethodna_cifra;
9     n = abs(n);

11    /* Izvan petlje se izdvaja poslednja cifra u zapisu broja da bi u
         petlji mogla da se poredi sa sledecom. */
13    prethodna_cifra = n % 10;
      n /= 10;
15
      /* U petlji se proverava poredak svake dve susedne cifre. Ukoliko
17       se detektuje da je poredak narusen, izlazi se iz funkcije i
         vraca se vrednost 0. */
```

```
  while (n)
  {
    tekuca_cifra = n % 10;

    if (tekuca_cifra > prethodna_cifra)
      return 0;

    /* Tekuca cifra postaje prethodna za narednu iteraciju. */
    prethodna_cifra = tekuca_cifra;
    n /= 10;
  }

  /* Nakon izlaska iz petlje povratna vrednost funkcije je 1 jer u
     slucaju da je poredak u nekom trenutku narusen iz funkcije bi
     se izaslo jos u petlji. */
  return 1;
}

int main()
{
  /* Deklaracija potrebne promenljive. */
  int n;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj: ");
  scanf("%d", &n);

  /* U zavisnosti od povratne vrednosti napisane funkcije vrsi
     se ispis odgovarajuce poruke. */
  if (neopadajuce(n))
    printf("Cifre su u neopadajucem poretku.\n");
  else
    printf("Cifre nisu u neopadajucem poretku.\n");

  return 0;
}
```

Rešenje 2.5.17

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

/* Funkcija proverava da li su cifre broja naizmenicno parne i
   neparne. Ako je uslov ispunjen vraca 1, u suprotnom vraca 0. */
int par_nepar(int x)
{
  int prethodna_cifra, tekuca_cifra;
  x = abs(x);

  /* Poslednja cifra broja se izdvaja van petlje da bi u petlji
     moglo da se vrsi poredjenje. */
```

```
13    prethodna_cifra = x % 10;
      x /= 10;
15
      while (x)
17    {
        tekuca_cifra = x % 10;
19
        /* Ukoliko su uzastopne cifre iste parnosti, uslov nije
21         ispunjen, rad petlje i funkcije se prekida i vraca se 0. */
        if (tekuca_cifra % 2 == prethodna_cifra % 2)
23        return 0;

25      /* Tekuca cifra postaje prethodna cifra za narednu iteraciju. */
        prethodna_cifra = tekuca_cifra;
27      x /= 10;
      }
29
      /* Sve uzastopne cifre su razlicite parnosti jer ni jednom u
31       petlji uslov da su cifre iste parnosti nije bio ispunjen. */
      return 1;
33  }

35  int main()
    {
37    /* Deklaracija potrebne promenljive. */
      int n;
39
      /* Ucitava se vrednost broja n. */
41    printf("Unesite broj n: ");
      scanf("%d", &n);
43
      /* U zavisnosti od povratne vrednosti napisane funkcije, vrsi
45       se ispis odgovarajuce poruke. */
      if (par_nepar(n))
47      printf("Broj ispunjava uslov.\n");
      else
49      printf("Broj ne ispunjava uslov.\n");

51    return 0;
    }
```

Rešenje 2.5.18

```
    #include<stdio.h>
2   #include<math.h>
    #include<stdlib.h>
4
    /* Funkcija racuna broj cifara celog broja n. */
6   int broj_cifara(int n)
    {
8     int brojac = 0;
```

```
  n = abs(n);

  if(n < 10)
    return 1;

  while(n)
  {
    brojac++;
    n /= 10;
  }

  return brojac;
}

/* Funkcija racuna broj koji se dobija rotacijom broja n za
   jedno mesto ulevo. */
int rotacija(int n)
{
  int znak, prva_cifra, n_bez_prve_cifre, br_cifara;

  znak = (n < 0) ? -1 : 1;
  n = abs(n);
  br_cifara = broj_cifara(n);

  /* Izdvajaju se prva cifra i deo broja bez prve cifre.
     Na primer: ako je n = 1234 onda je br_cifara = 4
     prva_cifra se dobija kao:
     n / (10^(br_cifara-1)) = 1234/1000 = 1.
     n_bez_prve_cifre se dobija kao: n%1000 = 234. */
  int tezina_pozicije = pow(10, br_cifara-1);
  prva_cifra = n / tezina_pozicije;
  n_bez_prve_cifre = n % tezina_pozicije;

  /* Rezultat se dobija nadovezivanjem prve cifre na kraj i
     mnozenjem sa znakom pocetnog broja. */
  return znak * (n_bez_prve_cifre*10 + prva_cifra);
}

int main()
{
  /* Deklaracija potrebne promenljive. */
  int n;

  /* Brojevi se ucitavaju sve do unosa broja nula i ispisuju
     se brojevi dobijeni kao rezultat izvrsavanja funcije rotacija
     nad unetim brojevima. */
  while (1)
  {
    printf("Unesite broj: ");
    scanf("%d", &n);

    if (n == 0)
```

```
      break;

    printf("%d\n", rotacija(n));
  }

  return 0;
}
```

Rešenje 2.5.19

```
#include<stdio.h>

/* Funkcija vraca zbir cifara datog broja x. */
int zbir_cifara(int x)
{
  int zbir = 0;
  while (x)
  {
    zbir += x%10;
    x /= 10;
  }
  return zbir;
}

/* Funkcija vraca 1 ako je broj srecan, a 0 u suprotnom. */
int srecan(int x)
{
  /* Dok god u broju x ima vise od 2 cifre, vrednost broja x se
     zamenjuje sa zbirom njegovih cifara. Na primer, za pocetno
     x = 7698, nakon prve iteracije x postaje 7+6+9+8 = 30, nakon
     druge iteracije x postaje 3 + 0 = 3 i zatim se izlazi iz
     petlje. */
  while(x <= 10)
    x = zbir_cifara(x);

  /* Broj je srecan ako na kraju x ima vrednost 1. */
  return (x == 1);
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, i;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if(n <= 0)
  {
```

```
    printf("Greska: neispravan unos.\n");
43    return -1;
  }
45
  /* Ispisuju se svi srecni brojevi manji ili jednaki n. */
47  printf("Srecni brojevi: ");
  for (i = 1; i <= n; i++)
49    if (srecan(i))
        printf("%d ", i);
51
  printf("\n");
53  return 0;
}
```

Rešenje 2.5.20

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

5 /* Funkcija racuna broj cifara celog broja n. */
  int broj_cifara(int n)
7 {
    int brojac = 0;
9   n = abs(n);

11  if(n < 10)
      return 1;
13
    while(n)
15  {
      brojac++;
17    n /= 10;
    }
19
    return brojac;
21 }

23 /* Funkcija proverava da li je broj Armstrongov. */
  int armstrong(int x)
25 {
    int suma = 0;
27  int n = broj_cifara(x);
    int x_pocetno = x;
29
    /* Racuna se suma n-tih stepena cifara broja x. */
31  while (x)
    {
33    suma += pow(x % 10, n);
      x /= 10;
35  }
```

```
37    /* Ako je suma jednaka pocetnoj vrednosti broja x, broj je
         Armstrongov, u suprotnom nije. */
39    return x_pocetno == suma;
}
41
int main()
43 {
    /* Deklaracija potrebne promenljive. */
45    int x;

47    /* Ucitava se vrednost broja x. */
    printf("Unesite broj: ");
49    scanf("%d", &x);

51    /* Proverava se da li je x Armstrongov broj i ispisuje se
         odgovarauca poruka. */
53    if (armstrong(x))
        printf("Broj je Armstrongov.\n");
55    else
        printf("Broj nije Armstrongov.\n");
57
    return 0;
59 }
```

Rešenje 2.5.21

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

4 /* Funkcija racuna vrednost e^x kao parcijalnu sumu reda
     suma(x^n/n!), gde indeks n ide od od 0 do beskonacno, pri cemu
6    se sumiranje vrsi dok je razlika sabiraka u redu po apsolutnoj
     vrednosti manja od eps. */
8 double e_na_x(double x, double eps)
{
10    double s = 1;
    double clan = 1;
12    int n = 1;

14    /* Parcijalnu suma se formira tako sto se u svakoj iteraciji
         petlje promenljivoj s doda jedan sabirak sume oblika (x^n)/n!
16       koji se cuva u promenljivoj clan.

18       Svaki sabirak se dobija na osnovu prethodnog tako sto se
         prethodni pomnozi sa x i podeli sa n (n predstavlja redni broj
20       sabirka u sumi).

22       Prvi sabirak (kome odgovara n=0) iznosi 1; zbog toga
         promenljive s i clan se inicijalizuju na vrednost 1.
24
```

```
     Sumiranje se sprovodi sve dok 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()
{
  /* Deklaracija potrebnih promenljivih. */
  double x, eps;

  /* Ucitavavaju se vrednosti x i eps. */
  printf("Unesite broj x: ");
  scanf("%lf", &x);
  printf("Unesite eps: ");
  scanf("%lf", &eps);

  /* Ispis rezultata. */
  printf("Rezultat: %f\n", e_na_x(x, eps));
  return 0;
}
```

Rešenje 2.5.22

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

/* Funkcija ispisuje vrednosti funckije sin(x) u n ravnomeno
   rasporedjenih tacaka na intervalu [a,b]. */
void ispis(float a, float b, int n)
{
  float i;
  float korak = (b - a) / (n - 1);

  for (i = a; i <= b; i += korak)
    printf("%.4f ", sin(i));

  printf("\n");
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  float a, b;
  int n;
```

```
22
   /* Ucitavaju se granice intervala i vrsi se provera ispravnosti
24    ulaza. */
   printf("Unesite dva realna broja:");
26 scanf("%f%f", &a, &b);
   if(b >= a)
28 {
     printf("Greska: neispravan unos.\n");
30   return -1;
   }
32
   /* Ucitava se broj n i vrsi se provera ispravnosti ulaza. */
34 printf("Unesite broj n:");
   scanf("%d", &n);
36 if (n <= 1)
   {
38   printf("Greska: neispravan unos.\n");
     return -1;
40 }

42 /* Ispis rezultata. */
   ispis(a, b, n);
44
   return 0;
46 }
```

Rešenje 2.5.23

```
1  #include <stdio.h>

3  /* Funkcija vraca karakter koji se u abecedi nalazi k mesta pre
      datog karaktera c. */
5  char sifra(char c, int k)
   {
7    /* Provera da li je karakter malo slovo. */
     if (c >= 'a' && c <= 'z')
9    {
       /* Ako karakter koji je k pozicija pre datog karaktera ispada
11        iz opsega malih slova. */
       if (c - k < 'a')
13       /* Od k se oduzima rastojanje izmedju c i 'a' (jer je za
            toliko karaktera vec vraceno u nazad), kako bi se odredilo
15          koliko preostali broj karaktera koji treba preskociti od
            karaktera 'z'. */
17       return 'z' - (k - (c - 'a') - 1);
       else
19       /* U suprotnom, karakter c-k ne ispada iz opsega malih slova,
            te je dovoljno njega vratiti. */
21       return c - k;
     }
23   else if (c >= 'A' && c <= 'Z')
```

```
  {
    /* Postupak se ponavlja i za velika slova. */
    if (c - k < 'A')
      return 'Z' - (k - (c - 'A') - 1);
    else
      return c - k;
  }

  /* Ako nije ni malo ni veliko slovo, karakter se ne menja. */
  return c;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int k;
  char c;

  /* Ucitava se vrednost k. */
  printf("Unesite broj k: ");
  scanf("%d", &k);

  /* Ucitavaju se karakteri sve do kraja ulaza i ispisuje se
     njihova sifra. */
  printf("Unesite tekst (CTRL + D za prekid): ");
  while ((c = getchar()) != EOF)
    putchar(sifra(c, k));

  return 0;
}
```

Rešenje 2.5.25

```
#include<stdio.h>

/* Funkcija 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 odredjuje broj dana u datom mesecu. */
int broj_dana(int mesec, int godina)
{
  switch (mesec) {
  case 1:
  case 3:
  case 5:
```

```
19    case 7:
      case 8:
21    case 10:
      case 12:
23      return 31;
      case 4:
25    case 6:
      case 9:
27    case 11:
        return 30;
29    case 2:
        if (prestupna(godina))
31        return 29;
        else
33        return 28;
    }
35  return -1;
}
37
/* Funkcija proverava da li je datum ispravan. Ako je datum
39     ispravan funkcija vraca 1, inace vraca 0. */
int ispravan(int dan, int mesec, int godina)
41 {
    /* Ako je godina negativna, datum nije ispravan. */
43  if (godina < 0)
      return 0;
45
    /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */
47  if (mesec < 1 || mesec > 12)
      return 0;
49
    /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,
51     datum nije ispravan. */
    if (dan < 1 || dan > broj_dana(mesec, godina))
53    return 0;

55  return 1;
}
57
/* Funkcija racuna sledeci dan. */
59 void sledeci_dan(int dan, int mesec, int godina)
{
61  /* Za kraj godine, odnosno za datum 31.12. sledeci datum je 1.1.
       i godina se uvecava za jedan. */
63  if (mesec == 12 && dan == 31)
      printf("1.1.%d.\n", godina + 1);
65  /* Ukoliko je dan jednak poslednjem danu u tom mesecu, odnosno
       ako je jednak broju dana u tom mesecu, onda je sledeci datum
67     kada se mesec uveca za 1, a dan postane 1. Bitan je redosled
       ovih naredbi. Ako bi ovo ispitivanje bilo prvo, onda bi se
69     mesec mogao uvecati na 13. sto ne bi bio ispravan datum. Zato
       se prvo proverava da li je kraj godine, pa tek onda da li je
```

```
71      kraj meseca. */
    else if (dan == broj_dana(mesec, godina))
73      printf("1.%d.%d.\n", mesec + 1, godina);
    /* Ako nije ni jedan od prethodna dva slucaja, onda se dan moze
75      uvecati na 1, bez bojazni da ce se prekoraciti broj dana u
        datom mesecu. */
77  else
      printf("%d.%d.%d.\n", dan + 1, mesec, godina);
79 }

81 int main()
   {
83   /* Deklaracija potrebnih promenljivih. */
     int dan, mesec, godina;
85
     /* Ucitavaju se vrednosti dana, meseca i godine. */
87   printf("Unesite datum:");
     scanf("%d.%d.%d.", &dan, &mesec, &godina);
89
     /* Vrsi se provera ispravnosti datuma. */
91   if (!ispravan(dan, mesec, godina))
     {
93     printf("Greska: neispravan unos.\n");
       return -1;
95   }

97   /* Poziva se funkcija za ispis sledeceg dana. */
     printf("Datum sledeceg dana je:");
99   sledeci_dan(dan, mesec, godina);

101  return 0;
   }
```

Rešenje 2.5.26

Za rešavanje ovog zadatka koristi se funkcija `od_nove_godine` koja je definisana u rešenju zadatka 2.5.28.

Rešenje 2.5.27

Za rešavanje ovog zadatka koristi se funkcija `do_kraja_godine` koja je definisana u rešenju zadatka 2.5.28.

Rešenje 2.5.28

```
#include<stdio.h>
2
/* Funkcija proverava da li je godina prestupna. */
4 int prestupna(int godina)
  {
```

```
6   if ((godina % 100 != 0 && godina % 4 == 0) || godina % 400 == 0)
      return 1;
8   else
      return 0;
10 }

12 /* Funkcija odredjuje broj dana u datom mesecu. */
   int broj_dana(int mesec, int godina)
14 {
   switch (mesec) {
16 case 1:
   case 3:
18 case 5:
   case 7:
20 case 8:
   case 10:
22 case 12:
     return 31;
24 case 4:
   case 6:
26 case 9:
   case 11:
28   return 30;
   case 2:
30   if (prestupna(godina))
       return 29;
32   else
       return 28;
34 }
   return -1;
36 }

38 /* Funkcija proverava da li je datum ispravan. Ako je datum
      ispravan funkcija vraca 1, inace vraca 0. */
40 int ispravan(int dan, int mesec, int godina)
   {
42   /* Ako je godina negativna, datum nije ispravan. */
     if (godina < 0)
44     return 0;

46   /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */
     if (mesec < 1 || mesec > 12)
48     return 0;

50   /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,
        datum nije ispravan. */
52   if (dan < 1 || dan > broj_dana(mesec, godina))
       return 0;
54
     return 1;
56 }
```

```
/* Funkcija odredjuje koliko dana je proteklo od pocetka godine. */
int od_nove_godine(int dan, int mesec, int godina)
{
  int suma_dana = 0, i;

  /* Za sve mesece pre datog datuma dodaje se broj dana za dati
     mesec. */
  for (i = 1; i < mesec; i++)
    suma_dana += broj_dana(mesec, godina);

  /* Na kraju se dodaje koliko je dana proteklo u datom mesecu, a
     to je zadato sa promenljivom dan. */
  return suma_dana + dan;
}

/* Funkcija odredjuje koliko dana ima do kraja godine. */
int do_kraja_godine(int dan, int mesec, int godina)
{
  int suma_dana = 0, i;

  /* Za sve mesece posle datog datuma dodaje se broj dana za dati
     mesec. */
  for (i = mesec + 1; i <= 12; i++)
    suma_dana += broj_dana(mesec, godina);

  /* Na kraju se dodaje koliko je dana je ostalo u datom mesecu. */
  return suma_dana + broj_dana(mesec, godina) - dan;
}

/* Funkcija vraca 1 ako je prvi datum pre drugog datuma. U
   suprotnom vraca 0. */
int prethodi(int dan1, int mesec1, int godina1, int dan2,
             int mesec2, int godina2)
{
  if (godina1 < godina2)
    return 1;
  else if (godina1 > godina2)
    return 0;
  else if (mesec1 < mesec2)
    return 1;
  else if (mesec1 > mesec2)
    return 0;
  else if (dan1 < dan2)
    return 1;
  else
    return 0;
}

/* Funkcija vraca broj dana u datoj godini. */
int broj_dana_u_godini(int godina)
{
  if (prestupna(godina))
```

```
110     return 366;
    else
112     return 365;
}
114
/* Funkcija racuna broj dana izmedju dva datuma. */
116 int broj_dana_izmedju(int dan1, int mesec1, int godina1, int dan2,
                        int mesec2, int godina2) {
118   int pom, i;
    int suma_dana = 0;
120
    /* Vrsi se provera koji od datuma je ranije i ukoliko je to
122      potrebno, razmenjuju se tako da broj 1 ide uz prvi datum. */
    if (!prethodi(dan1, mesec1, godina1, dan2, mesec2, godina2))
124 {
      pom = dan1;
126   dan1 = dan2;
      dan2 = pom;
128
      pom = mesec1;
130   mesec1 = mesec2;
      mesec2 = pom;
132
      pom = godina1;
134   godina1 = godina2;
      godina2 = pom;
136 }

138 /* Ako su godine razlicite. */
    if (godina1 != godina2)
140 {
      /* Za manji datum dodaje se broj dana do kraja godine. */
142   suma_dana = do_kraja_godine(dan1, mesec1, godina1);

144   /* Za sve godine koje su izmedju dve date godine dodaje se broj
         dana u tim godinama. */
146   for (i = godina1 + 1; i < godina2; i++)
        suma_dana += broj_dana_u_godini(i);
148
      /* Za veci datum dodaje se broj dana od pocetka godine. */
150   suma_dana += od_nove_godine(dan2, mesec2, godina2);
    }
152 /* Ako su godine iste, ali meseci razliciti. */
    else if (mesec1 != mesec2)
154 {
      /* Dodaje se broj dana do kraja prvog meseca. */
156   suma_dana = broj_dana(mesec1, godina1) - dan1;

158   /* Dodaje se broj dana za svaki mesec koji je izmedju dva data
         meseca. Kako su godina1 i godina2 jednake svejedno je koja
160      od ove dve promenljive se koristi u pozivu funkcije. */
      for (i = mesec1 + 1; i < mesec2; i++)
```

```
162       suma_dana += broj_dana(i, godina1);

164     /* Dodaje se broj dana od pocetka meseca. */
        suma_dana += dan2;
166   }
      /* Ako su i godine i meseci jednaki. */
168   else
        suma_dana = dan2 - dan1;
170
      return suma_dana;
172 }

174 int main()
    {
176   /* Deklaracija potrebnih promenljivih. */
      int dan1, mesec1, godina1, dan2, mesec2, godina2;
178
      /* Ucitavaju se datumi. */
180   printf("Unesite prvi datum:");
      scanf("%d.%d.%d.", &dan1, &mesec1, &godina1);
182
      printf("Unesite drugi datum:");
184   scanf("%d.%d.%d.", &dan2, &mesec2, &godina2);

186   /* Vrsi se provera ispravnosti unetih datuma. */
      if (!ispravan(dan1, mesec1, godina1)
188       || !ispravan(dan2, mesec2, godina2))
      {
190     printf("Greska: neispravan unos.\n");
        return -1;
192   }

194   /* Ispis rezultata. */
      printf("Broj dana izmedju dva datuma je: %d\n",
196          broj_dana_izmedju(dan1, mesec1, godina1, dan2, mesec2,
                               godina2));
198
      return 0;
200 }
```

Rešenje 2.5.29

```
1 #include<stdio.h>

3 /* Funkcija iscrtava romb. */
  void romb(int n)
5 {
    int i, j;
7
    /* Petlja iscrtava liniju po liniju romba. */
9   for (i = 0; i < n; i++)
```

```
  {
    /* U svakoj liniji prvo se ispisuje n-i-1 razmaka. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");

    /* Ispisuje se n zvezdica. */
    for (j = 0; j < n; j++)
      printf("*");

    /* Ispisuje se novi red. */
    printf("\n");
  }
}

int main()
{
  /* Deklaracija potrebne promenljive. */
  int n;

  /* Ucitava se vrednost broja n. */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Vrsi se provera ispravnosti ulaza. */
  if (n <= 0)
  {
    printf("Greska: neispravan unos.\n");
    return -1;
  }

  /* Iscrtava se romb. */
  romb(n);

  return 0;
}
```

Rešenje 2.5.30

```
#include<stdio.h>

/* Funkcija 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 crta grafikon. */
```

```
void grafikon_h(int a, int b, int c, int d)
{
  /* Prvo se ispisuje a zvezdica. */
  stampaj_zvezdice(a);

  /* Postupak se ponavlja za vrednosti b, c i d. */
  stampaj_zvezdice(b);
  stampaj_zvezdice(c);
  stampaj_zvezdice(d);
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b, c, d;

  /* Ucitavaju se vrednosti a,b,c,d. */
  printf("Unesite brojeve: ");
  scanf("%d%d%d%d", &a, &b, &c, &d);

  /* Vrsi se provera ispravnosti ulaza i ispisuje se rezultat. */
  if (a < 0 || b < 0 || c < 0 || d < 0)
  {
    printf("Greska: neispravan unos.\n");
    return -1;
  }
  else
    grafikon_h(a, b, c, d);

  return 0;
}
```

Rešenje 2.5.31

```
#include<stdio.h>

/* Funkcija racuna najveci od 4 prosledjena broja. */
int maksimum(int a, int b, int c, int d)
{
  int max;

  max = a;
  if (b > max)
    max = b;
  if (c > max)
    max = c;
  if (d > max)
    max = d;

  return max;
}
```

```
19 /* Pomocna funkcija za stampanje beline ili zvezdice. */
   void stampaj_znak(int polje, int granica)
21 {
     if (polje < granica)
23     printf(" ");
     else
25     printf("*");
   }
27
   /* Funkcija iscrtava vertikalni grafikon. */
29 void grafikon_v(int a, int b, int c, int d)
   {
31   int i, max;

33   /* Na pocetku je potrebno pronaci najvecu od ove cetiri
        vrednosti. */
35   max = maksimum(a, b, c, d);

37   /* Grafikon ukupno ima max horizontalnih linija. */
     for (i = 0; i < max; i++)
39   {
       /* U svakoj od horizontalnih linija se nalazi po 4 polja: polje
41        za a,b,c i d uspravnu liniju. U svako od polja treba da se
          upise ili * ili belina, u zavisnosti od vrednosti i toga
43        koja linija se trenutno ispisuje. */

45     /* Stampa se znak za polje a. */
       stampaj_znak(i, max - a);
47
       /* Stampa se znak za polje b. */
49     stampaj_znak(i, max - b);

51     /* Stampa se znak za polje c. */
       stampaj_znak(i, max - c);
53
       /* Stampa se znak za polje d. */
55     stampaj_znak(i, max - d);

57     /* Na kraju svake horizontalne linije stampa se novi red. */
       printf("\n");
59   }
   }
61
   int main()
63 {
     /* Deklaracija potrebnih promenljivih. */
65   int a, b, c, d;

67   /* Ucitavaju se vrednosti cetiri broja. */
     printf("Unesite brojeve: ");
69   scanf("%d%d%d%d", &a, &b, &c, &d);
```

```
71    /* Vrsi se provera ispravnosti ulaza i poziva se funkcija za
         ispis grafikona. */
73    if (a < 0 || b < 0 || c < 0 || d < 0)
      {
75      printf("Greska: neispravan unos.\n");
        return -1;
77    }
      else
79      grafikon_v(a, b, c, d);

81    return 0;
  }
```

3

Predstavljanje podataka

3.1 Nizovi

Zadatak 3.1.1 Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje:

(a) elemente niza koji se nalaze na parnim pozicijama.

(b) parne elemente niza.

Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: neispravan unos.
```

[Rešenje 3.1.1]

Zadatak 3.1.2 Napisati program koji učitava dimenziju niza, elemente niza i zatim menja uneti niz tako što kvadrira sve negativne elemente niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju 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 dimenziju 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 dimenziju niza: 4
 Unesite elemente niza:
 9.53 5 1 4.89
 9.53 5 1 4.89
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 104
 Greska: neispravan unos.
```

[Rešenje 3.1.2]

Zadatak 3.1.3 Ako su $a = (a_1, \ldots, a_n)$ i $b = (b_1, \ldots, b_n)$ vektori dimenzije n, njihov skalarni proizvod se definiše kao $a \cdot b = a_1 \cdot b_1 + \ldots + a_n \cdot b_n$. 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. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: 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: 0
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 0
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 1
 Unesite koordinate vektora a:
 -1
 Unesite koordinate vektora b:
 1
 Skalarni proizvod: -1
```

[Rešenje 3.1.3]

Zadatak 3.1.4 Napisati program koji učitava dimenziju niza, elemente niza i ceo broj k i ispisuje indekse elemenata koji su deljivi sa k. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 6
 Unesite elemente niza: 1 2 3 4 5 6
 Unesite broj k: 0
 Greska: neispravan unos.
```

[Rešenje 3.1.4]

Zadatak 3.1.5 Autobusi su označeni rednim brojevima (počevši od 1) i u nizu se čuva vreme putovanja svakog autobusa u minutama. Međutim, zbog radova na putu između Požege i Užica, svi autobusi koji saobraćaju na tom potezu (autobusi označeni rednim brojevima od k do t) saobraćaju m minuta duže. Napisati program koji učitava broj autobusa n, n celih brojeva koji označavaju vreme putovanja tih autobusa i vrednosti k, t i m i ispisuje vreme putovanja svih autobusa nakon unetih izmena. Maksimalan broj autobusa je 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj autobusa:
 8
 Unesite vreme putovanja:
 24 78 13 124 56 90 205 45
 Unesite vrednosti k, t i m:
 3 6 23
 Vreme putovanja nakon izmena:
 24 78 36 147 79 113 205 45
```

Primer 2

```
Interakcija sa programom:
 Unesite broj autobusa:
 8
 Unesite vreme putovanja:
 24 78 13 124 56 90 205 45
 Unesite vrednosti k, t i m:
 3 15 3
 Greska: neispravan unos.
```

[Rešenje 3.1.5]

Zadatak 3.1.6 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 3.1.6]

Zadatak 3.1.7 Napisati program koji učitava karaktere sve do unosa karaktera * i ispisuje ih u redosledu suprotnom od redosleda čitanja. Maksimalan broj karaktera je 500.

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 3.1.7]

Zadatak 3.1.8 Napisati program koji učitava karaktere 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 koristiti 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
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Programiranje 1 je zanimljivo!!
 Karakter 1 se pojavljuje 1 puta
 Karakter a se pojavljuje 3 puta
 Karakter e se pojavljuje 2 puta
 Karakter g se pojavljuje 1 puta
 Karakter i se pojavljuje 3 puta
 Karakter j se pojavljuje 3 puta
 Karakter l se pojavljuje 1 puta
 Karakter m se pojavljuje 2 puta
 Karakter n se pojavljuje 2 puta
 Karakter o se pojavljuje 2 puta
 Karakter r se pojavljuje 3 puta
 Karakter v se pojavljuje 1 puta
 Karakter z se pojavljuje 1 puta
 Karakter P se pojavljuje 1 puta
```

[Rešenje 3.1.8]

Zadatak 3.1.9 Napisati program koji učitava jednu liniju teksta 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:
 Tasi, tasi, TaNaNa i SVILENA marama.....
  a:9 b:0 c:0 d:0 e:1 f:0 g:0 h:0 i:4 j:0 k:0 l:1 m:2
  n:3 o:0 p:0 q:0 r:1 s:3 t:3 u:0 v:1 w:0 x:0 y:0 z:0
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Mihailo Petrovic Alas (6 Maj 1868 - 8 Jun 1943)
  a:4 b:0 c:1 d:0 e:1 f:0 g:0 h:1 i:3 j:2 k:0 l:2 m:2
  n:1 o:2 p:1 q:0 r:1 s:1 t:1 u:1 v:1 w:0 x:0 y:0 z:0
```

Primer 3

```
Interakcija sa programom:
 Alan Matison Tjuring (London, 23. jun 1912 - Cesir, 7. jun 1954)
  a:3 b:0 c:1 d:1 e:1 f:0 g:1 h:0 i:3 j:3 k:0 l:2 m:1
  n:7 o:3 p:0 q:0 r:2 s:2 t:2 u:3 v:0 w:0 x:0 y:0 z:0
```

[Rešenje 3.1.9]

Zadatak 3.1.10 Takmičari na Beogradskom maratonu su označeni rednim brojevima počevši od 0, a vreme za koje su istrčali maraton je izraženo u minutima. Ovi podaci zadati su nizom celih brojeva, pri čemu indeks niza označava redni broj takmičara, a vrednosti u nizu označavaju vreme trčanja. Napisati sledeće funkcije za obradu navedenih podataka:

(a) `void ucitaj(int a[], int n)` koja učitava elemente niza a dimenzije n.

(b) `void ispisi(int a[], int n)` koja ispisuje elemente niza a dimenzije n.

(c) `int suma(int a[], int n)` koja računa i vraća ukupno vreme trčanja svih takmičara.

(d) `float prosek(int a[], int n)` koja računa i vraća prosečno vreme (aritmetičku sredinu) trčanja takmičara.

(e) `int maksimum(int a[], int n)` koja izračunava i vraća najduže vreme trčanja takmičara.

(f) `int pozicija_minimum(int a[], int n)` koja vraća redni broj pobednika Beogradskog maratona, tj. onog takmičara koji je najkraće trčao. U slučaju da ima više takvih takmičara, vratiti onog sa najmanjim indeksom.

Napisati program koji učitava podatke o rezultatima takmičara na maratonu i ispisuje učitane podatke, ukupno, prosečno i maksimalno vreme trčanja, kao i redni broj pobednika maratona. Maksimalan broj takmičara je 1000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 5
 Unesite elemente niza: 140 126 170 220 130
 Vreme trcanja takmicara: 140 126 170 220 130
 Ukupno vreme: 786
 Prosecno vreme trcanja: 157.20
 Maksimalno vreme trcanja: 220
 Indeks pobednika: 1
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 -5
 Greska: neispravan unos.
```

[Rešenje 3.1.10]

Zadatak 3.1.11 Napisati funkciju koja izračunava broj elemenata celobrojnog niza koji su manji od poslednjeg elementa niza. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje broj elemenata koji zadovoljavaju pomenuti uslov. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 11 2 4 9
 2
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 7
 Unesite elemente niza: 7 2 1 14 65 2 8
 4
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 5
 Unesite elemente niza: 25 18 29 30 14
 0
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza:
 -45
 Greska: neispravan unos.
```

[Rešenje 3.1.11]

Zadatak 3.1.12 Napisati funkciju koja izračunava broj parnih elemenata celobrojnog niza koji prethode maksimalnom elementu niza. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje broj elemenata koji zadovoljavaju pomenuti uslov. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 11 2 4 9
 0
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 7
 Unesite elemente niza: 7 2 1 14 65 2 8
 2
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 5
 Unesite elemente niza: 25 18 29 30 14
 1
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 105
 Greska: neispravan unos.
```

[Rešenje 3.1.12]

Zadatak 3.1.13 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 program koji učitava dimenziju niza, elemente niza i vrednosti i i j i zatim ispisuje zbir u datom opsegu. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju 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 dimenziju niza: 3
 Unesite elemente niza: -2 8 1
 Unesite vrednosti za i i j: 1 12
 Greska: neispravan unos.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 7
 Unesite elemente niza: -2 5 9 11 6 -3 -4
 Unesite vrednosti za i i j: 2 5
 Zbir: 23
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 9 5 7 6
 Unesite vrednosti za i i j: 2 2
 Zbir: 7
```

[Rešenje 3.1.13]

Zadatak 3.1.14 Napisati funkciju `float zbir_pozitivnih(float a[], int n, int k)` koja izračunava zbir prvih k pozitivnih elemenata realnog niza a dužine n. Napisati program koji učitava dimenziju niza, elemente niza i broj k i zatim ispisuje zbir prvih k pozitivnih elemenata niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 2.34 1 -12.7 5.2 -8 -6.2 7 14.2
 Unesite vrednost k: 3
 Zbir je: 8.54
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 3
 Unesite elemente niza:
 -6.598 -8.14 -15
 Unesite vrednost k: 4
 Zbir je: 0.00
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 7
 Unesite elemente niza:
 -35.11 5.29 -1.98 12.1 12.2 -3.33 -4.17
 Unesite vrednost k: 15
 Zbir: 29.59
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 3
 Unesite elemente niza:
 -0.11 5.29 -4.17
 Unesite vrednost k: -15
 Greska: neispravan unos.
```

[Rešenje 3.1.14]

Zadatak 3.1.15 Napisati funkciju koja menja niz tako što razmenjuje mesta najmanjem i najvećem elementu niza. Ukoliko se neki od njih javlja više puta, u obzir uzeti prvo pojavljivanje. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje izmenjeni niz. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: neispravan unos.
```

[Rešenje 3.1.15]

Zadatak 3.1.16 Napisati program koji vrši pretragu niza nadmorskih visina.

(a) Napisati funkciju koja proverava da li niz sadrži zadatu nadmorsku visinu 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 nadmorsku visinu m ili -1 ukoliko element nije u nizu.

(c) Napisati funkciju koja vraća vrednost poslednje pozicije na kojoj se nalazi element koji ima nadmorsku visinu m ili -1 ukoliko element nije u nizu.

Program učitava podatke o nadmorskim visinama i ceo broj m i ispisuje da li u nizu postoji podatak o unetoj nadmorskoj visini. Ukoliko postoji, ispisuje i poziciju prvog i poslednjeg pojavljivanja vrednosti m u nizu. Pozicije se broje od 0. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 7
 Unesite podatke:
 800 1100 -200 1400 -200 1100 800
 Unesite vrednost m:
 1100
 Nadmorska visina 1100 se nalazi medju podacima.
 Pozicija prvog pojavljivanja: 1
 Pozicija poslednjeg pojavljivanja: 5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 -5
 Greska: neispravan unos.
```

[Rešenje 3.1.16]

Zadatak 3.1.17 Marko skuplja sličice za Svetsko prvenstvo u fudbalu. Marko je primetio da mu se neke sličice ponavljaju i rešio je da ih razmeni sa drugarima. Napisati funkciju `int duplikati(int a[], int n, int b[])` koja od niza a dimenzije n formira niz b koji sadrzi sve elemente niza a koji se pojavljuju bar dva puta u nizu. Funkcija kao povratnu vrednost vraća dimenziju niza b. Napisati program koji učitava Markove sličice i ispisuje sve duplikate. Maksimalan broj elemenata niza je 600. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju 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 dimenziju niza: 13
 Unesite elemente niza a:
 8 26 7 2 1 1 7 2 2 2 7 5 1
 Elementi niza b: 7 2 1
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza: 2
 Unesite elemente niza a:
 9 5
 Elementi niza b:
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza: 0
 Greska: neispravan unos.
```

[Rešenje 3.1.17]

Zadatak 3.1.18 Palindrom je tekst koji se isto čita i sa leve i sa desne strane. Napisati funkciju koja proverava da li elementi niza karaktera čine palindrom (zanemariti velika/mala slova). Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje da li je uneti niz karaktera palindrom. Maksimalan

broj elemenata niza je 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 15
 Unesite elemente niza:
 AnaVoliMilovana
 Niz jeste palindrom.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 26
 Unesite elemente niza:
 Zanimljivo je programirati!
 Niz nije palindrom.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza:
 1
 Unesite elemente niza:
 a
 Niz jeste palindrom.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza:
 226
 Greska: neispravan unos.
```

[Rešenje 3.1.18]

Zadatak 3.1.19 Napisati funkciju koja proverava da li su elementi celobrojnog niza uređeni neopadajuće. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje da li je pomenuti uslov ispunjen. Maksimalan broj elemenata niza je 300. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 8
 Unesite elemente niza:
 -40 -8 -8 2 30 30 46 200
 Niz jeste uredjen neopadajuce.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 4
 Unesite elemente niza:
 4 23 15 30
 Niz nije uredjen neopadajuce.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza:
 1
 Unesite elemente niza:
 5
 Niz jeste uredjen neopadajuce.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza:
 304
 Greska: neispravan unos.
```

[Rešenje 3.1.19]

Zadatak 3.1.20 Svaki indeks niza označava jedan dan u mesecu, a elementi celobrojnog niza predstavljaju broj artikala koji se prodao tog dana. Napisati funkciju koja računa najduži uzastopnu seriju dana za koju važi da broj prodatih artikala nije opao. Napisati program koji učitava broj dana u mesecu, broj prodatih artikala za svaki dan u mesecu i zatim ispisuje dužinu izračunate serije. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 30
 Unesite broj prodatih artikala:
 89 171 112 67 119 36 181 157
 49 96 73 116 21 172
 140 0 23 71 157 135 11 166 21
 56 56 87 103 183 148 174
 Duzina najduzeg neopadajuceg
 prodavanja je 6.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 31
 Unesite broj prodatih artikala:
 215 223 262 95 18 116 334 97
 146 146 19 314 270 115 21 40
 253 27 210 68 96 175 41 242
 98 163 8 218 107 102
 Duzina najduzeg neopadajuceg
 prodavanja je 3.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: -5
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 31
 Unesite broj prodatih artikala:
 -215 223 262 95 18 116 334 97
 146 146 19 314 -270 115 21 40
 253 27 210 68 96 175 41 242
 98 163 -8 218 107 102
 Greska: neispravan unos.
```

[Rešenje 3.1.20]

Zadatak 3.1.21 Napisati funkciju koja određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje dužinu najduže serije jednakih elemenata niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 9 -1 2 2 2 2 80 -200
 Duzina najduze serije je 4.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 9 9 0 -3 -3 -3 -3 72
 Duzina najduze serije je 4.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 1 2 3 4 5 6 7 8
 Duzina najduze serije je 1.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 108
 Greska: neispravan unos.
```

[Rešenje 3.1.21]

Zadatak 3.1.22 Napisati funkciju koja određuje da li se jedan niz javlja kao podniz uzastopnih elemenata drugog niza. Zadatak rešiti na dva načina:

(a) Tako da elementi jednog niza moraju da budu uzastopni u drugom nizu.

(b) Tako da elementi jednog niza ne moraju da budu uzastopni u drugom nizu, ali je redosled pojavljivanja isti.

Napisati program koji učitava dimenzije i elemente dva niza i zatim ispisuje da li je neki od pomenutih uslova ispunjen. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 -4 2 7 90 -22 15 14 7
 Unesite dimenziju niza: 4
 Unesite elemente niza: 90 -22 15 14
 Elementi drugog niza cine
 uzastopni podniz prvog niza.
 Elementi drugog niza cine
 podniz prvog niza.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 -4 2 7 90 -22 15 14 7
 Unesite dimenziju niza: 4
 Unesite elemente niza: 2 7 15 7
 Elementi drugog niza ne cine
 uzastopni podniz prvog niza.
 Elementi drugog niza cine
 podniz prvog niza.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 -4 2 7 90 -22 15 14 7
 Unesite dimenziju niza: 4
 Unesite elemente niza: 90 -22 200 1
 Elementi drugog niza ne cine
 uzastopni podniz prvog niza.
 Elementi drugog niza ne
 cine podniz prvog niza.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 -4 2 7 90 -22 15 14 7
 Unesite dimenziju niza: 1
 Unesite elemente niza: 90
 Elementi drugog niza cine
 uzastopni podniz prvog niza.
 Elementi drugog niza cine
 podniz prvog niza.
```

[Rešenje 3.1.22]

Zadatak 3.1.23 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 učitava dimenziju niza i elemente niza i ispisuje da li je uneti niz permutacija. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza: 5
 Unesite elemente niza:
 1 5 4 3 2
 Uneti niz je permutacija.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza: 6
 Unesite elemente niza:
 2 3 3 1 1 5
 Uneti niz nije permutacija.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza: 1
 Unesite elemente niza:
 1
 Uneti niz je permutacija.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza: 101
 Greska: neispravan unos.
```

[Rešenje 3.1.23]

Zadatak 3.1.24 Napisati program koji učitava dva cela broja x i y i proverava da li se uneti brojevi zapisuju pomoću istih cifara. UPUTSTVO: *Zadatak rešiti korišćenjem nizova..*

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.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dva broja: -1 1
 Brojevi se zapisuju istim ciframa.
```

[Rešenje 3.1.24]

Zadatak 3.1.25 Napisati sledeće funkcije koje vrše transformacije celobrojnog niza:

(a) Napisati funkciju koja obrće elemente niza.

(b) Napisati funkciju koja rotira niz ciklično za jedno mesto ulevo.

(c) Napisati funkciju koja rotira niz ciklično za k mesta ulevo.

Napisati program koji učitava dimenziju niza, elemente niza i pozitivan ceo broj k i ispisuje niz koji se dobije nakon obrtanja početnog niza, niz koji se dobije rotiranjem tog niza za jedno mesto ulevo i niz koji se dobije dodatnim rotiranjem tog niza za k mesta ulevo. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 6
 7 -3 11 783 26 -19
 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: neispravan unos.
```

[Rešenje 3.1.25]

Zadatak 3.1.26 Napisati funkciju `void ukrsti(int a[], int b[], int n, int c[])` koja formira niz c koji se dobija naizmeničnim raspoređivanjem elemenata nizova a i b, tj. $c = [a_0, b_0, a_1, b_1, \ldots, a_{n-1}, b_{n-1}]$. Napisati program koji učitava dimenziju i elemente dva niza i ispisuje niz koji se dobija ukrštanjem unetih nizova. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: neispravan unos.
```

[Rešenje 3.1.26]

Zadatak 3.1.27 Napisati funkciju `void spoji(int a[], int b[], int n, int c[])` koja od nizova a i b dimenzije n formira niz c čija prva polovina odgovara elemetima niza b, a druga polovina elementima niza a, tj. $c = [b_0, b_1, \ldots, b_{n-1}, a_0, a_1, \ldots, a_{n-1}]$. Napisati program koji učitava dimenziju i elemente dva niza i ispisuje niz koji se dobija spajanjem unetih nizova na pomenuti način. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju nizova: 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 dimenziju nizova: 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 nizova: 145
 Greska: neispravan unos.
```

[Rešenje 3.1.27]

*** Zadatak 3.1.28** Napisati funkciju `void spoji_sortirano(int a[], int b[], int n, int c[])` koja od nizova a i b dimenzije n koji su uređeni neopadajuće po vrednosti formira niz c koji je uređen na isti način. Napisati program koji učitava dimenziju i elemente uređenih nizova a i b i ispisuje niz koji se dobije spajanjem ovih nizova na pomenuti način. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju nizova: 5
 Unesite elemente sortiranog niza:
 2 11 28 40 63
 Unesite elemente sortiranog niza:
 -19 -5 5 11 52
 -19 -5 2 5 11 11 28 40 52 63
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju nizova: 3
 Unesite elemente sortiranog niza:
 -2 4 8
 Unesite elemente sortiranog niza:
 6 15 19
 -2 4 6 8 15 19
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju nizova: 145
 Greska: neispravan unos.
```

[Rešenje 3.1.28]

Zadatak 3.1.29 Napisati funkciju `void promeni_redosled(int a[], int n)` koja menja redosled elementima niza a dimenzije n tako da se parni elementi niza nalaze na početku niza, a neparni na kraju. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji je izmenjen na pomenuti način. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. Napomena: *Nije dozvoljeno koristiti pomoćne nizove.*

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 10
 -2 8 11 53 59 20 17 -8 3 14
 Rezultujuci niz:
 14 142 -6 -278 28 34 33 -69 -9 9
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 10
 9 142 -9 -278 -69 33 34 28 -6 14
 Rezultujuci niz:
 -2 8 14 -8 20 59 17 53 3 11
```

[Rešenje 3.1.29]

Zadatak 3.1.30 Napisati funkciju koja iz datog niza briše sve elemente koji su prosti brojevi. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem pomenutih elemenata. Zadatak rešiti uz korišćenje pomoćnog niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 5
 Unesite elemente niza: 11 5 6 48 8
 6 48 8
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 11 5 19 21
 21
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 5
 Unesite elemente niza: 12 18 9 31 7
 12 18 9
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 3
 Unesite elemente niza: -31 11 -19
```

Primer 5

```
Interakcija sa programom:
 Unesite dimenziju niza: 5
 Unesite elemente niza: -2 15 -11 8 7
 15 8
```

[Rešenje 3.1.30]

Zadatak 3.1.31 Napisati funkciju koja iz datog niza briše sve neparne elemente. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem neparnih elemenata. Zadatak rešiti bez korišćenja pomoćnog niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 8 9 15 12
 8 12
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 6
 Unesite elemente niza: 21 5 3 22 19 188
 22 188
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 133 129 121 101
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju niza: 8
 Unesite elemente niza:
 15 -22 -23 13 18 46 14 -31
 -22 18 46 14
```

[Rešenje 3.1.31]

Zadatak 3.1.32 Napisati funkciju koja iz datog niza briše sve elemente koji nisu deljivi svojom poslednjom cifrom. Izuzetak su elementi čija je poslednja cifra nula i koje zbog toga treba zadržati. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem pomenutih elemenata. Zadatak rešiti bez korišćenja pomoćnog niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 9
 Unesite elemente niza a:
 173 -25 23 7 17 25 34 61 -4612
 -25 7 25 61 -4612
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 0
 Greska: neispravan unos.
```

[Rešenje 3.1.32]

Zadatak 3.1.33 Napisati funkciju koja iz datog niza briše sve brojeve koji nisu deljivi svojim indeksom. Ne razmatrati da li je u novom nizu, nakon brisanja i pomeranja, element deljiv svojim indeksom. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem pomenutih elemenata. Zadatak rešiti bez korišćenja pomoćnog niza. Maksimalan broj elemenata niza je 700. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. Napomena: *Nulti element niza treba zadržati jer nije dozvoljeno deljenje nulom.*

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 10
 Unesite elemente niza:
 4 2 1 6 7 8 10 2 16 3
 Novi niz:
 4 2 6 16
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 10
 Unesite elemente niza:
 -8 5 10 6 7 10 8 2 16 27
 Novi niz:
 -8 5 10 6 10 16 27
```

[Rešenje 3.1.33]

Zadatak 3.1.34 Korišćenjem nizova moguće je predstaviti skupove podataka. Napisati program koji demonstrira osnovne operacije nad skupovima (uniju, presek i razliku). Pomoću dva niza predstaviti dva skupa celih brojeva. Maksimalan broj elemenata niza je 500. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza a: 5
 Unesite elemente niza a: 1 2 3 4 5
 Unesite broj elemenata niza b: 3
 Unesite elemente niza b: 5 4 9
 Unija: 1 2 3 4 5 9
 Presek: 4 5
 Razlika: 1 2 3
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza a: 3
 Unesite elemente niza a: 11 4 -5
 Unesite broj elemenata niza b: 2
 Unesite elemente niza b: 18 9
 Unija: 11 4 -5 18 9
 Presek:
 Razlika: 11 4 -5
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza a: 6
 Unesite elemente niza a: 12 7 9 12 5 1
 Greska: skup ne moze imati duplikate.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj elemenata niza a: -2
 Greska: neispravan unos.
```

[Rešenje 3.1.34]

Zadatak 3.1.35 Prilikom ulaska u banku klijent dobija redni broj, a u nizu se čuva redosled opsluživanja klijenata. Tako, prvi klijent u nizu će biti prvi uslužen, a klijent koji je poslednji došao se nalazi na kraju niza. Redni brojevi se izdaju počevši od 1 svakog radnog dana, ali se niz za redosled stalno menja. Dodatno, postoje specijalni klijenti (npr. oni koji podižu stambeni kredit) koji mogu dobiti i negativan redni broj da bi se razlikovali od uobičajenih usluga koje banka omogućava. Pomozite radniku obezbeđenja da lakše prati redosled opsluživanja klijenata.

(a) Napisati funkciju koja ubacuje datog klijenta sa rednim brojem x na kraj niza.

(b) Napisati funkciju koja ubacuje datog klijenta sa rednim brojem x na početak niza (lica sa posebnim potrebama, trudnice, stara lica i ostale ugrožene kategorije).

(c) Napisati funkciju koja ubacuje datog klijenta sa rednim brojem x na datu poziciju k (manje prioritetna lica, recimo službena lica ili roditelji sa decom, poziciju k bira radnik obezbeđenja).

(d) Napisati funkciju koja izbacuje prvi element niza (usluženi klijent).

(e) Napisati funkciju koja izbacuje poslednji element niza (klijent je odustao jer je shvatio da ima mnogo klijenata ispred njega).

(f) Napisati funkciju koja izbacuje element sa date pozicije k (klijent je odustao jer je dugo čekao).

Napisati program koji testira rad funkcija. Maksimalan broj klijenata u jednom danu je 2000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite trenutni broj klijenata:
 8
 Unesite niz sa rednim brojevima klijenata:
 2 5 -2 16 33 19 8 11
 Unesite klijenta kojeg treba ubaciti u niz:
 35
 Niz nakon ubacivanja klijenta: 2 5 -2 16 33 19 8 11 35
 Unesite prioritetnog klijenta kojeg treba ubaciti u niz:
 36
 Niz nakon ubacivanja klijenta: 36 2 5 -2 16 33 19 8 11 35
 Unesite prioritetnog klijenta kojeg treba ubaciti u niz i njegovu poziciju:
 -6 2
 Niz nakon ubacivanja klijenta: 36 2 -6 5 -2 16 33 19 8 11 35
 Niz nakon odlaska klijenta: 2 -6 5 -2 16 33 19 8 11 35
 Niz nakon odlaska poslednjeg klijenta: 2 -6 5 -2 16 33 19 8 11
 Unesite redni broj klijenta koji je napustio red:
 -2
 Niz nakon odlaska klijenta: 2 -6 5 16 33 19 8 11
```

[Rešenje 3.1.35]

3.2 Rešenja

Rešenje 3.1.1

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

/* Predprocesorska direktiva kojom se definise maksimalan broj
   elemenata niza. */
#define MAKS 100

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS];
  int n, i;

  /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
     ulaza. */
  printf("Unesite dimenziju niza:\n");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
```

```
20      printf("Greska: neispravan unos.\n");
        /* Za izlazak iz programa moze da se koristi i funkcija exit.
22         Argument EXIT_FAILURE oznacava da je doslo do neke greske
           pri izvrsavanju programa. Deklaracija ove funkcije se nalazi
24         u zaglavlju stdlib.h. */
      exit(EXIT_FAILURE);
26    }

28    /* Ucitavaju se elementi niza. */
      printf("Unesite elemente niza:\n");
30    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
32
      /* Ispisuju se elementi niza na parnim pozicijama. */
34    printf("Elementi niza na parnim pozicijama:\n");
      for (i = 0; i < n; i += 2)
36      printf("%d ", a[i]);
      printf("\n");
38
      /* Ispisuju se parni elementi niza. */
40    printf("Parni elementi niza:\n");
      for (i = 0; i < n; i++)
42      if (a[i] % 2 == 0)
          printf("%d ", a[i]);
44    printf("\n");

46    /* Kada se funkciji exit prosledi EXIT_SUCCESS to znaci da se
         program uspesno zavrsio. Efekat je isti kao i da je navedeno
48       return 0; na ovom mestu. */
      exit(EXIT_SUCCESS);
50  }
```

Rešenje 3.1.2

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    #define MAKS 100
5
    int main()
7   {
      /* Deklaracija potrebnih promenljivih. */
9     float brojevi[MAKS];
      int n, i;
11
      /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
13       ulaza. */
      printf("Unesite dimenziju niza: ");
15    scanf("%d", &n);
      if (n <= 0 || n > MAKS)
17    {
```

```
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* 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,
     kvadrira se tako sto se pomnozi 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");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.3

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

#define MAKS 100

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a[MAKS];
  int b[MAKS];
  int n, i, skalarni_proizvod;

  /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
     ulaza. */
  printf("Unesite dimenziju vektora: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* 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 zadatoj formuli. */
  skalarni_proizvod = 0;
  for (i = 0; i < n; i++)
    skalarni_proizvod += a[i] * b[i];

  /* Ispis rezultata. */
  printf("Skalarni proizvod: %d\n", skalarni_proizvod);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.4

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

#define MAKS 100

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int brojevi[MAKS];
  int n, i, k, indikator;

  /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
     ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &brojevi[i]);

  /* Ucitava se broj k i proverava se njegova ispravnost. */
  printf("Unesite broj k: ");
  scanf("%d", &k);
  if (k == 0)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }
```

```
35
   /* Promenljiva koja cuva informaciju o tome da li je u nizu
37    postojao element koji je deljiv brojem k. Inicijalna vrednost
      je 0. */
39 indikator = 0;

41 /* Ukoliko je element niza deljiv brojem k, indikator se
      postavlja na 1 i ispisuje se indeks tog elementa. */
43 for (i = 0; i < n; i++)
   {
45   if (brojevi[i] % k == 0)
     {
47     indikator = 1;
       printf("%d ", i);
49   }
   }
51
   /* Ukoliko je indikator jednak nuli to znaci da ne postoji
53    element u nizu koji je deljiv brojem k. */
   if (indikator == 0)
55   printf("U nizu nema elemenata koji su deljivi brojem %d.\n", k);

57 exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.5

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

4 /* Indeksiranje autobusa pocinje od 1, pa zato maksimalna
     dimenzija niza mora biti 201, a ne 200. */
6 #define MAKS 201

8 int main()
{
10   /* Deklaracija potrebnih promenljivih. */
   int n, niz[MAKS], i;
12   int k, t, m;

14   /* Ucitava se dimenzija niza. */
   printf("Unesite broj autobusa: ");
16   scanf("%d", &n);

18   /* Vrsi se provera ispravnosti ulaza. */
   if (n <= 0 || n > MAKS)
20   {
     printf("Greska: neispravan unos.\n");
22     exit(EXIT_FAILURE);
   }
24
```

```
  /* Ucitavaju se vremena putovanja. */
  printf("Unesite vreme putovanja:\n");
  for (i = 1; i <= n; i++)
    scanf("%d", &niz[i]);

  /* Ucitavaju se redni brojevi autobusa cije se vreme putovanja
     menja i vrednost kasnjenja. */
  printf("Unesite vrednosti k, t i m:\n");
  scanf("%d%d%d", &k, &t, &m);

  /* Vrsi se provera ispravnosti ulaza. */
  if (k <= 0 || k > n || t <= 0 || t > n || m < 0)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Azuriraju se vremena putovanja. */
  for (i = k; i <= t; i++)
    niz[i] += m;

  /* Ispis rezultata. */
  printf("Vreme putovanja nakon izmena:");
  for (i = 1; i <= n; i++)
    printf("%d ", niz[i]);
  printf("\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.6

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

#define BROJ_CIFARA 10

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int x, x_original, cifra, i;
  int brojaci[BROJ_CIFARA];

  /* Ucitava se ceo broj sa standardnog ulaza. */
  printf("Unesite ceo broj:\n");
  scanf("%d", &x);

  /* Cuva se njegova x_originalna vrednost zbog finalnog ispisa. */
  x_original = x;
  x = abs(x);

```

```
  /* Svaki element niza brojaci predstavlja brojac za jednu od
     cifara: brojac[0] predstavlja broj nula u zapisu broja x
     brojac[1] predstavlja broj jedinica u zapisu broja x ...
     brojac[9] predstavlja broj devetki u zapisu broja x. */

  /* Brojaci se na pocetku inicijalizuju nulama. */
  for (i = 0; i < BROJ_CIFARA; 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]++;

    /* 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 < BROJ_CIFARA; i++)
  {
    if (brojaci[i])
    {
      printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n",
             x_original, i, brojaci[i]);
    }
  }

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.7

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

#define MAKS 100

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  char karakteri[MAKS];
  char c;
  int i, n;

  /* Ucitava se karakter po karakter sa standardnog ulaza sve dok
     se ne unese * ili se ne prekoraci maksimalni broj karaktera. */
```

```
  for (i = 0; i < MAKS; i++)
  {
    printf("Unesite karakter: ");
    scanf("%c", &c);
    /* Cita se znak za novi red nakon unetog karaktera. */
    getchar();

    /* Ukoliko je unet karakter * izlazi se iz petlje. */
    if (c == '*')
      break;

    /* Procitani karakter se smesta u niz. */
    karakteri[i] = c;
  }

  /* Broj unetih karaktera nakon izlaska iz petlje je i. */
  n = i;

  /* Ispis karaktera u obrnutom redosledu. */
  for (i = n-1; i >= 0; i--)
    printf("%c ", karakteri[i]);
  printf("\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.8

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

#define BROJ_CIFARA 10
#define DUZINA_ALFABETA 26

/* Pomocna funkcija za ispis elemenata niza.
   Vrednost n oznacava broj elemenata niza
   (ima vrednost 10 ili 26).
   Karakter c oznacava prvi karakter za datu kategoriju
   ('a' za mala slova, 'A' za velika i '0' za cifre). */
void ispisi(int niz[], int n, char c)
{
  int i;
  for (i = 0; i < n; i++)
  {
    if (niz[i] != 0)
      printf("Karakter %c se pojavljuje %d puta\n", c + i, niz[i]);
  }
}

/* Funkcija inicijalizuje niz postavljajuci vrednosti svih
   elemenata na nulu. */
```

```
24 void inicijalizuj(int niz[], int n)
   {
26   int i;
     for (i = 0; i < n; i++)
28     niz[i] = 0;
   }
30
   int main()
32 {
     /* Deklaracije nizova brojaca za cifre, mala i velika slova. */
34   int cifre[BROJ_CIFARA];
     int mala_slova[DUZINA_ALFABETA];
36   int velika_slova[DUZINA_ALFABETA];

38   /* Deklaracije pomocnih promenljivih. */
     int c;
40
     /* Brojaci se na pocetku inicijalizuju nulama. */
42   inicijalizuj(cifre, BROJ_CIFARA);
     inicijalizuj(mala_slova, DUZINA_ALFABETA);
44   inicijalizuj(velika_slova, DUZINA_ALFABETA);

46   /* Ucitavaju se karakteri sve do kraja ulaza. */
     while ((c = getchar()) != EOF)
48   {
       if (c >= 'A' && c <= 'Z')
50     {
         /* Ako je procitani karakter veliko slovo uvecava se broj
52        pojavljivanja odgovarajuceg velikog slova. Indeks velikog
          slova u nizu se odredjuje oduzimanjem slova A.
54        Na taj nacin slovo 'A' ce imati indeks 0, slovo 'B' indeks
          1, itd.*/
56       velika_slova[c - 'A']++;
       }
58     else if (c >= 'a' && c <= 'z')
       {
60     /* Ako je procitani karakter malo slovo uvecava se broj
          pojavljivanja odgovarajuceg malog slova. */
62       mala_slova[c - 'a']++;
       }
64     else if (c >= '0' && c <= '9')
       {
66     /* Ako je procitani karakter cifra uvecava se broj
          pojavljivanja odgovarajuce cifre. */
68       cifre[c - '0']++;
       }
70   }

72   /* Ispisuju se trazene informacije. */
     ispisi(cifre, BROJ_CIFARA, '0');
74   ispisi(mala_slova, DUZINA_ALFABETA, 'a');
     ispisi(velika_slova, DUZINA_ALFABETA, 'A');
```

```
76
   exit(EXIT_SUCCESS);
78 }
```

Rešenje 3.1.9

```
#include <stdio.h>
2 #include <stdlib.h>
#include <ctype.h>
4
#define DUZINA_ALFABETA 26
6
/* Pomocna funkcija za ispis elemenata niza. */
8 void ispisi(int niz[], int n)
{
10   int i;
   for (i = 0; i < n; i++)
12     printf("%c:%d  ", 'a' + i, niz[i]);
   putchar('\n');
14 }

16 /* Funkcija inicijalizuje niz postavljajuci vrednosti svih
    elemenata na nulu. */
18 void inicijalizuj(int niz[], int n)
{
20   int i;
   for (i = 0; i < n; i++)
22     niz[i] = 0;
}
24
int main()
26 {
   /* Deklaracija potrebnih promenljivih. */
28   int mala_slova[DUZINA_ALFABETA];
   int c;
30
   /* Inicijalizacija niza brojaca na nule. */
32   inicijalizuj(mala_slova, DUZINA_ALFABETA);

34   /* Ucitavaju se karakteri sve do kraja ulaza. */
   while ((c = getchar()) != EOF)
36   {
     /* Ako je procitani karakter slovo broj pojavljivanja slova se
38        uvecava. Kako se zanemaruje velicina slova, svako slovo se
        pretvori u malo i potom se element na odgovarajucoj poziciji
40        u nizu uveca. */
     if (isalpha(c))
42       mala_slova[tolower(c) - 'a']++;
   }
44
   /* Ispis rezultata. */
```

```
46    ispisi(mala_slova, DUZINA_ALFABETA);

48    exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.10

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

4 #define MAKS 1000

6 /* Funkcija ucitava elemente niza. */
void ucitaj(int a[], int n)
8 {
    int i;
10  printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
12    scanf("%d", &a[i]);
}
14
/* Funkcija ispisuje elemente niza. */
16 void ispisi(int a[], int n)
{
18  int i;
    for (i = 0; i < n; i++)
20    printf("%d ", a[i]);
    printf("\n");
22 }

24 /* Funkcija racuna sumu elemenata niza. */
int suma(int a[], int n)
26 {
    int i;
28  int suma_elemenata = 0;
    for (i = 0; i < n; i++)
30    suma_elemenata += a[i];
    return suma_elemenata;
32 }

34 /* Funkcija racuna prosecnu vrednost elemenata niza. */
float prosek(int a[], int n)
36 {
    int suma_elemenata = suma(a, n);
38  return (float) suma_elemenata / n;
}
40
/* Funkcija izracunava maksimum elemenata niza. */
42 int maksimum(int a[], int n)
{
44  int najveci, i;
```

```
46    najveci = a[0];
      for (i = 1; i < n; i++)
48      if (a[i] > najveci)
          najveci = a[i];
50
      return najveci;
52  }

54  /* Funkcija izracunava poziciju maksimalnog elementa u nizu. */
    int pozicija_maksimuma(int a[], int n)
56  {
      int najveci, pozicija_najveceg;
58    int i;

60    najveci = a[0];
      pozicija_najveceg = 0;
62
      for (i = 1; i < n; i++)
64    {
        if (a[i] < najveci)
66      {
          najveci = a[i];
68        pozicija_najveceg = i;
        }
70    }

72    return pozicija_najveceg;
    }
74
    int main()
76  {
      /* Deklaracija potrebnih promenljivih. */
78    int a[MAKS];
      int n;
80
      /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
82       ulaza. */
      printf("Unesite dimenziju niza:");
84    scanf("%d", &n);
      if (n <= 0 || n > MAKS)
86    {
        printf("Greska: neispravan unos.\n");
88      exit(EXIT_FAILURE);
      }
90
      /* Ucitavaju se elementi niza. */
92    ucitaj(a, n);

94    /* Ispisuju se elementi niza. */
      printf("Vreme trcanja takmicara: ");
96    ispisi(a, n);
```

```
98    /* Ispis ukupnog, prosecnog i maksimalnog vremena. */
      printf("Ukupno vreme: %d\n", suma(a, n));
100   printf("Prosecno vreme trcanja: %.2f\n", prosek(a, n));
      printf("Maksimalno vreme trcanja: %d\n", maksimum(a, n));
102
      /* Ispis indeksa pobednika. */
104   printf("Indeks pobednika: %d\n", pozicija_maksimuma(a, n));

106   exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.11

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

4 #define MAKS 100

6 /* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
8 {
    int i;
10  printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
12    scanf("%d", &a[i]);
}
14
/* Funkcija prebrojavanje vraca broj elemenata niza koji su manji
16    od poslednjeg elementa. */
int prebrojavanje(int a[], int n)
18 {
    int i;
20  int broj_manjih = 0;

22  /* Prebrojavaju se elementi niza za koje vazi da su manji od
       poslednjeg elementa (a[n-1]). Petlja ide od prvog do
24     predposlednjeg elementa. */
    for (i = 0; i < n - 1; i++)
26  {
      if (a[i] < a[n - 1])
28      broj_manjih++;
    }
30
    return broj_manjih;
32 }

34 int main()
{
36  /* Deklaracija potrebnih promenljivih. */
    int a[MAKS];
```

```
38    int n;
40    /* Ucitava se dimenzija niza i vrsi se provera
         ispravnosti ulaza. */
42    printf("Unesite dimenziju niza: ");
      scanf("%d", &n);
44    if (n <= 0 || n > MAKS)
      {
46      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
48    }
50    /* Ucitavaju se elementi niza. */
      ucitaj(a, n);
52
      /* Ispis rezultata. */
54    printf("%d\n", prebrojavanje(a, n));
56    exit(EXIT_SUCCESS);
  }
```

Rešenje 3.1.12

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    #define MAKS 100
5
    /* Funkcija ucitava elemente niza dimenzije n. */
7   void ucitaj(int a[], int n)
    {
9     int i;
      printf("Unesite elemente niza: ");
11    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
13  }

15  /* Funkcija vraca poziciju najveceg elementa niza. */
    // int pozicija_najveceg(int a[], int n)
17  // {
    //   int i, pozicija = 0;
19  //   /* Prolazi se kroz niz i ako se naidje na element cija je
    //      vrednost veca od trenutno najveceg (a[pozicija]), vrsi
21  //      se azuriranje pozicije trenutno najveceg. */
    //   for(i=1; i<n; i++)
23  //     if(a[i] > a[pozicija])
    //       pozicija = i;
25  //
    //   return pozicija;
27  // }
```

```
/* Funkcija vraca broj parnih elemenata niza koji prethode
   maksimalnom elementu niza. */
// int prebrojavanje(int a[], int n)
// {
//   int i;
//   int pozicija_maksimuma = pozicija_najveceg(a,n);
//
//   int broj_parnih = 0;
//   for (i = 0; i < pozicija_maksimuma; i++) {
//     if (a[i] % 2 == 0) {
//       broj_parnih++;
//     }
//   }
//
//   return broj_parnih;
// }

/* Zadatak se moze resiti i jednom prolaskom kroz niz.
   Ideja je da se paralelno radi pretraga maksimalnog elementa
   i prebrojavanje parnih elemenata koji mu prethode.

   Ovo moze da se uradi sa dva brojaca parnih elemenata:
   1. broj_parnih - brojac koji cuva broj parnih koji prethode
      trenutnom maksimumu.
   2. broj_parnih_izmedju - brojac koji cuva broj parnih elemenata
      koji se nalaze iza trenutnog maksimuma

   Svaki put kada se maksimum azurira, na broj parnih se doda
   broj parnih koji se prebrojao izmedju dva azuriranja,
   a broj_parnih_izmedju se vraca na nulu. */
int prebrojavanje_jednim_prolazom(int a[], int n)
{
  int i;
  int pozicija_maksimuma = 0;
  int broj_parnih = 0;
  int broj_parnih_izmedju = 0;

  for (i = 0; i < n; i++)
  {
    if(a[i] > a[pozicija_maksimuma])
    {
      pozicija_maksimuma = i;
      broj_parnih += broj_parnih_izmedju;
      broj_parnih_izmedju = 0;
    }

    if (a[i] % 2 == 0)
      broj_parnih_izmedju++;
  }

  return broj_parnih;
}
```

```
81
int main()
83 {
   /* Deklaracija potrebnih promenljivih. */
85 int a[MAKS];
   int n;
87
   /* Ucitava se dimenzija niza i vrsi se provera
89    ispravnosti ulaza. */
   printf("Unesite dimenziju niza: ");
91 scanf("%d", &n);
   if (n <= 0 || n > MAKS)
93 {
     printf("Greska: neispravan unos.\n");
95   exit(EXIT_FAILURE);
   }
97
   /* Ucitavaju se elementi niza. */
99 ucitaj(a, n);

101 /* Ispis rezultata. */
   /* I nacin:
103    printf("%d\n", prebrojavanje(a, n)); */

105 /* II nacin: jedinim prolaskom kroz niz: */
   printf("%d\n", prebrojavanje_jednim_prolazom(a, n));
107
   exit(EXIT_SUCCESS);
109 }
```

Rešenje 3.1.13

```
1 #include <stdio.h>
  #include <stdlib.h>
3
  #define MAKS 100
5
  /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n)
  {
9   int i;
    printf("Unesite elemente niza: ");
11  for (i = 0; i < n; i++)
      scanf("%d", &a[i]);
13 }

15 /* Funkcija racuna zbir elemenata niza od pozicije i do
      pozicije j. */
17 int zbir(int a[], int i, int j)
   {
19   int k;
```

```
  int rezultat = 0;

  /* Obilaze se elementi niza iz zadatog opsega. */
  for (k = i; k <= j; k++) {
    rezultat += a[k];
  }

  /* Vraca se izracunata vrednost. */
  return rezultat;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n, i, j;
  int a[MAKS];

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
  ucitaj(a, n);

  /* Ucitavaju se vrednosti granica i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite vrednosti za i i j: ");
  scanf("%d%d", &i, &j);
  if (i < 0 || j < 0 || i > n - 1 || j > n - 1 || i > j)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ispis rezultata. */
  printf("Zbir je: %d", zbir(a, i, j));

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.14

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

```c
#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(float a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%f", &a[i]);
}

/* Funckija racuna zbir prvih k pozitivnih elemenata niza. */
float zbir_pozitivnih(float a[], int n, int k)
{
  int i;
  float zbir = 0;

  /* Obilazi se element po element niza. Postupak se zavrsava
     ukoliko se dodje do kraja niza ili ukoliko se sabere k
     pozitivnih elemenata. */
  for (i = 0; i < n && k > 0; i++) {
    if (a[i] >= 0) {
      zbir += a[i];
      /* Umanjuje se brojac pozitivnih elemenata. */
      k--;
    }
  }

  return zbir;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, k;
  float a[MAKS];

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
  ucitaj(a, n);

  /* Ucitava se broj k i vrsi se provera ispravnosti ulaza. */
  printf("Unesite vrednost k: ");
```

```
56    scanf("%d", &k);
      if (k < 0 || k > n)
58    {
        printf("Greska: neispravan unos.\n");
60      exit(EXIT_FAILURE);
      }
62
      /* Ispis rezultata. */
64    printf("Zbir je: %.2f\n", zbir_pozitivnih(a, n, k));

66    exit(EXIT_SUCCESS);
  }
```

Rešenje 3.1.15

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    #define MAKS 100
5
    /* Funkcija ucitava elemente niza dimenzije n. */
7   void ucitaj(int a[], int n)
    {
9     int i;
      printf("Unesite elemente niza: ");
11    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
13  }

15  /* Funkcija ispisuje elemente niza dimenzije n. */
    void ispisi(int a[], int n)
17  {
      int i;
19    for (i = 0; i < n; i++)
        printf("%d ", a[i]);
21    printf("\n");
    }
23
    /* Funkcija razmenjuje najmanji i najveci element niza. */
25  void razmeni_min_max(int brojevi[], int n)
    {
27    int i;
      /* Najvecim, kao i najmanjim elementom niza proglasava se nulti
29       element niza. Pozicije najveceg i najmanjeg elementa se
         postavljaju na 0. */
31    int najveci = brojevi[0];
      int najmanji = brojevi[0];
33    int pozicija_najveceg = 0;
      int pozicija_najmanjeg = 0;
35
      /* U prolazu kroz niz trazi se najveci i najmanji element i
```

```
37      pamte se njihove pozicije. */
    for (i = 1; i < n; i++)
39  {
      if (brojevi[i] > najveci)
41    {
        najveci = brojevi[i];
43      pozicija_najveceg = i;
      }
45
      if (brojevi[i] < najmanji)
47    {
        najmanji = brojevi[i];
49      pozicija_najmanjeg = i;
      }
51  }

53  /* Zamenjuju se elementi na pozicijama pozicija_najmanjeg i
      pozicija_najveceg. */
    brojevi[pozicija_najveceg] = najmanji;
55  brojevi[pozicija_najmanjeg] = najveci;
}
57
int main()
59{
  /* Deklaracija potrebnih promenljivih. */
61  int brojevi[MAKS];
  int n;
63
  /* Ucitava se dimenzija niza i vrsi se provera
65    ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
67  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
69  {
    printf("Greska: neispravan unos.\n");
71  exit(EXIT_FAILURE);
  }
73
  /* Ucitavaju se elementi niza. */
75  ucitaj(brojevi, n);

77  /* Razmenjuju se najmanji i najveci element. */
  razmeni_min_max(brojevi, n);
79
  /* Ispisuje se rezultujuci niz. */
81  ispisi(brojevi, n);

83  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.16

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

#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite podatke: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija proverava da li niz sadrzi zadatu vrednost m. */
int sadrzi(int a[], int n, int m)
{
  int i;
  /* Prolazi se kroz sve elemente niza i ukoliko se naidje na
     element cija je vrednost jednaka m, kao povratna vrednost
     funkcije se vraca 1. */
  for (i = 0; i < n; i++) {
    if (a[i] == m)
      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;
}

/* Funkcija vraca indeks prvog pojavljivanja elementa m
   u nizu a ili -1 ukoliko se m ne nalazi u nizu a. */
int prvo_pojavljivanje(int a[], int n, int m)
{
  int i;
  for (i = 0; i < n; i++)
    if (a[i] == m)
      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;
}

/* Funkcija vraca indeks poslednjeg pojavljivanja elementa m
   u nizu a ili -1 ukoliko se m ne nalazi u nizu a. */
int poslednje_pojavljivanje(int a[], int n, int m)
{
```

```
52    int i;

54    /* Polazi se od kraja niza i poredi se element po element sa
         zadatim brojem m. */
56    for (i = n - 1; i >= 0; i--)
        if (a[i] == m)
58        return i;

60    /* 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
62       funkcija treba da vrati -1. */
    return -1;
64  }

66  int main()
    {
68    /* Deklaracije potrebnih promenljivih. */
      int a[MAKS];
70    int n, m, i;

72    /* Ucitava se dimenzija niza i vrsi se provera
         ispravnosti ulaza. */
74    printf("Unesite dimenziju niza: ");
      scanf("%d", &n);
76    if (n <= 0 || n > MAKS)
      {
78      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
80    }

82    /* Ucitavaju se elementi niza. */
      ucitaj(a, n);
84
      /* Ucitava se vrednost za pretragu. */
86    printf("Unesite vrednost m:");
      scanf("%d", &m);
88
      /* Ispisuju se rezultati pretrage. */
90    if (sadrzi(a, n, m))
      {
92      printf("Nadmorska visina %d se nalazi medju podacima.\n", m);

94      i = prvo_pojavljivanje(a, n, m);
        printf("Pozicija prvog pojavljivanja: %d\n", i);
96
        i = poslednje_pojavljivanje(a, n, m);
98      printf("Pozicija poslednjeg pojavljivanja: %d\n", i);
      }
100   else
        printf("Nadmorska visina %d se ne nalazi medju podacima.\n", m);
102
      exit(EXIT_SUCCESS);
```

```
104 }
```

Rešenje 3.1.17

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 600
5
   /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int a[], int n)
   {
9    int i;
     printf("Unesite elemente niza a: ");
11   for (i = 0; i < n; i++)
       scanf("%d", &a[i]);
13 }

15 /* Funkcija ispisuje elemente niza dimenzije n. */
   void ispisi(int niz[], int n)
17 {
     int i;
19   for (i = 0; i < n; i++)
       printf("%d ", niz[i]);
21   printf("\n");
   }
23
   /* Funkcija proverava da li niz a dimenzije n sadrzi zadatu
25    vrednost x. Pretraga se vrsi od prosledjene pozicije. */
   int sadrzi(int niz[], int n, int od_pozicije, int x)
27 {
     int i;
29   for (i = od_pozicije; i < n; i++)
       if (niz[i] == x)
31       return 1;

33   return 0;
   }
35
   /* Funkcija formira niz b tako sto u njega ubacuje sve elemente
37    niza a koji se u tom nizu pojavljuju bar dva puta. */
   int duplikati(int a[], int n, int b[])
39 {
     /* Promenljiva j je brojac elemenata rezultujuceg niza. */
41   int i, j = 0;

43   /* Obilazi se element po element niza a.
        Trenutni element je duplikat ukoliko se javlja jos neki put u
45      nizu a. Dovoljno je gledati da li se nalazi iza tekuceg
        elementa jer ako se nalazi ispred, onda je on vec obradjen
47      (i duplikat je detektovan).
```

```
     Element a[i] se dodaje u niz duplikata ako vazi:
     1. a[i] je duplikat
     2. a[i] se ne nalazi u nizu duplikata
     Provera sadrzi(a, n, i+1, a[i]) proverava prvi uslov.
     Provera !sadrzi(b, j, 0, a[i]) proverava drugi uslov. */
  for (i = 0; i < n; i++)
  {
    if (sadrzi(a, n, i+1, a[i]) && !sadrzi(b, j, 0, a[i]))
    {
      b[j] = a[i];
      j++;
    }
  }

  /* Povratna vrednost funkcije je duzina niza b. */
  return j;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a[MAKS], b[MAKS];
  int n_a, n_b;

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite broj n: ");
  scanf("%d", &n_a);
  if (n_a <= 0 || n_a > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se podaci o slicicama. */
  ucitaj(a, n_a);

  /* Niz b se popunjava duplikatima iz a. */
  n_b = duplikati(a, n_a, b);

  /* Ispis rezultata. */
  printf("Elementi niza b: ");
  ispisi(b, n_b);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.18

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

```c
#include <ctype.h>

#define MAKS 200

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(char niz[], int n)
{
  int i;
  /* Ucitava se niz od n karaktera. */
  printf("Unesite elemete niza: ");
  for (i = 0; i < n; i++)
    scanf("%c", &niz[i]);
}

/* Funkcija proverava da li je niz karaktera palindrom. */
int je_palindrom(char niz[], int n)
{
  int i;
  /* U petlji se porede niz[0] i niz[n-1], zatim niz[1] i niz[n-2]
     itd. Ako se naidje na par koji se razlikuje - niz nije
     palindrom. */
  for (i = 0; i < n / 2; i++)
  {
    if (tolower(niz[i]) != tolower(niz[n - 1 - i]))
      return 0;
  }

  /* Izvrsila se cela petlja => niz je palindrom. */
  return 1;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  char niz[MAKS];
  int n;

  /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
     ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Preskace se novi red nakon unosa dimenzije. Ovo se radi
     jer sledi ucitavanje karaktera i bez ove linije, prvi
     karakter koji bi se upisao u niz bi bio novi red. */
  getchar();

```

```
  /* Ucitavaju se elementi niza. */
  ucitaj(niz, n);

  /* Ispis rezultata. */
  if(je_palindrom(niz, n))
    printf("Niz jeste palindrom.\n");
  else
    printf("Niz nije palindrom.\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.19

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

#define MAKS 300

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija vraca 1 ako je niz uredjen neopadajuce,
   a nulu inace. */
int uredjen_neopadajuce(int niz[], int n)
{
  int i;
  for (i = 0; i < n - 1; i++)
  {
    if (niz[i] > niz[i + 1])
      return 0;
  }

  return 1;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, niz[MAKS];

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
```

```
38    if (n <= 0 || n > MAKS)
      {
40      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
42    }

44    /* Ucitavaju se elementi niza. */
      ucitaj(niz, n);
46
      /* Ispis rezultata. */
48    if(uredjen_neopadajuce(niz, n))
        printf("Niz jeste uredjen neopadajuce.\n");
50    else
        printf("Niz nije uredjen neopadajuce.\n");
52
      exit(EXIT_SUCCESS);
54  }
```

Rešenje 3.1.20

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

4 /* Maksimalan broj dana u mesecu je 31, ali dani pocinju od 1, pa
     je potrebno odvojiti 32 mesta u nizu jer se nulti ne koristi. */
6 #define MAKS_DANA 32

8 /* Funkcija ucitava elemente niza dimenzije n. */
  void ucitaj(int a[], int n)
10 {
    int i;
12  printf("Unesite broj prodatih artikala: ");
    for (i = 0; i < n; i++)
14  {
      scanf("%d", &a[i]);
16    if(a[i] < 0)
      {
18      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
20    }
    }
22 }

24 /* Funkcija racuna duzinu najduzeg neopadajuceg podniza niza a. */
  int najduzi_neopadajuci(int a[], int n)
26 {
    int i;
28  /* Na pocetku i duzina trenutne serije i duzina maksimalne serije
       se inicijalizuju na 1. */
30  int duzina_trenutne_serije = 1;
    int duzina_najduze_serije = 1;
```

```
32
  for (i = 1; i < n; i++) {
34   /* Proverava se da li uzastopni elementi ispunjavaju
        neopadajuci uslov. Ako je to slucaj uvecava se duzina
36      serije, a ako nije, duzina trenutne serije se vraca na 1,
        kako bi se ispravno racunala duzina sledece serije. */
38   if (a[i] >= a[i - 1])
       duzina_trenutne_serije++;
40   else
       duzina_trenutne_serije = 1;
42
     /* Ukoliko je trenutna duzina serije veca od duzine do sada
44      najduze serije, parametar za duzinu najduze serije se
        postavlja na novu, vecu vrednost. */
46   if (duzina_trenutne_serije > duzina_najduze_serije)
       duzina_najduze_serije = duzina_trenutne_serije;
48 }

50 return duzina_najduze_serije;
}
52
int main()
54 {
  /* Deklaracija potrebnih promenljivih. */
56 int a[MAKS_DANA], n;

58 /* Ucitava se dimenzija niza i vrsi se provera
      ispravnosti ulaza. */
60 printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
62 if (n <= 0 || n > MAKS_DANA)
  {
64   printf("Greska: neispravan unos.\n");
     exit(EXIT_FAILURE);
66 }

68 /* Ucitavaju se elementi niza. */
  ucitaj(a, n);
70
  /* Ispis rezultata. */
72 printf("Duzina najduzeg neopadajuceg prodavanja je %d.\n",
         najduzi_neopadajuci(a, n));
74
  exit(EXIT_SUCCESS);
76 }
```

Rešenje 3.1.21

```
1 #include <stdio.h>
  #include <stdlib.h>
3
```

```c
#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija vraca duzinu najduze serije jednakih elemenata niza. */
int najduza_serija(int a[], int n)
{
  int i;
  /* Na pocetku i duzina trenutne serije i duzina maksimalne serije
     se inicijalizuju na 1. */
  int trenutna_serija = 1;
  int najduza_serija = 1;

  for (i = 1; i < n; i++) {
    /* Proverava se da li su uzastopni elementi jednaki. Ako je to
       slucaj uvecava se duzina serije. Ako uzastopni elementi nisu
       jednaki serija je prekinuta i paramtar za duzinu serije se
       postavlja ponovo na 1 da bi mogla da se racuna duzina
       sledece serije. */
    if (a[i] == a[i - 1])
      trenutna_serija++;
    else
      trenutna_serija = 1;

    /* Ukoliko je trenutna duzina serije veca od duzine do sada
       najduze serije, parametar za duzinu najduze serije se
       postavlja na novu, vecu vrednost. */
    if (trenutna_serija > najduza_serija)
      najduza_serija = trenutna_serija;
  }

  return najduza_serija;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, a[MAKS];

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
```

```
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
  ucitaj(a, n);

  /* Ispis rezultata. */
  printf("Duzina najduze serije je %d.\n", najduza_serija(a, n));

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.22

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

#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int niz[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &niz[i]);
}

/* Resenje pod a. */
int podniz_uzastopnih(int a[], int n, int b[], int m)
{
  int i, j;

  /* Prolaze se elementi prvog niza. Svaki element prvog niza moze
     biti pocetak podniza, odnosno pocetak drugog niza. */
  for (i = 0; i + m - 1 < n; i++)
  {
    /* Prolaze se elementi drugog niza. Za svaki element niza b
       proverava se da li je jednak odgovarajucem elementu niza a.
       Za niz a razmatra se da li podniz pocinje od pozicije i. Tako
       0-ti element niza b je na poziciji i, 1. element je na
       poziciji i+1, 2. na poziciji i+2, ..., j-ti na poziciji
       i+j. Ako uslov nije ispunjen, petlja se prekida i proverava
       se da li na sledecoj poziciji u nizu a pocinje podniz. */
    for (j = 0; j < m; j++)
      if (a[i + j] != b[j])
        break;
    /* Ako petlja nije prekinuta nakon ispitivanja, brojac za niz b
       je jedanak dimenziji niza b, odnosno svi elementi niza b se
       uzastopno nalaze u nizu a. */
```

```
    if (j == m)
      return 1;
  }

  /* Ukoliko niz b jeste uzastopni podniz uslov u petlji ce u nekom
     trenutku biti ispunjen i iz petlje i funkcije ce se izaci sa
     return naredbom. Ipak, ako se to nije desilo i dalje se
     izvrsava funkcija, onda niz b nije uzastopni podniz. */
  return 0;
}

/* Resenje pod b. */
int podniz(int a[], int n, int b[], int m)
{
  int i, j;

  /* Petljom se prolaze elementi niza a. */
  for (i = 0, j = 0; i < n && j < m; i++)
  {
    /* Svaki put kada se naidje na element niza b, brojac za niz b
       se uvecava i proverava se da li se sledeci element niza b
       nalazi u nizu a. */
    if (a[i] == b[j])
      j++;
  }

  /* Ukoliko se pronadju svi elementi niza b u nizu a, onda je
     brojac za niz b jednak dimenziji niza b. U tom slucaju se
     vraca vrednost 1, odnosno da niz jeste podniz. */
  return j == m;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, a[MAKS];
  int m, b[MAKS];

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi prvog niza. */
  ucitaj(a, n);

  /* Ucitava se dimenzija niza i vrsi se provera
```

```
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &m);
  if (m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi drugog niza. */
  ucitaj(b, m);

  /* a) */
  if (podniz_uzastopnih(a, n, b, m))
    printf("Elementi drugog niza cine uzastopni podniz "
           "prvog niza.\n");
  else
    printf("Elementi drugog niza ne cine uzastopni podniz "
         "prvog niza.\n");

  /* b) */
  if (podniz(a, n, b, m))
    printf("Elementi drugog niza cine podniz prvog niza.\n");
  else
    printf("Elementi drugog niza ne cine podniz prvog niza.\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.23

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

#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int niz[], int n)
{
  int i;

  /* Ucitavaju se elementi niza. */
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
  {
    scanf("%d", &niz[i]);

    /* Niz moze sadrzati elemente koji nisu u opsegu od 1 do n.
       U tom slucaju taj niz nije permutacija. */
    if (niz[i] <= 0 || niz[i] > n)
    {
```

```
21        printf("Uneti niz nije permutacija.\n");
          exit(EXIT_SUCCESS);
23      }
    }
25  }

27  /* Funkcija prebrojava koliko puta se pojavljuje svaki element
       niza a. */
29  void brojanje(int a[], int b[], int n)
    {
31    int i;

33    /* Niz b se inicijalizuje nulama jer se za svaki element postavi
         da se poljavuljuje 0 puta u nizu a. */
35    for (i = 1; i <= n; i++)
        b[i] = 0;
37
      /* Peljom se prolazi kroz niz a i za svaki element a[i] uvecava
39       se broj njegovog pojavljivanja u nizu b. Na primer, ako je
         a[3] = 7, onda treba uvecati broj pojavljivanja broja 7, a to
41       je b[7]++, sto se krace moze zapisati kao b[a[3]]++.
         Pretpostavlja se da je niz a dobro zadat, odnosno da su sve
43       njegove vrednosti u intervalu od 1 do n. */
      for (i = 0; i < n; i++)
45      b[a[i]]++;
    }
47
    /* Funkcija proverava da li je niz a permutacija. */
49  int permutacija(int a[], int n)
    {
51     /* Niz b moze imati index MAKS (jer niz b se posmatra od 1 do
         MAKS), pa zato njegova dimenzija mora biti za jedan veca. */
53    int b[MAKS + 1];
      int i;
55
      /* Racuna se broj pojavljivanja svakog broja niza a. */
57    brojanje(a, b, n);

59    /* Ukoliko se svaki element niza a javlja tacno jednom u nizu a,
         onda niz a jeste permutacija. Ovo svojstvo se proverava
61       koriscenjem dobijenog niza b. */
      for (i = 1; i <= n; i++)
63    {
        if (b[i] != 1)
65        return 0;
      }
67
      return 1;
69  }

71  int main()
    {
```

```
  /* Deklaracija potrebnih promenljivih. */
  int a[MAKS], n;

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza a. */
  ucitaj(a, n);

  /* Ispis rezultta. */
  if(permutacija(a, n))
    printf("Uneti niz je permutacija.\n");
  else
    printf("Uneti niz nije permutacija.\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.24

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

#define BROJ_CIFARA 10

/* Funkcija inicijalizuje niz postavljajuci vrednosti svih
   elemenata na nulu. */
void inicijalizuj(int niz[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    niz[i] = 0;
}

/* Funkcija izdvaja cifru po cifru broja i uvecava odgovarajuci
   element niza koji odgovara brojacu za tu cifru. Na primer,
   za broj=1123, po zavrsetku ove funkcije niz[1] ce imati vrednost
   2 jer se cifra 1 pojavljuje 2 puta, niz[2] i niz[3] ce imati
   vrednost 1, a svi ostali elementi niza ce imati vrednost 0. */
void analiza_cifara(int broj, int niz[])
{
  int c;

  /* Inicijalizacija svih brojaca na nule. */
```

```
25    inicijalizuj(niz, BROJ_CIFARA);

27    /* Uvecavanje odgovarajucih brojaca. */
      do {
29      c = broj % 10;
        niz[c]++;
31      broj /= 10;
      }
33    while (broj);
}
35
int main()
37{
      /* Niz cifrex predstavlja brojace za cifre broja x. Niz cifrey
39       predstavlja brojace za cifre broja y. */
      int cifrex[BROJ_CIFARA], cifrey[BROJ_CIFARA];
41    int x, y, i, indikator;

43    /* Ucitavaju se brojevi x i y. */
      printf("Unesite dva broja: ");
45    scanf("%d%d", &x, &y);

47    /* Za slucaj da su unete vrednosti negativne, posmatra se njihova
         apsolutna vrednost. Ovo je opravdano iz razloga sto se brojevi
49       x i -x zapisuju istim ciframa. */
      x = abs(x);
51    y = abs(y);

53    /* Popunjavaju se nizovi sa brojacima cifara. */
      analiza_cifara(x, cifrex);
55    analiza_cifara(y, cifrey);

57    /* Promenljiva indikator sluzi za pracenje da li su oba broja
         sastavljena od istih cifara. */
59    indikator = 1;

61    for (i = 0; i < BROJ_CIFARA; i++) {
        /* Ako se broj pojavljivanja cifre i u zapisu broja x razlikuje
63         od broja pojavljivanja cifre i u zapisu broja y, brojevi se
           ne zapisuju istim ciframa. Zato se vrednost indikatora moze
65         postaviti na 0 i prekinuti dalje uporedjivanje broja
           pojavljivanja. */
67      if (cifrey[i] != cifrex[i]) {
          indikator = 0;
69        break;
        }
71    }

73    /* Ako je vrednost promenljive indikator ostala 1, to znaci da u
         petlji nije pronadjena cifra koja se ne pojavljuje isti broj
75       puta u zapisima brojeva x i y. Zato se moze zakljuciti da se
         brojevi zapisuju istim ciframa. */
```

```
77    if (indikator)
        printf("Brojevi se zapisuju istim ciframa.\n");
79    else
        printf("Brojevi se ne zapisuju istim ciframa.\n");
81
83    exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.25

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

#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija ispisuje elemente niza dimenzije n. */
void ispisi(int a[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", a[i]);
  printf("\n");
}

/* Funkcija obrce elemente niza. */
void obrni(int a[], int n)
{
  int t, i, j;

  /* 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]. 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. */
  for (i = 0, j = n - 1; i < j; i++, j--)
  {
    t = a[i];
    a[i] = a[j];
    a[j] = t;
  }
}
```

```
42 /* Funkcija rotira niz ciklicno za jedno mesto u levo. */
   void rotiraj1(int a[], int n)
44 {
     int i, prvi;
46
     /* Izdvaja se prvi element niza. */
48   prvi = a[0];

50   /* Pomeraju se preostali elementi niza za jedno mesto u levo. */
     for (i = 0; i < n - 1; i++)
52     a[i] = a[i + 1];

54   /* Poslednjem elementu se dodeljuje sacuvana vrednost prvog
        elementa. */
56   a[n - 1] = prvi;
   }
58
   /* Funkcija rotira niz ciklicno za k mesta u levo. */
60 void rotirajk(int a[], int n, int k)
   {
62   int i;

64   /* Odredjuje se vrednost broja k koja je u opsegu od 0 do n-1
        kako bi se izbegla suvisna prvieranja. */
66   k = k % n;

68   /* Niz se rotira za jednu poziciju ulevo k puta. */
     for (i = 0; i < k; i++)
70     rotiraj1(a, n);
   }
72
   int main()
74 {
     /* Deklaracija potrebnih promenljivih. */
76   int a[MAKS];
     int n, k;
78
     /* Ucitava se dimenzija niza i vrsi se provera
80      ispravnosti ulaza. */
     printf("Unesite dimenziju niza: ");
82   scanf("%d", &n);
     if (n <= 0 || n > MAKS)
84   {
       printf("Greska: neispravan unos.\n");
86     exit(EXIT_FAILURE);
     }
88
     /* Ucitavaju se elementi niza. */
90   ucitaj(a, n);

92   /* Obrtanje niza. */
```

```
  printf("Elementi niza nakon obrtanja:\n");
  obrni(a, n);
  ispisi(a, n);

  /* Rotiranje za jedno mesto u levo. */
  printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
  rotiraj1(a, n);
  ispisi(a, n);

  /* Rotiranje za k mesta u levo. */
  printf("Unesite jedan pozitivan ceo broj:");
  scanf("%d", &k);
  if (k <= 0)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }
  rotirajk(a, n, k);
  printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n", k);
  ispisi(a, n);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.26

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

#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int niz[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    scanf("%d", &niz[i]);
}

/* Funkcija ispisuje elemente niza. */
void ispisi(int niz[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", niz[i]);
  printf("\n");
}

/* Funkcija formira niz c ukrstanjem nizova a i b. */
void ukrsti(int a[], int b[], int n, int c[])
{
```

```
  int i, j;
  /* 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];
  }
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a[MAKS], b[MAKS], c[2 * MAKS];
  int n;

  /* Ucitava se dimenzija nizova i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju nizova: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi nizova. */
  printf("Unesite elemente niza a: ");
  ucitaj(a, n);
  printf("Unesite elemente niza b: ");
  ucitaj(b, n);

  /* Formira se niz c. */
  ukrsti(a, b, n, c);

  /* Ispisuju se elementi rezultujuceg niza. */
  printf("Rezultujuci niz:\n");
  ispisi(c, 2*n);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.27

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

```
3
#define MAKS 100
5
/* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int niz[], int n)
{
9   int i;
  for (i = 0; i < n; i++)
11    scanf("%d", &niz[i]);
}
13
/* Funkcija ispisuje elemente niza dimenzije n. */
15 void ispisi(int niz[], int n)
{
17   int i;
  for (i = 0; i < n; i++)
19    printf("%d ", niz[i]);
  printf("\n");
21 }

23 /* Funkcija formira niz c nadovezivanjem nizova a i b. */
void spoji(int a[], int b[], int n, int c[])
25 {
  int i;
27
  /* Niz c ima 2*n elemenata: prvih n elemenata su elementi niza b,
29     a narednih n elemenata elementi niza a. Elementi niza b se
     nalaze na pozicijama 0,1,2,...n-1, a elementi niza a na
31     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
33     poziciju n+i element a[i] niza a. */
  for (i = 0; i < n; i++) {
35    c[i] = b[i];
    c[n + i] = a[i];
37  }
}
39
int main()
41 {
  /* Deklaracija potrebnih promenljivih. */
43  int a[MAKS], b[MAKS], c[2 * MAKS];
  int n;
45
  /* Ucitava se dimenzija nizova i vrsi se provera
47     ispravnosti ulaza. */
  printf("Unesite dimenziju nizova: ");
49  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
51  {
    printf("Greska: neispravan unos.\n");
53    exit(EXIT_FAILURE);
  }
```

```
55
   /* Ucitavaju se elementi nizova. */
57 printf("Unesite elemente niza a: ");
   ucitaj(a, n);
59 printf("Unesite elemente niza b: ");
   ucitaj(b, n);
61
   /* Formira se niz c. */
63 spoji(a, b, n, c);

65 /* Ispisuju se elementi niza c. */
   ispisi(c, 2*n);
67
   exit(EXIT_SUCCESS);
69 }
```

Rešenje 3.1.28

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 100
5
   /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int niz[], int n)
   {
9    int i;
     printf("Unesite elemente sortiranog niza:\n");
11   for (i = 0; i < n; i++)
       scanf("%d", &niz[i]);
13 }

15 /* Funkcija za ispis niza. */
   void ispisi(int niz[], int n)
17 {
     int i;
19   for (i = 0; i < n; i++)
       printf("%d ", niz[i]);
21   printf("\n");
   }
23
   int main()
25 {
     /* Deklaracija potrebnih promenljivih. */
27   int a[MAKS], b[MAKS], c[2 * MAKS];
     int n;
29   /* Brojac u petlji za elemente niza a. */
     int i = 0;
31   /* Brojac u petlji za elemente niza b. */
     int j = 0;
33   /* Brojac u petlji za elemente niza c. */
```

```
  int k = 0;

  /* Ucitava se dimenzija nizova i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju nizova: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi nizova. */
  ucitaj(a, n);
  ucitaj(b, n);

  /* Vrsi se spajanje nizova. */
  while (i < n && j < n)
  {
    /* Porede se elementi nizova a i b i u niz c upisuje se samo
       onaj koji je manji. Ako je upisan element iz niza a, onda se
       vrsi i uvecavanje brojaca i (prelazak na sledeci element
       niza a), a ako je upisan element iz niza b, onda se vrsi
       uvecavanje brojaca j (prelazak na sledeci element niza b). */
    if (a[i] < b[j])
    {
      c[k] = a[i];
      i++;
    }
    else
    {
      c[k] = b[j];
      j++;
    }

    /* U nizu c na poziciju k je upisan ili a[i] ili b[j]. Brojac k
       se uvecava. */
    k++;
  }

  /* Ukoliko je ostalo elemenata u nizu a, upisuju se u niz c. */
  while (i < n)
  {
    c[k] = a[i];
    k++;
    i++;
  }

  /* Ukoliko je ostalo elemenata u nizu b, upisuju se u niz c. */
  while (j < n)
  {
    c[k] = b[j];
```

```
    k++;
87    j++;
  }
89
  /* Ispis elemenata niza c cija dimenzija je zbir dimenzija nizova
91    a i b. */
  ispisi(c, 2 * n);
93
  exit(EXIT_SUCCESS);
95}
```

Rešenje 3.1.29

```
1#include <stdio.h>
#include <stdlib.h>
3
#define MAKS 100
5
/* Funkcija ucitava elemente niza dimenzije n. */
7void ucitaj(int a[], int n)
{
9  int i;
  printf("Unesite elemente niza: ");
11  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
13}

15/* Funkcija ispisuje elemente niza dimenzije n. */
void ispisi(int niz[], int n)
17{
  int i;
19  for (i = 0; i < n; i++)
    printf("%d ", niz[i]);
21  printf("\n");
}
23
/* Funkcija razmenjuje elemente niza tako da se na pocetku niza
25   nalaze svi parni elementi niza, nakon kojih slede svi neparni
   elementi niza. */
27void promeni_redosled(int niz[], int n)
{
29  int i = 0, j = n - 1;
  int pom;
31
  /* Krece se sa pocetka niza (po brojacu i) i sa kraja niza (po
33     brojacu j) i svaki put kada se naidje na elemente koji po
     parnosti ne odgovaraju delu niza u kome treba da budu, ti
35     elementi se zamene. */
  while (i < j && i < n && j >= 0)
37  {
    if (niz[i] % 2 != 0 && niz[j] % 2 == 0)
```

```
39      {
          pom = niz[i];
41        niz[i] = niz[j];
          niz[j] = pom;
43      }

45      /* Ukoliko je element na poziciji i paran, prelazi se na
           sledeci element niza, brojac i se uvecava. */
47      if (niz[i] % 2 == 0)
          i++;
49
        /* Ukoliko je element na poziciji j neparan, prelazi se na
51         sledeci element niza, brojac j se smanjuje. */
        if (niz[j] % 2 != 0)
53        j--;
    }
55  }

57  int main()
    {
59    /* Deklaracija potrebnih promenljivih. */
      int niz[MAKS];
61    int n;

63    /* Ucitava se dimenzija niza i vrsi se provera
         ispravnosti ulaza. */
65    printf("Unesite dimenziju niza: ");
      scanf("%d", &n);
67    if (n <= 0 || n > MAKS)
      {
69      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
71    }

73    /* Ucitavaju se elementi niza. */
      ucitaj(niz, n);
75
      /* Menja se niz na trazeni nacin. */
77    promeni_redosled(niz, n);

79    /* Ispis rezultata. */
      printf("Rezultujuci niz:\n");
81    ispisi(niz, n);

83    exit(EXIT_SUCCESS);
    }
```

Rešenje 3.1.30

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

```c
#include <math.h>

#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija ispisuje elemente niza dimenzije n. */
void ispisi(int a[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", a[i]);
  printf("\n");
}

/* Funkcija vraca 1 ako je broj prost, a 0 u suprotnom.  */
int prost(int x)
{
  int i;

  /* Brojevi 2 i 3 su prosti. */
  if (x == 2 || x == 3)
    return 1;

  /* Parni brojevi nisu prosti. */
  if (x % 2 == 0)
    return 0;

  /* Ako se naidje na broj koji deli broj x, onda broj x nije
     prost. Provera se vrsi za sve neparne brojeve izmedju 3 i
     sqrt(x), jer kada bi x imao parnog delioca, onda bi i broj 2
     delio x, a taj uslov je vec proveren. */
  int koren_x = sqrt(x);
  for (i = 3; i <= koren_x; i += 2)
    if (x % i == 0)
      return 0;

  /* Ako nijedan od prethodnih uslova nije bio ispunjen, to znaci
     da nijedan broj ne deli x, pa je on prost. */
  return 1;
}

/* Funkcija od niza a formira niz b koji sadrzi sve elemente
   niza a koji nisu prosti brojevi. Povratna vrednost funkcije
   je broj elemenata niza b. */
```

```
int obrisi_proste(int a[], int n, int b[])
{
  int i, j;

  /* Kada se u nizu a naidje na prost element, on se upisuje u niz
    b i uvecava se brojac za niz b. */
  for (i = 0, j = 0; i < n; i++)
  {
    if (prost(a[i]) == 0)
    {
      b[j] = a[i];
      j++;
    }
  }

  return j;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS], b[MAKS];
  int n_a, n_b;

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n_a);
  if (n_a <= 0 || n_a > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
  ucitaj(a, n_a);

  /* Formira se niz b brisanjem prostih iz niza a. */
  n_b = obrisi_proste(a, n_a, b);

  /* Ispisuju se elementi niza b. */
  ispisi(b, n_b);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.31

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

```

```c
#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija ispisuje elemente niza dimenzije n. */
void ispisi(int niz[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", niz[i]);
  printf("\n");
}

/* Funkcija brise sve neparne elemente niza. */
int obrisi_neparne(int a[], int n)
{
  int i, j;
  /* Promenljiva j predstavlja brojac prve slobodne pozicije na koju
     se moze upisati element niza koji treba da ostane u nizu. 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. */
  for (i = 0, j = 0; i < n; i++)
  {
    /* Ako je tekuci element niza a paran. */
    if (a[i] % 2 == 0)
    {
      /* Premesta se na poziciju j. */
      a[j] = a[i];

      /* Vrednost brojaca j se priprema za narednu iteraciju. */
      j++;
    }
    /* Ako je tekuci element niza a neparan, sa njim nista ne treba
       raditi. */
  }

  /* Rezultujuci niz ima j elemenata. */
  return j;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a[MAKS];
```

```
  int n;

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
  ucitaj(a, n);

  /* Brisu se neparni elementi. */
  n = obrisi_neparne(a, n);

  /* Ispisuju se elementi modifikovanog niza a. */
  ispisi(a, n);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.32

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

#define MAKS 100

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija ispisuje elemente niza dimenzije n. */
void ispisi(int a[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", a[i]);
  printf("\n");
}

/* Funkcija brise sve elemente niza koji nisu deljivi
   svojom poslednjom cifrom. Povratna vrednost funkcije je
```

```
26    broj elemenata rezultujuceg niza. */
int brisanje(int a[], int n)
28 {
  int i, j, poslednja_cifra;
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 nula ili je element deljiv svojom
38       poslednjom cifrom, taj element se zadrzava i smesta na
       poziciju j. */
40    if (poslednja_cifra == 0 || a[i] % poslednja_cifra == 0)
    {
42      a[j] = a[i];
      j++;
44    }
  }
46
  return j;
48 }

50 int main()
{
52  /* Deklaracija potrebnih promenljivih. */
  int a[MAKS];
54  int n;

56  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
58  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
60  if (n <= 0 || n > MAKS)
  {
62    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
64  }

66  /* Ucitavaju se elementi niza. */
  ucitaj(a, n);
68
  /* Brisu se svi elementi koji nisu deljivi svojom poslednjom
70     cifrom. */
  n = brisanje(a, n);
72
  /* Ispisuje se rezultujuci niz. */
74  ispisi(a, n);

76  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.33

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

#define MAKS 700

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
  int i;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
    scanf("%d", &a[i]);
}

/* Funkcija ispisuje elemente niza dimenzije n. */
void ispisi(int a[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", a[i]);
  printf("\n");
}

/* Funkcija pomera za jedno mesto u levo elemente niza a pocevsi od
   pozicije j. Element na poziciji j se brise i na njegovo mesto se
   upisuje element na poziciji j+1, a u skladu sa tim svi ostali
   elementi posle njega u nizu se pomeraju. */
void pomeri_za_jedno_mesto(int a[], int n, int j)
{
  int i;
  for (i = j; i < n; i++)
    a[i] = a[i + 1];
}

/* Funkcija brise sve elemente niza koji nisu deljivi svojim
   indeksom. Povratna vrednost funkcije je broj elemenata
   rezultujuceg niza. */
int brisanje(int niz[], int n)
{
  int i;

  /* Potrebno je krenuti od poslednjeg elementa niza i petljom ici
     ka pocetku niza (element na poziciji 0 se ne razmatra).
     Proverava se da li je element potrebno obrisati i ako jeste
     vrsi se pomeranje elemenata niza za jedno mesto u levo.
     Prednost ovog resenja u odnosu na resenje kada se krene od
     pocetka niza je u tome sto element koji se ispituje sigurno
     nije promenio svoju poziciju usled pomeranja zbog brisanja.
     Problem se moze resiti i koriscenjem pomocnog niza (uraditi za
     vezbu). To resenje je efikasnije, ali trosi vise resursa. */
```

```
  for (i = n - 1; i > 0; i--)
  {
    if (niz[i] % i != 0)
    {
      pomeri_za_jedno_mesto(niz, n, i);
      /* Nakon brisanja elementa, smanjuje se i dimenzija niza. */
      n--;
    }
  }

  return n;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, niz[MAKS];

  /* Ucitava se dimenzija niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
  ucitaj(niz, n);

  /* Iz niza se brisu odgovarajuci elementi. */
  n = brisanje(niz, n);

  /* Ispis novog niza. */
  printf("Novi niz:\n");
  ispisi(niz, n);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.34

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

#define MAKS 500

/* Funkcija vraca 1 ukoliko broj x postoji u nizu, 0 inace. */
int postoji(int niz[], int n, int x)
{
```

```
  int i;
  for (i = 0; i < n; i++)
    if (niz[i] == x)
      return 1;

  return 0;
}

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int niz[], int n)
{
  int i, element;
  printf("Unesite elemente niza: ");
  for (i = 0; i < n; i++)
  {
    scanf("%d", &element);
    if(postoji(niz, i, element))
    {
      printf("Greska: skup ne moze imati duplikate.\n");
      exit(EXIT_FAILURE);
    }
    niz[i] = element;
  }
}

/* Funkcija ispisuje elemente niza dimenzije n. */
void ispisi(int niz[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", niz[i]);
  printf("\n");
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a[MAKS], b[MAKS], unija[2 * MAKS], presek[MAKS],
      razlika[MAKS];
  int i, n_a, n_b, n_unija, n_presek, n_razlika;

  /* Ucitava se dimenzija prvog niza i vrsi se provera
     ispravnosti ulaza. */
  printf("Unesite dimenziju niza: ");
  scanf("%d", &n_a);
  if (n_a <= 0 || n_a > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavaju se elementi niza. */
```

```
61  ucitaj(a, n_a);

63  /* Ucitava se dimenzija drugog niza i vrsi se provera
       ispravnosti ulaza. */
65  printf("Unesite dimenziju niza: ");
    scanf("%d", &n_b);
67  if (n_b <= 0 || n_b > MAKS)
    {
69    printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
71  }

73  /* Ucitavaju se elementi niza. */
    ucitaj(b, n_b);
75
    /* Brojaci elemenata u nizovima unija, presek i razlika. */
77  n_unija = 0;
    n_presek = 0;
79  n_razlika = 0;

81  for (i = 0; i < n_a; i++)
    {
83    /* Svi elementi niza a se dodaju u uniju. */
        unija[n_unija] = a[i];
85      n_unija++;

87    /* Ukoliko se element a[i] nalazi u nizu b i ne postoji u nizu
         presek, dodaje se presek i povecava se brojac elemenata u
89       nizu presek. */
      if (postoji(b, n_b, a[i]) == 1
91        && postoji(presek, n_presek, a[i]) == 0) {
        presek[n_presek] = a[i];
93      n_presek++;
      }
95
      /* Ukoliko element a[i] ne postoji u nizu b i ne postoji u nizu
97       razlika, dodaje se u razliku i povecava se brojac elemenata
         u nizu razlika. */
99    if (postoji(b, n_b, a[i]) == 0
          && postoji(razlika, n_razlika, a[i]) == 0) {
101     razlika[n_razlika] = a[i];
        n_razlika++;
103   }
    }
105
    /* Elemente niza b koji nisu uneti u uniju dodaju se u uniju. */
107 for (i = 0; i < n_b; i++)
    {
109   if (postoji(unija, n_unija, b[i]) == 0)
      {
111     unija[n_unija] = b[i];
        n_unija++;
```

```
    }
  }
  /* Ispis rezultata. */
  printf("Unija: ");
  ispisi(unija, n_unija);

  printf("Presek: ");
  ispisi(presek, n_presek);

  printf("Razlika: ");
  ispisi(razlika, n_razlika);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.1.35

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

#define MAKS 2000

/* Funkcija ispisuje elemente niza dimenzije n. */
void ispis(int niz[], int n)
{
  int i;
  for (i = 0; i < n; i++)
    printf("%d ", niz[i]);
  printf("\n");
}

/* Funkcija ubacuje element x na kraj niza. Vraca novu dimenziju
   niza. */
int ubaci_na_kraj(int niz[], int n, int x)
{
  if(n == MAKS)
  {
    printf("Greska: prekoracen je maksimalan broj elemenata niza.");
    exit(EXIT_FAILURE);
  }

  niz[n] = x;
  return n + 1;
}

/* Funkcija ubacuje element x na pocetak niza. Vraca novu dimenziju
   niza. */
int ubaci_na_pocetak(int niz[], int n, int x)
{
  if(n == MAKS)
  {
```

```
35      printf("Greska: prekoracen je maksimalan broj elemenata niza.");
        exit(EXIT_FAILURE);
37    }

39    int i;
      /* Prvo se svi elementi niza pomere za jednu poziciju u desno da
41       bi se oslobodio prostor za prvi element niza. Poslednji
         element niza se pomera sa pozicije (n-1) na poziciju (n).
43       Slicno se pomeraju i ostali elementi. */
      for (i = n; i > 0; i--)
45      niz[i] = niz[i - 1];

47    /* Na prvu poziciju se upisuje novi element. Bitan je redosled
         naredbi: ako bi prvo bio upisan novi element, a tek onda
49       izvrseno pomeranje, element na poziciji niz[0] bi bio obrisan
         i ne bi mogao biti upisan na poziciju niz[1]. */
51    niz[0] = x;

53    return n + 1;
  }
55
  /* Funkcija ubacuje element x na neku poziciju u nizu. Vraca
57     novu dimenziju niza. */
  int ubaci_na_poziciju(int niz[], int n, int x, int pozicija)
59{
    if(n == MAKS)
61  {
      printf("Greska: prekoracen je maksimalan broj elemenata niza.");
63    exit(EXIT_FAILURE);
    }
65
    int i;
67  /* Prvo se svi elementi niza od pozicije do kraja pomere za jedno
       mesto u desno da bi se oslobodio prostor za novi element niza.
69   */
    for (i = n; i > pozicija; i--)
71    niz[i] = niz[i - 1];

73  /* Na poziciju se upisuje novi element. */
    niz[pozicija] = x;
75
    return n + 1;
77}

79/* Funkcija brise prvi element niza. Vraca novu dimenziju
     niza. */
81int brisi_prvog(int niz[], int n)
{
83  if(n == 0)
    {
85    printf("Greska: nije moguce brisanje iz praznog niza.\n");
      exit(EXIT_FAILURE);
```

```
87    }

89    int i;
      /* Svi elementi niza pomeraju se za jedno mesto u levo. */
91    for (i = 0; i < n - 1; i++)
        niz[i] = niz[i + 1];
93
      return n - 1;
95  }

97  /* Funkcija brise poslednji element niza. Vraca novu
       dimenziju niza. */
99  int brisi_poslednjeg(int niz[], int n)
    {
101   if(n == 0)
      {
103     printf("Greska: nije moguce brisanje iz praznog niza.\n");
        exit(EXIT_FAILURE);
105   }

107   /* Dovoljno je smanjiti dimenziju niza, elemente niza nije
         potrebno brisati. */
109   return n - 1;
    }
111
    /* Funkcija brise element x. Pretpostavlja se da element
113    ima samo jedno pojavljivanje (za vezbu napisati funkciju koja
       brise sva pojavljivanja, ako ih ima vise). Vraca novu dimenziju
115    niza. */
    int brisi_element(int niz[], int n, int x)
117 {
      int i, j;
119
      /* Prvo treba pronaci poziciju elementa u nizu. */
121   for (i = 0; i < n; i++)
        if (niz[i] == x)
123       break;

125   /* Provera da li element postoji u nizu. Ako je brojac stigao do
         kraja niza, onda element ne postoji u nizu. */
127   if (i == n) {
        printf("Klijent sa rednim brojem %d ne postoji u nizu.\n", x);
129     return n;
      }
131
      /* Ukoliko element postoji u nizu, svi elementi niza nakon njega
133      se pomeraju za jedno mesto u levo. */
      for (j = i; j < n - 1; j++)
135     niz[j] = niz[j + 1];

137   return n - 1;
    }
```

```
139
    int main()
141 {
      int n, niz[MAKS], i, klijent, pozicija;
143
      /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
145       ulaza. */
      printf("Unesite trenutni broj klijenata: ");
147   scanf("%d", &n);
      if (n <= 0 || n > MAKS) {
149     printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
151   }

153   /* Ucitavaju se elementi niza. */
      printf("Unesite niz sa rednim brojevima klijenata: ");
155   for (i = 0; i < n; i++)
        scanf("%d", &niz[i]);
157
      /* Ubacivanje klijenta na kraj. */
159   printf("Unesite klijenta kojeg treba ubaciti u niz: ");
      scanf("%d", &klijent);
161   n = ubaci_na_kraj(niz, n, klijent);
      printf("Niz nakon ubacivanja klijenta:\n");
163   ispis(niz, n);

165   /* Ubacivanje klijenta na pocetak. */
      printf("Unesite prioritetnog klijenta kojeg treba"
167          "ubaciti u niz: ");
      scanf("%d", &klijent);
169   n = ubaci_na_pocetak(niz, n, klijent);
      printf("Niz nakon ubacivanja klijenta:\n");
171   ispis(niz, n);

173   /* Ubacivanje klijenta na zadatu poziciju. */
      printf("Unesite prioritetnog klijenta kojeg treba ubaciti "
175          "u niz i njegovu poziciju:");
      scanf("%d%d", &klijent, &pozicija);
177   if (pozicija < 0 || pozicija > n) {
        printf("Greska: neispravan unos.\n");
179     exit(EXIT_FAILURE);
      } else {
181     n = ubaci_na_poziciju(niz, n, klijent, pozicija);
        printf("Niz nakon ubacivanja klijenta:\n");
183     ispis(niz, n);
      }
185
      /* Brisanje prvog klijenta. */
187   n = brisi_prvog(niz, n);
      printf("Niz nakon odlaska klijenta:\n");
189   ispis(niz, n);
```

```
191   /* Brisanje poslednjeg klijenta. */
      n = brisi_poslednjeg(niz, n);
193   printf("Niz nakon odlaska klijenta:\n");
      ispis(niz, n);
195
      /* Brisanje klijenta sa datim rednim brojem. */
197   printf("Unesite redni broj klijenta koji je napustio red: ");
      scanf("%d", &klijent);
199   n = brisi_element(niz, n, klijent);
      printf("Niz nakon odlaska klijenta:\n");
201   ispis(niz, n);

203   exit(EXIT_SUCCESS);
}
```

3.3 Pokazivači

Zadatak 3.3.1 Napisati funkciju `void uredi(int *pa, int *pb)` koja uređuje svoja dva celobrojna argumenta tako da se u prvom nalazi manja vrednost, a u drugom veća. Napisati program koji učitava dva cela broja i ispisuje uređene brojeve.

Primer 1

```
Interakcija sa programom:
 Unesite dva broja:  2 5
 Uredjene promenljive: 2, 5
```

Primer 2

```
Interakcija sa programom:
 Unesite dva broja:  11 -4
 Uredjene promenljive: -4, 11
```

[Rešenje 3.3.1]

Zadatak 3.3.2 Napisati funkciju `void rgb_u_cmy(int r, int g, int b, float* c, float* m, float* y)` koja datu boju u *rgb* formatu konvertuje u boju u *cmy* formatu po sledećim formulama:

$$c = 1 - r/255$$

$$m = 1 - g/255$$

$$y = 1 - b/255$$

Napisati program koji učitava boju u *rgb* formatu i ispisuje vrednosti unete boje u *cmy* formatu. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. Napomena: *Vrednosti boja u rgb formatu su u opsegu* $[0, 255]$.

Primer 1

```
Interakcija sa programom:
 Unesite boju u rgb formatu: 56 111 24
 cmy: (0.78, 0.56, 0.91)
```

Primer 2

```
Interakcija sa programom:
 Unesite boju u rgb formatu: 156 -90 5
 Greska: neispravan unos.
```

Primer 3

```
Interakcija sa programom:
 Unesite boju u rgb formatu: 9 0 237
 cmy: (0.96, 1.00, 0.07)
```

Primer 4

```
Interakcija sa programom:
 Unesite boju u rgb formatu: 300 11 27
 Greska: neispravan unos.
```

[Rešenje 3.3.2]

Zadatak 3.3.3 Napisati funkciju `int presek(float k1, float n1, float k2, float n2, float *px, float *py)` koja za dve prave date svojim koeficijentima pravca i slobodnim članovima određuje njihovu tačku preseka. Funkcija treba da vrati jedinicu ako se prave seku, a nulu ako nemaju tačku preseka (ako su paralelne). Napisati program koji učitava podatke o pravama i ukoliko prave imaju presek, ispisuje koordinate tačke preseka, a ako nemaju, ispisuje odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite k i n za prvu pravu: 4 5
 Unesite k i n za drugu pravu: 11 -4
 Prave se seku u tacki (1.29,10.14).
```

Primer 2

```
Interakcija sa programom:
 Unesite k i n za prvu pravu: 0.5 -4.7
 Unesite k i n za drugu pravu: 0.5 9.1
 Prave su paralelne.
```

[Rešenje 3.3.3]

Zadatak 3.3.4 Napisati funkciju koja za dva cela broja izračunava njihov količnik i ostatak pri deljenju. Funkcija treba da vrati jedinicu ukoliko je uspešno izračunala vrednosti, a nulu ukoliko deljenje nije moguće. Napisati program koji učitava dva cela broja i ispisuje njihov količnik i ostatak pri deljenju. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 4 5
 Kolicnik: 0
 Ostatak: 4
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: 4 0
 Greska: neispravan unos.
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve: -123 11
 Kolicnik: -11
 Ostatak: -2
```

[Rešenje 3.3.4]

Zadatak 3.3.5 Napisati funkciju koja za dužinu trajanja filma koje je dato u sekundama, određuje ukupno trajanje filma u satima, minutama i sekundama. Napisati program koji učitava trajanje filma u sekundama i ispisuje odgovarajuće vreme trajanja u formatu *broj_sati:broj_minuta:broj_sekundi*. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Trajanje fima u sekundama: 5000
 1h:23m:20s
```

Primer 2

```
Interakcija sa programom:
 Trajanje fima u sekundama: -300
 Greska: neispravan unos.
```

Primer 3

```
Interakcija sa programom:
 Trajanje fima u sekundama: 2500
 0h:41m:40s
```

Primer 4

```
Interakcija sa programom:
 Trajanje fima u sekundama: 7824
 2h:10m:24s
```

[Rešenje 3.3.5]

Zadatak 3.3.6 Napisati funkciju koja sa ulaza učitava karakter po karakter sve do kraja ulaza i prebrojava sva pojavljivanja karaktera tačka i sva pojavljivanja karaktera zarez. Napisati program koji za uneti tekst ispisuje koliko puta se pojavila tačka, a koliko puta se pojavio zarez.

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 3

```
Interakcija sa programom:
 Unesite tekst:
 sunce
 Broj tacaka: 0
 Broj zareza: 0
```

[Rešenje 3.3.6]

Zadatak 3.3.7 Napisati funkciju `void par_nepar(int a[], int n, int parni[], int* np, int neparni[], int* nn)` koja razbija niz a na niz parnih i niz neparnih brojeva. Pokazivači np i nn redom treba da sadrže broj elemenata niza parnih tj. niza neparnih elemenata. Maksimalan broj elemenata niza je 50. Napisati program koji učitava dimenziju niza, a zatim i elemente niza i ispisuje odgovarajuće nizove parnih, odnosno neparnih elemenata unetog niza. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 4

```
Interakcija sa programom:
 Unesite broj elemenata niza: 0
 Greska: neispravan unos.
```

[Rešenje 3.3.7]

Zadatak 3.3.8 Napisati funkciju koja izračunava najmanji i najveći element niza realnih brojeva. Napisati program koji učitava niz realnih brojeva maksimalne dužine 50 i ispisuje vrednosti najmanjeg i najvećeg elementa niza, zaokružene na tri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza:  5
 Unesite elemente niza:
 24.16 -32.11 999.25 14.25 11
 Najmanji: -32.110
 Najveci: 999.250
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza:  4
 Unesite elemente niza:
 -5.126 -18.29 44 29.268
 Najmanji: -18.290
 Najveci: 44.000
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza:  1
 Unesite elemente niza:
 4.16
 Najmanji: 4.160
 Najveci: 4.160
```

Primer 4

```
Interakcija sa programom:
 Unesite broj elemenata niza:  -3
 Greska: neispravan unos.
```

[Rešenje 3.3.8]

3.4 Rešenja

Rešenje 3.3.1

```c
#include <stdio.h>
#include <stdlib.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 ne vide u
   glavnom programu.
void uredi_pogresno(int a, int b)
{
  int pom;
  if (a > b) {
    pom = a;
    a = b;
    b = pom;
  }
}
*/

/* Argumenti funkcije uredi, 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 b pristupamo sa *pb.

   Vrednosti promenljivih *pa i *pb razmenjujemo kao i vrednosti
   bilo koje dve celobrojne promenljive. */
void uredi(int *pa, int *pb)
{
  int pom;
  if (*pa > *pb)
  {
    pom = *pa;
    *pa = *pb;
    *pb = pom;
  }
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b;

  /* Ucitavaju se vrednosti dva cela broja. */
  printf("Unesite dva broja:");
  scanf("%d%d", &a, &b);

  /* Neispravan nacin:
```

```
  uredi_pogresno(a, b);
  printf("Uredjene promenljive: %d, %d\n", a, b); */

  /* Funkcija uredi kao argumente prima dve pokazivacke
     promenljive (int*,int*). Zbog toga joj je u pozivu funkcije
     neophodno proslediti adrese promenljivih koje zelimo da
     uredimo rastuce, &a i &b. */
  uredi(&a, &b);
  printf("Uredjene promenljive: %d, %d\n", a, b);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.3.2

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

#define MIN_RGB 0
#define MAKS_RGB 255

/* Funkcija koja vrsi konverziju boje iz rgb formata u cmy format.
   Kako se pomocu naredbe return ne moze vratiti vise od jedne
   vrednosti, neophodno je da se promenljive cije se vrednosti
   racunaju prenesu preko pokazivaca. */
void rgb_u_cmy(int r, int g, int b, float* c, float* m, float* y)
{
  *c = 1 - r / 255.0;
  *m = 1 - g / 255.0;
  *y = 1 - b / 255.0;
}

/* Funkcija koja proverava da li je vrednost boje u ispravnom
   opsegu. */
int ispravna_rgb_vrednost(int a)
{
  if (a < MIN_RGB || a > MAKS_RGB)
    return 0;
  return 1;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int r, g, b;
  float c, m, y;

  /* Ucitava se vrednost boje u rgb formatu. */
  printf("Unesite boju u rgb formatu: ");
  scanf("%d%d%d", &r, &g, &b);
```

```
  /* Vrsi se provera ispravnosti ulaza. */
  if (!ispravna_rgb_vrednost(r) || !ispravna_rgb_vrednost(g) ||
      !ispravna_rgb_vrednost(b))
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Konverzija boje i ispis rezultata. Funkciji se kao argumenti
     prosledjuju vrednosti brojeva r, g, i b, kao i adrese na koje
     treba da se upisu izracunate c, m, y vrednosti. */
  rgb_u_cmy(r, g, b, &c, &m, &y);
  printf("cmy: (%.2f,%.2f,%.2f)\n", c, m, y);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.3.3

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

/* Funkcija koja racuna presek pravih y = k1 * x + n1 i
   y = k2 * x + n2. Koordinate preseka (ako postoji) se upisuju
   na adrese px i py. Kao povratna vrednost funkcije se vraca
   jedinica ukoliko presek postoji, a nula inace. */
int presek(float k1, float n1, float k2, float n2, float *px,
           float *py)
{
  /* Ako je koeficijent pravca jednak, prave su paralelne.
     Povratna vrednost funkcije je 0, kao indikator da
     nema presecne tacke. */
  if (k1 == k2)
    return 0;

  /* Koordinate preseka se upisuju na adrese (px, py). */
  *px = -(n1 - n2) / (k1 - k2);
  *py = k1 * (*px) + n1;

  /* Funkcija vraca 1 kao indikator da je presek uspesno
     izracunat. */
  return 1;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  float k1, k2, n1, n2;
  float x, y;

  /* Ucitavaju se parametri za dve prave. */
```

```
  printf("Unesite k i n za prvu pravu:");
  scanf("%f%f", &k1, &n1);
  printf("Unesite k i n za drugu pravu:");
  scanf("%f%f", &k2, &n2);

  /* Ispis rezultata. */
  if (presek(k1, n1, k2, n2, &x, &y))
    printf("Prave se seku u tacki (%.2f,%.2f).\n", x, y);
  else
    printf("Prave su paralelne.\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.3.4

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

/* Funkcija koja racuna kolicnik i ostatak pri deljenju a sa b.
   Ukoliko su ove vrednosti uspesno izracunate, funkcija vraca 1,
   a inace vraca nulu. */
int kolicnik_ostatak(int a, int b, int* pk, int* po)
{
  if(b == 0)
    return 0;

  *pk = a/b;
  *po = a%b;
  return 1;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int a, b, kolicnik, ostatak;

  /* Ucitavaju se vrednosti a i b. */
  printf("Unesite brojeve: ");
  scanf("%d%d", &a, &b);

  /* Ispis rezultata. */
  if(kolicnik_ostatak(a, b, &kolicnik, &ostatak))
  {
    printf("Kolicnik: %d\n", kolicnik);
    printf("Ostatak: %d\n", ostatak);
  }
  else
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
```

```
36    }

38    exit(EXIT_SUCCESS);
}
```

Rešenje 3.3.5

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

/* Funkcija koja dato trajanje izrazeno u ukupnom broju sekundi
   konvertuje u trajanje koje je izrazeno u broju sati, minuta
   i sekundi. */
void konverzija(int trajanje, int* psati, int* pminuti,
                int* psekunde)
{
  *psati = trajanje / 3600;
  trajanje -= *psati * 3600;

  *pminuti = trajanje/60;
  trajanje -= *pminuti * 60;

  *psekunde = trajanje;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int trajanje, sati, minuti, sekunde;

  /* Ucitava se trajanje u sekundama. */
  printf("Trajanje filma u sekundama: ");
  scanf("%d", &trajanje);

  /* Vrsi se provera ispravnosti ulaza. */
  if(trajanje < 0)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Racunanje rezultata. */
  konverzija(trajanje, &sati, &minuti, &sekunde);

  /* Ispis rezultata. */
  printf("%dh:%dm:%ds\n", sati, minuti, sekunde);
  exit(EXIT_FAILURE);
}
```

Rešenje 3.3.6

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

/* Funkcija koja ucitava karakter po karakter sa ulaza i prebrojava
   koliko puta se pojavio karakter '.' i koliko puta se pojavio
   karakter ','. Ucitavanje se zaustavlja kada se dodje do kraja
   ulaza (EOF-a). */
void interpunkcija(int *br_tacaka, int *br_zareza)
{
  /* Deklaracije i inicijalizacije pomocnih promenljivih. */
  int tacke = 0, zarezi = 0;
  char c;

  /* Ucitavanje i prebrojavanje trazenih karaktera. */
  while ((c = getchar()) != EOF)
  {
    if (c == '.')
      tacke++;

    if (c == ',')
      zarezi++;
  }

  /* Smestanje rezultata na prosledjene adrese. */
  *br_tacaka = tacke;
  *br_zareza = zarezi;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int br_tacaka, br_zareza;

  /* Ucitavanje i obrada teksta. */
  printf("Unesite tekst: \n");
  interpunkcija(&br_tacaka, &br_zareza);

  /* Ispis rezultata. */
  printf("Broj tacaka: %d\n", br_tacaka);
  printf("Broj zareza: %d\n", br_zareza);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.3.7

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

```
3
#define MAKS 50
5
/* Funkcija koja od niza a formira dva niza: niz parnih elemenata
7    niza a i niz neparnih elemenata niza a. Duzine rezultujucih
     nizova se upisuju na adrese np i nn. */
9 void par_nepar(int a[], int n, int parni[], int *np,
                 int neparni[], int *nn)
11 {
   int i, j, k;
13
   /* Promenljiva i je brojac u originalnom nizu i on se uvecava u
15      svakoj iteraciji.
        Promenljiva j je projac za niz parnih brojeva i on treba da se
17      uveca svaki put kada se naidje na novi element ovog niza.
        Promenljiva k je brojac za niz neparnih brojeva i on treba da
19      se uveca sveki put kada se naidje na novi element ovog niza. */
   for (i = 0, j = 0, k = 0; i < n; i++)
21 {
     if (a[i] % 2 == 0)
23   {
       parni[j] = a[i];
25     j++;
     }
27   else
     {
29     neparni[k] = a[i];
       k++;
31   }
   }
33
   /* Na kraju petlje, u promenljivoj j se nalazi podatak o broju
35      elementa niza parni[], a u promenljivoj k podatak o broju
        elementa niza neparni[]. Ove vrednosti se upisuju na adrese
37      np i nn. */
   *np = j;
39 *nn = k;
}
41
void ispisi(int niz[], int n)
43 {
   int i;
45 for (i = 0; i < n; i++)
     printf("%d ", niz[i]);
47 printf("\n");
}
49
int main()
51 {
   /* Deklaracije potrebnih promenljivih. */
53 int n, n1, n2;
   int a[MAKS], parni[MAKS], neparni[MAKS];
```

```
55    int i;

57    /* Ucitava se dimenzija niza. */
      printf("Unesite broj elemenata niza: ");
59    scanf("%d", &n);

61    /* Vrsi se provera ispravnosti ulaza. */
      if (n < 0 || n > MAKS)
63    {
        printf("Greska: neispravan unos.\n");
65      exit(EXIT_FAILURE);
      }
67
      /* Ucitavaju se elementi niza. */
69    printf("Unesite elemente niza: ");
      for (i = 0; i < n; i++)
71      scanf("%d", &a[i]);

73    /* Nizovi parni[] i neparni[] se popunjavaju odgovarajucim
         vrednostima. */
75    par_nepar(a, n, parni, &n1, neparni, &n2);

77    /* Ispis niza parni[] koji ima n1 elemenata. */
      printf("Niz parnih brojeva: ");
79    ispisi(parni, n1);

81    /* Ispis niza neparni[] koji ima n2 elemenata. */
      printf("Niz neparnih brojeva: ");
83    ispisi(neparni, n2);

85    exit(EXIT_SUCCESS);
  }
```

Rešenje 3.3.8

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    #define MAKS 50
5
    /* Funkcija koja racuna najmanji i najveci element niza a. */
7   void min_max(float a[], int n, float *najmanji, float *najveci)
    {
9     int i;

11    /* Vrednosti minimuma i maksimuma se inicijalizuju na vrednost
         prvog clana niza. */
13    *najmanji = a[0];
      *najveci = a[0];
15
      /* U petlji se prolazi kroz ostale clanove niza i po potrebi se
```

```
17      vrsi azuriranje najmanje i najvece vrednosti. */
    for (i = 1; i < n; i++)
19  {
      if (a[i] > *najveci)
21      *najveci = a[i];

23    if (a[i] < *najmanji)
        *najmanji = a[i];
25  }

27  /* Na kraju petlje, na adresama najmanji i najveci se nalaze
       trazene vrednosti. */
29 }

31 int main()
   {
33   /* Deklaracija potrebnih promenljivih. */
     int i, n;
35   float a[MAKS], min, max;

37   /* Ucitava se dimenzija niza. */
     printf("Unesite broj elemenata niza: ");
39   scanf("%d", &n);

41   /* Vrsi se provera ispravnosti ulaza. */
     if (n < 0 || n > MAKS)
43   {
       printf("Greska: neispravan unos.\n");
45     exit(EXIT_FAILURE);
     }
47

     /* Ucitavaju se elementi niza. */
49   printf("Unesite elemente niza:\n");
     for (i = 0; i < n; i++)
51     scanf("%f", &a[i]);

53   /* Racunaju se vrednosti najmanjeg i najveceg elementa. */
     min_max(a, n, &min, &max);
55

     /* Ispis rezultata. */
57   printf("Najmanji: %.3f\n", min);
     printf("Najveci: %.3f\n", max);
59

     exit(EXIT_SUCCESS);
61 }
```

3.5 Niske

Zadatak 3.5.1 Napisati funkciju `void konvertuj(char s[])` koja menja nisku s tako što mala slova zamenjuje odgovarajućim velikim slovima, a velika

slova zamenjuje odgovarajućim malim slovima. Napisati program koji učitava nisku maksimalne dužine 10 karaktera i ispisuje konvertovanu nisku.

Primer 1

```
Interakcija sa programom:
 Unesite nisku: BeoGrad
 bEOgRAD
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku: A+B+C
 a+b+c
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: 12345
 12345
```

[Rešenje 3.5.1]

Zadatak 3.5.2 Napisati funkciju `void ubaci_zvezdice(char s[])` koja menja nisku s tako što u njoj svaki drugi karakter zameni zvezdicom. Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje izmenjenu nisku.

Primer 1

```
Interakcija sa programom:
 Unesite nisku: *a*b*c*
 Izmenjena niska: *******
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku: zimA
 Izmenjena niska: z*m*
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: 123abc789
 Izmenjena niska: 1*3*b*7*9
```

[Rešenje 3.5.2]

Zadatak 3.5.3 Napisati program koji vrši poređenje niski. Napisati funkcije:

(a) `int jednake(char s1[], char s2[])` koja vraća jedinicu ako su s_1 i s_2 jednake niske, a nulu inače.

(b) `void u_velika_slova(char s[])` koja pretvara sva slova niske s u velika slova, a ostale karaktere ne menja.

Program učitava dve reči maksimalne dužine 20 karaktera i ispituje da li su unete reči jednake. Pri poređenju treba zanemariti razliku između malih i velikih slova.

Primer 1

```
Interakcija sa programom:
 Unesite niske:
 isPit2010
 IsPiT2010
 Niske su jednake.
```

Primer 2

```
Interakcija sa programom:
 Unesite niske:
 Prog1
 prog2
 Niske nisu jednake.
```

Primer 3

```
Interakcija sa programom:
 Unesite niske:
 jun
 JUNSKI
 Niske nisu jednake.
```

[Rešenje 3.5.3]

Zadatak 3.5.4 Napisati program koji proverava da li se uneta niska završava samoglasnikom. Napisati funkcije:

(a) `int samoglasnik(char c)` — ispituje da li je karakter c samoglasnik;

(b) `int samoglasnik_na_kraju(char s[])` — ispituje da li se niska s završava samoglasnikom.

Pretpostaviti da je uneta niska maksimalne dužine 20 karaktera.

Primer 1

```
Interakcija sa programom:
 Unesite nisku: kestenje
 Niska se zavrsava samoglasnikom.
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku: vetar
 Niska se ne zavrsava samoglasnikom.
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: OLUJA
 Niska se zavrsava samoglasnikom.
```

Primer 4

```
Interakcija sa programom:
 Unesite nisku: Programiranje1
 Niska se ne zavrsava samoglasnikom.
```

[Rešenje 3.5.4]

Zadatak 3.5.5 Napisati funkciju `int sadrzi_veliko(char s[])` koja proverava da li niska s sadrži veliko slovo. Napisati program koji za učitanu nisku maksimalne dužine 20 karaktera proverava da li sadrži veliko slovo i ispisuje odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite nisku:
 naocare
 Ne sadrzi veliko slovo.
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku:
 DiopTrija0.75
 Sadrzi veliko slovo.
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku:
 21.06.2017.
 Ne sadrzi veliko slovo.
```

[Rešenje 3.5.5]

Zadatak 3.5.6 Napisati program koji za učitanu nisku s i karakter c ispituje da li se c pojavljuje u niski s. Ako se pojavljuje, program treba da ispiše indeks prvog pojavljivanja karaktera c u niski s, a u suprotnom -1. Pretpostaviti da niska može da ima najviše 20 karaktera.

Primer 1

```
Interakcija sa programom:
 Unesite nisku:
 bazen
 Unesite karakter:
 z
 2
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku:
 lezaljka
 Unesite karakter:
 a
 3
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku:
 limunada
 Unesite karakter:
 b
 -1
```

[Rešenje 3.5.6]

Zadatak 3.5.7 Napisati funkciju `int podniska(char s[], char t[])` koja proverava da li je niska t podniska niske s. Napisati program koji učitava niske s i t maksimalne dužine 10 karaktera i ispisuje da li je niska t podniska niske s.

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 3.5.7]

Zadatak 3.5.8 Napisati funkciju `void skrati(char s[])` koja uklanja beline sa kraja date niske. Napisati program koji učitava liniju maksimalne dužine 100 karaktera i ispisuje učitanu i izmenjenu nisku između zvezdica.

Primer 1

```
Interakcija sa programom:
 Unesite nisku:
 rep belina
 Ucitana niska:
 *rep belina                *
 Izmenjena niska:
 *rep belina*
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku:
 tri tabulatora na kraju
 Ucitana niska:
 *tri tabulatora na kraju        *
 Izmenjena niska:
 *tri tabulatora na kraju*
```

[Rešenje 3.5.8]

Zadatak 3.5.9 Napisati funkciju `void ukloni_slova(char s[])` koja iz niske s uklanja sva mala i sva velika slova. Napisati program koji za učitanu nisku maksimalne dužine 20 karaktera ispisuje odgovarajuću izmenjenu nisku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 a1b2c3def
 123
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 1+2=3
 1+2=3
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 malaVELIKA
```

[Rešenje 3.5.9]

Zadatak 3.5.10 Napisati funkciju `void ukloni(char *s)` koja iz niske uklanja sva slova iza kojih neposredno sledi slovo koje je u abecedi nakon njih, pri čemu se veličina slova zanemaruje. Napisati program koji učitava liniju teksta koja ima najviše 100 karaktera i ispisuje liniju koja se dobije nakon uklanjanja pomenutih karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Zdravo svima!
 Zrvo vma!
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Danas je 10 stepeni.
 Dns j 10 tpni.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Ima vetra, kise i hladnoce.
 ma vtra, se i loe.
```

[Rešenje 3.5.10]

Zadatak 3.5.11 Napisati program koji učitava nisku s maksimalne dužine 30 karaktera i formira nisku t trostrukim nadovezivanjem niske s.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: dan
 dandandan
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: 3sesira
 3sesira3sesira3sesira
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: a-b=5
 a-b=5a-b=5a-b=5
```

[Rešenje 3.5.11]

Zadatak 3.5.12 Napisati program koji za unetu reč maksimalne dužine 20 karaktera i pozitivan broj n manji od 10, formira rezultujuću reč tako što unetu reč kopira n puta, pri čemu se između svaka dva kopiranja umeće crtica. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: ana
 Unesite broj n: 4
 ana-ana-ana-ana
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: 123
 Unesite broj n: 1
 123
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: x*y
 Unesite broj n: 3
 x*y-x*y-x*y
```

[Rešenje 3.5.12]

Zadatak 3.5.13 Napisati funkciju `void kopiraj_n(char t[], char s[], int n)` koja kopira najviše n karaktera niske s u nisku t. Napisati program koji testira rad napisane funkcije. Pretpostaviti da je maksimalna dužina niske s 20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: petar
 Unesite broj n: 3
 Rezultujuca niska: pet
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: gromobran
 Unesite broj n: 4
 Rezultujuca niska: grom
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku: abc
 Unesite broj n: 15
 Rezultujuca niska: abc
```

[Rešenje 3.5.13]

Zadatak 3.5.14 Napisati funkciju `void dupliranje(char t[], char s[])` koja na osnovu niske s formira nisku t tako što duplira svaki karakter niske s. Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje nisku koja se dobije nakon dupliranja karaktera.

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 3.5.14]

Zadatak 3.5.15 Napisati program koji učitava nisku cifara sa eventualnim vodećim znakom i pretvara je u ceo broj. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
Interakcija sa programom:
 Unesite nisku:
 -1238
 -1238
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku:
 73
 73
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku:
 +1
 1
```

[Rešenje 3.5.15]

Zadatak 3.5.16 Napisati program koji učitava ceo broj, pretvara ga u nisku i ispisuje dobijenu nisku.

Primer 1

```
Interakcija sa programom:
 Unesite ceo broj:
 -6543
 -6543
```

Primer 2

```
Interakcija sa programom:
 Unesite ceo broj:
 84
 84
```

Primer 3

```
Interakcija sa programom:
 Unesite ceo broj:
 5
 5
```

[Rešenje 3.5.16]

Zadatak 3.5.17 Napisati funkciju `int heksadekadni_broj(char s[])` koja proverava da li je niskom s zadat korektan heksadekadni broj. Funkcija treba da vrati vrednost 1 ukoliko je uslov ispunjen, odnosno 0 ako nije. Napisati program koji za učitanu nisku maksimalne dužine 7 karaktera ispisuje da li je korektan heksadekadni broj. Uputstvo: *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.

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 3.5.17]

Zadatak 3.5.18 Napisati funkciju `int dekadna_vrednost(char s[])` koja izračunava dekadnu vrednost heksadekadnog broja zadatog niskom s. Napisati program koji za učitanu nisku maksimalne dužine 7 karaktera ispisuje odgovarajuću dekadnu vrednost. 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 3.5.18]

Zadatak 3.5.19 Napisati funkciju `int ucitaj_liniju(char s[], int n)` koja učitava liniju maksimalne dužine n u nisku s i vraća dužinu učitane linije. Napisati program koji učitava linije do `EOF` i ispisuje najdužu liniju i njenu dužinu. Ukoliko ima više linija maksimalne dužine, ispisati prvu. Pretpostaviti da svaka linija sadrži najviše 80 karaktera. Napomena: *Linija može da sadrži blanko znakove, ali ne sadrži znak za novi red ili `EOF`.*

Primer 1

```
Interakcija sa programom:
 Unesite linije:
 Dobar dan!
 Kako ste, sta ima novo?
 Ja sam dobro.
 Kako ste, sta ima novo?
 23
```

Primer 2

```
Interakcija sa programom:
 Unesite linije:
 Prva linija
 Druga linija
 Treca linija
 Druga linija
 12
```

Primer 3

```
Interakcija sa programom:
 Unesite linije:
 Danas je lep dan.
 Danas je lep dan.
 17
```

[Rešenje 3.5.19]

*** Zadatak 3.5.20** Napisati funkcije za rad sa rečenicama:

(a) `int procitaj_recenicu(char s[], int n)` koja učitava rečenicu i smešta je u nisku s. Funkcija vraća dužinu učitane rečenice. Učitavanje se završava nakon učitanog karaktera `.`, nakon n učitanih karaktera ili ako se dođe do kraja ulaza.

(b) `void prebroj(char s[], int *broj_malih, int *broj_velikih)` koja prebrojava mala i velika slova u niski s.

Napisati program koji učitava rečenice do kraja ulaza i ispisuje onu rečenicu kod koje je apsolutna razlika broja malih i velikih slova najveća. Pri učitavanju rečenica zanemariti sve beline koje se nalaze između dve rečenice. Pretpostaviti da jedna rečenica sadrži najviše 80 karaktera.

Primer 1

```
Interakcija sa programom:
Unesite tekst:
U ovom poglavlju se govori o niskama. Niske su nizovi karaktera ciji je
poslednji element terminalna nula.
U ovom zadatku je potrebno ucitati recenice. Svaka recenica pocinje sa bilo
kojim karakterom koji nije belina. Na kraju recenice se nalazi tacka.
Niske su nizovi karaktera ciji je poslednji element terminalna nula.
```

Primer 2

```
Interakcija sa programom:
Unesite tekst:
bbbbAAA. AbAbAb. AbAbAbab.
AbAbAbab.
```

Primer 3

```
Interakcija sa programom:
Unesite tekst:

Nije uneta nijedna recenica.
```

Primer 4

```
Interakcija sa programom:
Unesite tekst:
AAAbbb. bbbAAA. AbAbAb.
AAAbbb
```

[Rešenje 3.5.20]

Zadatak 3.5.21 Napisati funkciju `char* strchr_klon(char s[], char c)` koja vraća pokazivač na prvo pojavljivanje karaktera c u niski s ili $NULL$ ukoliko se karakter c ne pojavljuje u niski s. [1] Napisati program koji za učitanu nisku maksimalne dužine 20 karaktera i karakter c ispisuje indeks prvog pojavljivanja karaktera c u okviru učitane niske ili -1 ukoliko učitana niska ne sadrži uneti karakter.

Primer 1

```
Interakcija sa programom:
Unesite nisku s: programiranje
Unesite karakter c: a
Pozicija: 5
```

Primer 2

```
Interakcija sa programom:
Unesite nisku s: 123456789
Unesite karakter c: y
Pozicija: -1
```

Primer 3

```
Interakcija sa programom:
Unesite nisku s: leto2017
Unesite karakter c: 0
Pozicija: 5
```

Primer 4

```
Interakcija sa programom:
Unesite nisku s: jedrilica
Unesite karakter c: I
Pozicija: -1
```

[Rešenje 3.5.21]

[1] Funkcija *strchr_klon* odgovara funkciji *strchr* čija se deklaracija nalazi u zaglavlju string.h. Slično važi i za ostale *klon* funkcije iz narednih zadataka.

Zadatak 3.5.22 Napisati funkciju `int strspn_klon(char t[], char s[])` koja izračunava dužinu prefiksa niske t sastavljenog od karaktera niske s. Napisati program koji za učitane dve niske maksimalne dužine 20 karaktera ispisuje rezultat poziva napisane funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite nisku t: program
 Unesite nisku s: pero
 3
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku t: Barselona
 Unesite nisku s: Brazil
 3
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku t: 24.10.2017.
 Unesite nisku s: 0123456789
 2
```

Primer 4

```
Interakcija sa programom:
 Unesite nisku t: 12345
 Unesite nisku s: 9876543210
 5
```

[Rešenje 3.5.22]

Zadatak 3.5.23 Napisati funkciju `int strcspn_klon(char t[], char s[])` koja izračunava dužinu prefiksa niske t sastavljenog isključivo od karaktera koji se ne nalaze u niski s. Napisati program koji testira ovu funkciju za dve unete niske maksimalne dužine 100 karaktera.

Primer 1

```
Interakcija sa programom:
 Unesite nisku t:
 programiranje
 Unesite nisku s:
 pero
 0
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku t:
 programiranje
 Unesite nisku s:
 analiza
 5
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku t:
 programiranje
 Unesite nisku s:
 1.10.
 13
```

[Rešenje 3.5.23]

Zadatak 3.5.24 Napisati funkciju `char* strstr_klon(char s[], char t[])` koja vraća pokazivač na prvo pojavljivanje niske t u niski s ili $NULL$ ukoliko se niska t ne pojavljuje u niski s. Napisati program koji testira napisanu funkciju tako što učitava pet linija i ispisuje sve redne brojeve linija koje sadrže nisku *program*. Ukoliko ne postoji linija sa niskom *program*, ispisati odgovarajuću poruku. Pretpostaviti da je svaka linija maksimalne dužine 100 karaktera kao i da se linije numerišu od broja 1.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
tv program
c prog. jezik
c++ programskih jezik
Programski odbor
<b>program</b>
1 3 5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
Programske paradigme
su predmet na
trecoj godini
programerskih
smerova.
4
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
U narednim
linijama
necemo navoditi
nisku koja se
trazi.
Nijedna linija ne sadrzi
nisku program.
```

[Rešenje 3.5.24]

Zadatak 3.5.25 Napisati funkciju `int strcmp_klon(char s[], char t[])` koja vraća 0 ako su niske s i t jednake, neku pozitivnu vrednost ako je s leksikografski iza t, a neku negativnu vrednost inače. Napisati program koji učitava dve niske maksimalne dužine 20 karaktera i ako su različite, ispisuje učitane niske u rastućem leksikografskom poretku, a ako su jednake, ispisuje samo jednu nisku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku s:
Beograd
Unesite nisku t:
Amsterdam
Amsterdam
Beograd
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku s:
Beograd
Unesite nisku t:
Beograd
Beograd
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku s:
radnik
Unesite nisku t:
radnica
radnica
radnik
```

[Rešenje 3.5.25]

Zadatak 3.5.26 Napisati funkciju `void obrni(char s[])` koja obrće nisku s. Napisati program koji obrće učitanu nisku maksimalne dužine 20 karaktera i ispisuje obrnutu nisku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
kisobran
narbosik
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Aleksandar
radnaskelA
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
kajak
kajak
```

[Rešenje 3.5.26]

Zadatak 3.5.27 Napisati funkciju `void rotiraj(char s[], int k)` koja rotira nisku s za k mesta ulevo. Napisati program koji učitava nisku maksimalne dužine 20 karaktera i nenegativan ceo broj k i ispisuje rotiranu nisku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 sveska
 2
 eskasv
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 olovka
 6
 olovka
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 rezac
 8
 acrez
```

[Rešenje 3.5.27]

Zadatak 3.5.28 Napisati program koji šifrira unetu nisku tako što svako slovo zamenjuje sledećim slovom abecede (slova 'z' i 'Z' zamenjuje redom sa 'a' i 'A'), a ostale karaktere ostavlja nepromenjene. Pretpostaviti da uneta niska nije duža od 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 bundeva
 cvoefwb
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 zimzelen
 ajnafmfo
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 Oktobar17
 Plupcbs17
```

[Rešenje 3.5.28]

Zadatak 3.5.29 Napisati funkciju `void sifruj(char rec[], char sifra[])` koja na osnovu date reči formira šifru tako što se svako slovo u reči zameni sa naredna tri slova u abecedi. Napisati program koji testira napisanu funkciju za reč maksimalne dužine 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 tamo
 uvwbcdnoppqr
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 Zec
 ABCfghdef
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 a+b=c
 bcd+cde=def
```

[Rešenje 3.5.29]

Zadatak 3.5.30 Napisati funkciju `void formiraj(char s1[], char s2[], char c1, char c2)` koja na osnovu niske s_1 formira nisku s_2 udvajanjem svih karaktera c_1 u niski s_1 i izbacivanjem svih karaktera c_2 iz niske s_1, dok ostali karakteri ostaju nepromenjeni. Napisati program koji testira ovu funkciju za unetu nisku i dva uneta karaktera. Pretpostaviti da uneta niska nije duža od 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 flomaster
 Unesite prvi karakter:
 m
 Unesite drugi karakter:
 s
 floasster
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 bojica
 Unesite prvi karakter:
 b
 Unesite drugi karakter:
 a
 bbojic
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 patentara
 Unesite prvi karakter:
 t
 Unesite drugi karakter:
 a
 pttenttr
```

[Rešenje 3.5.30]

*** Zadatak 3.5.31** Napisati program za rad sa brojevima zapisanim u različitim brojevnim sistemima.

(a) Napisati funkciju `unsigned int u_dekadni_sistem(char broj[], unsigned int osnova)` koja određuje dekadnu vrednost zapisa datog neoznačenog broja *broj* u datoj osnovi.

(b) Napisati funkciju `void iz_dekadnog_sistema(unsigned int broj, unsigned int osnova, char rezultat[])` koja datu dekadnu vrednost *broj* zapisuje u datoj osnovi *osnovi* i smešta rezultat u nisku *rezultat*. Pretpostaviti da je $0 < b \leq 16$.

Napisati program koji učitava broj n i osnove o_1 i o_2 i ispisuje dekadnu vrednost broja n u osnovi o_1, kao i vrednost koja se dobije kada se ta dekadna vrednost zapiše u osnovi o_2. Pretpostaviti da je ulaz ispravan i da će svi brojevi biti u opsegu tipa `unsigned`.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite n, o1 i o2: 10101011 2 16
 Dekadna vrednost broja 10101011: 171
 Vrednost broja 171 u osnovi 16: AB
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite n, o1 i o2: 1067 8 3
 Dekadna vrednost broja 1067: 567
 Vrednost broja 567 u osnovi 3: 210000
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite n, o1 i o2: 1010111001010 2 3
 Dekadna vrednost broja 1010111001010: 5578
 Vrednost broja 5578 u osnovi 3: 21122121
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite n, o1 i o2: 111 3 5
 Dekadna vrednost broja 111: 13
 Vrednost broja 13 u osnovi 5: 23
```

[Rešenje 3.5.31]

3.6 Rešenja

Rešenje 3.5.1

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

/* Poslednji karakter svake niske je terminirajuca nula '\0',
   specijalni karakter ciji je ASCII kod 0.

   Ukoliko je pretpostavka da niska sadrzi najvise 10 karaktera,
   neophodno je deklarisati niz od 11 karaktera, pri cemu se
   dodatni izdvaja za terminirajucu nulu. */
#define MAKS_NISKA 11

/* Funkcija vrsi konverziju svakog malog slova niske u odgovarajuce
   veliko i obrnuto. Ostali karakteri ostaju nepromenjeni. */
void konvertuj(char s[])
{
  int i;

  /* Prolazi se kroz nisku, karakter po karakter, sve dok se ne
     dodje do terminalne nule koja sluzi kao oznaka da se doslo
     do kraja niske. */
  for (i = 0; s[i] != '\0'; i++)
  {
    /* Svako malo slovo se pretvara u veliko i obrnuto. */
    if (islower(s[i]))
      s[i] = toupper(s[i]);
    else if (isupper(s[i]))
      s[i] = tolower(s[i]);
  }

  /* II nacin: Uslov u petlji moze krace da se zapise sa s[i]
     jer ASCII kod terminalne nule ima vrednost 0.
  for (i = 0; s[i]; i++)
  {
```

```
35      if (islower(s[i]))
          s[i] = toupper(s[i]);
37      else if (isupper(s[i]))
          s[i] = tolower(s[i]);
39    } */
  }
41
  int main()
43{
    char s[MAKS_NISKA];
45  printf("Unesite nisku: ");

47  /* Za razliku od nizova koji se ucitavaju i stampaju element po
       element, niske se mogu ucitati i odstampati pomocu jedne
49     scanf/printf naredbe koriscenjem specifikatora %s. */
    scanf("%s", s);
51
    /* Izmena niske. */
53  konvertuj(s);

55  /* Ispis rezultata. */
    printf("%s\n", s);
57
    exit(EXIT_SUCCESS);
59}
```

Rešenje 3.5.2

```
1 #include <stdio.h>
  #include <stdlib.h>
3
  #define MAKS_NISKA 21
5
  /* Funkcija ubacuje zvezdice na svako drugo mesto niske s. */
7 void ubaci_zvezdice(char s[])
  {
9   int i;

11  for(i=0; s[i] != '\0' && s[i+1] != '\0'; i+=2)
      s[i+1] = '*';
13
    /* II nacin:
15   for(i=0; s[i]; i++)
       if(i%2 == 1)
17       s[i] = '*';
     */
19}

21int main()
  {
23  /* Deklaracija potrebnih promenljivih. */
```

```
  char s[MAKS_NISKA];

  /* Ucitavanje niske. */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Izmena niske. */
  ubaci_zvezdice(s);

  /* Ispis rezultata. */
  printf("Izmenjena niska: %s\n", s);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.3

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

#define MAKS_NISKA 21

/* Funckija pretvara sva slova niske s u velika slova. */
void u_velika_slova(char s[])
{
  int i;

  for(i=0; s[i]; i++)
    s[i] = toupper(s[i]);
}

/* Funkcija vraca 1 ako su niske s1 i s2 jednake, a nulu inace. */
int jednake(char s1[], char s2[]){
  int i;

  /* Prolazi se kroz obe niske dok god ima neobradjenih karaktera
     u bilo kojoj od njih. Ukoliko se naidje na karaktere koji
     su razliciti, kao povratna vrednost se vraca 0 jer u tom
     slucaju niske nisu jednake. */
  for(i=0; s1[i] || s2[i]; i++)
    if(s1[i] != s2[i])
      return 0;

  /* Ako se doslo do kraja petlje znaci da su se svi karakteri
     poklopili, a da se pri tom doslo do kraja obe niske, tako da
     se kao povratna vrednost funkcije vraca 1 jer su niske
     s1 i s2 jednake. */
  return 1;
}
```

```
int main(){
  /* Deklaracija potrebnih promenljivih. */
  char s1[MAKS_NISKA], s2[MAKS_NISKA];

  /* Ucitavaju se niske s1 i s2. */
  printf("Unesite niske:\n");
  scanf("%s%s", s1, s2);

  /* Kako bi se pri poredjenju zanemarila razlika izmedju malih i
     velikih slova, sva slova obe niske se pretvaraju u velika. */
  u_velika_slova(s1);
  u_velika_slova(s2);

  /* Ispis rezultata. */
  if(jednake(s1, s2))
    printf("Niske su jednake.\n");
  else
    printf("Niske nisu jednake.\n");

  exit(EXIT_FAILURE);
}
```

Rešenje 3.5.4

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

#define MAKS_NISKA 21

/* Funkcija proverava da li je karakter c samoglasnik. */
int samoglasnik(char c)
{
  /* Karakter se pretvara u veliko slovo kako bi se izbegle posebne
     provere za mala i velika slova. */
  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 proverava da li se niska s zavrsava samoglasnikom. */
int samoglasnik_na_kraju(char s[])
{
  /* Funkcija strlen racuna duzinu date niske. Njena deklaracija se
     nalazi u zaglavlju string.h. */
  int duzina = strlen(s);
```

```
29    /* Ako je niska prazna, ne zavrsava se samoglasnikom. */
      if (duzina == 0)
31      return 0;

33    /* Proverava se da li je poslednji karakter niske samoglasnik. */
      return samoglasnik(s[duzina - 1]);
35  }

37  int main()
    {
39    /* Deklaracija niske. */
      char s[MAKS_NISKA];
41
      /* Ucitava se niska. */
43    printf("Unesite nisku: ");
      scanf("%s", s);
45
      /* Ispis rezultata. */
47    if (samoglasnik_na_kraju(s))
        printf("Niska se zavrsava samoglasnikom.\n");
49    else
        printf("Niska se ne zavrsava samoglasnikom.\n");
51
      exit(EXIT_SUCCESS);
53  }
```

Rešenje 3.5.5

```
#include <stdio.h>
2   #include <stdlib.h>
    #include <ctype.h>
4
    #define MAKS_NISKA 21
6
    /* Funkcija proverava da li niska s sadrzi bar jedno
8      veliko slovo. */
    int sadrzi_veliko(char s[])
10  {
      int i;
12
      for(i=0; s[i]; i++)
14      if(isupper(s[i]))
          return 1;
16
      return 0;
18
      /* Cesta greska:
20         for(i=0; s[i]; i++)
           {
22            if(isupper(s[i]))
```

```
            return 1;
          else
            return 0;
      }
    Cim se naidje na prvi karakter koji nije veliko
    slovo, vraca se 0 kao oznaka da niska ne sadrzi
    veliko slovo, sto nije tacno. Nula moze da se vrati
    tek kada se prodju svi karakteri niske s. */
}

int main()
{
  /* Deklaracija niske. */
  char s[MAKS_NISKA];

  /* Ucitava se niska. */
  printf("Unesite nisku:");
  scanf("%s", s);

  /* Ispis rezultata. */
  if(sadrzi_veliko(s))
    printf("Sadrzi veliko slovo.\n");
  else
    printf("Ne sadrzi veliko slovo.\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.6

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

#define MAKS_NISKA 21

/* Funkcija vraca indeks prvog pojavljivanja karaktera c u okviru
   niske s. Ukoliko se ne pojavljuje, funkcija vraca -1. */
int pozicija(char s[], char c)
{
  int i;

  for(i=0; s[i]; i++)
    if(s[i] == c)
      return i;

  return -1;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
```

```
22    char s[MAKS_NISKA];
      char c;
24
      /* Ucitavaju se niska i karakter. */
26    printf("Unesite nisku:");
      scanf("%s", s);
28    getchar();
      printf("Unesite karakter:");
30    c = getchar();

32    /* I nacin: */
       printf("%d\n", pozicija(s,c));
34
      /* II nacin:
36       Funkcija strchr(s,c) je funkcija koja vraca adresu prvog
         pojavljivanja karaktera c u niski s, ako se c pojavljuje u s,
38       a NULL inace.

40       Vrednost promenljive s je zapravo vrednost adrese prvog
         karaktera niske s.
42
         Ako treba da se ispise indeks prvog pojavljivanja, to moze
44       da se uradi tako sto se od adrese koji je vratila funkcija
         strchr oduzme adresa prvog karaktera.
46
         Na primer:
48       s = "koliba"     ==> s je adresa karaktera 'k'
         p = strchr(s, 'l') ==> p je adresa karaktera 'l'
50       |k|o|l|i|b|a|
          ^   ^
52        |   |
          s   p
54       Izraz p-s ima vrednost 2 (jer je rastojanje izmedju
         ove dve adrese 2).
56
         Tip promenljive p je char* jer predstavlja adresu
58       jednog karaktera.

60    char *p = strchr(s, c);
      if (p != NULL)
62      printf("%d\n", p - s);
      else
64      printf("-1\n");
      */
66
      exit(EXIT_SUCCESS);
68  }
```

Rešenje 3.5.7

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

#define MAKS_NISKA 11

/* Funkcija proverava da li je niska t podniska niske s. */
int podniska(char s[], char t[])
{
  int i, j;

  /* Spoljasnja petlja ide redom po niski s. */
  for (i = 0; s[i] != '\0'; i++)
  {
    /* Unutrasnja petlja ide redom po niski t pomocu brojaca j
       i proverava da li se cela niska t poklapa sa delom niske
       s koji pocinje na poziciji i.

       Cim se naidje na situaciju da se karakteri ne poklapaju,
       izlazi se iz unutrasnje petlje. */
    for (j = 0; t[j] != '\0'; j++)
      if (s[i+j] != t[j])
        break;

    /* Ako je unutrasnja petlja dosla do kraja niske t, to
       znaci da su se svi karakteri iz t poklopili sa karakterima
       iz s i t je podniska od s. */
    if (t[j] == '\0')
      return 1;
  }

  return 0;
}


int main()
{
  /* Deklaracija niski s i t. */
  char s[MAKS_NISKA], t[MAKS_NISKA];

  /* Ucitavaju se niske s i t. */
  printf("Unesite nisku s: ");
  scanf("%s", s);
  printf("Unesite nisku t: ");
  scanf("%s", t);

  /* Ispis rezultata. */
  if (podniska(s, t))
    printf("t je podniska niske s.\n");
  else
    printf("t nije podniska niske s.\n");

  /* II nacin:
```

```
     Funkcija strstr(t, s) proverava da li je t podniska od s
     i kao povratnu vrednost vraca adresu prvog pojavljivanja t u s
     ili NULL ukoliko se t ne pojavljuje u s.
     Deklaracija ove funkcije se nalazi u zaglavlju string.h

     char* p = strstr(t, s);
     if(p == NULL)
       printf("t nije podniska od s.\n");
     else
       printf("t je podniska od s.\n");
   */

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.8

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

#define MAKS_LINIJA 101

/* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
   Funkcija ne smesta znak za novi red na kraj linije. */
void ucitaj_liniju(char s[], int n)
{
  int i = 0;
  int c;

  /* Ucitava se karakter po karakter dok se ne unese novi red
     ili oznaka za kraj ulaza ili dok se ne dostigne maksimalan
     broj karaktera. */
  while ((c = getchar()) != '\n' && i < n-1 && c != EOF)
  {
    s[i] = c;
    i++;
  }

  /* Maksimalan broj karaktera za liniju je n-1 jer na kraju
     treba ostaviti i jedno mesto za terminalnu nulu. */
  s[i] = '\0';
}

/* Funkcija uklanja beline sa kraja niske s. */
void skrati(char s[])
{
  int i;
  /* Vrsi se prolazak kroz nisku sa desna na levo i trazi se pozicija
     prvog karaktera koji nije belina.
```

```
35
      Funkcija isspace proverava da li je dati karakter neka od belina
37    (blanko, tab ili novi red) i njena deklaracija se nalazi u
      zaglavlju ctype.h. */
39  for (i = strlen(s) - 1; i >= 0; i--)
      if (!isspace(s[i]))
41      break;

43  /* Nakon izlaska iz petlje, brojac i se nalazi na poziciji prvog
      karaktera sa desne strane koji nije belina. Iz tog razloga se
45    na poziciju i+1 upisuje terminalna nula kao oznaka da se sada
      tu nalazi kraj niske. */
47  s[i + 1] = '\0';
}
49
int main()
51 {
  /* Deklaracija niske. */
53  char s[MAKS_LINIJA];

55  /* Ucitava se cela linija sa ulaza. */
  printf("Unesite nisku:\n");
57  ucitaj_liniju(s, MAKS_LINIJA);

59  /* NAPOMENA:
      Postoji vise nacina za ucitavanje cele linije sa standardnog
61    ulaza koriscenjem funkcija iz standardne c biblioteke. Jedan od
      njih je koriscenjem funkcije gets:
63    gets(s);
      Postoje razlozi zasto ova funkcija nije bezbedna za koriscenje
65    i oni ce biti objasnjeni u kasnijim poglaljima.
      U poglavlju "Datoteke" ce biti predstavljeni i bezbedni nacini
67    da se to uradi koriscenjem nekih drugoh funkcija. */

69  /* Ispis rezultata. */
  printf("Ucitana niska:\n*%s*\n", s);
71  skrati(s);
  printf("Izmenjena niska:\n*%s*\n", s);
73
  exit(EXIT_SUCCESS);
75 }
```

Rešenje 3.5.9

```
#include <stdio.h>
2 #include <stdlib.h>
#include <string.h>
4 #include <ctype.h>

6 #define MAKS_LINIJA 21
```

```c
/* Funkcija uklanja sva slova iz niske s. */
void ukloni_slova(char s[])
{
  int i, j;

  /* Prolazi se kroz nisku s karakter po karakter i vrsi se provera
     da li trenutni karakter treba da se zadrzi. Karakter treba da
     se zadrzi ukoliko nije ni malo ni veliko slovo.

     Brojac j sluzi da pamti gde se upisuje sledeci karakter koji
     treba da se zadrzi i svaki put kada se naidje na takav karaker,
     on se upisuje na poziciju j, a brojac j se uvecava. */
  for(i=0, j=0; s[i]; i++){
    if(!islower(s[i]) && !isupper(s[i]))
    {
      s[j] = s[i];
      j++;
    }
  }

  /* Na kraju se na poziciji j upisuje i terminalna nula, kako bi se
     naznacilo da se kraj niske nalazi nakon poslednjeg zadrzanog
     karaktera. */
  s[j] = '\0';
}

int main()
{
  /* Deklaracija niske. */
  char s[MAKS_LINIJA];

  /* Ucitava se niska s. */
  printf("Unesite nisku:\n");
  scanf("%s", s);

  /* Ispis rezultata. */
  ukloni_slova(s);
  printf("%s\n", s);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.10

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

#define MAKS_LINIJA 101

/* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
```

```
   Funkcija ne smesta znak za novi red na kraj linije. */
void ucitaj_liniju(char s[], int n)
{
  int i = 0, c;

  while ((c = getchar()) != '\n' && i < n-1 && c != EOF)
  {
    s[i] = c;
    i++;
  }
  s[i] = '\0';
}

/* Pomocna funkcija koja proverava da li karakter c1 treba zadrzati
   ako vazi da se iza njega nalazi c2. */
int treba_zadrzati(char c1, char c2)
{
  /* Ako neki od karaktera nije slovo, c1 se ne izbacuje. */
  if(!isalpha(c1) || !isalpha(c2))
    return 1;

  /* Oba karaktera se pretvaraju u veliko slovo kako bi se smanjio
     broj poredjenja. */
  c1 = toupper(c1);
  c2 = toupper(c2);

  /* c1 se zadrzava ako se c2 ne nalazi iza njega u abecedi. */
  return c2 <= c1;
}

/* Funkcija uklanja sva slova za koja vazi da se neposredno
   nakon njih nalazi slovo koje je u abecedi iza njih. */
void ukloni(char s[])
{
  int i, j;
  for(i=0, j=0; s[i]; i++){
    if(treba_zadrzati(s[i], s[i+1]))
    {
      s[j] = s[i];
      j++;
    }
  }
  s[j] = '\0';
}

int main()
{
  /* Deklaracija niske. */
  char s[MAKS_LINIJA];

  /* Ucitava se cela linija sa ulaza. */
  printf("Unesite nisku:\n");
```

```
  ucitaj_liniju(s, MAKS_LINIJA);

  /* Ispis rezultata. */
  ukloni(s);
  printf("%s\n", s);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.11

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

#define MAKS_NISKA 31
#define MAKS_REZULTAT 91

/* Niske se ne kopiraju naredbom dodele. Ukoliko je potrebno da neka
   niska ima isti sadrzaj kao i neka druga niska, moze se koristiti
   funkcija strcpy(t, s) koja kopira karaktere niske s u nisku t
   zajedno za terminirajucom nulom. Deklaracija ove funkcije se
   nalazi u zaglavlju string.h.

   Funkcija strcpy_klon predstavlja jednu implementaciju funkcije
   strcpy. */
void strcpy_klon(char kopija[], char original[])
{
  int i;
  for (i = 0; original[i]; i++)
    kopija[i] = original[i];

  kopija[i] = '\0';
}

int main()
{
  /* Deklaracija niski. */
  char s[MAKS_NISKA];
  char t[MAKS_REZULTAT];

  /* Ucitava se niska. */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Niska s se kopira u nisku t. */
  strcpy_klon(t, s);

  /* Funkcija strcat(s,t) nadovezuje karaktere niske s na kraj
     niske t i novu nisku terminira karakterom '\0'. Deklaracija
     ovde funkcije se nalazi u zaglavlju string.h. */
```

```
41
  /* Niska s se jos dva puta nadovezuje na nisku t. */
43  strcat(t, s);
  strcat(t, s);
45
  /* Ispis rezultata. */
47  printf("%s\n", t);

49  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.12

```
#include <stdio.h>
2 #include <stdlib.h>
#include <string.h>
4
#define MAKS_NISKA 21
6 #define MAKS_N 10
/* Rezultat se dobija nadovezivanjem niske maksimalne duzine
8    MAKS_NISKA-1 i karaktera '-' najvise MAKS_N puta.
   Odavde je maksimalna duzina rezultata:
10   (MAKS_NISKA - 1 + 1) * MAKS_N = MAKS_NISKA*MAKS_N.
   Na ovo treba dodati jos jedan karakter zbog terminalne nule. */
12 #define MAKS_REZULTAT (MAKS_NISKA*MAKS_N + 1)

14 int main()
{
16   /* Deklaracija niski. */
  char s[MAKS_NISKA];
18   char t[MAKS_REZULTAT];
  int i, n;
20
  /* Ucitava se niska. */
22  printf("Unesite nisku: ");
  scanf("%s", s);
24
  /* Ucitava se broj ponavljanja i vrsi se provera ispravnosti
26     ulaza. */
  printf("Unesite broj n: ");
28  scanf("%d", &n);
  if(n <= 0 || n > MAKS_N)
30  {
    printf("Greska: neispravan unos.\n");
32    exit(EXIT_FAILURE);
  }
34
  /* Formira se rezultat. Prvi karakter rezultujuce niske se
36     postavlja na terminalnu nulu. Ovo se radi jer strcat
     funkcionise tako sto krene od pocetka niske, ide do
38     terminalne nule i zatim pocevsi od tog mesta nadovezuje
```

```
     nisku koja je prosledjena kao drugi argument. Na ovaj nacin
     je obezbedjeno da ce prvi poziv funkcije strcat krenuti da
     nadovezuje od pocetka niske t.
     U petlji se na t nadovezuje prvo niska s, a zatim niska "-".
     Ovo se ponavlja n-1 puta jer nakon poslednjeg nadovezivanja
     niske s ne treba da se nadje "-". Iz tog razloga se po
     zavrsetku petlje vrsi jos jedno nadovezivanje niske s, ali ne
     i niske "-". */
  t[0] = '\0';
  for(i=0; i<n-1; i++)
  {
    strcat(t, s);
    strcat(t, "-");
  }
  strcat(t, s);

  /* Ispis rezultata. */
  printf("%s\n", t);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.13

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

#define MAKS_NISKA 21

/* Funkcija kopira prvih n karaktera niske s u nisku t. */
void kopiraj_n(char t[], char s[], int n)
{
  int i;
  /* Kopiranje se vrsi ili dok se ne dodje do terminalne nule u s
     ili dok se ne prekopira n karaktera. */
  for (i = 0; i < n && s[i] != '\0'; i++) {
    t[i] = s[i];
  }

  /* Na kraju rezultujuce niske se upisuje terminalna nula. */
  t[i] = '\0';
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n;
  char s[MAKS_NISKA], t[MAKS_NISKA];

  /* Ucitava se niska. */
```

```
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Ucitava se broj n i vrsi se provera ispravnosti
     ulaza. */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if (n < 0 || n > MAKS_NISKA-1) {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Formira se rezultat. */
  kopiraj_n(t, s, n);

  /* II nacin:
     Koriscenjem funkcije strncpy(t, s, n), cija se deklaracija
     nalazi u zaglavlju string.h, kopira najvise n karaktera niske
     s u nisku t.

     strncpy(t,s,n);
     */

  /* Ispis rezultata. */
  printf("%s\n", t);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.14

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

/* Duzina niske koja se ucitava, bez terminalne nule. */
#define MAKS_DUZINA 20

/* Duzine originalne i rezultujuce niske. */
#define MAKS_NISKA (MAKS_DUZINA + 1)
#define MAKS_REZULTAT (2 * MAKS_DUZINA + 1)

/* Funkcija formira nisku t od niske s dupliranjem svakog
   karaktera. Npr. abc postaje aabbcc. */
void dupliranje(char t[], char s[])
{
  int i, j;

  /* Brojac i oznacava tekucu poziciju u niski s, a
     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];
  }

  /* Upisuje se terminalna nula na kraj rezultujuce niske. */
  t[j] = '\0';
}

int main()
{
  /* Deklaracija niski. */
  char s[MAKS_NISKA], t[MAKS_REZULTAT];

  /* Ucitava se niska. */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Poziva se funkcija za dupliranje. */
  dupliranje(t, s);

  /* Ispis rezultata. */
  printf("%s\n", t);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.15

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

#define MAKS_NISKA 10

/* Funkcija formiraj_broj na osnovu niske b formira ceo broj
   ciji je to zapis.

   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 broj = 0, znak;

  /* Odredjivanje znaka broja i pozicije prve cifre. */
  if(b[0] == '-')
  {
    znak = -1;
    i = 1;
```

```
  }
  else if(b[0] == '+')
  {
    znak = 1;
    i = 1;
  }
  else
  {
    i = 0;
    znak = 1;
  }

  /* Prolazak kroz cifre broja i racunanje vrednosti broja
     koriscenjem Hornerove seme. Vrednost trenutne cifre se
     dobija kada se od trenutnog karaktera (b[i]) oduzme
     karakter '0'.
     Ako se naidje na karakter koji nije cifra, petlja se
     prekida. Na primer za b="123abc", rezultat treba da
     bude 123. */
  for (; b[i] != '\0'; i++)
  {
    if(isdigit(b[i]))
      broj = broj * 10 + (b[i] - '0');
    else
      break;
  }

  return broj*znak;
}

int main()
{
  /* Deklaracija niske. */
  char s[MAKS_NISKA];

  /* Broj se ucitava kao niska. */
  scanf("%s", s);

  /* Ispis rezultata. */
  printf("%d\n", formiraj_broj(s));

  /* II nacin:
     Koriscenjem funkcije atoi. Deklaracija ove funkcije se nalazi
     u zaglavlju stdlib.h.

   printf("%d\n", atoi(s)); */

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.16

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

#define MAKS_NISKA 10

/* Funkcija racuna broj cifara broja n. */
int broj_cifara(int n)
{
  int i = 0;
  do
  {
    i++;
    n/=10;
  }while(n);

  return i;
}

/* Funkcija od prosledjenog broja formira nisku. */
void broj_u_nisku(int broj, char s[])
{
  int n, cifra, i;

  /* Promenljiva n cuva informaciju o duzini niske. Duzina niske
     odgovara broju cifara prosledjenog broja. Ukoliko je broj
     negativan, onda se duzina uvecava za 1 i na prvo mesto se
     upisuje znak '-'. */
  n = broj_cifara(broj);
  if(broj < 0)
  {
    s[0] = '-';
    n++;
  }

  /* U nastavku se radi sa apsolutnom vrednoscu broja. */
  broj = abs(broj);

  /* Cifre broja se upisuju u nisku s sa desna na levo. */
  s[n] = '\0';
  i = n-1;
  do
  {
    /* Karakter koji odgovara trenutnoj cifri se dobija izrazom
       '0' + cifra. Na primer, '0' + 5 je '5' jer se karakter '5'
       nalazi 5 mesta nakon karaktera '0' u ASCII tablici. */
    cifra = broj % 10;
    broj = broj / 10;
    s[i] = '0' + cifra;
    i--;
  } while(broj);
}
```

```
52
int main()
54 {
  /* Deklaracija broja i niske. */
56  int n;
  char s[MAKS_NISKA];
58
  /* Ucitava se broj. */
60  printf("Unesite ceo broj: ");
  scanf("%d", &n);
62
  /* Formira se niska. */
64  broj_u_nisku(n, s);

66  /* Ispis rezultata. */
  printf("%s\n", s);
68
  exit(EXIT_SUCCESS);
70 }
```

Rešenje 3.5.17

```
1 #include <stdio.h>
#include <stdlib.h>
3 #include <ctype.h>

5 #define MAKS_NISKA 8

7 /* Funkcija proverava da li je prosledjeni karakter ispravna
     heksadekadna cifra. */
9 int heksa_cifra(char c)
{
11  c = toupper(c);

13  /* Cifra je ispravna ako je cifra ili ako je neko od slova:
       A, B, C, D, E ili F. */
15  return isdigit(c) || (c >= 'A' && c <= 'F');
}
17
/* Funkcija proverava da li prosledjena niska s predstavlja
19    ispravan heksadekadni broj. */
int heksadekadni_broj(char s[])
21 {
  int i;
23
  /* Svaki heksadekasni broj pocinje sa 0x ili 0X. */
25  if (s[0] != '0' || toupper(s[1]) != 'X')
    return 0;
27
  /* Za svaki karakter niske s se proverava da li predstavlja
29     ispravnu heksadekadnu cifru. Ako se naidje na neku cifru koja
```

```
     ne zadovoljava taj uslov, onda se kao povratna vrednost
     vraca nula. */
  for (i = 2; s[i]; i++)
    if (!heksa_cifra(s[i]))
      return 0;

  /* Ako su sve cifre isravne heksadekadne cifre, onda je i s
     ispravan heksadekadni broj i funkcija vraca jedninicu. */
  return 1;
}

int main()
{
  /* Deklaracija niske. */
  char s[MAKS_NISKA];

  /* Ucitava se niska. */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Ispis rezultata. */
  if (heksadekadni_broj(s))
    printf("Korektan heksadekadni broj.\n");
  else
    printf("Nekorektan heksadekadni broj.\n");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.18

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

#define MAKS_NISKA 8

/* Funkcija racuna dekadnu vrednost jedne heksadekadne cifre.
   Ako je c broj, vrednost se dobija oduzimanjem '0'.
   Ako je c slovo, vrednost se dobija oduzimanjem 'A' i dodavanjem
   10 (npr. vrednost karaktera 'B' je 10 + 'B' - 'A' = 11). */
int vrednost_heksa_cifre(char c)
{
  if(isdigit(c))
    return c - '0';
  else
    return 10 + toupper(c) - 'A';
}

/* Funkcija racuna dekadnu vrednost heksadekadnog broja. */
```

```c
int dekadna_vrednost(char s[])
{
  int i, tezina_pozicije = 1, rezultat = 0;
  int n = strlen(s);

  /* Vrsi se prolazak kroz nisku sa desna na levo. Heksadekadna
     cifra najvece tezine se nalazi na poziciji n-1, a ona najvece
     tezine se nalazi na poziciji 2 (jer su prva dva karaktera 0x).

     U svakoj iteraciji, na rezultat se dodaje vrednost tekuce
     cifre, pomnozena sa vrednoscu tezine njene pozicije.
     Na primer, za s = "0x1a8e", n=6
     i = 5, rezultat += vrednost('e')*1 => rezultat += 11*1
     i = 4, rezultat += vrednost('8')*16 => rezultat += 8*16
     i = 3, rezultat += vrednost('a')*256 => rezultat += 10*256
     i = 2, rezultat += vrednost('1')*4096 => rezultat += 1*4096 */
  for (i = n - 1; i >= 2; i--)
  {
    rezultat += tezina_pozicije * vrednost_heksa_cifre(s[i]);
    tezina_pozicije *= 16;
  }

  return rezultat;
}

int main()
{
  /* Deklaracija niske. */
  char s[MAKS_NISKA];

  /* Ucitava se niska. */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Ispis rezultata. */
  printf("%d\n", dekadna_vrednost(s));

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.19

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

#define MAKS_LINIJA 81

/* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
   Funkcija ne smesta znak za novi red na kraj linije. */
int ucitaj_liniju(char s[], int n)
```

```
{
  int i = 0;
  int c;

  /* Ucitava se karakter po karakter dok se ne unese novi red
     ili oznaka za kraj ulaza ili dok se ne dostigne maksimalan
     broj karaktera. */
  while ((c = getchar()) != '\n' && i < n-1 && c != EOF)
  {
    s[i] = c;
    i++;
  }

  /* Maksimalan broj karaktera za liniju je n-1 jer na kraju
     treba ostaviti i jedno mesto za terminalnu nulu. */
  s[i] = '\0';

  return i;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  char linija[MAKS_LINIJA], najduza_linija[MAKS_LINIJA];
  int duzina_najduze = 0, duzina;

  /* U petlji se ucitavaju linije sve dok se ne unese prazna linija.
     Ukoliko se unese linija koja je duza od trenutno najduze,
     vrsi se azuriranje duzine najduze linije, kao i same linije. */
  while ((duzina = ucitaj_liniju(linija, MAKS_LINIJA)) > 0)
  {
    if (duzina_najduze < duzina)
    {
      duzina_najduze = duzina;
      strcpy(najduza_linija, linija);
    }
  }

  /* Ispis rezultata. */
  if(duzina_najduze == 0)
    printf("Nije uneta nijedna linija.\n");
  else
    printf("%s\n%d\n", najduza_linija, duzina_najduze);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.20

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

```c
#include <string.h>
#include <ctype.h>

#define MAKS_RECENICA 81

/* Funkcija ucitava recenicu maksimalne duzine n. */
int ucitaj_recenicu(char s[], int n)
{
  int i = 0, c;

  /* Preskacu se beline sa pocetka ako ih ima. Po zavrsetku ove
     petlje u c se nalazi prvi sledeci karakter koji nije belina. */
  do{
    c = getchar();
  } while(isspace(c));

  /* Ako je taj karakter EOF, zavrsava se ucitavanje. */
  if(c == EOF)
    return 0;

  /* U nisku se smesta karakter, prelazi se na sledeci karakter i
     postupak se ponavlja sve dok se ne unese tacka, EOF ili dok
     se ne popuni maksimalan broj karaktera koje recenica moze
     da sadrzi. */
  do{
    s[i] = c;
    i++;
    c = getchar();
  } while (c != '.' && i < n-2 && c != EOF);

  /* Ako je poslednji uneti karakter EOF, zavrsava se ucitavanje. */
  if(c == EOF)
    return 0;

  /* Na kraju svake recenice stoji tacka za kojom sledi '\0'. */
  s[i] = '.';
  s[i+1] = '\0';

  return i+1;
}

/* Funkcija prebrojava mala i velika slova. */
void prebroj(char s[] , int* broj_malih, int* broj_velikih)
{
  int i, mala = 0, velika = 0;

  for(i=0; s[i]; i++)
  {
    if(islower(s[i]))
      mala++;
    else if(isupper(s[i]))
      velika++;
```

```
  }

  *broj_malih = mala;
  *broj_velikih = velika;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  char recenica[MAKS_RECENICA];
  char rezultujuca_recenica[MAKS_RECENICA];
  int najveca_razlika = -1, trenutna_razlika;
  int mala, velika;
  int ucitana_bar_jedna = 0;

  /* U petlji se ucitavaju recenice sve dok se ne unese EOF. */
  while (ucitaj_recenicu(recenica, MAKS_RECENICA) > 0) {

    /* Prebrojavaju se mala i velika slova. */
    prebroj(recenica, &mala, &velika);

    /* Racuna se njihova apsolutna razlika. */
    trenutna_razlika = abs(mala-velika);

    /* Ako je razlika veca od trenutno najvece, azurira se vrednost
       najvece razlike i pamti se trenutna recenica. */
    if(trenutna_razlika > najveca_razlika){
      najveca_razlika = trenutna_razlika;
      strcpy(rezultujuca_recenica, recenica);
    }

    /* Indikator koji oznacava da se petlja bar jednom izvrsila, tj.
       da korisnik nije odmah zadao EOF. */
    ucitana_bar_jedna = 1;
  }

  /* Ispis rezultata. */
  if(ucitana_bar_jedna)
    printf("%s\n", rezultujuca_recenica);
  else
    printf("Nije uneta nijedna recenica. ");

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.21

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

#define MAKS_NISKA 21
```

```c
/* Funkcija vraca adresu prvog pojavljivanja karaktera c u niski s
   ili NULL ukoliko se c ne pojavljuje u s.

   Trazeni rezultat moze se dobiti koriscenjem funkcije strchr
   cija se deklaracija nalazi u zaglavlju string.h.
   Funkcija strchr_klon predstavlja jednu mogucu implementaciju
   ove funkcije. */
char* strchr_klon(char s[], char c)
{
  int i;

  /* Za svaki karakter se proverava da li je jednak karakteru c.
     Ako se naidje na takav karakter, kao povratna vrednost funkcije
     se vraca njegova adresa (&s[i]). */
  for(i=0; s[i]; i++)
    if(s[i] == c)
      return &s[i];

  /* Ako je petlja zavrsena, znaci da nije pronadjen karakter koji
     je jednak karakteru c i kao povratna vrednost funkcije se vraca
     NULL pokazivac kao oznaka da se c ne nalazi u s. */
  return NULL;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  char s[MAKS_NISKA];
  char c;

  /* Ucitava se niska s. */
  printf("Unesite nisku s: ");
  scanf("%s", s);

  /* Preskace se novi red koji je unet nakon niske s i
     ucitava se karakter c. */
  getchar();
  printf("Unesite karakter c: ");
  scanf("%c", &c);

  /* Racunanje i ispis rezultata. */
  char* p = strchr_klon(s, c);
  if(p == NULL)
    printf("Pozicija: -1\n");
  else
    printf("Pozicija: %ld\n", p-s);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.22

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

#define MAKS_NISKA 21

/* Funkcija racuna duzinu prefiksa niske t koji se moze
   zapisati pomocu karaktera niske s.
   Na primer, t="programiranje", s="grupacija", rezultat je 2 jer
   niska s sadrzi prva dva karaktera niske t, ali ne i treci.

   Trazeni rezultat moze se dobiti koriscenjem funkcije strspn
   cija se deklaracija nalazi u zaglavlju string.h.
   Funkcija strspn_klon predstavlja jednu mogucu implementaciju
   ove funkcije. */
int strspn_klon(char t[], char s[])
{
  int i, brojac = 0;

  /* Ide se redom po karakterima niske t i za svaki karakter se
     vrsi provera da li se on nalazi u zapisu niske s. Za ovo se
     koristi funkcija strchr. Ako se nalazi, uvecava se brojac,
     a ako se ne nalazi, prekida se petlja. */
  for(i=0; t[i]; i++)
  {
    if(strchr(s, t[i]) != NULL)
      brojac++;
    else
      break;
  }

  return brojac;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  char s[MAKS_NISKA];
  char t[MAKS_NISKA];

  /* Ucitavaju se niske. */
  printf("Unesite nisku t: ");
  scanf("%s", t);
  printf("Unesite nisku s: ");
  scanf("%s", s);

  /* Racunanje i ispis rezultata. */
  printf("%d\n", strspn_klon(t,s));

  exit(EXIT_SUCCESS);
```

```
}
```

Rešenje 3.5.23

Rešenje ovog zadatka se svodi na rešenje zadatka 3.5.22, uz razliku da se ovde prebrojavaju karakteri koji se ne nalaze u zapisu niske s.

Rešenje 3.5.24

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

#define MAKS_NISKA 101

/* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
   Funkcija ne smesta znak za novi red na kraj linije. */
void ucitaj_liniju(char s[], int n)
{
  int i = 0, c;

  while ((c = getchar()) != '\n' && i < n-1 && c != EOF) {
    s[i] = c;
    i++;
  }

  s[i] = '\0';
}

/* Funkcija vraca pokazivac na prvo pojavljivanje niske
   t u okviru niske s ili NULL ukoliko se t ne nalazi u s.

   Trazeni rezultat moze se dobiti koriscenjem funkcije strstr
   cija se deklaracija nalazi u zaglavlju string.h.
   Funkcija strstr_klon predstavlja jednu mogucu implementaciju
   ove funkcije. */
char* strstr_klon(char s[], char t[])
{
  int i, j;

  /* Spoljasnja petlja ide redom po niski s. */
  for (i = 0; s[i] != '\0'; i++) {
    /* Unutrasnja petlja ide redom po niski t pomocu brojaca j
       i proverava da li se cela niska t poklapa sa delom niske
       s koji pocinje na poziciji i.

       Cim se naidje na situaciju da se karakteri ne poklapaju,
       izlazi se iz unutrasnje petlje. */
    for (j = 0; t[j] != '\0'; j++)
      if (s[i+j] != t[j])
```

```
        break;

    /* Ako je unutrasnja petlja dosla do kraja niske t, to
       znaci da su se svi karakteri iz t poklopili sa karakterima
       iz s i t je podniska od s.
       Kao povratna vrednost se vraca mesto gde t pocinje u s. */
    if (t[j] == '\0')
      return &s[i];
  }

  return NULL;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  char linija[MAKS_NISKA];
  int i, bar_jedna = 0;

  /* Ucitavanje linija i ispis rednih brojeva linija
     koje sadrze rec "program". */
  for(i=1; i<=5; i++)
  {
    ucitaj_liniju(linija, MAKS_NISKA);
    if(strstr_klon(linija, "program") != NULL){
      printf("%d ", i);
      bar_jedna = 1;
    }
    /* II nacin: koriscenjem funkcije strstr cija se deklaracija
       nalazi u zaglavlju string.h:
      if(strstr(linija, "program") != NULL){
        printf("%d ", i);
        bar_jedna = 1;
      } */
  }
  printf("\n");

  /* Ako indikator bar_jedna i dalje ima vrednost 0, znaci da
     nije uneta nijedna linija koja sadrzi rec "program". */
  if(!bar_jedna)
  {
    printf("Nijedna linija ne sadrzi nisku program.\n");
  }

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.25

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

```
3
#define MAKS_NISKA 21
5
/* Funkcija poredi dve niske i vraca nulu ukoliko su jednake,
7    nesto pozitivno ukoliko je niska s1 leksikografski iza s2,
   a neku negativnu vrednost inace.
9
   Trazeni rezultat moze se dobiti koriscenjem funkcije strcmp
11   cija se deklaracija nalazi u zaglavlju string.h.
   Funkcija strcmp_klon predstavlja jednu mogucu implementaciju
13   ove funkcije. */
int strcmp_klon(char s1[], char s2[])
15 {
  int i;
17
  /* Prolazi se kroz obe niske dok god se odgovarajuci karakteri
19     poklapaju. Ako se u ovom prolasku desi da je petlja dosla
     do kraja obe niske, onda su one jednake i kao povratna vrednost
21     funkcije se vraca 0. */
  for (i = 0; s1[i] == s2[i]; i++)
23    if (s1[i] == '\0')
       return 0;
25
  /* Ako niske nisu jednake, znaci da je brojac i stao na prvom
27     mestu gde se niske s1 i s2 razlikuju. Posto funkcija treba da
     vrati pozitivnu vrednost ako je niska s1 laksikografski iza
29     s2, a negativnu u suprotnom, ovo moze biti realizovano vracanjem
     razlike ASCII kodova.
31     Na primer: s1 = "pero", s2 = "program"
     Nakon petlje, brojac i ima vrednost 1 (jer je tu prva razlika).
33     Kao povratna vrednost se vraca s1[1] - s2[1] = 'e' - 'r' = -13
     sto kao negativna vrednost govori da se s1 nalazi
35     leksikografski ispred s2. */
  return s1[i] - s2[i];
37 }

39 int main()
{
41   /* Deklaracije potrebnih promenljivih. */
  char s[MAKS_NISKA];
43   char t[MAKS_NISKA];
  int rezultat;
45
  /* Ucitavaju se niske s i t. */
47   printf("Unesite nisku s: ");
  scanf("%s", s);
49   printf("Unesite nisku t: ");
  scanf("%s", t);
51
  /* Vrsi se poredjenje niski i ispisuje se rezultat. */
53   rezultat = strcmp_klon(s,t);
```

```
55    /* II nacin: Koriscenjem funkcije strcmp cija se deklaracija
         nalazi u zaglavlju string.g:
57    rezultat = strcmp(s, t); */

59    if(rezultat == 0)
        printf("%s\n", s);
61    else if(rezultat < 0)
        printf("%s\n%s\n", s, t);
63    else
        printf("%s\n%s\n", t, s);
65
      exit(EXIT_SUCCESS);
67  }
```

Rešenje 3.5.26

```
1   #include <stdio.h>
    #include <stdlib.h>
3   #include <string.h>

5   #define MAKS_NISKA 21

7   /* Funkcija obrce nisku s. */
    void obrni(char s[])
9   {
      int i, j;
11    int n = strlen(s);
      char c;
13
      /* Brojac i ide od prvog karaktera niske s, a brojac j
15       od poslednjeg i dok god se ne sretnu, vrsi se zamena
         karaktera koji se nalaze na njihovim pozicijama. */
17    for(i=0, j=n-1; i<j; i++,j--)
      {
19        c = s[i];
          s[i] = s[j];
21        s[j] = c;
      }
23  }

25  int main()
    {
27    /* Deklaracija niske. */
      char s[MAKS_NISKA];
29
      /* Ucitava se niska. */
31    printf("Unesite nisku: ");
      scanf("%s", s);
33
      /* Racunanje i ispis rezultata. */
35    obrni(s);
```

```
  printf("%s\n", s);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.27

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

#define MAKS_NISKA 21

/* Funkcija rotira nisku za jedno mesto ulevo. */
void rotiraj1(char s[], int n)
{
  int i;
  /* Pamti se prvi karakter. */
  char prvi = s[0];

  /* Svaki sledeci karakter se pomera za jedno mesto ulevo. */
  for(i=0; i<n-1; i++)
    s[i] = s[i+1];

  /* Prvi karakter se upisuje na kraj niske. */
  s[n-1] = prvi;
}

/* Funkcija rotira nisku s za k mesta ulevo. */
void rotiraj(char s[], int k){
  int i;
  int n = strlen(s);

  for(i=0; i<k; i++)
    rotiraj1(s, n);
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  char s[MAKS_NISKA];
  int k;

  /* Ucitavaju se niska i vrednost k. */
  printf("Unesite nisku: ");
  scanf("%s", s);
  printf("Unesite k: ");
  scanf("%d", &k);

  /* Vrsi se provera ispravnosti ulaza. */
  if(k < 0)
```

```
  {
46    printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
48  }

50  /* Racunanje i ispis rezultata. */
    rotiraj(s, k);
52  printf("%s\n", s);

54  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.28

```
#include <stdio.h>
2 #include <stdlib.h>
  #include <string.h>
4 #include <ctype.h>

6 #define MAKS_NISKA 21

8 /* Funkcija svako slovo niske s menja sa slovom
     koje se u ASCII tablici nalazi neposredno iza njega.
10   Specijalan slucaj je slovo z koje treba da se zameni
     sa slovom a. Ostali karakteri ostaju nepromenjeni. */
12 void sifruj(char s[])
{
14   int i;

16   for(i=0; s[i]; i++)
     {
18     if(isalpha(s[i]))
       {
20       if(s[i] == 'z')
           s[i] = 'a';
22       else if(s[i] == 'Z')
           s[i] = 'A';
24       else
           s[i] = s[i] + 1;
26     }
   }
28 }

30 int main()
{
32   /* Deklaracija niske. */
     char s[MAKS_NISKA];
34
     /* Ucitava se niska. */
36   printf("Unesite nisku: ");
     scanf("%s", s);
```

```
38
   /* Racunanje i ispis rezultata. */
40 sifruj(s);
   printf("%s\n", s);
42
   exit(EXIT_SUCCESS);
44 }
```

Rešenje 3.5.29

```
#include <stdio.h>
2 #include <stdlib.h>
#include <ctype.h>
4
#define MAKS_DUZINA 20
6 #define MAKS_NISKA (MAKS_DUZINA + 1)
#define MAKS_REZULTAT (3*MAKS_DUZINA + 1)
8
/* Pomocna funkcija koja za prosledjeno slovo
10    vraca slovo koje ide posle njega. */
char sledeci(char c)
12 {
  if(c == 'z')
14   return 'a';

16 if(c == 'Z')
    return 'A';
18
  return c+1;
20 }

22 /* Funkcija od niske s formira rezultujucu nisku koja se dobija
   na sledeci nacin:
24  1. ako je s[i] slovo, onda se u rezultujucu nisku upisuju naredna
   tri slova alfabeta (kada se stigne do kraja alfabeta, ide se u
26  krug, tj. nakon slova z sledi slovo a)
   2. ako s[i] nije slovo, samo se s[i] prepisuje u rezultat. */
28 void sifruj(char s[], char rezultat[])
{
30 int i, j;

32 /* Brojac i se koristi za nisku s, a brojac j za
     rezultujucu nisku. */
34 for(i=0, j=0; s[i]; i++)
  {
36   if(isalpha(s[i]))
    {
38     /* Ako je s[i] slovo, onda se u rezultat upisuju 3 slova koja
          slede nakon njega. */
40     rezultat[j] = sledeci(s[i]);
       rezultat[j+1] = sledeci(rezultat[j]);
```

```
      rezultat[j+2] = sledeci(rezultat[j+1]);
      j += 3;
    }
    else
    {
      /* Ako s[i] nije slovo, onda se samo prepisuje u rezultat. */
      rezultat[j] = s[i];
      j++;
    }
  }

  /* Na kraj rezultata se dopisuje terminalna nula. */
  rezultat[j] = '\0';
}

int main()
{
  /* Deklaracija niske. */
  char s[MAKS_NISKA];
  char rezultat[MAKS_REZULTAT];

  /* Ucitava se niska. */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Racunanje i ispis rezultata. */
  sifruj(s, rezultat);
  printf("%s\n", rezultat);

  exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.30

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

#define MAKS_DUZINA 20
#define MAKS_NISKA (MAKS_DUZINA + 1)
#define MAKS_REZULTAT (2*MAKS_DUZINA + 1)

/* Funkcija od niske s formira rezultujucu nisku na sledeci nacin:
   1. Svi karakteri niske s koji su jednaki c1 se dupliraju.
   2. Svi karakteri niske s koji su jednaki c2 se brisu.
   3. Ostali karakteri se samo prepisuju. */
void formiraj(char s[], char rezultat[], char c1, char c2)
{
  int i, j;

  /* Brojac i se koristi za nisku s, a brojac j za
```

```
18      rezultujucu nisku. */
   for(i=0, j=0; s[i]; i++)
20 {
     if(s[i] == c1)
22   {
       /* Ako je s[i] jednako c1, duplira se u rezultatu. */
24     rezultat[j] = s[i];
       rezultat[j+1] = s[i];
26     j += 2;
     }
28   else if(s[i] != c2)
     {
30     /* Ako s[i] razlicito od c2, upisuje se u rezultat. */
       rezultat[j] = s[i];
32     j++;
     }
34 }

36 /* Na kraj rezultata se dopisuje terminalna nula. */
   rezultat[j] = '\0';
38 }

40 int main()
{
42 /* Deklaracija potrebnih promenljivih. */
   char s[MAKS_NISKA];
44 char rezultat[MAKS_REZULTAT];
   char c1, c2;
46
   /* Ucitavaju se niska i karakteri. */
48 printf("Unesite nisku: ");
   scanf("%s", s);
50 getchar();
   printf("Unesite prvi karakter: ");
52 scanf("%c", &c1);
   getchar();
54 printf("Unesite drugi karakter: ");
   scanf("%c", &c2);
56
   /* Vrsi se provera ispravnosti ulaza. */
58 if(c1 == c2)
   {
60   printf("Greska: neispravan unos.\n");
     exit(EXIT_FAILURE);
62 }

64 /* Racunanje i ispis rezultata. */
   formiraj(s, rezultat, c1, c2);
66 printf("%s\n", rezultat);

68 exit(EXIT_SUCCESS);
}
```

Rešenje 3.5.31

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

#define MAKS_NISKA 20

/* Pomocna funkcija koja racuna dekadnu vrednost prosledjenog
   karaktera ('1' ima vrednost 1, 'C' ima vrednost 12). */
unsigned vrednost_cifre(char c)
{
  c = toupper(c);
  if(isdigit(c))
    return c - '0';
  else
    return c - 'A' + 10;
}

/* Funkcija racuna dekadnu vrednost neonacenog broja
   zapisanog u datoj osnovi. */
unsigned int u_dekadni_sistem(char broj[], unsigned int osnova)
{
  int i, n = strlen(broj);
  int rezultat = 0;
  int tezina_pozicije = 1;

  for(i=n-1; i>=0; i--)
  {
    rezultat += vrednost_cifre(broj[i])*tezina_pozicije;
    tezina_pozicije *= osnova;
  }

  return rezultat;
}

/* Funkcija obrce nisku s. */
void obrni(char s[])
{
  int i, j;
  int n = strlen(s);
  char c;

  for(i=0, j=n-1; i<j; i++,j--){
      c = s[i];
      s[i] = s[j];
      s[j] = c;
  }
}

/* Pomocna funkcija koja dekadnu vrednost cifre pretvara u
```

```
   odgovarajuci karakter (12 u 'C', 5 u '5', itd.). */
char ostatak_u_char(int ostatak)
{
  if(ostatak < 10)
    return '0' + ostatak;
  else
    return 'A' + ostatak - 10;
}

/* Funkcija datu dekadnu vrednost broja prebacuje u broj u
   datoj osnovi. */
void iz_dekadnog_sistema(unsigned int broj, unsigned int osnova,
                         char rezultat[])
{
  int i = 0;
  int ostatak;

  do{
    ostatak = broj % osnova;
    broj = broj / osnova;
    rezultat[i] = ostatak_u_char(ostatak);
    i++;
  }while(broj);

  rezultat[i] = '\0';

  obrni(rezultat);
}

int main()
{
  char broj[MAKS_NISKA];
  char broj2[MAKS_NISKA];
  unsigned int osnova1, osnova2;

  printf("Unesite n, o1 i o2: ");
  scanf("%s%u%u", broj, &osnova1, &osnova2);

  unsigned dekadna_vrednost = u_dekadni_sistem(broj, osnova1);
  printf("Dekadna vrednost broja %s: %u\n", broj, dekadna_vrednost);

  iz_dekadnog_sistema(dekadna_vrednost, osnova2, broj2);
  printf("Vrednost broja %u u osnovi %u: %s\n", dekadna_vrednost,
    osnova2, broj2);

  exit(EXIT_SUCCESS);
}
```

3.7 Višedimenzioni nizovi

Zadatak 3.7.1 Napisati program koji učitava i zatim ispisuje vrednosti učitane matrice. Pretpostaviti da je maksimalna dimenzija matrice 50×50, a da se sa ulaza najpre učitavaju dva cela broja m i n, a potom i elementi matrice celih brojeva dimenzije $m \times n$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Matrica je:
1 2 3 4
5 6 7 8
9 10 11 12
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Matrica je:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice:
500 3
Greska: neispravan unos.
```

[Rešenje 3.7.1]

Zadatak 3.7.2 Napisati program koji za učitanu celobrojnu matricu dimenzije $m \times n$ izračunava njenu Ekulidsku normu. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. Uputstvo: *Euklidska norma matrice je kvadratni koren sume kvadrata svih elemenata matrice.*

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Euklidska norma je: 25.495
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Euklidska norma je: 15.875
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice:
500 3
Greska: neispravan unos.
```

[Rešenje 3.7.2]

Zadatak 3.7.3 Napisati funkcije za rad sa celobrojnim matricama:

(a) `void ucitaj(int a[][MAKS], int n, int m)` kojom se učitavaju vrednosti matrice celih brojeva a dimenzije $m \times n$,

(b) `void ispisi(int a[][MAKS], int n, int m)` kojom se ispisuju vrednosti matrice a dimenzije $m \times n$.

Napisati program koji najpre učitava, a zatim i ispisuje vrednosti učitane matrice. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 3 4
 Unesite elemente matrice:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Matrica je:
 1 2 3 4
 5 6 7 8
 9 10 11 12
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 5 3
 Unesite elemente matrice:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Matrica je:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 500 3
 Greska: neispravan unos.
```

[Rešenje 3.7.3]

Zadatak 3.7.4 Napisati funkciju `void transponovana(int a[][MAKS], int m, int n, int b[][MAKS])` koja određuje matricu b koja je dobijena transponovanjem matrice a. Napisati program koji za učitanu matricu celih brojeva[2] ispisuje odgvarajuću transponovanu matricu. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

[2] Pod pojmom *učitati matricu* ili *za datu matricu* uvek se podrazumeva da se prvo unose dimenzije matrice, a potom i sama matrica.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 3 4
 Unesite elemente matrice:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Transponovana matrica je:
 1 5 9
 2 6 10
 3 7 11
 4 8 12
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 5 3
 Unesite elemente matrice:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Transponovana matrica je:
 1 5 7 1 0
 1 0 8 2 1
 2 2 9 4 1
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 500 3
 Greska: neispravan unos.
```

[Rešenje 3.7.4]

Zadatak 3.7.5 Napisati funkciju `void razmeni(int a[][MAKS], int m, int n, int k, int t)` u kojoj se razmenjuju elemeti k–te i t–te vrste matrice a dimezije $m \times n$. Napisati program koji za učitanu matricu celih brojeva, i dva cela broja k i t ispisuje matricu dobijenu razmenjivanjem k–te i t–te vrste ulazne matrice. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 3 4
 Unesite elemente matrice:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Unesite indekse vrsta:
 0 2
 9 10 11 12
 5 6 7 8
 1 2 3 4
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 5 3
 Unesite elemente matrice:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Unesite indekse vrsta:
 1 3
 1 1 2
 1 2 4
 7 8 9
 5 0 2
 0 1 1
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 5 3
 Unesite elemente matrice:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Unesite indekse vrsta:
 -1 50
 Greska: neispravan unos.
```

[Rešenje 3.7.5]

Zadatak 3.7.6 Napisati program koji za učitanu matricu celih brojeva ispisuje indekse onih elemenata matrice koji su jednaki zbiru svih svojih susednih

```
- - - - - - s b s -
- s s s - - s s s -
- s a s - - - - - -
- s s s - - - - - -
- - - - - - - - s s
- - - - - - - - s c
```

Slika 3.1: *Susedni elementi u matrici.*

elemenata. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Uputstvo: *Broj susednih elemenata matrice zavisi od položaja elementa u matrici. Na slici 3.1 su slovom* `s` *obeleženi susedni elementi matrice za elemente* `a`*,* `b` *i* `c`*.*

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 4 5
 Unesite elemente matrice:
 1 1 2 1 3
 0 8 1 9 0
 1 1 1 0 0
 0 3 0 2 2
 Indeksi elemenata koji su
 jednaki zbiru suseda su:
 1 1
 3 1
 3 4
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 3 4
 Unesite elemente matrice:
 7 10 12 20
 -1 -3 1 7
 0 -47 2 0
 Indeksi elemenata koji su
 jednaki zbiru suseda su:
 0 3
 1 2
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice:
 5 -3
 Greska: neispravan unos.
```

[Rešenje 3.7.6]

Zadatak 3.7.7 Napisati funkciju koja formira niz $b_0, b_1, \ldots, b_n$ od matrice tako što element niza b_i izračunava kao srednju vrednost elemenata i-te vrste matrice. Napisati program koji za učitanu matricu celih brojeva ispisuje dobijeni niz. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice:
 4 5
 Unesite elemente matrice:
 1 1 2 1 3
 0 8 1 9 0
 1 1 1 0 0
 0 3 0 2 2
 Dobijeni niz je:
 1.6 3.6 0.6 1.4
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice:
 3 4
 Unesite elemente matrice:
 7 10 12 20
 -1 -3 1 7
 0 -47 2 0
 Dobijeni niz je:
 12.25 1 -11.25
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice:
 51 13
 Greska: neispravan unos.
```

[Rešenje 3.7.7]

Zadatak 3.7.8 Relacija se može predstaviti kvadratnom matricom nula i jedinica na sledeći način: element i je u relaciji sa elementom j ukoliko se u preseku i–te vrste i j–te kolone nalazi jedinica, a nije u relaciji ukoliko se tu nalazi nula. Napisati funkcije:

(a) `int refleksivna(int a[][MAKS], int n)` kojom se za relaciju zadatom matricom a dimenzije $n \times n$ ispituje da li je refleksivna;

(b) `int simetricna(int a[][MAKS], int n)` kojom se za relaciju zadatom matricom a dimenzije $n \times n$ ispituje da li je simetrična;

(c) `int tranzitivna(int a[][MAKS], int n)` kojom se za relaciju zadatom matricom a ispituje dimenzije $n \times n$ da li je tranzitivna;

(d) `int ekvivalencija(int a[][MAKS], int n)` kojom se za relaciju koja je zadata matricom a dimenzije $n \times n$ ispituje da li je relacija ekvivalencije.

Napisati program koji za učitanu dimenziju n i kvadratnu matricu dimenzije $n \times n$ ispisuje osobine odgovarajuće relacije. Pretpostaviti da je maksimalna dimenzija matrice 50×50 i da matrica za vrednosti elemenata može imati samo nule i jedinice. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 4
 Unesite elemente matrice:
 1 0 0 0
 0 1 1 0
 0 0 1 0
 0 0 0 0
 Relacija nije refleksivna.
 Relacija nije simatricna.
 Relacija jeste tranzitivna.
 Relacija nije ekvivalencija.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 4
 Unesite elemente matrice:
 1 1 0 0
 1 1 1 0
 0 0 1 0
 0 0 0 1
 Relacija jeste refleksivna.
 Relacija jeste simatricna.
 Relacija nije tranzitivna.
 Relacija nije ekvivalencija.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 54
 Greska: neispravan unos.
```

[Rešenje 3.7.8]

Zadatak 3.7.9 Data je kvadratna matrica dimenzije $n \times n$.

(a) Napisati funkciju `float trag(float a[][MAKS], int n)` koja računa trag matrice, odnosno zbir elemenata na glavnoj dijagonali matrice.

(b) Napisati funckciju `float suma_sporedna(float a[][MAKS], int n)` koja računa zbir elemenata na sporednoj dijagonali matrice.

(c) Napisati funkciju `float suma_iznad(float a[][MAKS], int n)` koja određuje sumu elememenata iznad glavne dijagonale.

(d) Napisati funkciju `float suma_ispod(float a[][MAKS], int n)` koja određuje sumu elememenata ispod sporedne dijagonale matrice.

Napisati program koji za učitanu matricu realnih brojeva ispisuje na tri decimale trag matrice, sumu na sporednoj dijagonali, sumu iznad glavne dijagonale i sumu elemenata ispod sporedne dijagonale. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice: 4
 Unesite elemente matrice:
 6 12.08 -1 20.5
 8 90 -33.4 19.02
 7.02 5 -20 14.5
 8.8 -1 3 -22.8
 Trag je 53.20.
 Suma na sporednoj dijagonali je 0.90.
 Suma iznad glavne dijagonale je 31.70.
 Suma ispod sporedne dijagonale je -1.82.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice: 5
 Unesite elemente matrice:
 1 2 3 5 5
 7 8 9 0 1
 6 4 3 2 2
 8 9 1 3 4
 0 3 1 8 6
 Trag je 21.00.
 Suma na sporednoj dijagonali je 17.00.
 Suma iznad glavne dijagonale je 33.00.
 Suma ispod sporedne dijagonale je 24.00.
```

[Rešenje 3.7.9]

Zadatak 3.7.10 Kvadratna matrica je donje trougaona ako se u gornjem trouglu (iznad glavne dijagonale, ne uključujući dijagonalu) nalaze sve nule. Napisati program koji za učitanu kvadratnu matricu proverava da li je ona donje trougaona i ispisuje odgovarajuću poruku. Pretpostaviti da je maksimalna dimenzija matrice 100×100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice: 5
 Unesite elemente matrice:
 -1 0 0 0 0
 2 10 0 0 0
 0 1 5 0 0
 7 8 20 14 0
 -23 8 5 1 11
 Matrica jeste donje trougaona.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice: 3
 Unesite elemente matrice:
 2 -2 1
 1 2 2
 2 1 -2
 Matrica nije donje trougaona.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju matrice: 200
 Greska: neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju matrice: 4
 Unesite elemente matrice:
 2 0 0 0
 7 80 0 0
 -9 4 4 0
 14 23 -8 1
 Matrica jeste donje trougaona.
```

[Rešenje 3.7.10]

Zadatak 3.7.11 Napisati program koji za učitanu celobrojnu kvadratnu matricu ispisuje redni broj kolone koja ima najveći zbir elemenata. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 3
 Unesite elemente matrice:
 1 2 3
 7 3 4
 5 3 1
 Indeks kolone je: 0
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 4
 Unesite elemente matrice:
 7 8 9 10
 7 6 11 4
 3 1 2 -2
 8 3 9 9
 Indeks kolone je: 2
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 104
 Greska: neispravan unos.
```

[Rešenje 3.7.11]

Zadatak 3.7.12 Napisati program koji za učitanu kvadratnu matricu realnih brojeva izračunava i ispisuje na dve decimale razliku između zbira elemenata gornjeg trougla i zbira elemenata donjeg trougla matrice. Gornji trougao čine svi elementi matrice koji su iznad glavne i sporedne dijagonale (ne računajući dijagonale), a donji trougao čine svi elementi ispod glavne i sporedne dijagonale (ne računajući dijagonale). Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 3
 Unesite elemente matrice:
 2 3.2 4
 7 8.8 1
 2.3 1 1
 Razlika je: 2.20
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 5
 Unesite elemente matrice:
 2.3 1 12 8 -20
 4 -8.2 7 14.5 19
 1 -2.5 9 11 33
 3 4.3 -5.7 2 8
 9 56 1.08 7 5.5 19.01
 Razlika je:-30.38
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 52
 Greska: neispravan unos.
```

[Rešenje 3.7.12]

Zadatak 3.7.13 Napisati program koji za učitanu celobrojnu matricu dimenzije $m \times n$ i uneta dva broja p i k ($p \leq m$, $k \leq n$) ispisuje sume svih podmatrica dimenzije $p \times k$ unete matrice. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice:  3 4
 Unesite elemente matrice:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Unesite dva cela broja:   3 3
 Sume podmatrica su: 54 63
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice:  3 4
 Unesite elemente matrice:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Unesite dva cela broja:   2 3
 Sume podmatrica su: 24 30 48 54
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice: 5 3
 Unesite elemente matrice:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Unesite dva cela broja:  2 2
 Sume podmatrica su: 7 5 20 19 18 23 4 8
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenzije matrice: -3 200
 Greska: neispravan unos.
```

[Rešenje 3.7.13]

Zadatak 3.7.14 Napisati program koji za učitanu celobrojnu kvadratnu matricu ispituje da li su njeni elementi po kolonama, vrstama i dijagonalama (glavnoj i sporednoj) sortirani strogo rastuće. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesi dimenziju matrice: 3
 Unesi elemente matrice:
 1 2 3
 4 5 6
 7 8 9
 Elementi su sortirani po kolonama.
 Elementi su sortirani po vrstama.
 Elementi su sortirani po dijagonalama.
```

Primer 2

```
Interakcija sa programom:
 Unesi dimenziju matrice: 2
 Unesi elemente matrice:
 6 9
 4 10
 Elementi nisu sortirani po kolonama.
 Elementi su sortirani po vrstama.
 Elementi nisu sortirani po dijagonalama.
```

Primer 3

```
Interakcija sa programom:
 Unesi dimenziju matrice: 4
 Unesi elemente matrice:
 5 5 7 9
 6 10 11 13
 8 12 14 15
 13 15 16 20
 Elementi su sortirani po kolonama.
 Elementi nisu sortirani po vrstama.
 Elementi su sortirani po dijagonalama.
```

Primer 4

```
Interakcija sa programom:
 Unesi dimenziju matrice: 1
 Unesi elemente matrice:
 5
 Elementi su sortirani po kolonama.
 Elementi su sortirani po vrstama.
 Elementi su sortirani po dijagonalama.
```

[Rešenje 3.7.14]

Zadatak 3.7.15 Napisati program koji za učitanu celobrojnu kvadratnu matricu ispituje da li su zbirovi elemenata njenih kolona uređeni u strogo rastućem poretku. Pretpostaviti da je maksimalna dimenzija matrice 10×10. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice: 4
 Unesite elemente matrice:
 1 0 0 0
 0 0 1 0
 0 0 0 1
 0 1 0 0
 Sume nisu uredjenje strogo rastuce.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice: 3
 Unesite elemente matrice:
 1 2 3
 4 5 6
 7 8 9
 Sume jesu uredjenje strogo rastuce.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju matrice: 3
 Unesite elemente matrice:
 2 -2 1
 1 2 2
 2 1 -2
 Sume nisu uredjenje strogo rastuce.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju matrice: 5
 Unesite elemente matrice:
 -1 0 3 0 20
 0 0 0 10 0
 0 0 -1 0 0
 0 1 0 0 0
 0 0 0 0 -1
 Sume jesu uredjenje strogo rastuce.
```

[Rešenje 3.7.15]

Zadatak 3.7.16 Matrica je *ortonormirana* ako je vrednost skalarnog proizvod svakog para različitih vrsta jednak nuli, a vrednost skalarnog proizvoda vrste sa samom sobom jednak jedinici. Napisati program koji za unetu celobrojnu kvadratnu matricu proverava da li je ortonormirana. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. Napomena: *Skalarni proizvod vektora* $a = (a_1, a_2, \ldots, a_n)$ *i* $b = (b_1, b_2, \ldots, b_n)$ *je* $a_1 \cdot b_1 + a_2 \cdot b_2 + \ldots + a_n \cdot b_n$.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice: 4
 Unesite elemente matrice:
 1 0 0 0
 0 0 1 0
 0 0 0 1
 0 1 0 0
 Matrica jeste ortonormirana.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice: 3
 Unesite elemente matrice:
 1 2 3
 4 5 6
 7 8 9
 Matrica nije ortonormirana.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju matrice: 3
 Unesite elemente matrice:
 2 -2 1
 1 2 2
 2 1 -2
 Matrica nije ortonormirana.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenziju matrice: 5
 Unesite elemente matrice:
 -1 0 0 0 0
 0 0 0 1 0
 0 0 -1 0 0
 0 1 0 0 0
 0 0 0 0 -1
 Matrica jeste ortonormirana.
```

[Rešenje 3.7.16]

Zadatak 3.7.17 Kvadratna matrica je *magični kvadrat* ako je suma elemenata u svakoj koloni i svakoj vrsti jednaka. Napisati program koji proverava da li je data celobrojna kvadratna matrica magični kvadrat i ispisuje odgovarajuću poruku na standardni izlaz. Pretpostaviti da je maksimalna dimenzija matrice je 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice: 4
 Unesite elemente matrice:
 1 5 3 1
 2 1 2 5
 3 2 2 3
 4 2 3 1
 Matrica jeste magicni kvadrat.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice: 3
 Unesite elemente matrice:
 1 2 3
 4 5 6
 -1 3 3
 Matrica nije magicni kvadrat.
```

[Rešenje 3.7.17]

*** Zadatak 3.7.18** Napisati program koji učitava celobrojnu kvadratnu matricu i ispisuje elemente matrice u grupama koje su paralelne sa njenom sporednom dijagonalom, počevši od gornjeg levog ugla. Pretpostaviti da je maksimalna dimenzija matrice 100×100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 3
 Unesite elemente matrice:
 1 2 3
 4 5 6
 7 8 9
 Ispis je:
 1
 2 4
 3 5 7
 6 8
 9
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 5
 Unesite elemente matrice:
 7 -8 1 2 3
 90 11 0 5 4
 12 -9 14 23 8
 80 6 88 17 62
 -22 10 44 57 -200
 Ispis je:
 7
 -8 90
 1 11 12
 2 0 -9 80
 3 5 14 6 -22
 4 23 88 10
 8 17 44
 62 57
 -200
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 -5
 Greska: neispravan unos.
```

[Rešenje 3.7.18]

*** Zadatak 3.7.19** Napisati funkciju `void mnozenje(int a[][MAKS], int m, int n, int b[][MAKS], int k, int t, int c[][MAKS])` koja računa matricu c kao proizvod matrica a i b. Dimenzija matrice a je $n \times m$, a dimenzija matrice b je $k \times t$. Napisati program koji ispisuje proizvod učitanih matrica. Pretpostaviti da je maksimalna dimenzija matrica 50×50. Ukoliko množenje matrica nije moguće ili je došlo do greške prilikom unosa podataka ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice A: 3 4
 Unesite elemente matrice A:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Unesite dimenzije matrice B: 4 2
 Unesite elemente matrice B:
 11 5
 6 7
 8 9
 0 -3
 Rezultat mnozenja je:
 87 64
 2 24
 145 83
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice A: 5 2
 Unesite elemente matrice A:
 1 7
 9 0
 -10 2
 92 3
 14 -8
 Unesite dimenzije matrice B: 2 4
 Unesite elemente matrice B:
 7 8 9 10
 -11 2 34 78
 Rezultat mnozenja je:
 -70 22 247 556
 63 72 81 90
 -92 -76 -22 56
 611 742 930 1154
 186 96 -146 -484
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice A: 3 4
 Unesite elemente matrice A:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Unesite dimenzije matrice B: 5 2
 Mnozenje matrica nije moguce.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice A: -3 4
 Greska: neispravan unos.
```

[Rešenje 3.7.19]

*** Zadatak 3.7.20** Element matrice naziva se *sedlo* ako je istovremeno najmanji u svojoj vrsti, a najveći u svojoj koloni. Napisati program koji ispisuje indekse i vrednosti onih elemenata matrice realnih brojeva koji su sedlo. Maksimalna dimenzija matrice je 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice:
 2 3
 Unesite elemente matrice:
 1 2 3
 0 5 6
 Sedlo: 0 0 1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice:
 3 3
 Unesite elemente matrice:
 10 3 20
 15 5 100
 30 -1 200
 Sedlo: 1 1 5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenzije matrice:
 3 -3
 Greska: neispravan unos.
```

[Rešenje 3.7.20]

*** Zadatak 3.7.21** Napisati program koji ispisuje elemente matrice celih brojeva u spriralnom redosledu počevši od gornjeg levog ugla krećući se u smeru suprotnom od smera kazaljke na satu. Maksimalna dimenzija matrice je 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju matrice:
 3 3
 Unesite elemente matrice:
 1 2 3
 4 5 6
 7 8 9
 Ispis je:
 1 2 3 6 9 8 7 4 5
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju matrice: 5 7
 Unesite elemente matrice:
 7 -8 1 2 3 -54 87
 90 11 0 5 4 9 18
 12 -9 14 23 8 -22 74
 80 6 88 17 62 38 41
 -22 10 44 57 -200 39 55
 Ispis je:
 7 -8 1 2 3 -54 87 18 74 41 55
 39 -200 57 44 10 -22 80 12 90
 11 0 5 4 9 -22 38 62 17 88 6
 -9 14 23 8
```

[Rešenje 3.7.21]

*** Zadatak 3.7.22** Matrica a se sadrži u matrici b ukoliko postoji podmatrica matrice b identična matrici a. Napisati program koji za dve učitane matrice celih brojeva proverava da li se druga matrica sadrži u prvoj učitanoj matrici. Maksimalna dimenzija obe matrice je 50×50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije matrice: 3 4
 Unesite elemente matrice:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Unesite dimenzije matrice: 2 2
 Unesite elemente matrice:
 2 3
 4 10
 Druga matrica je sadrzana u prvoj matrici.
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenzije matrice: 3 4
 Unesite elemente matrice:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Unesite dimenzije matrice: 2 2
 Unesite elemente matrice:
 2 8
 6 4
 Druga matrica nije sadrzana
 u prvoj matrici.
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenzije matrice: 5 5
 Unesite elemente matrice:
 7 -8 1 2 3
 90 11 0 5 4
 12 -9 14 23 8
 80 6 88 17 62
 -22 10 44 57 -200
 Unesite dimenzije matrice: 3 4
 Unesite elemente matrice:
 90 11 0 5
 12 -9 14 23
 80 6 88 17
 Druga matrica je sadrzana u prvoj matrici.
```

Primer 4

```
Interakcija sa programom:
 Unesite dimenzije matrice: 5 5
 Unesite elemente matrice:
 7 -8 1 2 3
 90 11 0 5 4
 12 -9 14 23 8
 80 6 88 17 62
 -22 10 44 57 -200
 Unesite dimenzije matrice: 53 4
 Greska: neispravan unos.
```

[Rešenje 3.7.22]

3.8 Rešenja

Rešenje 3.7.1

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

#define MAKS 50

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int m, n;
  int i, j;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
```

```
      scanf("%d", &a[i][j]);

  /* Ispis elemenata matrice. */
  for (i = 0; i < m; i++)
  {
    for (j = 0; j < n; j++)
      printf("%d ", a[i][j]);
    printf("\n");
  }

  return 0;
}
```

Rešenje 3.7.2

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

#define MAKS 50

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int m, n;
  int suma = 0;
  int i, j;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);

  /* Racunanje sume kvadrata svih elemenata. */
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      suma += a[i][j] * a[i][j];

  /* Ispis rezultata. */
  printf("Euklidska norma je %.3lf.\n", sqrt(suma));

```

```
  return 0;
}
```

Rešenje 3.7.3

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija ispisuje elemente matrice dimenzije m*n. */
void ispisi(int a[][MAKS], int m, int n)
{
  int i, j;

  for (i = 0; i < m; i++)
  {
    for (j = 0; j < n; j++)
      printf("%d ", a[i][j]);
    printf("\n");
  }
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int m, n;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, m, n);
```

```
48    /* Ispis ucitane matrice. */
      ispisi(a, m, n);
50
      return 0;
52  }
```

Rešenje 3.7.4

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

4   #define MAKS 50

6   /* Funkcija ucitava elemente matrice dimenzije m*n. */
    void ucitaj(int a[][MAKS], int m, int n)
8   {
      int i, j;
10
      printf("Unesite elemente matrice:\n");
12    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
14        scanf("%d", &a[i][j]);
    }
16
    /* Funkcija ispisuje elemente matrice dimenzije m*n. */
18  void ispisi(int a[][MAKS], int m, int n)
    {
20    int i, j;

22    for (i = 0; i < m; i++) {
        for (j = 0; j < n; j++)
24        printf("%d ", a[i][j]);
        printf("\n");
26    }
    }
28
    /* Funkcija formira maticu t transponovanjem matrice a. */
30  void transponovana(int a[][MAKS], int m, int n, int t[][MAKS])
    {
32    int i, j;

34    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
36        t[j][i] = a[i][j];
    }
38
    int main()
40  {
      /* Deklaracije potrebnih promenljivih. */
42    int a[MAKS][MAKS], t[MAKS][MAKS];
```

```
  int m, n;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, m, n);

  /* Formiranje transponovane matrice. */
  transponovana(a, m, n, t);

  /* Ispis rezultata. */
  ispisi(t, n, m);

  return 0;
}
```

Rešenje 3.7.5

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija ispisuje elemente matrice dimenzije m*n. */
void ispisi(int a[][MAKS], int m, int n)
{
  int i, j;

  for (i = 0; i < m; i++)
  {
    for (j = 0; j < n; j++)
      printf("%d ", a[i][j]);
    printf("\n");
```

```
  }
}

/* Funkcija razmenjuje elemente k-te i t-te vrste. */
void razmeni(int a[][MAKS], int m, int n, int k, int t)
{
  int j, pom;

  for (j = 0; j < n; j++)
  {
    pom = a[k][j];
    a[k][j] = a[t][j];
    a[t][j] = pom;
  }
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int m, n;
  int k, t;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, m, n);

  /* Ucitavanje indeksa vrsta i provera ispravnosti ulaza. */
  printf("Unesite indekse vrsta: ");
  scanf("%d%d", &k, &t);
  if (k < 0 || k >= m || t < 0 || t >= m)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Razmena k-te i t-te vrste. */
  razmeni(a, m, n, k, t);

  /* Ispis rezultata. */
  ispisi(a, m, n);

  return 0;
}
```

Rešenje 3.7.6

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int m, n, i, j, suma_suseda;
  int k, t;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, m, n);

  /* Izracunavanje i ispis rezultata. */
  printf("Indeksi elemenata koji su jednaki zbiru suseda su:\n");
  for (i = 0; i < m; i++)
  {
    for (j = 0; j < n; j++)
    {
      suma_suseda = 0;

      /* Vrsi se racunanje sume elemenata podmatrice velicine 3*3
         ciji je centralni element a[i][j]. Pri racunanju ove sume
         vodi se racuna da se ne izadje iz okvira matice a. */
      for (k = i - 1; k <= i + 1; k++)
        for (t = j - 1; t <= j + 1; t++)
          if (k >= 0 && k < m && t >= 0 && t < n)
            suma_suseda += a[k][t];
```

```
52        /* Od ukupne sume se oduzima tekuci element kako bi se dobio
             zbir elemenata koji su njegovi susedi. */
54        suma_suseda -= a[i][j];

56        /* Ukoliko je suma suseda jednaka tekucem elementu, ispisuju
             se indeksi tekuceg elementa matrice. */
58        if (suma_suseda == a[i][j])
            printf("%d %d\n", i, j);
60      }
    }
62  return 0;
}
```

Rešenje 3.7.7

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

4 #define MAKS 50

6 /* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
8 {
  int i, j;
10
  printf("Unesite elemente matrice:\n");
12  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
14    scanf("%d", &a[i][j]);
}
16
/* Funkcija formira niz b tako sto element b[i] ima vrednost
18    prosecne vrednosti i-te vrste matrice. */
void kreiraj_niz(int a[][MAKS], int m, int n, double b[])
20 {
  int i, j, suma;
22
  for (i = 0; i < m; i++)
24  {
    suma = 0;
26    for (j = 0; j < n; j++)
      suma += a[i][j];
28
    b[i] = (double) suma / n;
30  }
}
32
int main()
34 {
  /* Deklaracije potrebnih promenljivih. */
```

```
36    int a[MAKS][MAKS];
      double b[MAKS];
38    int m, n, i;

40    /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
      printf("Unesite dimenzije matrice: ");
42    scanf("%d%d", &m, &n);
      if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
44    {
        printf("Greska: neispravan unos.\n");
46      exit(EXIT_FAILURE);
      }
48
      /* Ucitavanje elemenata matrice. */
50    ucitaj(a, m, n);

52    /* Formira se niz b. */
      kreiraj_niz(a, m, n, b);
54
      /* Ispis rezultata. */
56    printf("Dobijeni niz je:\n");
      for (i = 0; i < m; i++)
58      printf("%g ", b[i]);
      printf("\n");
60
      return 0;
62  }
```

Rešenje 3.7.8

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

4   #define MAKS 50

6   /* Funkcija ucitava elemente matrice dimenzije n*n. */
    void ucitaj(int a[][MAKS], int n)
8   {
      int i, j;
10
      printf("Unesite elemente matrice:\n");
12    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
14        scanf("%d", &a[i][j]);
    }
16
    /* Relacija je refleksivna ukoliko je za svako i, a[i][i] = 1.
18     Funkcija proverava da li je relacija zadata matricom a
       refleksivna i vraca 1 ukoliko jeste, a 0 inace. */
20  int refleksivna(int a[][MAKS], int n)
    {
```

```
22    int i;

24    for (i = 0; i < n; i++)
        if (a[i][i] != 1)
26        return 0;

28    return 1;
}
30
/* Relacija je simetricna ukoliko za svaki par i, j vazi da je
32    a[i][j] = a[j][i]. Funkcija proverava da li je relacija zadata
      matricom a simetricna i vraca 1 ukoliko jeste, a 0 inace. */
34 int simetricna(int a[][MAKS], int n)
{
36    int i, j;

38    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
40        if (a[i][j] != a[j][i])
            return 0;
42
    return 1;
44 }

46 /* Relacija je tranzitivna ukoliko za svaku trojku i, j, k vazi da
      ako je a[i][j] = 1 i a[j][k] = 1, onda je i a[i][k] = 1.
48    Funkcija proverava da li je relacija zadata matricom a
      tranzitivna i vraca 1 ukoliko jeste, a 0 inace. */
50 int tranzitivna(int a[][MAKS], int n)
{
52    int i, j, k;

54    for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
56        for (k = 0; k < n; k++)
            if (a[i][j] == 1 && a[j][k] == 1 && a[i][k] == 0)
58            return 0;

60    return 1;
}
62
/* Relacija je relacija ekvivalencije ukoliko je refleksivna,
64    tranzitivna i simetricna. Funkcija proverava da li je relacija
      zadata matricom a relacija ekvivalencije i vraca 1 ukoliko
66    jeste, a 0 inace. */
68 int ekvivalencija(int a[][MAKS], int n)
{
70    if (refleksivna(a, n) && simetricna(a, n) && tranzitivna(a, n))
        return 1;

72    return 0;
}
```

```c

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int n;

  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenziju matrice: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Racunanje i ispis rezultata. */
  if (refleksivna(a, n))
    printf("Relacija jeste refleksivna.\n");
  else
    printf("Relacija nije refleksivna.\n");

  if (simetricna(a, n))
    printf("Relacija jeste simetricna.\n");
  else
    printf("Relacija nije simatricna.\n");

  if (tranzitivna(a, n))
    printf("Relacija jeste tranzitivna.\n");
  else
    printf("Relacija nije tranzitivna.\n");

  if (ekvivalencija(a, n))
    printf("Relacija jeste ekvivalencija.\n");
  else
    printf("Relacija nije ekvivalencija.\n");

  return 0;
}
```

Rešenje 3.7.9

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije n*n. */
```

```
void ucitaj(float a[][MAKS], int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
      scanf("%f", &a[i][j]);
}

/* Funkcija racuna trag matrice. */
float trag(float a[][MAKS], int n)
{
  float suma = 0;
  int i;

  for (i = 0; i < n; i++)
    suma += a[i][i];

  return suma;
}

/* Funkcija racuna sumu elemenata koji se nalaze na sporednoj
   dijagonali matrice. */
float suma_sporedna(float a[][MAKS], int n)
{
  float suma = 0;
  int i;

  for (i = 0; i < n; i++)
    suma += a[i][n - i - 1];

  return suma;
}

/* Funckija racuna sumu elemenata koji se nalaze iznad glavne
   dijagonale matrice. */
float suma_iznad(float a[][MAKS], int n)
{
  float suma = 0;
  int i, j;

  for (i = 0; i < n; i++)
    for (j = i + 1; j < n; j++)
      suma += a[i][j];

  return suma;
}

/* Funkcija racuna sumu elemenara koji se nalaze ispod sporedne
   dijagonale matrice. */
float suma_ispod(float a[][MAKS], int n)
```

```
{
  float suma = 0;
  int i, j;

  for (i = 0; i < n; i++)
    for (j = n - i - 1; j > i; j--)
      suma += a[i][j];

  return suma;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  float a[MAKS][MAKS];
  int n;

  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenziju matrice: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Ispis rezultata. */
  printf("Trag je %.2f.\n", trag(a, n));
  printf("Suma na sporednoj dijagonali je %.2f.\n",
         suma_sporedna(a, n));
  printf("Suma iznad glavne dijagonale je %.2f.\n",
         suma_iznad(a, n));
  printf("Suma ispod sporedne dijagonale je %.2f.\n",
         suma_ispod(a, n));

  return 0;
}
```

Rešenje 3.7.10

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije n*n. */
void ucitaj(int a[][MAKS], int n)
{
```

```
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija proverava da li je matrica donje trougaona i vraca
   jedinicu ukoliko jeste, a nulu inace. */
int donje_trougaona(int a[][MAKS], int n)
{
  int i, j;

  /* Prolazi se kroz sve elemente iznad glavne dijagonale i ukoliko
     se naidje na element koji je razlicit od nule, onda matrica
     nije donje trougaona. */
  for (i = 0; i < n; i++)
    for (j = i + 1; j < n; j++)
      if (a[i][j] != 0)
        return 0;

  /* Ukoliko su svi elementi iznad glavne dijagonale nule, matrica
     jeste donje trougaona. */
  return 1;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int n;

  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenziju matrice: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Ispis rezultata. */
  if (donje_trougaona(a, n))
    printf("Matrica jeste donje trougaona.\n");
  else
    printf("Matrica nije donje trougaona.\n");

  return 0;
```

```
}
```

Rešenje 3.7.11

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije n*n. */
void ucitaj(int a[][MAKS], int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int n, i, j;
  int maksimalni_zbir, trenutni_zbir = 0, indeks_kolone;

  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenziju matrice: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Maksimalni zbir se inicijalizuje na vrednost zbira prve
     kolone. U ovom slucaju bi bilo pogresno da se maksimalni zbir
     inicijalizuje na nulu jer moze da se desi da su svi elementi
     matrice negativni. Drugi nacin da se ispravno inicijalizuje
     maksimalni zbir jeste da mu se dodeli vrednost konstante
     INT_MIN cija se definicija nalazi u zaglavlju limits.h. */
  for (i = 0; i < n; i++)
    trenutni_zbir += a[i][0];

  maksimalni_zbir = trenutni_zbir;
  indeks_kolone = 0;
```

```
48    /* Racuna se zbir svake sledece kolone i azurira se vrednost
         maksimalnog zbira. */
50    for (j = 1; j < n; j++) {
        /* Racuna se zbir kolone j. */
52      trenutni_zbir = 0;
        for (i = 0; i < n; i++)
54        trenutni_zbir += a[i][j];

56      /* Ukoliko je taj zbir veci od trenutno maksimalnog zbira,
           azurira se vrednost maksimalnog zbira i pamti se tekuca
58         kolona. */
        if (trenutni_zbir > maksimalni_zbir) {
60        maksimalni_zbir = trenutni_zbir;
          indeks_kolone = j;
62      }
    }
64
    /* Ispis rezultata. */
66  printf("Indeks kolone je: %d\n", indeks_kolone);

68  return 0;
}
```

Rešenje 3.7.12

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

4 #define MAKS 50

6 /* Funkcija ucitava elemente matrice dimenzije n*n. */
  void ucitaj(float a[][MAKS], int n)
8 {
    int i, j;
10
    printf("Unesite elemente matrice:\n");
12  for (i = 0; i < n; i++)
      for (j = 0; j < n; j++)
14      scanf("%f", &a[i][j]);
  }
16
  int main()
18 {
    /* Deklaracije potrebnih promenljivih. */
20  float a[MAKS][MAKS];
    int n, i, j;
22  float gornji_trougao = 0, donji_trougao = 0;

24  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
    printf("Unesite dimenziju matrice: ");
26  scanf("%d", &n);
```

```
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Racuna se suma gornjeg trougla. */
  for (i = 0; i < n / 2; i++)
    for (j = i + 1; j < n - i - 1; j++)
      gornji_trougao += a[i][j];

  /* Racuna se suma donjeg trougla. */
  for (i = n / 2; i < n; i++)
    for (j = n - i; j < i; j++)
      donji_trougao += a[i][j];

  /* Ispis rezultata. */
  printf("Razlika je: %.2f\n", gornji_trougao - donji_trougao);

  return 0;
}
```

Rešenje 3.7.13

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int n, i, j, m, x, y, p, k;
  int suma;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
```

```
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, m, n);

  /* Ucitavanje dimenzija p i k i provera ispravnosti ulaza. */
  printf("Unesite dva cela broja: ");
  scanf("%d%d", &p, &k);
  if (p <= 0 || p > m || k <= 0 || k > n)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Racunanje i ispis rezultata. */
  printf("Sume podmatrica su: ");
  for (i = 0; i <= m - p; i++)
  {
    for (j = 0; j <= n - k; j++)
    {
      /* Za svaku poziciju (i,j), racuna se suma podmatrice
         dimenzije p*k, ciji je gornji levi ugao a[i][j]. */
      suma = 0;
      for (x = 0; x < p; x++)
        for (y = 0; y < k; y++)
          suma += a[i + x][j + y];

      printf("%d ", suma);
    }
  }
  printf("\n");

  return 0;
}
```

Rešenje 3.7.14

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije n*n. */
void ucitaj(int a[][MAKS], int n)
{
```

```
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija proverava da li je kolona j sortirana rastuce i vraca
   jedinicu ukoliko jeste, a nulu inace. */
int sortirana_kolona(int a[][MAKS], int n, int j)
{
  int i;

  for (i = 0; i < n - 1; i++)
    if (a[i][j] >= a[i + 1][j])
      return 0;

  return 1;
}

/* Funkcija proverava da li je svaka kolona matrice sortirana
   rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
int sortirani_po_kolonama(int a[][MAKS], int n)
{
  int j;

  for (j = 0; j < n; j++)
    if (!sortirana_kolona(a, n, j))
      return 0;

  return 1;
}

/* Funkcija proverava da li je i-ta vrsta sortirana rastuce i vraca
   jedinicu ukoliko jeste, a nulu inace. */
int sortirana_vrsta(int a[][MAKS], int n, int i)
{
  int j;

  for (j = 0; j < n - 1; j++)
    if (a[i][j] >= a[i][j + 1])
      return 0;

  return 1;
}

/* Funkcija proverava da li je svaka vrsta matrice sortirana
   rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
int sortirani_po_vrstama(int a[][MAKS], int n)
{
  int i;
```

```
61
  for (i = 0; i < n; i++)
63   if (!sortirana_vrsta(a, n, i))
       return 0;
65
  return 1;
67 }

69 /* Funkcija proverava da li je glavna dijagonala matrice sortirana
     rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
71 int sortirana_glavna(int a[][MAKS], int n)
   {
73   int i;

75   for (i = 0; i < n - 1; i++)
       if (a[i][i] >= a[i + 1][i + 1])
77       return 0;

79   return 1;
   }
81
   /* Funkcija proverava da li je sporedna dijagonala matrice
83    sortirana rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
   int sortirana_sporedna(int a[][MAKS], int n)
85 {
     int i;
87
     for (i = 0; i < n - 1; i++)
89     if (a[i][n - i - 1] >= a[i + 1][n - i - 2])
         return 0;
91
     return 1;
93 }

95 /* Funkcija proverava da li su obe dijagonale matrice sortirane
     rastuce i vraca jedinicu ukoliko jesu, a nulu inace. */
97 int sortirani_po_dijagonalama(int a[][MAKS], int n)
   {
99   return sortirana_glavna(a, n) && sortirana_sporedna(a, n);
   }
101
   int main()
103 {
     /* Deklaracije potrebnih promenljivih. */
105  int a[MAKS][MAKS];
     int n;
107
     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
109  printf("Unesite dimenziju matrice: ");
     scanf("%d", &n);
111  if (n <= 0 || n > MAKS)
     {
```

```
113     printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
115   }

117   /* Ucitavanje elemenata matrice. */
      ucitaj(a, n);
119
      /* Ispis rezultata. */
121   if (sortirani_po_kolonama(a, n))
        printf("Elementi su sortirani po kolonama.\n");
123   else
        printf("Elementi nisu sortirani po kolonama.\n");
125
      if (sortirani_po_vrstama(a, n))
127     printf("Elementi su sortirani po vrstama.\n");
      else
129     printf("Elementi nisu sortirani po vrstama.\n");

131   if (sortirani_po_dijagonalama(a, n))
        printf("Elementi su sortirani po dijagonalama.\n");
133   else
        printf("Elementi nisu sortirani po dijagonalama.\n");
135
      return 0;
137 }
```

Rešenje 3.7.15

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 10
5
   /* Funkcija ucitava elemente matrice dimenzije n*n. */
7  void ucitaj(int a[][MAKS], int n)
   {
9    int i, j;

11   printf("Unesite elemente matrice:\n");
     for (i = 0; i < n; i++)
13     for (j = 0; j < n; j++)
         scanf("%d", &a[i][j]);
15 }

17 /* Funkcija racuna sumu elemenata kolone j. */
   int suma_kolone(int a[][MAKS], int n, int j)
19 {
     int suma = 0, i;
21
     for (i = 0; i < n; i++)
23     suma += a[i][j];
```

```
  return suma;
}

/* Funkcija proverava da li su sume kolona uredjene rastuce i vraca
   jedinicu ako jesu, a nulu inace. */
int uredjene_sume(int a[][MAKS], int n)
{
  int prethodna_suma, trenutna_suma;
  int j;

  /* Prva suma se inicijalizuje na sumu prve kolone. */
  prethodna_suma = suma_kolone(a, n, 0);

  for (j = 1; j < n; j++)
  {
    /* Racuna se suma trenutne kolone. */
    trenutna_suma = suma_kolone(a, n, j);

    /* Ukoliko je ta suma manja ili jednaka prethodnoj, poredak
       suma nije rastuci. */
    if (trenutna_suma <= prethodna_suma)
      return 0;

    /* Suma trenutne kolone postaje suma prethodne kolone za
       narednu iteraciju. */
    prethodna_suma = trenutna_suma;
  }

  return 1;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int n;

  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenziju matrice: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Ispis rezultata. */
  if (uredjene_sume(a, n))
```

```
    printf("Sume jesu uredjenje strogo rastuce.\n");
  else
    printf("Sume nisu uredjenje strogo rastuce.\n");

  return 0;
}
```

Rešenje 3.7.16

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

#define MAKS 200

/* Funkcija ucitava elemente matrice dimenzije n*n. */
void ucitaj(int a[][MAKS], int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija racuna skalarni proizvod i-te i j-te vrste matrice. */
int skalarni_proizvod(int a[][MAKS], int n, int i, int j)
{
  int suma = 0, k;

  for (k = 0; k < n; k++)
    suma += a[i][k] * a[j][k];

  return suma;
}

/* Matrica je ortonormirana ukoliko je skalarni proizvod svakog
   para razlicitih vrsta jednak nuli, a skalarni proizvod svake
   vrste same sa sobom jednak jedinici. Funkcija proverava da li
   je matrica ortorormirana i vraca jedinicu ukoliko jeste, a nulu
   inace. */
int ortonormirana(int a[][MAKS], int n)
{
  int i, j;

  /* Za svaki par vrsta se racuna skalarni proizvod i proverava da
     li je uslov ispunjen. Ukoliko nije, kao povratna vrednost
     funkcije se vraca nula. */
  for (i = 0; i < n; i++)
  {
    for (j = i; j < n; j++)
```

```
43      {
          /* Provera za slucaj kada se racuna skalarni proizvod vrste
45           same sa sobom. */
          if (i == j && skalarni_proizvod(a, n, i, i) != 1)
47          return 0;

49        /* Provera za par razlicitih vrsta. */
          if (i != j && skalarni_proizvod(a, n, i, j) != 0)
51          return 0;
        }
53    }

55    /* Ako je izvrsavanje stiglo do kraja petlje, znaci da je uslov
         ispunjen za sve vrste, tj. da je matrica ortonormirana. */
57    return 1;
  }
59
  int main()
61{
    /* Deklaracije potrebnih promenljivih. */
63  int a[MAKS][MAKS];
    int n;
65
    /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
67  printf("Unesite dimenziju matrice: ");
    scanf("%d", &n);
69  if (n <= 0 || n > MAKS)
    {
71    printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
73  }

75  /* Ucitavanje elemenata matrice. */
    ucitaj(a, n);
77
    /* Ispis rezultata. */
79  if (ortonormirana(a, n))
      printf("Matrica jeste ortonormirana.\n");
81  else
      printf("Matrica nije ortonormirana.\n");
83
    return 0;
85}
```

Rešenje 3.7.17

```
1 #include <stdio.h>
  #include <stdlib.h>
3
  #define MAKS 50
5
```

```c
/* Funkcija ucitava elemente matrice dimenzije n*n. */
void ucitaj(int a[][MAKS], int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < n; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija racuna sumu kolone j. */
int suma_kolone(int a[][MAKS], int n, int j)
{
  int i, suma = 0;

  for (i = 0; i < n; i++)
    suma += a[i][j];

  return suma;
}

/* Funkcija racuna sumu i-te vrste. */
int suma_vrste(int a[][MAKS], int n, int i)
{
  int j, suma = 0;

  for (j = 0; j < n; j++)
    suma += a[i][j];

  return suma;
}

/* Funkcija proverava da li elementi matrice predstavljaju magicni
   kvadrat. */
int magicni_kvadrat(int a[][MAKS], int n)
{
  /* Da bi matrica bila magicni kvadrat, sume svih vrsta i kolona
     treba da budu jednke. Suma se zato inicijalizuje na sumu prve
     kolone. */
  int suma = suma_kolone(a, n, 0);
  int i, j;

  /* Proverava se da li su sume ostalih kolona jednake izracunatoj
     sumi. Ukoliko se naidje na kolonu koja ne zadovoljava ovaj
     uslov, matrica nije magicni kvadrat. */
  for (j = 1; j < n; j++)
    if (suma_kolone(a, n, j) != suma)
      return 0;

  /* Proverava se i da li su sume svih vrsta jednake izracunatoj
     sumi. Ukoliko se naidje na vrstu koja ne zadovoljava ovaj
```

```
    uslov, matrica nije magicni kvadrat. */
  for (i = 0; i < n; i++)
    if (suma_vrste(a, n, i) != suma)
      return 0;

  /* Ako sve vrste i kolone imaju jednake sume, matrica je magicni
     kvadrat. */
  return 1;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int n;

  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenziju matrice: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Ispis rezultata. */
  if (magicni_kvadrat(a, n))
    printf("Matrica jeste magicni kvadrat.\n");
  else
    printf("Matrica nije magicni kvadrat.\n");

  return 0;
}
```

Rešenje 3.7.18

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

#define MAKS 100

/* Funkcija ucitava elemente matrice dimenzije n*n. */
void ucitaj(int a[][MAKS], int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < n; i++)
```

```
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int n;
  int i, j, k;

  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenziju matrice: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, n);

  /* Petlja kojom se ispisuju dijagonale iznad sporedne dijagonale,
     ukljucujuci i sporednu dijagonalu.
     Npr. za n=4, indeksi elemenata u matrici su:
     (0,0) (0,1) (0,2) (0,3)
     (1,0) (1,1) (1,2) (1,3)
     (2,0) (2,1) (2,2) (2,3)
     (3,0) (3,1) (3,2) (3,3)
     Dakle, ispis elemenata ide u sledecem redosledu:
     (0,0)
     (0,1) (1,0)
     (0,2) (1,1) (2,0)
     (0,3) (1,2) (2,1) (3,0)
     Za k-ti ispis vazi da indeksi vrsta imaju vrednosti od nula do
     k, a indeksi kolona od k do nula. */
  for (k = 0; k < n; k++)
  {
    /* Indeks kolone se inicijalizuje na k. */
    j = k;
    /* Indeks vrste se inicijalizuje na 0. */
    i = 0;

    /* Ispisuju se odgovarajuci elementi, indeks vrste se povecava,
       a indeks kolone se smanjuje. */
    while (j >= 0)
    {
      printf("%d ", a[i][j]);
      i++;
      j--;
    }
```

```
    printf("\n");
  }

  /* Petlja kojom se ispisuju dijagonale ispod sporedne dijagonale.
     Npr. za n=4, indeksi elemenata u matrici su:
     (0,0) (0,1) (0,2) (0,3)
     (1,0) (1,1) (1,2) (1,3)
     (2,0) (2,1) (2,2) (2,3)
     (3,0) (3,1) (3,2) (3,3)
     Dakle, ispis elemenata ide u sledecem redosledu:
     (1,3) (2,2) (3,1)
     (2,3) (3,2)
     (3,3)
     Za k-ti ispis vazi da indeksi vrsta imaju vrednosti od k do
     n-1, a indeksi kolona od n-1 do 1. */
  for (k = 1; k < n; k++)
  {
    i = k;
    j = n - 1;

    while (i < n)
    {
      printf("%d ", a[i][j]);
      i++;
      j--;
    }
    printf("\n");
  }

  return 0;
}
```

Rešenje 3.7.19

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija ispisuje elemente matrice dimenzije m*n. */
```

```
void ispisi(int a[][MAKS], int m, int n)
{
  int i, j;

  for (i = 0; i < m; i++)
  {
    for (j = 0; j < n; j++)
      printf("%d ", a[i][j]);
    printf("\n");
  }
}

/* Funkcija vrsi mnozenje matrica a i b i rezultat smesta u matricu
   c. */
void mnozenje(int a[][MAKS], int m, int n,
              int b[][MAKS], int k, int t, int c[][MAKS])
{
  int i, j, w;

  for (i = 0; i < m; i++)
  {
    for (j = 0; j < t; j++)
    {
      /* Element c[i][j] se dobija kao skalarni proizvod i-te vrste
         matrice a i j-te kolone matrice b. */
      c[i][j] = 0;
      for (w = 0; w < n; w++)
        c[i][j] += a[i][w] * b[w][j];
    }
  }
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS], b[MAKS][MAKS], c[MAKS][MAKS];
  int m, n;
  int k, t;

  /* Ucitavanje dimenzija prve matrice i provera ispravnosti
     ulaza. */
  printf("Unesite dimenzije matrice A: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata prve matrice. */
  ucitaj(a, m, n);

```

```
  /* Ucitavanje dimenzija druge matrice i provera ispravnosti
     ulaza. */
  printf("Unesite dimenzije matrice B: ");
  scanf("%d%d", &k, &t);
  if (k <= 0 || k > MAKS || t <= 0 || t > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Provera da li se odgovarajuce dimenzije matrica poklapaju. */
  if (n != k)
  {
    printf("Mnozenje matrica nije moguce.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata druge matrice. */
  ucitaj(b, k, t);

  /* Racunanje proizvoda. */
  mnozenje(a, m, n, b, k, t, c);

  /* Ispis rezultata. */
  printf("Rezultat mnozenja je:\n");
  ispisi(c, m, t);

  return 0;
}
```

Rešenje 3.7.20

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(double a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%lf", &a[i][j]);
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
```

```
20  double a[MAKS][MAKS];
    int m, n, k, i, j;
22
    int indeks_kolone;
24  double maks_kolone, min_vrste;

26  /* Ucitavanje dimenzija prve matrice i provera ispravnosti ulaza.
     */
28  printf("Unesite dimenzije matrice: ");
    scanf("%d%d", &m, &n);
30  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
    {
32    printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
34  }

36  /* Ucitavanje elemenata prve matrice. */
    ucitaj(a, m, n);
38
    /* Pronalazak elemenata koji su sedlo. */
40  for (i = 0; i < m; i++)
    {
42    /* Pronalazi se najmanji element u tekucoj vrsti. Pamti se
         kolona kojoj taj element pripada. */
44    min_vrste = a[i][0];
      indeks_kolone = 0;
46
      for (j = 1; j < n; j++)
48    {
        if (a[i][j] < min_vrste)
50      {
          min_vrste = a[i][j];
52        indeks_kolone = j;
        }
54    }

56    /* Pronalazi se najveci element u zapamcenoj koloni. */
      maks_kolone = a[0][indeks_kolone];
58
      for (k = 1; k < m; k++)
60      if (a[k][indeks_kolone] > maks_kolone)
          maks_kolone = a[k][indeks_kolone];
62
      /* Element je sedlo ukoliko je on istovremeno najmanji u svojoj
64       vrsti i najveci u svojoj koloni. */
      if (min_vrste == maks_kolone)
66      printf("Sedlo: %d %d %g\n", i, indeks_kolone, min_vrste);
    }
68
    return 0;
70 }
```

Rešenje 3.7.21

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS];
  int m, n, brojac, i, j, pravac;
  int gornja_granica, donja_granica, leva_granica, desna_granica;

  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
  printf("Unesite dimenzije matrice: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata matrice. */
  ucitaj(a, m, n);

  /* Ciklicni ispis elemenata matrice:
     Npr. za n=4, indeksi elemenata u matrici su:
     (0,0) (0,1) (0,2) (0,3)
     (1,0) (1,1) (1,2) (1,3)
     (2,0) (2,1) (2,2) (2,3)
     (3,0) (3,1) (3,2) (3,3)
     Ispis treba da ide sledecim redosledom:
     1. krece se sa leva na  desno (0,0) (0,1) (0,2) (0,3)
     2. zatim se ide na dole (1,3) (2,3) (3,3)
     3. zatim na levo (3,2) (3,1) (3,0)
     4. zatim na gore (2,0) (1,0)
     (ovde se staje jer je (0,0) vec ispisano) i prelazi se opet
     na levo. Koraci 1-4 se ponavljaju dok god se ne ispisu svi
     elementi. Ideja je da kada se ispisu elementi
     prve vrste (kada se ide sa leva na desno), da se pomeri
```

```
51      "gornja granica ispisa" za 1, kako bi se naznacilo da je taj
        red vec ispisan. Slicno, kada se vrsi ispis odozgo na dole,
53      uspesno je ispisana jedna kolona pa je potrebno pomeriti
        "desnu granicu ispisa" za jedan u levo. Kada se ispise jedna
55      vrsta sa desna na levo, vrsi se pomeranje donje granice ispisa
        za jedan na gore. Slicno, kada se ispise jedna kolona odozdo
57      na gore, pomera se leva granica ispisa za jedan u desno. */
    gornja_granica = 0;
59  donja_granica = m - 1;
    leva_granica = 0;
61  desna_granica = n - 1;

63  /* Promenljiva pravac govori u kom smeru ispis ide. */
    pravac = 1;
65
    /* Promenljive i i j su indeksi elementa koji se ispisuje. */
67  i = 0;
    j = 0;
69
    for (brojac = 0; brojac < m * n; brojac++) {
71    printf("%d ", a[i][j]);

73    switch (pravac) {
        /* Ako je pravac = 1, trenutni smer ispisa je sa leva na
75         desno. */
      case 1:
77      /* Ako je ispisan element na desnoj granici, onda se menja
           pravac ispisa. */
79      if (j == desna_granica)
        {
81        /* Prelazi se na pravac odozgo na dole. */
          pravac = 2;
83        /* Pomera se gornja granica za jedan na dole. */
          gornja_granica++;
85        /* Pomera se vrednost vrste za jedan na dole. */
          i++;
87      }
        else
89      {
          /* Ako jos uvek nije ispisan element na desnoj granici,
91           vrsi se pomeranje na sledeci element u trenutnoj vrsti. */
          j++;
93      }
        break;
95
        /* Ako je pravac = 2, trenutni smer ispisa je odozgo na dole.
97         Slicno kao i u prethodnom slucaju, ako se dodje do donje
           granice, menja se pravac i pomera se desna granica za
99         jedno mesto u levo. A ako nije, samo se prelazi na narednu
           vrstu. */
101   case 2:
        if (i == donja_granica)
```

```
103         {
              pravac = 3;
105           desna_granica--;
              j--;
107         }
            else
109         {
              i++;
111         }
            break;
113
            /* Ako je pravac = 3, trenutni smer ispisa je sa desna na
115            levo. Slicno kao i u prethodnom slucaju, ako se dodje do
               leve granice, menja se pravac i pomera se donja granica za
117            jedno mesto na gore. A ako nije, samo se prelazi na
               narednu kolonu. */
119       case 3:
            if (j == leva_granica)
121         {
              pravac = 4;
123           donja_granica--;
              i--;
125         }
            else
127         {
              j--;
129         }
            break;
131
            /* Ako je pravac = 4, trenutni smer ispisa je odozdo na gore.
133            Slicno kao i u prethodnim slucajevima, ako se dodje do
               gornje granice, menja se pravac i pomera se leva granica
135            za jedno mesto u desno. A ako nije, samo se prelazi na
               narednu vrstu. */
137       case 4:
            if (i == gornja_granica)
139         {
              pravac = 1;
141           leva_granica++;
              j++;
143         }
            else
145         {
              i--;
147         }
        }
149   }

151   return 0;
}
```

Rešenje 3.7.22

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
  int i, j;

  printf("Unesite elemente matrice:\n");
  for (i = 0; i < m; i++)
    for (j = 0; j < n; j++)
      scanf("%d", &a[i][j]);
}

/* Funkcija proverava da li je matrica b podmatrica matrice a i
   vraca jedinicu ukoliko jeste, a nulu inace. */
int podmatrica(int a[][MAKS], int m, int n,
               int b[][MAKS], int k, int t)
{
  int i, j, x, y;
  int jeste_podmatrica;

  for (i = 0; i <= m - k; i++)
  {
    for (j = 0; j <= n - t; j++)
    {
      /* Za svaku poziciju (i,j) se proverava da li je podmatrica
         dimenzije k*t ciji je gornji levi ugao a[i][j] jednaka
         matrici b. */
      jeste_podmatrica = 1;
      for (x = 0; x < k && jeste_podmatrica; x++)
        for (y = 0; y < t && jeste_podmatrica; y++)
          if (a[i + x][j + y] != b[x][y])
            jeste_podmatrica = 0;

      if (jeste_podmatrica)
        return 1;
    }
  }

  return 0;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int a[MAKS][MAKS], b[MAKS][MAKS];
  int m, n;
```

```
  int k, t;

  /* Ucitavanje dimenzija prve matrice i provera ispravnosti
     ulaza. */
  printf("Unesite dimenzije matrice A: ");
  scanf("%d%d", &m, &n);
  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata prve matrice. */
  ucitaj(a, m, n);

  /* Ucitavanje dimenzija druge matrice i provera ispravnosti
     ulaza. */
  printf("Unesite dimenzije matrice B: ");
  scanf("%d%d", &k, &t);
  if (k <= 0 || k > MAKS || t <= 0 || t > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje elemenata druge matrice. */
  ucitaj(b, k, t);

  /* Ispis rezultata. */
  if (podmatrica(a, m, n, b, k, t))
    printf("Druga matrica je sadrzana u prvoj matrici.\n");
  else
    printf("Druga matrica nije sadrzana u prvoj matrici.\n");

  return 0;
}
```

3.9 Strukture

Zadatak 3.9.1 Definisati strukturu kojom se opisuje kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja. Napisati program koji za učitana dva kompleksna broja ispisuje vrednost zbira, razlike, proizvoda i količnika. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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 3.9.1]

Zadatak 3.9.2 Definisati strukturu kojom se opisuje razlomak. Napisati funkcije koje izračunavaju zbir i proizvod dva razlomka. Napisati program koji za uneti ceo broj n i unetih n razlomaka ispisuje njihov ukupan zbir i proizvod. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj razlomaka:  5
 Unesite razlomke:
 1 2
 7 8
 3 4
 5 6
 2 9
 Suma svih razlomaka je 229/72.
 Proizvod svih razlomaka je 35/576.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj razlomaka:  10
 Unesite razlomke:
 4 3
 12 25
 3 8
 1 3
 8 9
 2 3
 5 6
 -24 50
 7 18
 -7 19
 Suma svih razlomaka je 6089/1368.
 Proizvod svih razlomaka je 1568/577125.
```

[Rešenje 3.9.2]

Zadatak 3.9.3 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 funkcije:

(a) `int ucitaj(Vocka niz[])` koja učitava voćke sa standardnog ulaza sve do unosa reči KRAJ i kao povratnu vrednost vraća broj učitanih voćki;

(b) `Vocka vocka _sa_najvise_vitamina(Vocka niz[], int n)` koja pronalazi voćku koja ima najviše C vitamina.

Napisati program koji učitava podatke o voćkama i ispisuje ime voćke sa najviše vitamina C. Pretpostaviti da broj voćki neće biti veći od 50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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 83.5
 Unesite ime voćke i njenu količinu vitamina C:   kivi 71
 Unesite ime voćke i njenu količinu vitamina C:   banana 8.7
 Unesite ime voćke i njenu količinu vitamina C:   pomorandza 70.8
 Unesite ime voćke i njenu količinu vitamina C:   KRAJ
 Voce sa najvise C vitamina je: limun
```

[Rešenje 3.9.3]

Zadatak 3.9.4 Definisati strukturu `Grad` koja sadrži ime grada (niska dužine 20 karaktera) i prosečnu temperaturu u toku decembra (realan broj). Napisati funkcije:

(a) `void ucitaj(Grad gradovi[], int n)` koja učitava podatke o gradovima sa standardnog ulaza.

(b) `void ispisi(Grad gradovi[], int n)` koja ispisuje podatke o gradovima koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni.

Napisati program koji učitava imena n $(0 < n < 50)$ gradova i njihove prosečne temperature, a zatim ispisuje imena gradova sa idealnom temperaturom za klizanje. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj gradova:  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 gradova:  2
 Unesite grad i temperaturu:   Varsava 11
 Unesite grad i temperaturu:   Prag 2
 Gradovi sa idealnom temperaturom za klizanje u decembru:
```

[Rešenje 3.9.4]

Zadatak 3.9.5 Definisati strukturu `ParReci` koja sadrži reč na srpskom jeziku i odgovarajući prevod na engleski jezik. Napisati program koji do kraja ulaza učitava sve parove reči, a potom za rečenicu koja se zadaje u jednoj liniji ispisati prevod. Ako je reč u rečenici nepoznata umesto nje ispisati odgovarajući broj zvezdica. Maksimalna dužina reči je 50 karaktera, maksimalan broj parova reči je 100, a maksimalna dužina rečenice je 100 karaktera.

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
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 zima winter
 pas dog
 sreca happiness
 prijatelj friend
 solja cup
 covek man
 Unesite recenicu za prevod:
 pas je covekov najbolji prijatelj
 dog is ******* best friend
```

[Rešenje 3.9.5]

Zadatak 3.9.6 Cenoteka pomaže kupcima da pronađu najpovoljniju cenu za proizvod koji žele da kupe. Napisati program koji učitava najpre broj različitih prodavnica (ceo broj manji od 50) a zatim i podatke o ceni traženog artikla – zadaje se naziv prodavnice (niske maksimalne dužine 20 karaktera) i cena u toj prodavnici (realan broj). Korisnik zadaje željenu cenu proizvoda, a program ispisuje imena svih onih prodavnica u kojima je cena proizvoda jednaka ili manja od željene. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj prodavnica:  5
 idea 58.9
 maxi 58.2
 roda 55.1
 tempo 54.5
 interex 57.99
 Unesite zeljenu cenu:  57.0
 Povoljne prodavnice su:
 roda
 tempo
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj prodavnica:  4
 dm 43.2
 lily 45.99
 benu_apoteke 43.99
 sephora 50.99
 Unesite zeljenu cenu:  47.00
 Povoljne prodavnice su:
 dm
 lily
 benu_apoteke
```

[Rešenje 3.9.6]

Zadatak 3.9.7 Statistički zavod Srbije istražuje kako rade obdaništa u Srbiji. Za dato obdanište dobija spisak n dece sa kolonama: pol (`m` ili `z`), broj godina (od 3 do 6) i ocena koju je dete dalo radu obdaništa (od 1 do 5). Maksimalan broj dece u obdaništu je 200. Napisati program koji za decu datog pola i broja godina ispisuje na tri decimale prosečnu ocenu obdaništa. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj dece:  5
 Unesite podatke za svako dete, pol,
 broj godina i ocenu:
 m 3 5
 z 3 4
 m 4 2
 m 5 4
 m 3 4
 Unesite pol i broj godina:  m 3
 Prosecna ocena je: 4.500.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj dece:  10
 Unesite podatke za svako dete, pol,
 broj godina i ocenu:
 m 3 5
 z 4 4
 m 5 4
 z 4 3
 z 3 2
 z 4 5
 m 6 5
 z 4 4
 z 4 5
 m 6 3
 Unesite pol i broj godina:  z 4
 Prosecna ocena je: 4.200.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj dece:  15
 Unesite podatke za svako dete, pol,
 broj godina i ocenu:
 m 3 2
 z 7 5
 Greska: neispravan broj godina.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj dece:  2
 Unesite podatke za svako dete, pol,
 broj godina i ocenu:
 m 3 2
 z 3 5
 Unesite pol i broj godina:  h 5
 Greska: neispravan pol.
```

[Rešenje 3.9.7]

Zadatak 3.9.8 Definisati strukturu kojom se opisuje student. Student je zadat svojim imenom i prezimenom (oba su maksimalne dužine 30 karaktera), smerom (R, I, V, N, T, O) i prosečnom ocenom. Napisati program koji učitava podatke o n studenata, zatim učitava smer i ispisuje imena i prezimena onih studenta koji su sa datog smera. Potom ispisati podatke za studenta koji ima najveći prosek. Ako ima više takvih studenata ispisati ih sve. Maksimalan broj studenata je 2000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj studenata:  5
 Unesite podatke o studentima:
 0. student:  Kocic Marija R 9.14
 1. student:  Tanja Mratinkovic R 7.88
 2. student:  Mihailo Simic N 8.44
 3. student:  Milena Medar I 9.14
 4. student:  Ljubica Mihic N 9.00
 Unesite smer:  R
 Studenti sa R smera:
 Kocic Marija
 Tanja Mratinkovic
 ---------------------
 Svi studenti koji imaju maksimalni prosek:
 Kocic Marija, R, 9.14
 Milena Medar, I, 9.14
```

Primer 2

```
Interakcija sa programom:
 Unesite broj studenata:  4
 Unesite podatke o studentima:
 0. student:  Djordje Lazarevic N 9.05
 1. student:  Minja Peric W 7.70
 Greska: neispravan unos smera.
```

[Rešenje 3.9.8]

Zadatak 3.9.9 Program učitava podatke o učenicima do kraja unosa. Učenika može biti najviše 30. Definisati strukturu `Djak` koja sadrži ime đaka (maksimalne dužine 20 karaktera) i 9 ocena (ocene su celi brojevi od 1 do 5). Napisati program koji učitava podatke o đacima sve do kraja ulaza i na standardni izlaz ispisuje prvo imena nedovoljnih đaka, a zatim imena odličnih đaka. Đak je nedovoljan ako ima barem jednu jedinicu, a odličan ako ima prosek ocena veći ili jednak 4.5. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite podatke o djaku:  Maja 4 5 2 3 4 4 3 3 4
 Unesite podatke o djaku:  Nikola 5 4 5 5 5 4 4 5 5
 Unesite podatke o djaku:  Jasmina 2 2 1 1 2 3 3 1 3
 Unesite podatke o djaku:  Pera 5 4 5 3 5 5 1 5 5
 Unesite podatke o djaku:  Pavle 4 3 2 4 3 2 4 3 2
 Unesite podatke o djaku:

 NEDOVOLJNI: Jasmina Pera
 ODLICNI: Nikola
```

Primer 2

```
Interakcija sa programom:
 Unesite podatke o djaku:  Uros 3 4 2 3 4 2 3 4 4
 Unesite podatke o djaku:  Nebojsa 4 5 5 5 4 5 5 5 5
 Unesite podatke o djaku:  Sreten 2 3 2 4 5 4 4 4 2
 Unesite podatke o djaku:

 NEDOVOLJNI:
 ODLICNI: Nebojsa
```

Primer 3

```
Interakcija sa programom:
 Unesite podatke o djaku:   Mirko 2 3 4 4 4 3 3 3 4
 Unesite podatke o djaku:   Mihailo 2 3 10 5 5 2 3 4 2
 Greska: neispravna ocena.
```

[Rešenje 3.9.9]

Zadatak 3.9.10 Defnisati strukturu `Osoba` kojom se opisuje jedan unos u imenik. Za svaku osobu su dati podaci: ime (maksimalne dužine 20 karaktera), prezime (maksimalne dužine 30 karaktera) i email adresa (maksimalne dužine 50 karaktera). Napisati program koji učitava ceo broj n ($0 < n \leq 50$) a zatim podatke o n osoba. Ispisati imena i prezimena svih osoba koje imaju gmail adresu (čija se email adresa završava sa `@gmail.com`). U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. Napomena: *Može se smatrati da je svaka email adresa dobro zadata i sadrži samo jedno pojavljivanje znaka `@`.*

Primer 1

```
Interakcija sa programom:
 Unesite broj osoba:  3
 Unesite podatke o osobama:
 ime, prezime i email.
 Dusko Dugousko dusko@yahoo.com
 Pink Panter panter@gmail.com
 Pera Detlic pd@gmail.com
 Vlasnici gmail naloga su:
 Pink Panter
 Pera Detlic
```

Primer 2

```
Interakcija sa programom:
 Unesite broj osoba:  3
 Unesite podatke o osobama:
 ime, prezime i email.
 Homer Simpson homer@yahoo.com
 Mardz Simpson mardz@matf.bg.ac.rs
 Vlasnici gmail naloga su:
```

[Rešenje 3.9.10]

*** Zadatak 3.9.11** Napisati program koji izračunava prosečnu cenu jedne potrošačke korpe. Potrošačka korpa se sastoji od broja kupljenih artikala i niza kupljenih artikala. Svaki artikal određen je svojim nazivom, količinom i cenom. Program treba da učita broj potrošača n (najviše 100), zatim podatke za n potrošačkih korpi i da na osnovu učitanih podataka izračuna prosečnu cenu potrošačke korpe. Program ispisuje na dve decimale račune svake potrošačke korpe i na kraju ispisuje prosečnu cenu potrošačke korpe. Može se pretpostaviti da nijedan potrošač neće kupiti više od 20 artikala, kao i da naziv svakog artikla sadrži maksimalno 30 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj potrosackih korpi:  3
 Unesite podatke o korpi:
 Broj artikala:  4
 Unesite artikal, naziv, kolicinu i cenu:  jabuke 10 22.4
 Unesite artikal, naziv, kolicinu i cenu:  dezodorans 1 120.99
 Unesite artikal, naziv, kolicinu i cenu:  C_supa 3 36.56
 Unesite artikal, naziv, kolicinu i cenu:  sunka 1 230.99
 Unesite podatke o korpi:
 Broj artikala:  2
 Unesite artikal, naziv, kolicinu i cenu:  Jafa_keks 55.78
 Unesite artikal, naziv, kolicinu i cenu:  Najlepse_zelje 62.99
 Unesite podatke o korpi:
 Broj artikala:  3
 Unesite artikal, naziv, kolicinu i cenu:  prasak_za_ves 1 1199.99
 Unesite artikal, naziv, kolicinu i cenu:  omeksivac 1 279.99
 Unesite artikal, naziv, kolicinu i cenu:  protiv_kamenca 1 699.99

 Korpa 0:
        jabuke 10 22.40
        dezodorans 1 120.99
        C_supa 3 36.56
        sunka 1 230.99
 ------------------------
        ukupno: 685.66

 Korpa 1:
        Jafa_keks 55 0.78
        Najlepse_zelje 62 0.99
 ------------------------
        ukupno: 104.28

 Korpa 2:
        prasak_za_ves 1 1199.99
        omeksivac 1 279.99
        protiv_kamenca 1 699.99
 ------------------------
        ukupno: 2179.97

 Prosecna cena potrosacke korpe: 989.97
```

[Rešenje 3.9.11]

Zadatak 3.9.12 Definisati strukturu `Lopta` sa poljima `poluprecnik` (ceo broj u centimetrima) i `boja` (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Napisati funkcije

(a) `void ucitaj(Lopta niz[], int n)` koja učitava podatke o n lopti u niz.

(b) `double ukupna_zapremina(Lopta niz[], int n)` koja računa ukupnu zapreminu svih lopti.

(c) `int broj_crvenih(Lopta niz[], int n)` koja prebrojava koliko ima crvenih lopti u nizu.

Napisati program koji učitava informacije o n lopti ($0 < n < 50$) i ispisuje ukupnu zapreminu i broj crvenih lopti. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 2

```
Interakcija sa programom:
 Unesite broj lopti:  8
 Unesite dalje poluprecnike i boje lopti
 (1-plava, 2-zuta, 3-crvena, 4-zelena):
 1. lopta:  2 1
 2. lopta:  30 3
 3. lopta:  7 3
 4. lopta:  4 1
 5. lopta:  5 2
 6. lopta:  6 2
 7. lopta:  12 3
 8. lopta:  14 2
 Ukupna zapremina: 134996.34
 Ukupno crvenih lopti: 3
```

Primer 3

```
Interakcija sa programom:
 Unesite broj lopti:  8
 Unesite dalje poluprecnike i boje lopti
 (1-plava, 2-zuta, 3-crvena, 4-zelena):
 1. lopta:  1 2
 2. lopta:  2 10
 Greska: neispravan unos.
```

[Rešenje 3.9.12]

Zadatak 3.9.13 Napisati program za predstavljanje poligona i izračunavanje njegovog obima i dužine stranica.

(a) Definisati strukturu `Tacka` kojom se opisuje tačka Dekartovske ravni čije su x i y koordinate podaci tipa `double`.

(b) Definisati funkciju `double rastojanje(const Tacka* a, const Tacka* b)` koja izračunava rastojanje između dve tačke.

(c) Definisati funkciju `int ucitaj_poligon(Tacka* tacke, int n)` koja učitava maksimalno 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.

(d) Definisati funkciju `double obim(Tacka* poligon, int n)` koja izračunava obim poligona sa n tačaka u zadatom nizu NAPOMENA: *Prilikom računanja obima ne zaboraviti stranicu koja spaja poslednje i prvo teme.*

(e) Definisati funkciju `double maksimalna_stranica(Tacka* poligon, int n)` koja izračunava dužinu najduže stranice poligona sa n tačaka u zadatom nizu.

(f) Napisati funkciju `double povrsina_trougla(const Tacka* A, const Tacka* B, const Tacka* C)` za računanje površine trougla.

(g) Napisati funkciju `double povrsina(Tacka* poligon, int n)` za računanje površine konveksnog poligona. NAPOMENA: *Zadatak se može rešiti korišćenjem funkcije* ***povrsina_trougla****.*

(h) Napisati program koji učitava poligon sa maksimalno n temena ($0 < n \leq 1000$) i za učitani poligon ispisuje na tri decimale obim, dužinu maksimalnu stranice i površinu. Pretpostaviti da je uneti poligon konveksan. Poligon mora imati barem tri temena. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite maksimalan broj tacaka poligona:  10
 0 0
 0 6
 3 3
 Obim poligona je 14.485.
 Duzina maksimalne stranice je 6.000.
 Povrsina poligona je 9.000.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite maksimalan broj tacaka poligona:  10
 0 0
 12 0
 13 2
 16 5
 20 10
 18 15
 15 20
 10 20
 8 15
 3 4
 Obim poligona je 63.566.
 Duzina maksimalne stranice je 12.083.
 Povrsina poligona je 247.500.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite maksimalan broj tacaka poligona:  4
 0 0
 Greska: poligon mora imati bar tri tacke.
```

[Rešenje 3.9.13]

*** Zadatak 3.9.14** Definisati strukturu `Izraz` kojom se opisuje numerički izraz nad celim brojevima koji se sastoji od dva celobrojna operanda i numeričke operacije (sabiranje, oduzimanje, množenje ili celobrojno deljenje) nad celim brojevima.

(a) Napisati funkciju koja ispituje da li je dati izraz korektno zadat i vraća jedinicu ako jeste a nulu inace. Podrazumeva se da je izraz korektno zadat ako operacija odgovara $+$, $-$, $*$ ili / i u slučaju deljenja drugi operand je različit od 0.

(b) Napisati funkciju koja za dati izraz određuje vrednost izraza.

(c) Napisati funkciju koja učitava izraze. Funkcija treba da učita sa standardnog ulaza n izraza koji su zadati prefiksno — prvo operacija, a potom dva operanda.

Napisati program koji učitava prirodan broj n, $(n < 1000)$ a zatim n izraza u prefiksnoj notaciji. Program treba da ispiše maksimalnu vrednost unetih izraza i sve izraze čija vrednost je manja od polovine maksimalne vrednosti. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj izraza:  4
 Unesite izraze u prefiksnoj notaciji:
 + 10 4
 - 9 2
 * 11 2
 / 7 3
 Maksimalna vrednost izraza: 22
 Izrazi cija je vrednost manja
 od polovine maksimalne vrednosti:
 9 - 2 = 7
 7 / 3 = 2
```

Primer 2

```
Interakcija sa programom:
 Unesite broj izraza:  10
 Unesite izraze u prefiksnoj notaciji:
 + 10 2
 - -678 34
 * 77 2
 + 1000 -23
 + 102 4
 - 200 23
 / 67 12
 / 1000 2
 * 44 6
 / 13 1
 Maksimalna vrednost izraza: 977
 Izrazi cija je vrednost manja
 od polovine maksimalne vrednosti:
 10 + 2 = 12
 -678 - 34 = -712
 77 * 2 = 154
 102 + 4 = 106
 200 - 23 = 177
 67 / 12 = 5
 44 * 6 = 264
 13 / 1 = 13
```

Primer 3

```
Interakcija sa programom:
 Unesite broj izraza:  3
 Unesite izraze u prefiksnoj notaciji:
 * 1 2
 / 3 0
 Greska: deljenje nulom.
```

[Rešenje 3.9.14]

*** Zadatak 3.9.15** Definisati strukturu kojom se opisuje polinom. Polinom je dat svojim stepenom (može biti najviše 10) i realnim koeficijentima.

(a) Napisati funkciju koja sa standardnog ulaza učitava polinome sve do kraja ulaza. Polinomi su zadati stepenom i koeficijentima. Funkcija kao povratnu vrednost vraća broj učitanih polinoma.

(b) Napisati funkciju koja ispisuje polinom u obliku $k_0 \pm k_1 * x \pm k_2 * x\hat{}2 \pm k_3 * x\hat{}3 \pm \ldots \pm k_n * x\hat{}n$ (pri čemu je n stepen polinoma). Koeficijente ispisati na dve decimale. Ne ispisivati koeficijente koji su jednaki 0 i na mesto znaka $\pm$ zapisati odgovarajući znak, $+$ ili $-$, u zavisnosti od znaka odgovarajućeg koeficijenta.

(c) Napisati funkciju koja za dati polinom određuje njegov integral.

(d) Učitati polinome do kraja ulaza i za svaki učitani polinom odrediti i ispisati integral tog polinoma. Maksimalan broj polinoma je 100.

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite stepen:  3
 Unesite koeficijente polinoma:
 1 0 3 1
 Unesite stepen:  4
 Unesite koeficijente polinoma:
 7 9 4 0 4
 Unesite stepen:

 Integrali su:
 1.00*x + 1.00*x^3 + 0.25*x^4
 7.00*x + 4.50*x^2 + 1.33*x^3 + 0.80*x^5
```

Primer 2

```
Interakcija sa programom:
 Unesite stepen:  3
 Unesite koeficijente polinoma:
 1 0 -4 1
 Unesite stepen:  2
 Unesite koeficijente polinoma:
 1 2 -3
 Unesite stepen:  1
 Unesite koeficijente polinoma:
 0 -1
 Unesite stepen:

 Integrali su:
 1.00*x -1.33*x^3 + 0.25*x^4
 1.00*x + 1.00*x^2 -1.00*x^3
 -0.50*x^2
```

[Rešenje 3.9.15]

3.10 Rešenja

Rešenje 3.9.1

```
#include <stdio.h>

/* Struktura koja opisuje kompleksni broj. */
typedef struct
{
  float re;
  float im;
} KompleksanBroj;

/* Kada se neka promenljiva zadaje kao argument funkcije, obicno
   se prenosi po vrednosti (bez pokazivaca), ako se ona nece
   menjati u funkciji ili po adresi (preko pokazivaca), ako ce se
   njena vrednost promeniti u funkciji.

   Prilikom poziva funkcije, za svaki argument funkcije kreira se
   promenljiva koja predstavlja lokalnu kopiju argumenta i koja
   prestaje da postoji po zavrsetku funkcije. S obzirom da se
   strukuture sastoje od vise polja, zauzimaju vise memorije nego
   nestrukturne promenljive. Zbog toga je za njihovo kopiranje
   potrebno vise vremena i vise memorijskih resursa nego za
   kopiranje nestrukturnih promenljivih.

   Da bi program bio efikasniji, korisno je da se struktura uvek
   prenosi po adresi (preko pokazivaca), bez obzira da li ce se
   ona 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 se strukturna promenljiva prenosi u funkciju po adresi
   (preko pokazivaca), tada postoji mogucnost da se njena polja
   menjaju u funkciji. Ukoliko to nije potrebno, uz argument se
   dodaje kljucna rec const. Na taj nacin, u slucaju pokusaja
   izmene strukturne promenljive koja je prosledjena kao const,
   kompajler ce prijaviti gresku. Na ovaj nacin se obezbedjuje
   da promenljiva koja je preneta po adresi ne bude cak ni slucajno
   izmenjena u funkciji.

   Funkcija izracunava zbir kompleksnih brojeva. */
KompleksanBroj saberi(const KompleksanBroj * a,
                      const KompleksanBroj * b)
{
```

```
43    KompleksanBroj c;
      c.re = a->re + b->re;
45    c.im = a->im + b->im;
      return c;
47  }

49  /* Funkcija izracunava razliku kompleksnih brojeva. */
    KompleksanBroj oduzmi(const KompleksanBroj * a,
51                        const KompleksanBroj * b)
    {
53    KompleksanBroj c;
      c.re = a->re - b->re;
55    c.im = a->im - b->im;
      return c;
57  }

59  /* Funkcija izracunava proizvod kompleksnih brojeva. */
    KompleksanBroj pomnozi(const KompleksanBroj * a,
61                         const KompleksanBroj * b)
    {
63    KompleksanBroj c;
      c.re = a->re * b->re - a->im * b->im;
65    c.im = b->re * a->im + a->re * b->im;
      return c;
67  }

69  /* Funkcija izracunava kolicnik kompleksnih brojeva. */
    KompleksanBroj podeli(const KompleksanBroj * a,
71                        const KompleksanBroj * b)
    {
73    KompleksanBroj c;
      c.re = (a->re * b->re + a->im * b->im) /
75         (b->re * b->re + b->im * b->im);
      c.im = (b->re * a->im - a->re * b->im) /
77         (b->re * b->re + b->im * b->im);
      return c;
79  }

81  int main()
    {
83    /* Deklaracije potrebnih promenlivih. */
      KompleksanBroj a, b;
85    KompleksanBroj c;

87    /* Ucitavanje kompleksnih brojeva. */
      printf("Unesite realni i imaginarni deo prvog broja: ");
89    scanf("%f%f", &a.re, &a.im);

91    printf("Unesite realni i imaginarni deo drugog broja: ");
      scanf("%f%f", &b.re, &b.im);
93
      /* Ispis zbira. */
```

```
  c = saberi(&a, &b);
  /* Ukoliko je imaginarni deo negativan, njegov zapis vec
     ukljucuje znak, u suprotnom, broj je oblika a + b*i. */
  printf("Zbir: %.2f%c%.2f*i\n",
         c.re, c.im > 0 ? '+' : ' ', c.im);

  /* Ispis razlike. */
  c = oduzmi(&a, &b);
  printf("Razlika: %.2f%c%.2f*i\n",
         c.re, c.im > 0 ? '+' : ' ', c.im);

  /* Ispis proizvoda. */
  c = pomnozi(&a, &b);
  printf("Proizvod: %.2f%c%.2f*i\n",
         c.re, c.im > 0 ? '+' : ' ', c.im);

  /* Ispis kolicnika. */
  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");

  return 0;
}
```

Rešenje 3.9.2

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

typedef struct {
  int brojilac;
  int imenilac;
} Razlomak;

/* Funkcija Euklidovim algoritmom racuna najveci zajednicki
   delilac brojeva a i b. */
int nzd(int a, int b)
{
  int ostatak;

  while (b != 0) {
    ostatak = a % b;
    a = b;
    b = ostatak;
  }
```

```
21    return a;
}

/* Funkcija vraca razlomak koji se dobija deljenjem imenioca i
25     brojioca sa njihovim najvecim zajednickim deliocem. */
void uprosti(Razlomak * r)
27 {
    int nzd_razlomka = nzd(r->brojilac, r->imenilac);
29  r->brojilac /= nzd_razlomka;
    r->imenilac /= nzd_razlomka;
31 }

33 /* Funkcija racuna zbir razlomaka a i b. */
Razlomak saberi(const Razlomak* a, const Razlomak* b)
35 {
    Razlomak c;
37
    c.brojilac = a->brojilac * b->imenilac + b->brojilac * a->imenilac;
39  c.imenilac = a->imenilac * b->imenilac;
    uprosti(&c);
41
    return c;
43 }

45 /* Funkcija racuna proizvod razlomaka a i b. */
Razlomak pomnozi(const Razlomak* a, const Razlomak* b)
47 {
    Razlomak c;
49
    c.brojilac = a->brojilac * b->brojilac;
51  c.imenilac = a->imenilac * b->imenilac;
    uprosti(&c);
53
    return c;
55 }

57 int main()
{
59  /* Deklaracije potrebnih promenljivih. */
    int n, i;
61  Razlomak suma, proizvod, r;

63  /* Ucitavanje broja razlomaka i provera ispravnosti ulaza. */
    printf("Unesite broj razlomaka: ");
65  scanf("%d", &n);
    if (n <= 0)
67  {
      printf("Greska: neispravan unos.\n");
69    exit(EXIT_FAILURE);
    }
71
    /* Inicijalizacija sume i proizvoda. */
```

```
73    suma.brojilac = 0;
      suma.imenilac = 1;
75    proizvod.brojilac = 1;
      proizvod.imenilac = 1;
77
      /* Ucitavanje razlomaka i racunanje rezultata. */
79    printf("Unesite razlomke:\n");
      for (i = 0; i < n; i++)
81    {
        scanf("%d%d", &r.brojilac, &r.imenilac);
83
        suma = saberi(&suma, &r);
85      proizvod = pomnozi(&proizvod, &r);
      }
87
      /* Ispis rezultata. */
89    printf("Suma svih razlomaka je %d/%d.\n", suma.brojilac,
             suma.imenilac);
91    printf("Proizvod svih razlomaka je %d/%d.\n", proizvod.brojilac,
             proizvod.imenilac);
93
      return 0;
95  }
```

Rešenje 3.9.3

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

4 #define MAKS_IME 21
  #define MAKS_VOCKI 50
6
  /* Struktura koja opisuje vocku. */
8 typedef struct
  {
10  char ime[MAKS_IME];
    float vitamin;
12 } Vocka;

14 /* Funkcija ucitava podatke o vockama u niz struktura.
     Kao povratnu vrednost vraca broj ucitanih vocki. */
16 int ucitaj(Vocka niz[])
  {
18  int i=0;
    char ime[MAKS_IME];
20
    /* Vocke se ucitavaju sve dok se ne unese rec "KRAJ". */
22  do
    {
24    printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
```

```
    /* Ucitavanje imena vocke. */
    scanf("%s", ime);
    if (strcmp(ime, "KRAJ") == 0)
      break;
    strcpy(niz[i].ime, ime);

    /* Ucitavanje kolicine vitamina C. */
    scanf("%f", &niz[i].vitamin);

    i++;
  }
  while (i < MAKS_VOCKI);

  return i;
}

/* Funkcija pronalazi vocku sa najvise C vitamina. */
Vocka vocka_sa_najvise_vitamina(Vocka niz[], int n)
{
  /* Pronalazak pozicije vocke sa najvise vitamina c. */
  int maks_i = 0, i;
  for (i = 1; i < n; i++)
    if (niz[i].vitamin > niz[maks_i].vitamin)
      maks_i = i;

  /* Kao povratna vrednost se vraca vocka na poziciji maks_i. */
  return niz[maks_i];
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  Vocka vocke[MAKS_VOCKI], najzdravija;
  int n;

  /* Ucitavanje ulaza. */
  n = ucitaj(vocke);

  /* Ispis rezultata. */
  najzdravija = vocka_sa_najvise_vitamina(vocke, n);
  printf("Voce sa najvise C vitamina je: %s\n", najzdravija.ime);

  return 0;
}
```

Rešenje 3.9.4

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

#define MAKS_IME 21
```

```c
#define MAKS_GRADOVA 50
#define DONJA_GRANICA 3
#define GORNJA_GRANICA 8

typedef struct
{
  char ime_grada[MAKS_IME];
  float temperatura;
} Grad;

/* Funkcija ucitava podatke o gradovima u niz. */
void ucitaj(Grad gradovi[], int n)
{
  int i;
  for (i = 0; i < n; i++)
  {
    printf("Unesite grad i temperaturu: ");
    scanf("%s %f", gradovi[i].ime_grada, &gradovi[i].temperatura);
  }
}

/* Funkcija ispisuje gradove sa idealnom temperaturom za klizanje
   u decembru. */
void ispisi(Grad gradovi[], int n)
{
  int i;

  printf("Gradovi sa idealnom temperaturom za "
         "klizanje u decembru:\n");
  for (i = 0; i < n; i++)
  {
    if (gradovi[i].temperatura >= DONJA_GRANICA &&
        gradovi[i].temperatura <= GORNJA_GRANICA)
    {
      printf("%s\n", gradovi[i].ime_grada);
    }
  }
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n;
  Grad gradovi[MAKS_GRADOVA];

  /* Ucitavanje broja gradova i provera ispravnosti ulaza. */
  printf("Unesite broj gradova: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS_GRADOVA)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
```

```
  }

  /* Ucitavanje podataka o gradovima. */
  ucitaj(gradovi, n);

  /* Ispis rezultata. */
  ispisi(gradovi, n);

  return 0;
}
```

Rešenje 3.9.5

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

#define MAKS_REC 21
#define MAKS_BROJ_RECI 100

typedef struct
{
  char sr[MAKS_REC];
  char en[MAKS_REC];
} ParReci;

/* Funkcija ucitava parove reci u recnik. */
int ucitaj(ParReci recnik[])
{
  int i = 0;
  char sr[MAKS_REC];
  char en[MAKS_REC];

  /* Ucitavaju se parovi reci sa standardnog ulaza sve do kraja
     ulaza. */
  while (scanf("%s %s", sr, en) != EOF)
  {
    if (i == MAKS_BROJ_RECI)
      break;

    strcpy(recnik[i].sr, sr);
    strcpy(recnik[i].en, en);

    i++;
  }

  return i;
}


/*
   Funkcija 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;

  /* Pretraga reci. */
  for (i = 0; i < n; i++)
  {
    if (strcmp(recnik[i].sr, rec) == 0)
    {
      strcpy(prevod, recnik[i].en);
      return;
    }
  }

  /* Ukoliko rec nije pronadjena, formira se prevod reci koji se
     sastoji od zvezdica. */
  for (i = 0; rec[i]; i++)
    prevod[i] = '*';
  prevod[i] = '\0';
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  ParReci recnik[MAKS_BROJ_RECI];
  int n;

  char rec[MAKS_REC];
  char prevod[MAKS_REC];
  char c;

  /* Ucitavaju se parovi reci u recnik. */
  n = ucitaj(recnik);

  /* Ucitava se recenica i ispisuje se njen prevod. */
  printf("Unesite recenicu za prevod: \n");
  do {
    /* Ucitava se rec po rec date recenice i pronalazi se njen
       prevod. */
    scanf("%s", rec);
    pronadji_prevod(recnik, n, rec, prevod);
    printf("%s ", prevod);

    /* Ukoliko je karakter iza reci znak za novi red, onda se
       prekida sa unosom, a ako nije ucitava se sledeca rec. */
    c = getchar();
  } while (c != '\n');

```

```
  putchar('\n');

  return 0;
}
```

Rešenje 3.9.6

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

#define MAKS_PODATAKA 50
#define MAKS_NAZIV 21

typedef struct
{
  char naziv_prodavnice[MAKS_NAZIV];
  double cena_artikla;
} Artikal;

/* Funkcija ucitava podatke o ceni artikla u razlicitim
   prodavnicama. */
void ucitaj(Artikal niz[], int n)
{
  int i;
  for (i = 0; i < n; i++)
  {
    scanf("%s%lf", niz[i].naziv_prodavnice, &niz[i].cena_artikla);
    if (niz[i].cena_artikla <= 0)
    {
      printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
    }
  }
}

/* Funkcija ispisuje imena svih prodavnica u kojima je cena
   artikla manja ili jednaka zeljenoj ceni. */
void ispisi(Artikal niz[], int n, double zeljena_cena)
{
  int i;
  printf("Povoljne prodavnice su:\n");
  for (i = 0; i < n; i++)
    if (niz[i].cena_artikla <= zeljena_cena)
      printf("%s\n", niz[i].naziv_prodavnice);
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  Artikal niz[MAKS_PODATAKA];
  double zeljena_cena;
```

```
45    int n;
47    /* Ucitavanje broja prodavnica i provera ispravnosti ulaza. */
      printf("Unesite broj prodavnica: ");
49    scanf("%d", &n);
      if (n <= 0 || n > MAKS_PODATAKA)
51    {
        printf("Greska: neispravan unos.\n");
53      exit(EXIT_FAILURE);
      }
55
      /* Ucitavanje podataka o cenama. */
57    ucitaj(niz, n);
59    /* Ucitavanje zeljene cene. */
      printf("Unesite zeljenu cenu: ");
61    scanf("%lf", &zeljena_cena);
63    /* Ispis rezultata. */
      ispisi(niz, n, zeljena_cena);
65
      return 0;
67  }
```

Rešenje 3.9.7

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

4 #define MAKS_DECE 200

6 typedef struct
  {
8   char pol;
    int broj_godina;
10  int ocena;
  } Dete;
12
/* Funkcija ucitava podatke o deci i proverava ispravnost unetih
14    podataka. */
  void ucitaj(Dete niz[], int n)
16{
    char blanko;
18  int i;
    printf("Unesite podatke za svako dete, pol, broj godina i "
20         "ocenu:\n");
    for (i = 0; i < n; i++)
22  {
      scanf("%c%c%d%d", &blanko, &niz[i].pol, &niz[i].broj_godina,
24                      &niz[i].ocena);
```

```
    /* Ispitivanje pogresnog unosa. */
    if (niz[i].pol != 'm' && niz[i].pol != 'z')
    {
      printf("Greska: neispravan pol.\n");
      exit(EXIT_FAILURE);
    }
    if (niz[i].broj_godina > 6 || niz[i].broj_godina < 3)
    {
      printf("Greska: neispravan broj godina.\n");
      exit(EXIT_FAILURE);
    }
    if (niz[i].ocena < 1 || niz[i].ocena > 5)
    {
      printf("Greska: neispravna ocena.\n");
      exit(EXIT_FAILURE);
    }
  }
}



int main()
{
  /* Deklaracija potrebnih promenljivih. */
  int n, i, broj_godina;
  Dete niz[MAKS_DECE];
  char blanko, pol;
  int suma, broj_dece;

  /* Ucitavanje broja dece i provera ispravnosti ulaza. */
  printf("Unesite broj dece: ");
  scanf("%d", &n);
  if (n <= 0 || n > MAKS_DECE)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje podataka o deci. */
  ucitaj(niz, n);

  /* Ucitavanje trazenih podataka. */
  printf("Unesite pol i broj godina: ");
  scanf("%c%c%d", &blanko, &pol, &broj_godina);

  /* Ispitivanje ispravnosti unetih podataka. */
  if (pol != 'm' && pol != 'z')
  {
    printf("Greska: neispravan pol.\n");
    exit(EXIT_FAILURE);
  }
  if (broj_godina > 6 || broj_godina < 3)
```

```
78    {
        printf("Greska: neispravan broj godina.\n");
80      exit(EXIT_FAILURE);
      }
82
      /* Racuna se prosecna ocena dece ciji se pol i broj godina
84       poklapaju sa unetim. */
      suma = 0;
86    broj_dece = 0;
      for (i = 0; i < n; i++)
88    {
        if (niz[i].pol == pol && niz[i].broj_godina == broj_godina)
90      {
          suma += niz[i].ocena;
92        broj_dece++;
        }
94    }

96    /* Ispis rezultata. */
      if (broj_dece == 0)
98      printf("Ne postoje deca sa takvim karakteristikama.\n");
      else
100     printf("Prosecna ocena je: %.3lf.\n",(double)suma / broj_dece);

102   return 0;
}
```

Rešenje 3.9.8

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

4 #define MAKS_STUDENATA 2000
  #define MAKS_NISKA 31
6
  typedef struct Student {
8   char ime[MAKS_NISKA];
    char prezime[MAKS_NISKA];
10  char smer;
    float prosek;
12 } Student;

14 /* Funkcija ucitava podatke o studentima u niz. */
  void ucitaj(Student niz[], int n)
16 {
    int i;
18
    printf("Unesite podatke o studentima:\n");
20  for (i = 0; i < n; i++)
    {
22    printf("%d. student: ", i);
```

```
    scanf("%s %s %c %f", niz[i].ime, niz[i].prezime,
          &niz[i].smer, &niz[i].prosek);

    if (niz[i].smer != 'R' && niz[i].smer != 'I' &&
        niz[i].smer != 'V' && niz[i].smer != 'N' &&
        niz[i].smer != 'T' && niz[i].smer != 'O')
    {
      printf("Greska: neispravan unos smera.\n");
      exit(EXIT_FAILURE);
    }
  }
}

/* Funkcija ispisuje podatke o studentu. */
void ispisi(const Student * s)
{
  printf("%s %s, %c, %.2f\n", s->ime, s->prezime, s->smer,
         s->prosek);
}

/* Funkcija racuna najveci prosek. */
float najveci_prosek(Student studenti[], int n)
{
  float maks_prosek;
  int i;

  maks_prosek = studenti[0].prosek;
  for (i = 1; i < n; i++)
    if (maks_prosek < studenti[i].prosek)
      maks_prosek = studenti[i].prosek;

  return maks_prosek;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  Student studenti[MAKS_STUDENATA];
  int n, i;
  float maks_prosek;
  char smer;

  /* Ucitavanje broja studenata i provera ispravnosti ulaza. */
  printf("Unesite broj studenata: ");
  scanf("%d", &n);
  if (n < 0 || n > MAKS_STUDENATA)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje podataka o studentima. */
```

```
  ucitaj(studenti, n);

  /* Ucitavanje smera. Pre smera se preskace novi red koji je unet
     nakon podataka o poslednjem studentu. */
  printf("Unesite smer: ");
  getchar();
  scanf("%c", &smer);
  if (smer != 'R' && smer != 'I' && smer != 'V' && smer != 'N' &&
      smer != 'T' && smer != 'O')
  {
    printf("Greska: neispravan unos smera.\n");
    exit(EXIT_FAILURE);
  }

  /* Ispis studenata sa unetog smera. */
  printf("Studenti sa %c smera:\n", smer);
  for (i = 0; i < n; i++)
    if (studenti[i].smer == smer)
      printf("%s %s\n", studenti[i].ime, studenti[i].prezime);
  printf("---------------------\n");

  /* Racunanje najveceg proseka. */
  maks_prosek = najveci_prosek(studenti, n);

  /* Ispis svih studenata sa najvecim prosekom. */
  printf("Svi studenti koji imaju maksimalni prosek:\n");
  for (i = 0; i < n; i++)
    if (studenti[i].prosek == maks_prosek)
      ispisi(&studenti[i]);

  return 0;
}
```

Rešenje 3.9.9

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

#define MAKS_IME 21
#define BROJ_OCENA 9
#define MAKS_DJAKA 30

typedef struct
{
  char ime[MAKS_IME];
  int ocena[BROJ_OCENA];
}
Djak;

/* Funkcija proverava ispravnost date ocene. */
void provera_ocene(int ocena)
```

```
{
  if (ocena < 1 || ocena > 5)
  {
    printf("Greska: neispravna ocena.\n");
    exit(EXIT_FAILURE);
  }
}

/* Funkcija ucitava podatke o djacima u niz. */
int ucitaj(Djak niz[])
{
  int i,j;

  while (i < MAKS_DJAKA)
  {
    printf("Unesite podatke o djaku: ");
    /* Ucitavanje imena. */
    if(scanf("%s", niz[i].ime) == EOF)
        break;

    /* Ucitavanje ocena. */
    for (j = 0; j < BROJ_OCENA; j++)
    {
      scanf("%d", &niz[i].ocena[j]);
      provera_ocene(niz[i].ocena[j]);
    }

    i++;
  }

  return i;
}

/* Funkcija racuna prosecnu ocenu datog djaka. */
float prosecna_ocena(const Djak* djak)
{
  int j;
  float suma = 0;
  for (j = 0; j < BROJ_OCENA; j++)
    suma += djak->ocena[j];

  return suma / BROJ_OCENA;
}

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  Djak niz[MAKS_DJAKA];
  int i = 0, n, j;
  float prosek;

  /* Ucitavanje podataka o djacima. */
```

```
  n = ucitaj(niz);

  /* Ispisivanje imena nedovoljnih ucenika. */
  printf("\n\nNEDOVOLJNI: ");
  for (i = 0; i < n; i++)
    for (j = 0; j < 9; j++)
      if (niz[i].ocena[j] == 1) {
        printf("%s ", niz[i].ime);
        break;
      }
  printf("\n");

  /* Ispisivanje imena odlicnih ucenika. */
  printf("ODLICNI: ");
  for (i = 0; i < n; i++)
  {
    prosek = prosecna_ocena(&niz[i]);
    if (prosek >= 4.5)
      printf("%s ", niz[i].ime);
  }
  printf("\n");

  return 0;
}
```

Rešenje 3.9.10

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

#define MAKS_IME 21
#define MAKS_PREZIME 31
#define MAKS_EMAIL 51
#define MAKS_OSOBA 50

typedef struct
{
  char ime[MAKS_IME];
  char prezime[MAKS_PREZIME];
  char email[MAKS_EMAIL];
} Osoba;

/* I nacin:
   Funkcija proverava da li se prosledjeni email zavrsava sa
   "gmail.com" koriscenjem funkcije strtok. */
int gmail(char email[])
{
  /* Funkcija strtok "deli" nisku u podniske tako sto ih razdvaja
     na mestu na kom se nalazi prosledjeni delimiter (u ovom slucaju
     je to "@").
```

```
25      Na primer, ukoliko je email="pera.peric@gmail.com", funkcija
        deli ovu nisku na "pera.peric" i "gmail.com". */
27    char *deo = strtok(email, "@");

29    /* Kada se funkcija sledeci put pozove i pri tom pozivu se kao
        prvi argument navede NULL, tada funkcija vraca sledeci token u
       nizu,
31      a to je u ovom slucaju "gmail.com". */
      deo = strtok(NULL, "");
33
      /* Ako se email zavrsava na "gmail.com", funkcija vraca 1, a
35      u suprotnom 0. */
      return strcmp(deo, "gmail.com") == 0;
37  }

39  // /* II nacin:
    //    Funkcija proverava da li se prosledjeni email zavrsava sa
41  //    "gmail.com" koriscenjem funkcije strchr. */
    // int gmail2(char email[])
43  // {
    //   /* Pronalazi se pokazivac na znak @. */
45  //   char* desni_deo = strchr(email, '@');
    //
47  //   /* Poredi se niska koja pocinje jedan karakter posle @ sa
    //      niskom "gmail.com". */
49  //   return strcmp(desni_deo+1, "gmail.com") == 0;
    // }
51
    int main()
53  {
      /* Deklaracije potrebnih promenljivih. */
55    int n, i;
      Osoba osobe[MAKS_OSOBA];
57
      /* Ucitavanje broja osoba i provera ispravnosti ulaza. */
59    printf("Unesite broj osoba: ");
      scanf("%d", &n);
61    if (n < 0 || n >= MAKS_OSOBA)
      {
63      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
65    }

67    /* Ucitavanje podataka o osobama. */
      printf("Unesite podatke o osobama, ime, prezime i email.\n");
69    for (i = 0; i < n; i++)
        scanf("%s%s%s", osobe[i].ime, osobe[i].prezime, osobe[i].email);
71
      /* Ispis rezultata. */
73    printf("Vlasnici gmail naloga su:\n");
      for (i = 0; i < n; i++)
75      if (gmail(osobe[i].email))
```

```
      printf("%s %s\n", osobe[i].ime, osobe[i].prezime);

  return 0;
}
```

Rešenje 3.9.11

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

#define MAKS_ARTIKALA 20
#define MAKS_KORPI 100
#define MAKS_NAZIV 31

typedef struct
{
  char naziv[MAKS_NAZIV];
  int kolicina;
  float cena;
} Artikal;

typedef struct
{
  int broj_artikala;
  Artikal artikli[MAKS_ARTIKALA];
} Korpa;


/* Funkcija ucitava jedan artikal i proverava ispravnost
   ucitanih podataka. */
void ucitaj_artikal(Artikal * a)
{
  printf("Unesite artikal, naziv, kolicinu i cenu: ");
  scanf("%s%d%f", a->naziv, &a->kolicina, &a->cena);

  if (a->kolicina <= 0)
  {
    printf("Greska: neispravan unos kolicine (%d).\n", a->kolicina);
    exit(EXIT_FAILURE);
  }

  if (a->cena < 0)
  {
    printf("Greska: neispravan unos cene (%f).\n", a->cena);
    exit(EXIT_FAILURE);
  }
}

/* Funkcija ucitava podatke o jednoj potrosackoj korpi. */
void ucitaj_korpu(Korpa * k)
{
```

```
45    int i;
      printf("Unesite podatke o korpi: \n");
47
      /* Ucitavanje broja artikala u korpi. */
49    printf("Broj artikala: ");
      scanf("%d", &k->broj_artikala);
51    if (k->broj_artikala <= 0)
      {
53      printf("Greska: neispravan unos broja artikala (%d).\n",
              k->broj_artikala);
55      exit(EXIT_FAILURE);
      }
57
      /* Ucitavanje podataka o svakom artiklu. */
59    for (i = 0; i < k->broj_artikala; i++)
        ucitaj_artikal(&k->artikli[i]);
61  }

63  /* Funkcija ucitava podatke o n potrosackih korpi. */
    void ucitaj_niz_korpi(Korpa korpe[], int n)
65  {
      int i;
67    for (i = 0; i < n; i++)
        ucitaj_korpu(&korpe[i]);
69  }

71  /* Funkcija racuna ukupan racun za datu korpu. */
    float izracunaj_racun(const Korpa * k)
73  {
      int i;
75    float racun = 0;

77    for (i = 0; i < k->broj_artikala; i++)
        racun += k->artikli[i].kolicina * k->artikli[i].cena;
79
      return racun;
81  }

83  /* Funkcija ispisuje racun za datu korpu.*/
    void ispisi_racun(const Korpa * k)
85  {
      int i;
87    for (i = 0; i < k->broj_artikala; i++)
        printf("\t%s %d %.2f\n", k->artikli[i].naziv,
89            k->artikli[i].kolicina, k->artikli[i].cena);
      printf("------------------------\n");
91    printf("\tukupno: %.2f\n", izracunaj_racun(k));
    }
93
    /* Funkcija ispisuje racune za sve potrosacke korpe u nizu. */
95  void ispisi_racune_za_korpe(Korpa korpe[], int n)
    {
```

```
97    int i;
      for (i = 0; i < n; i++)
99    {
        printf("\nKorpa %d:\n", i);
101     ispisi_racun(&korpe[i]);
      }
103 }

105 /* Funkcija racuna prosecnu cenu potrosacke korpe za dati
       niz potrosackih korpi. */
107 float prosek(Korpa korpe[], int n)
    {
109   int i;
      float prosecna_cena = 0;
111
      for (i = 0; i < n; i++)
113     prosecna_cena += izracunaj_racun(&korpe[i]);

115   return prosecna_cena / n;
    }
117
    int main()
119 {
      /* Deklaracije potrebnih promenljivih. */
121   int n;
      Korpa korpe[MAKS_KORPI];
123
      /* Ucitavanje broja potrosackih korpi i provera ispravnosti
125      ulaza. */
      printf("Unesite broj potrosackih korpi:");
127   scanf("%d", &n);
      if (n < 0 || n > MAKS_KORPI)
129   {
        printf("Greska: neispravan unos.\n");
131     exit(EXIT_FAILURE);
      }
133
      /* Ucitavanje podataka o potrosackim korpama. */
135   ucitaj_niz_korpi(korpe, n);

137   /* Ispis svih racuna. */
      ispisi_racune_za_korpe(korpe, n);
139
      /* Ispis prosecne cene potrosacke korpe. */
141   printf("Prosecna cena potrosacke korpe: %.2f\n",
             prosek(korpe, n));
143
      return 0;
145 }
```

Rešenje 3.9.12

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

#define MAKS 50

typedef struct lopta {
  int poluprecnik;
  enum { plava, zuta, crvena, zelena } boja;
} Lopta;

/* Funkcija racuna zapreminu lopte. */
float zapremina(const Lopta* l)
{
  return pow(l->poluprecnik, 3) * 4 / 3 * M_PI;
}

/* Funkcija racuna zbir zapremina svih lopti u nizu. */
float ukupna_zapremina(Lopta lopte[], int n)
{
  int i;
  float ukupno = 0;

  for (i = 0; i < n; i++)
    ukupno += zapremina(&lopte[i]);

  return ukupno;
}

/* Funkcija broji lopte cija je boja jednaka boji koja je
   prosledjena kao argument funkcije. */
int broj_lopti_u_boji(Lopta lopte[], int n, unsigned boja)
{
  int br = 0;
  int i;

  for (i = 0; i < n; i++)
    if (lopte[i].boja == boja)
      br++;

  return br;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  Lopta lopte[MAKS];
  int n;
  int i;
  unsigned boja;

```

```
  /* Ucitavanje broja lopti i provera ispravnosti ulaza. */
  printf("Unesite broj lopti: ");
  scanf("%d", &n);
  if (n < 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Ucitavanje lopti u niz. */
  printf("Unesite dalje poluprecnike i boje lopti "
         "(1-plava, 2-zuta, 3-crvena, 4-zelena):\n");
  for (i = 0; i < n; i++)
  {
    printf("%d. lopta: ", i + 1);
    scanf("%d%u", &lopte[i].poluprecnik, &boja);
    if(boja < 1 || boja>4)
    {
      printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
    }
    lopte[i].boja = boja;
  }

  /* Ispis rezultata. */
  printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));
  printf("Ukupno crvenih lopti: %d\n",
         broj_lopti_u_boji(lopte, n, crvena));

  return 0;
}
```

Rešenje 3.9.13

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

#define MAKS_TACAKA 1000

typedef struct
{
  int x, y;
} Tacka;

/* Funkcija racuna rastojanje izmedju dve tacke. */
double rastojanje(const Tacka* a, const Tacka* b)
{
  return sqrt(pow(a->x - b->x, 2) + pow(a->y - b->y, 2));
}
```

```
/* Funkcija ucitava tacke poligona. */
int ucitaj_poligon(Tacka poligon[], int maks_tacaka)
{
  int i = 0;

  while (scanf("%d%d", &poligon[i].x, &poligon[i].y) != EOF)
  {
    i++;
    if(i >= maks_tacaka)
      break;
  }

  return i;
}

/* Funkcija racuna obim poligona. */
double obim_poligona(Tacka poligon[], int n)
{
  double obim = 0;
  int i;

  for (i = 0; i < n - 1; i++)
    obim += rastojanje(&poligon[i], &poligon[i + 1]);

  obim += rastojanje(&poligon[n - 1], &poligon[0]);

  return obim;
}

/* Funkcija racuna najduzu stranicu poligona. */
double maksimalna_stranica(Tacka poligon[], int n)
{
  double maks = rastojanje(&poligon[0], &poligon[n - 1]);
  double stranica;
  int i;

  for (i = 0; i < n - 1; i++) {
    stranica = rastojanje(&poligon[i], &poligon[i + 1]);
    if (stranica > maks)
      maks = stranica;
  }

  return maks;
}

/* Funkcija racuna povrsinu trougla cija su temena A, B i C. */
double povrsina_trougla(const Tacka* A, const Tacka* B, const Tacka*
    C)
{
  double a = rastojanje(B, C);
  double b = rastojanje(A, C);
  double c = rastojanje(A, B);
```

```
70    double s = (a + b + c) / 2;
72    return sqrt(s * (s - a) * (s - b) * (s - c));
}
74
/* Funkocija racuna povrsinu poligona. */
76 double povrsina_poligona(Tacka * poligon, int n)
{
78    double P = 0;
    int i;
80
    for (i = 1; i < n - 1; i++)
82      P += povrsina_trougla(&poligon[0], &poligon[i], &poligon[i + 1]);
84    return P;
}
86
int main()
88 {
    /* Deklaracije potrebnih promenljivih. */
90    int maks_tacaka, n;
    Tacka poligon[MAKS_TACAKA];
92
    /* Ucitavanje maksimalnog broja tacaka i provera ispravnosti. */
94    printf("Uneti maksimalan broj tacaka poligona: ");
    scanf("%d", &maks_tacaka);
96    if (maks_tacaka < 3 || maks_tacaka > MAKS_TACAKA)
    {
98      printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
100   }
102   /* Ucitavanje poligona. */
    n = ucitaj_poligon(poligon, maks_tacaka);
104   if (n < 3)
    {
106     printf("Greska: poligon mora imati bar tri tacke.\n");
      exit(EXIT_FAILURE);
108   }
110   /* Ispis rezultata. */
    printf("Obim poligona je %.3lf.\n",
112           obim_poligona(poligon, n));
    printf("Duzina maksimalne stranice je %.3lf.\n",
114           maksimalna_stranica(poligon, n));
    printf("Povrsina poligona je %.3lf.\n",
116           povrsina_poligona(poligon, n));
118   return 0;
}
```

Rešenje 3.9.14

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

#define MAKS 1000

typedef struct
{
  char o;
  int x;
  int y;
} Izraz;

/* Funkcija proverava da li je izraz ispravno zadat. */
int korektan_izraz(const Izraz* izraz)
{
  if (izraz->o != '+' && izraz->o != '-' &&
      izraz->o != '*' && izraz->o != '/')
  {
    printf("Greska: neispravna operacija.\n");
    return 0;
  }

  if (izraz->o == '/' && izraz->y == 0)
  {
    printf("Greska: deljenje nulom.\n");
    return 0;
  }

  return 1;
}

/* Funkcija ucitava n izraza sa standardnog ulaza. */
void ucitaj(Izraz izrazi[], int n)
{
  int i;

  printf("Unesite izraze u prefiksnoj notaciji:\n");
  for (i = 0; i < n; i++)
  {
    scanf("%c%d%d", &izrazi[i].o, &izrazi[i].x, &izrazi[i].y);
    /* Preskace se novi red koji se nalazi nakon izraza, kako bi
       naredni izraz bio ispravno ucitan. */
    getchar();

    /* Provera ispravnosti ucitanog izraza. */
    if (!korektan_izraz(&izrazi[i]))
    {
      printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
    }
```

```
  }

}

/* Funkcija racuna vrednost izraza. */
int vrednost(const Izraz* izraz)
{
  switch (izraz->o)
  {
    case '+':
      return izraz->x + izraz->y;
    case '-':
      return izraz->x - izraz->y;
    case '*':
      return izraz->x * izraz->y;
    case '/':
      return izraz->x / izraz->y;
    default:
      printf("Greska: neispravna operacija.\n");
      exit(EXIT_FAILURE);
  }
}

/* Funkcija racuna najvecu vrednost izraza. */
int najveca_vrednost(Izraz izrazi[], int n)
{
  int i;
  int maks_vrednost, tr_vrednost;

  maks_vrednost = vrednost(&izrazi[0]);

  for (i = 1; i < n; i++)
  {
    tr_vrednost = vrednost(&izrazi[i]);
    if (tr_vrednost > maks_vrednost)
      maks_vrednost = tr_vrednost;
  }

  return maks_vrednost;
}

int main()
{
  /* Deklaracije potrebnih promenljivih. */
  int n;
  Izraz izrazi[MAKS];
  int maks, trenutna_vrednost;
  float polovina;
  int i;

  /* Ucitavanje broja izraza i provera ispravnosti ulaza. */
  printf("Unesite broj izraza: ");
```

```
  scanf("%d", &n);
  if (n < 0 || n > MAKS)
  {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
  }

  /* Preskace se belina koja se unosi nakon broja izraza.
     Ovaj korak je neophodan jer se izraz zadaje u formatu
     <operacija> <operand> <operand>
     A <operacija> je tipa char i kada bi ovaj korak bio
     izostavljen, ta belina bi bila ucitana kao <operacija>
     za prvi izraz. */
  getchar();
  ucitaj(izrazi, n);

  /* Pronalazak polovine maksimalne vrednosti. */
  maks = najveca_vrednost(izrazi, n);
  printf("Maksimalna vrednost izraza:%d\n", maks);
  polovina = maks / 2.0;

  /* Ispis rezultata. */
  printf("Izrazi cija je vrednost manja od polovine maksimalne "
         "vrednosti:\n");
  for (i = 0; i < n; i++)
  {
    trenutna_vrednost = vrednost(&izrazi[i]);
    if (trenutna_vrednost < polovina)
    {
      printf("%d %c %d = %d\n", izrazi[i].x, izrazi[i].o,
             izrazi[i].y, trenutna_vrednost);
    }
  }

  return 0;
}
```

Rešenje 3.9.15

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

#define MAKS_STEPEN 10
#define MAKS_POLINOMA 100

typedef struct
{
  int stepen;
  float koef[MAKS_STEPEN + 1];
} Polinom;
```

```c

/* Funkcija ucitava podatke o polinomima. */
int ucitaj(Polinom niz[])
{
  int i=0,j;

  while(i<MAKS_POLINOMA)
  {
    printf("Unesite stepen: ");
    if (scanf("%d", &(niz[i].stepen)) == EOF)
      break;

    if (niz[i].stepen > MAKS_STEPEN || niz[i].stepen < 0)
    {
      printf("Greska: neispravan unos stepena.\n");
      exit(EXIT_FAILURE);
    }

    printf("Unesite koeficijente polinoma:\n");
    for (j = 0; j <= niz[i].stepen; j++)
      scanf("%f", &(niz[i].koef[j]));

    i++;
  }
  return i;
}

/* Prvi monom je specijalan jer se ispred njega ne vrsi
   eksplicitan ispis znaka.
   Na primer, za polinom x + 3*x^2, prvi monom je x.
   Svakom sledecem monomu (u ovom slucaju samo 3*x^2)
   u ispisu prethodi znak (+ ili -).
   Funkcija ispisuje prvi monom. */
void ispis_prvog_monoma(float koef, int stepen)
{
    printf("%.2f", koef);

    if (stepen == 1)
      printf("*x ");
    else if (stepen > 1)
      printf("*x^%d ", stepen);
}

/* Funkcija ispisuje monom koji nije prvi. */
void ispis_monoma(float koef, int stepen)
{
  /* Monomi ciji je koeficijent nula se ne ispisuju. */
  if (koef != 0)
  {
    /* Ispis znaka. */
    if (koef > 0)
      printf("+ ");
```

```
    else
      printf("- ");

    /* Ispis koeficijenta. */
    printf("%.2f", fabs(koef));

    /* Ispis ostatka. */
    if (stepen == 1)
      printf("*x ");
    else if (stepen > 1)
      printf("*x^%d ", stepen);
  }
}

/* Funkcija ispisuje ceo polinom p. */
void ispis(const Polinom * p)
{
  int i;

  /* Vrsi se ispis prvog monoma. Posto je moguce da prvi monom
     ima koeficijent 0, trazi se prvi monom sa ne-nula
     koeficijentom. */
  for(i=0; i <= p->stepen; i++)
  {
    if(p->koef[i] != 0)
    {
      ispis_prvog_monoma(p->koef[i], i);
      break;
    }
  }

  /* Ispis ostalih monoma. Nastavlja se od mesta gde se stalo u
     prethodnoj petlji i iz tog razloga je preskocen korak
     inicijalizacije brojaca i. */
  for (;i <= p->stepen; i++)
      ispis_monoma(p->koef[i], i);

  printf("\n");
}

/* Funkcija racuna integral polinoma p. */
void integral(const Polinom * p, Polinom * integ)
{
  int i;

  integ->stepen = p->stepen + 1;
  integ->koef[0] = 0;

  for (i = 1; i <= integ->stepen; i++)
    integ->koef[i] = (float) p->koef[i - 1] / i;
}
```

```
int main()
{
  /* Deklaracija potrebnih promenljivih. */
  Polinom polinomi[MAKS_POLINOMA], integ;
  int n, i;

  /* Ucitavanje polinoma. */
  n = ucitaj(polinomi);

  /* Ispis integrala. */
  printf("\n\nIntegrali su:\n");
  for (i = 0; i < n; i++)
  {
    integral(&polinomi[i], &integ);
    ispis(&integ);
  }

  return 0;
}
```

4

Ulaz i izlaz programa

4.1 Datoteke

Zadatak 4.1.1 Napisati program koji prepisuje sadržaj datoteke *ulaz.txt* u datoteku *izlaz.txt* karakter po karakter. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: v301, karakter po karakter, prepisivanje, fiksni naziv

Primer 1

```
ULAZ.TXT
 Danas je 21. mart.
 To je prvi dan proleca.
IZLAZ.TXT
 Danas je 21. mart.
 To je prvi dan proleca.
```

Primer 2

```
ULAZ.TXT
 Ispit iz Programiranja 1 je
 zakazan za 10. jun.
IZLAZ.TXT
 Ispit iz Programiranja 1 je
 zakazan za 10. jun.
```

Primer 3

```
ULAZ.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
 Greska: datoteka ulaz.txt
 ne postoji.
```

[Rešenje 4.1.1]

Zadatak 4.1.2 Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p302, karakter po karakter, prepisivanje, fiksni naziv

Primer 1

```
ULAZ.TXT
  Volim programiranje.
IZLAZ.TXT
  Vipgmae
```

Primer 2

```
ULAZ.TXT
  abcdefghi
  123456789
IZLAZ.TXT
  adg
  147
```

Primer 3

```
ULAZ.TXT
  U Beogradu ce biti
  suncan i lep
  dan.
IZLAZ.TXT
  Ueruei
  nn pa
```

[Rešenje 4.1.2]

Zadatak 4.1.3 Napisati program koji šifrira sadržaj datoteke *podaci.txt* tako što svako slovo ciklično zamenjuje njegovim prethodnikom suprotne veličine i upisuje u datoteku *sifra.txt*. Na primer, `b` se zamenjuje sa `A`, `B` sa `a`, `a` sa `Z`, `A` sa `z`, itd. Ostali karakteri ostaju nepromenjeni. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3id17, karakter po karakter, prepisivanje, fiksno ime, nema resenje

Primer 1

```
PODACI.TXT
  ABC.123.xyz
SIFRA.TXT
  zab.123.WXY
```

Primer 2

```
PODACI.TXT
  a=x+y;
  x=b+5;
SIFRA.TXT
  Z=W+X;
  W=A+5;
```

Primer 3

```
PODACI.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
  Greska: datoteka podaci.txt
  ne postoji.
```

[Rešenje 4.1.3]

Zadatak 4.1.4 Sa standarnog ulaza učitavaju se imena dve datoteke i jedan karakter. Napisati program koji prepisuje sadržaj prve datoteke u drugu na sledeći način:

- ukoliko je učitan karakter `u`, sva mala slova zamenjuje velikim
- ukoliko je učitan karakter `l`, sva velika slova zamenjuje malim

Maksimalna dužina naziva datoteka je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3iv3, karakter po karakter, prepisivanje, ima resenje

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 ZA GREŠKE:
  Greska: neispravan poziv.
```

[Rešenje 4.1.4]

Zadatak 4.1.5 Napisati program koji za dve datoteke čija se imena unose sa standarnog ulaza, radi sledeće:

- za svaku cifru u prvoj datoteci, u drugu datoteku upisuje 0
- za svako slovo u prvoj datoteci, u drugu datoteku upisuje 1
- za sve ostale karaktere u prvoj datoteci, u drugu datoteku upisuje 2

Maksimalna dužina naziva datoteka je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3id16, karakter po karakter, prepisivanje, nema resenje
Jovana: Dodat je deo zadatka za slucaj da datoteka ne postoji.
Jovana: ispraviti ovoliki prored izmedju stavki u okviru itemize (to se menja u zavisnosti od prostora na stranici)

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  prva.dat druga.dat
PRVA.DAT
  abc.123.[]
  567.ABC.
DRUGA.DAT
  111200022220002111222
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  ulaz.txt izlaz.txt
ULAZ.TXT
  18. februar 2019.
IZLAZ.TXT
  11220000000211112
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
  in.txt out.txt
IN.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
  Greska: datoteka in.txt
  ne postoji.
```

[Rešenje **??**]

Zadatak 4.1.6 Napisati program koji prebrojava mala slova u datoteci *test.txt* i dobijeni rezultat ispisuje na standardni izlaz. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p301, karakter po karakter, brojanje, fiksni naziv

Primer 1

```
TEST.TXT
  Abcd EFGH+ijKLMN

IZLAZ:
  Broj malih slova je: 5
```

Primer 2

```
TEST.TXT
  PrograMiranje

IZLAZ:
  Broj malih slova je: 11
```

Primer 3

```
TEST.TXT
  123456
  ABCDEF

IZLAZ:
  Broj malih slova je: 0
```

[Rešenje 4.1.6]

Zadatak 4.1.7 Napisati program koji u datoteci čije se ime unosi sa standardnog ulaza prebrojava koliko se puta svaka cifra pojavljuje i na standardni izlaz ispisuje cifru sa najvećim brojem pojavljivanja. Ukoliko ima više takvih cifara, ispisati sve. Ukoliko datoteka ne sadrži nijednu cifru, ispisati odgovarajuću poruku. Maksimalna dužina naziva datoteka je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p2id14, karakter po karakter, brojanje, nema resenje (domaci)

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  ulaz.txt
ULAZ.TXT
  danas je lep dan
  i ja zelim
  da postanem programer
IZLAZ:
  Datoteka ne sadrzi cifre.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  prva.dat
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
  1 22 333.444
IZLAZ:
  3 4
```

[Rešenje 4.1.7]

Zadatak 4.1.8 Napisati program koji u datoteci čije je ime dato kao argument komandne linije proverava da li su zagrade pravilno uparene. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3x2, argumenti, karakter po karakter, brojanje, nema resenja

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 ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.8]

Zadatak 4.1.9 Napisati program koji prebrojava slova i cifre u datoteci.

a) Napisati C funkciju `int unesi_skup(char s[], FILE* f)` kojom se unosi skup elemenata iz datoteke `F`. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiđe na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspešno učitani.

b) Napisati funkciju `void prebroj(char s[], int *br_slova,int *br_cifara)` kojom se određuje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.

c) Napisati program koji koristeći prethodne funkcije prebrojava cifre i slova u datoteci čije se ime unosi kao argument komandne linije i ispisuje dobijene vrednosti na standardni izlaz. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: x4, karakter, brojanje, argumenti, nema resenje

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 skup4.txt
SKUP4.TXT
 11.2.2019.
IZLAZ:
 broj slova: 0
 broj cifara: 2
```

Primer 5

```
POKRETANJE: ./a.out skup5.txt
SKUP4.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
 Greska: datoteka skup4.txt
 ne postoji.
```

Primer 6

```
POKRETANJE: ./a.out
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.9]

Zadatak 4.1.10 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. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p308, rec po rec, fiksno ime

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite rec:  abcde

ROTACIJE.TXT
 abcde
 bcdea
 cdeab
 deabc
 eabcd
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite rec:  1234

ROTACIJE.TXT
 1234
 2341
 3412
 4123
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite rec:  a=3*x+5;

ROTACIJE.TXT
 a=3*x+5;
 =3*x+5;a
 3*x+5;a=
 *x+5;a=3
 x+5;a=3*
 +5;a=3*x
 5;a=3*x+
 ;a=3*x+5
```

[Rešenje 4.1.10]

Zadatak 4.1.11 Sa standarnog ulaza se učitava ime datoteke i nenegativan ceo broj k. Napisati program koji učitava reči iz datoteke, i svaku pročitanu reč rotira za k mesta u levo i da tako dobijenu reč upisuje u datoteku čije je ime *rotirano.txt*. Maksimalna dužina naziva datoteke je 20 karaktera. Može se pretpostaviti da datoteka sadrži samo slova i beline i da je maksimalna dužina jedne reči u datoteci 100 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3id18, rec po rec, ime jedne datoteke sa standardnog ulaza a drugo fiksno ime, nema resenje

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite ime datoteke:  ulaz.txt
 Unesite broj k:  3
ULAZ.TXT
 jedan dva
 tri cetiri
ROTIRANO.TXT
 anjed dva tri iricet
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite ime datoteke:  in.dat
 Unesite broj k:  5
IN.DAT
 Popodne ce biti kise
ROTIRANO.TXT
 nePopod ec itib isek
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite ime datoteke:  input.txt
 Unesite broj k:  0
INPUT.TXT
 Popodne ce
 biti kise
ROTIRANO.TXT
 Popodne ce biti kise
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite ime datoteke:  tekst.dat
 Unesite broj k:  7
TEKST.DAT NE POSTOJI
IZLAZ ZA GREŠKE:
 Greska: datoteka tekst.dat ne postoji.
```

[Rešenje 4.1.11]

Zadatak 4.1.12 Napisati program koji iz datoteke čije se ime zadaje sa standardnog ulaza prepisuje reči na standardni izlaz a one reči koje sadrže prvu reč iz datoteke i podvlaku upisuje u datoteku *rez.txt*. Maksimalna dužina naziva datoteke je 20 karaktera a reči u datoteci 50 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3iv6, rec po rec, fiksno ime, ima resenje

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 dat1.txt
DAT1.TXT
 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
 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
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 dat3.txt
DAT3.TXT
 4 abc 1234 (5+3)*12_4 11-k
IZLAZ:
 4 abc 1234 (5+3)*12_4 11-k
REZ.TXT:
 (5+3)*12_4
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 dat4.txt
DAT4.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
 Greska: datoteka dat4.txt ne postoji.
```

[Rešenje 4.1.12]

Zadatak 4.1.13 Napisati program koji iz datoteke *razno.txt* u datoteku *palindromi.txt* prepisuje sve palindrome. Podrazumeva se da je reč palindrom

ako se čita isto sa leve i desne strane bez obzira na veličinu slova. Maksimalna dužina reči je 200 karaktera a maksimalan broj reči nije poznat. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3id20, rec po rec, fiksno ime, nema resenje

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
  Oko kapAk radar
```

Primer 3

```
RAZNO.TXT
  ovde nema palindroma
PALINDROMI.TXT:
```

Primer 4

```
RAZNO.TXT
  Ana voli Milovana.
PALINDROMI.TXT:
  Ana
```

[Rešenje 4.1.13]

Zadatak 4.1.14 U datoteci čije se ime zadaje sa standardnog ulaza nalazi se broj n ($n \leq 256$), a zatim i n reči dužine najviše 50 karaktera. Napisati program koji učitava reči iz datoteke u niz i:

(a) ispisuje ga na standardni izlaz

(b) iz niza uklanja sve duplikate i upisuje transformisani niz u datoteku *rez.txt*

Maksimalna dužina naziva datoteka je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3iv5, rec po rec, fiksno ime, ime se unosi sa standardnog ulaza, ima resenje

Jovana: ima dve varijante resenja, koristi se u obe dvodimenzioni niz

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
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 dat3.txt
DAT3.TXT
 2
 4 abc 1234 (5+3)*12.4 11-k
IZLAZ:
 abc 1234 (5+3)*12.4 11-k
REZ.TXT:
 abc 1234 (5+3)*12.4 11-k
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 dat4.txt
DAT4.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
 Greska: datoteka dat4.txt ne postoji.
```

[Rešenje 4.1.14]

Zadatak 4.1.15 Napisati program koji u datoteku *izlaz.txt* prepisuje sve reči iz datoteke *ulaz.txt* čiji je zbir ASCII kodova karaktera strogo veći od 1000. Reči su odvojene praznina i nisu duže od 200 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3id19, karakter po karakter, prepisivanje, fiksno ime, nema resenje

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 4.1.15]

Zadatak 4.1.16 U datoteci čije se ime zadaje kao prvi argument komandne linije nalazi se ceo pozitivan broj n a zatim i n celih brojeva. Napisati program koji na standardni izlaz ispisuje koliko k-tocifrenih brojeva postoji u datoteci, pri čemu se pozitivan ceo broj k zadaje kao drugi argument komandne linije. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: v303, brojevi, argumenti

Primer 1

```
POKRETANJE: ./a.out ulaz.txt 2
ULAZ.TXT
 6
 15
 193
 -27
 9790
 35
 1
IZLAZ:
 3
```

Primer 2

```
POKRETANJE: ./a.out in.dat 5
ULAZ.TXT
 4
 15
 193
 -27
 9790
IZLAZ:
 0
```

Primer 3

```
POKRETANJE: ./a.out in.txt 3
IN.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
 Greska: datoteka in.txt ne postoji.
```

Primer 4

```
POKRETANJE: ./a.out in.txt
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.16]

Zadatak 4.1.17 Napisati program koji na standardni izlaz ispisuje maksimum brojeva iz datoteke *brojevi.txt*. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p305, brojevi

Primer 1

```
BROJEVI.TXT
 2.36 -16.11 5.96 8.88
 -265.31 54.96 38.4

IZLAZ:
 Najveci broj je: 54.96
```

Primer 2

```
BROJEVI.TXT
 10.5 183.111 -90.2 3.167

IZLAZ:
 Najveci broj je: 183.111
```

Primer 3

```
BROJEVI.TXT
 -62.7 -190.2 -2.3 -1000
 -198.25 -8

IZLAZ:
 Najveci broj je: -2.3
```

[Rešenje 4.1.17]

Zadatak 4.1.18 Prvi red datoteke *matrice.txt* sadrži dva 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)`. Pretpostaviti da je sadržaj datoteke ispravan. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3iv4, matrice, ima resenje

Primer 1

```
MATRICE.TXT
 3 4
 1 2 3 4
 7 2 15 -3
 -1 3 1 3
IZLAZ:
 (1, 0, 7)
 (1, 2, 15)
```

Primer 2

```
MATRICE.TXT
 2 2
 1 1
 -2 2
IZLAZ:
 (0, 0, 1)
 (0, 1, 1)
```

Primer 3

```
MATRICE.TXT
 1 4
 9 3 5 2
IZLAZ:
 (0, 2, 5)
```

[Rešenje 4.1.18]

Zadatak 4.1.19 Prvi red datoteke *ulaz.txt* sadrži dva cela broja između 2 i 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. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3x6, matrice, fiksno ime, nema resenje
Jovana: preciziran tekst, uskladiti resenje

Primer 1

```
ULAZ.TXT
 3 4
 1 2 3 4
 7 2 15 -3
 -1 3 1 3
IZLAZ:
 (3, 15, 4, -3)
 (7, -1, 2, 3)
 (2, 3, 15, 1)
 (15, 1, -3, 3)
```

Primer 2

```
MATRICE.TXT
 2 2
 1 1
 -2 2
IZLAZ:
```

Primer 3

```
MATRICE.TXT
 1 4
 9 3 5 2
IZLAZ ZA GREŠKE:
 Greska: neispravna dimenzija.
```

[Rešenje 4.1.19]

Jovana: Prva tri zadatka nisu resena, trebalo bi resiti bar prvi

Zadatak 4.1.20 U datoteci *tacke.txt* se nalazi broj tačaka, a zatim u posebnim linijama za svaku tačku njene x i y koordinate. Napisati program koji u datoteku *rastojanja.txt* upisuje rastojanje svake od učitanih tačaka od koordinatnog početka, a na standardni izlaz koordinate tačke koja je od njega najudaljenija. Koristiti strukturu *Tacka* sa poljima x i y, kao i funkciju kojom

se računa rastojanje. Maksimalan broj tačaka u datoteci je 50. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p307, strukture, fiksno ime

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 2

```
TACKE.TXT
 -2
 0 0
 9 -8

IZLAZ ZA GREŠKE:
 Greska: neispravan broj tacaka.
```

[Rešenje 4.1.20]

Zadatak 4.1.21 Data je struktura kojom se opisuje trodimenzioni vektor:

```
typedef struct{
   int x;
   int y;
   int z;
} vektor;
```

U datoteci *vektori.txt* nalazi se nepoznati broj vektora (najviše 200). Napisati program koji učitava vektore iz ove datoteke i na standardni izlaz ispisuje koordinate vektora sa najvećom dužinom. Dužina vektora se izračunava po formuli:

$$|v| = \sqrt{x^2 + y^2 + z^2}$$

U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3x5, struktura, fiksno ime, nema resenje

Primer 1

```
VEKTORI.TXT
 2
 4 -1 7
 3 1 2
IZLAZ:
 4 -1 7
```

Primer 2

```
VEKTORI.TXT
 670
IZLAZ ZA GREŠKE:
 Greska: neispravan broj
     vektora.
```

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
```

Primer 5

```
VEKTORI.TXT
 4
 3 0 1
 4 5 2
 1
 2 -1 2
IZLAZ ZA GREŠKE:
 Greska: neispravan
 sadrzaj datoteke.
```

Primer 6

```
VEKTORI.TXT

IZLAZ ZA GREŠKE:
 Greska: neispravan
 sadrzaj datoteke.
```

[Rešenje 4.1.21]

Zadatak 4.1.22 Data je struktura koja opisuje pravougaonik dužinama svojih stranica i imenom:

```
typedef struct{
   unsigned int a, b;
   char ime[5];
}Pravougaonik;
```

Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava podatke o pravougaonicima (nepoznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine među pravougaonicima koji nisu kvadrati. Maksimalan broj pravougaonika je 200. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3x1, strukture, argumenti, nema resenja

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 empty.dat
EMPTY.DAT
IZLAZ:
```

[Rešenje 4.1.22]

Zadatak 4.1.23 U prvom redu datoteke *studenti.txt* se nalazi broj studenata, a zatim u posebnim linijama za svakog studenta korisničko ime na Alasu i poslednjih pet ocena koje je dobio. Napisati program koji pronalazi studenta koji je ostvario najbolji uspeh i ispisuje njegove podatke. Ukoliko više studenata ima maksimalni prosek, ispisati prvog. Pretpostaviti da broj studenata neće biti veći od 100. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p306, strukture, fiksno ime

Primer 1

```
STUDENTI.TXT
 5
 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
```

Primer 2

```
STUDENTI.TXT
 3
 mr16156 10 9 9 10 10
 mi17234 9 9 10 10 10
 ml17084 9 8 8 8 8

IZLAZ:
 korisnicko ime: mr16156, prosek ocena: 9.6
```

[Rešenje 4.1.23]

Zadatak 4.1.24 Kreirati strukturu *Student* koja sadrži:

- *ime_i_prezime* (u polju se čuva ime i prezime studenta, napr. "Marko Markovic", maksimalna dužina polja je 100 karaktera),
- *ocene* (sadrži najviše 10 ocena studenta)
- *br_ocena* (ukupan broj ocena za studenata)
- *prosek* (prosečna ocena)

U datoteci čije se ime zadaje kao argument komandne linije se nalaze podaci o studentima. Za svakog studenta dato je ime, prezime i niz ocena koji se završava nulom. Svi podaci su razdvojeni razmacima. Napisati program koji učitava podatke o studentima i na standardni izlaz ispisuje podatke za studenta sa najvećim prosekom (prosek ispisati na 2 decimale). Maksimalan broj studenta je 100. UPUTSTVO: *Ime i prezime studenta se mogu pročitati pomoću specifikatora %s a potom se za kreiranje niske ime_i_prezime u traženom formatu može iskoristiti funkcija strcat.* U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p3x3, struktura, argumenti, nema resenja

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 0
IZLAZ:
 Jelena Jankovic 10 10 10 0 10.00
```

Primer 2

```
POKRETANJE: ./a.out
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.24]

Zadatak 4.1.25 Imena ulazne i izlazne datoteke se redom navode kao argumenti komandne linije. U ulaznoj datoteci se nalaze podaci o razlomcima: u prvom redu se nalazi broj razlomaka , a u svakom sledećem redu brojilac i imenilac jednog razlomka. Definisati strukturu koja opisuje razlomak i napisati program koji učitava niz razlomaka iz datoteke, a potom:

(a) ukoliko je navedena opcija x, upisati u izlaznu datoteku recipročni razlomak za svaki razlomak iz niza (npr. za 2/3 treba upisati 3/2)

(b) ukoliko je navedena opcija y, upisati u izlaznu datoteku realnu vrednost recipročnog razlomka svakog razlomka iz niza (npr. za 2/3 treba upisati 1.5)

Pretpostaviti da se u ulaznoj datoteci nalazi najviše 100 razlomaka. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: v305, strukture, argumenti, opcije

Primer 1

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt -x
ULAZ.TXT
 4
 1 5
 19 3
 -2 7
 97 90
IZLAZ.TXT
 5/1
 3/19
 -7/2
 90/97
```

Primer 2

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt -y
ULAZ.TXT
 4
 1 5
 19 3
 -2 7
 97 90
IZLAZ.TXT
 5.000000
 0.157894
 -3.500000
 0.927835
```

Primer 3

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt -y
ULAZ.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
 Greska: datoteka ulaz.txt ne postoji.
```

Primer 4

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.25]

Zadatak 4.1.26 Definisati strukturu `Automobil` koja sadrži marku, model i cenu. Napisati program koji iz datoteke čije se ime zadaje sa standardnog ulaza učitava broj automobila i podatke za svaki automobil i zatim:

(a) ispisuje prosečnu cenu po marki kola

(b) za maksimalnu cenu koju je kupac spreman da plati, a koja se zadaje kao argument komandne linije, ispisuje automobile u tom cenovnom rangu zajedno sa prosečnom cenom odgovarajuće marke

Pretpostaviti da se model i marka sastoje od jedne reči i da svaka od njih sadrži najviše 30 karaktera kao i da se u datoteci nalaze podaci za najviše 100 automobila. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: v306, strukture, argumenti

Primer 1

```
POKRETANJE: ./a.out 4000
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT
  7
  renault twingo 2900
  renault megan 6250
  renault clio 3650
  dacia logan 5400
  dacia sandero 7800
  fiat bravo 4900
  fiat linea 4290
IZLAZ:
  Informacije o prosecnoj
  ceni po markama:
  renault 4266.67 3
  dacia 6600.00 2
  fiat 4595.00 2
  Kola u vasem cenovnom rangu:
  renault twingo 4266.67
  renault clio 4266.67
```

Primer 2

```
POKRETANJE: ./a.out 5000
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
  Greska: datoteka dat1.txt
  ne postoji.
```

Primer 3

```
POKRETANJE: ./a.out
IZLAZ ZA GREŠKE:
  Greska: neispravan poziv.
```

[Rešenje 4.1.26]

Zadatak 4.1.27 Napisati program koji u datoteci čije se ime navodi kao argument komandne linije određuje liniju maksimalne dužine i ispisuje je na standarni izlaz. Ukoliko ima više takvih linija, ispisati onu koja je leksikografski prva. Pretpostaviti da su linije maksimalne dužine 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: v302, linije, duzina, argumenti

Primer 1

```
POKRETANJE: ./a.out test.txt
TEST.TXT
 Danas je veoma hladno decembarsko
 popodne. Ne pada sneg, kazu mozda
 ce sutra.

IZLAZ:
 Danas je veoma hladno decembarsko
```

Primer 2

```
POKRETANJE: ./a.out in.txt
IN.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
 Greska: datoteka
 in.txt ne postoji.
```

[Rešenje 4.1.27]

Zadatak 4.1.28 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. Maksimalna dužina linije je 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p303, duzina, argumenti, linije

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 ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.28]

Zadatak 4.1.29 U datoteci čije se ime navodi kao prvi argument komandne linije navedena je reč r i niz linija. Napisati program koji u datoteku čije se ime navodi kao drugi argument komandne linije upisuje sve linije u kojima se reč r pojavljuje bar n puta gde je n pozitivan ceo broj koji se unosi sa standardnog ulaza. Računaju se i samostalna pojavljivanja reči r i pojavljivanja u okviru neke

druge reči. Ispis treba da bude u formatu `broj_pojavljivanja:linija`. Maksimalna dužina reči je 100 karaktera, a maksimalna dužina linije je 500 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: v304, linije, podniska, argumenti

Primer 1

```
POKRETANJE: ./a.out input.txt output.txt
INTERAKCIJA SA PROGRAMOM:
 Unesite prirodan broj: 2
INPUT.TXT
 sto
 stolica lampa
 postotak Stopiranje stopa
 presto Ostoja stotina prostorija
OUTPUT.TXT
 2: postotak Stopiranje stopa
 4: presto Ostoja stotina prostorija
```

Primer 2

```
POKRETANJE: ./a.out input.txt output.txt
INTERAKCIJA SA PROGRAMOM:
 Unesite prirodan broj: 3
INPUT.TXT
 red
 redar za ovu nedelju
 redosled ured
 odrediti raspored
OUTPUT.TXT
```

Primer 3

```
POKRETANJE: ./a.out in.txt out.txt
IN.TXT NE POSTOJI

INTERAKCIJA SA PROGRAMOM:
IZLAZ ZA GREŠKE:
 Greska: datoteka in.txt ne postoji.
```

Primer 4

```
POKRETANJE: ./a.out in.txt

INTERAKCIJA SA PROGRAMOM:
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.29]

Zadatak 4.1.30 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. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p304, linije, podniska

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 4.1.30]

Zadatak 4.1.31 Napisati program koji linije koje 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. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: p309, linije, prepisivanje, opcije, fiksno ime

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:
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

Primer 4

```
POKRETANJE: ./a.out
INTERAKCIJA SA PROGRAMOM:
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.31]

Zadatak 4.1.32 Napisati program koji poredi dve datoteke i ispisuje redni broj linija u kojima se datoteke razlikuju. Imena datoteka se zadaju kao argumenti komandne linije. Pretpostaviti da je maksimalna dužina linije 200 karaktera. Linije brojati počevši od 1. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Jovana: iv10, linije, poredjenje, argumenti, ima resenje

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 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
```

Primer 4

```
POKRETANJE: ./a.out prva.dat
IZLAZ ZA GREŠKE:
 Greska: neispravan poziv.
```

[Rešenje 4.1.32]

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;
12  FILE *ulaz, *izlaz;

14  /*
       Promenljive ulaz i izlaz predstavljaju
16     pokazivace na ugradjenu strukturu FILE.
       Unutar ove strukture nalaze se polja neophodna
18     za rad sa datotekama.

20     Kada zelimo da radimo sa nekom datotekom,
       moramo je prvo otvoriti. Ugradjena funkcija
22     fopen(dat, mode) otvara datoteku sa nazivom
       dat. Datoteka moze biti otvorena za citanje,
24     pisanje ili nadovezivanje, sto odredjuje
       argument mode koji moze imati vrednost "r" (read),
26     "w"(write) ili "a"(append).
    */
28
    ulaz=fopen("ulaz.txt","r");
30
    /*
32     Do neuspesnog otvaranja datoteke moze doci
       ukoliko ne postoji datoteka sa datim nazivom
34     ili je putanja do datoteke pogresna. U tom
       slucaju, funkcija fopen vraca pokazivac na NULL
36     i tada treba prijaviti gresku. Datoteka stderr
       predstavlja standardnu datoteku u koju se upisuju
38     greske. Stderr je podrazumevano postavljen
       na standardni izlaz.
40
       Ugradjena funkcija exit prouzrokuje zavrsetak programa.
42     Argument ove funkcije je jedna od konstanti definisanih
       u biblioteci stdlib.h koje pokazuju da li se program
44     zavrsio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).

46  */
    if(ulaz==NULL)
48  {
       fprintf(stderr,"error fopen(): Neuspelo otvoranje datoteke ulaz
     .txt za citanje.\n");
50     exit(EXIT_FAILURE);
    }
52
    izlaz= fopen("izlaz.txt", "w");
54  if(izlaz==NULL)
    {
56     fprintf(stderr,"error fopen():  Neuspelo otvoranje datoteke
     izlaz.txt za citanje.\n");
       exit(EXIT_FAILURE);
58  }

60  /*
```

```
      Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
      Povratna vrednost ove funkcije je ascii kod unetog
      karaktera.

      Funkcija fputc ispisuje karakter c u datoteku izlaz.
   */

   while((c=fgetc(ulaz))!=EOF)
      fputc(c,izlaz);

   /*
      Nakon zavrsetka rada sa datotekama, neophodno ih je
      zatvoriti pomocu ugradjene funkcije fclose.
   */
   fclose(ulaz);
   fclose(izlaz);
   return 0;
}
```

Rešenje 4.1.2

```
/* Napisati program koji prepisuje svaki treci karakter datoteke ulaz
    :txt u datoteku izlaz.txt */

#include<stdio.h>

int main(){

  FILE *in, *out;
  int c;
  int rbr_karaktera;


  /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
    uspesno otvorili*/
  in = fopen("ulaz.txt", "r");
  if(in == NULL){
  printf("Greska!");
  return 0;
  }

  /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo dali smo je
    uspesno otvorili*/
  out = fopen("izlaz.txt", "w");
  if(out == NULL){
  printf("Greska!");
  return 0;
  }

  /* Inicijalizujemo redni broj karaktera koji se cita */
  rbr_karaktera=0;
```

```
29    /* Citamo karakter po karakter iz datoteke sve dok ne stignemo do
        kraja datoteke */
      while((c=fgetc(in)) != EOF){
31
        /* Ukoliko je procitani karakter na poziciji koja je deljiva sa 3
         prepisujemo ga */
33      if(rbr_karaktera%3==0)
        fputc(c, out);
35
      /* Uvecavamo redni broj karaktera */
37    rbr_karaktera++;
      }
39
      /*Zatvaramo obe datoteke koje smo otvorili*/
41    fclose(out);
      fclose(in);
43    return 0;
  }
```

Rešenje 4.1.3

```
  #include <stdio.h>
2 #include <stdlib.h>
  #include <ctype.h>
4
  int main()
6 {
    /* Otvaranje datoteka za citanje i pisanje. */
8   FILE *f = fopen("plain.txt", "r");
    FILE *g = fopen("sifra.txt", "w");
10  char c;

12  if (f == NULL) {
      printf("Ne postoji datoteka plain.txt\n");
14    exit(EXIT_FAILURE);
    }
16
    if (g == NULL) {
18    printf("Nije moguce kreirati datoteku sifra.txt\n");
      exit(EXIT_FAILURE);
20  }

22  /* Karakter po karakter se ucitava iz datoteke. */
    while((c = fgetc(f)) != EOF) {
24    /* Karakter koji je malo slovo se pretava u veliko slovo. */
      if (islower(c)) {
26      c = toupper(c);
        if (c == 'A')
28        c = 'Z';
        /* Karakter postavljamo na prethodni. */
```

```
30        else
            c = c - 1;
32      }
        else if (isupper(c)) {
34        c = tolower(c);
          if (c == 'a')
36          c = 'z';
          else
38          c = c - 1;
        }
40
        fputc(c, g);
42    }


44


46    fclose(f);
      fclose(g);
48    return 0;
  }
```

Rešenje ??

```
1  #include<stdio.h>
   #include<stdlib.h>
3  #include<ctype.h>

5  #define MAX_NAZIV 21

7  void greska(){
     fprintf(stderr, "-1\n");
9    exit(EXIT_FAILURE);
   }
11
   int main(){
13   char ime1[MAX_NAZIV], ime2[MAX_NAZIV];
     char c;
15
     /*ulaz.txt izlaz.txt u*/
17   scanf("%s%s %c", ime1, ime2, &c);
     if(c != 'u' && c != 'l')
19     greska();

21   FILE* ulaz, *izlaz;
     ulaz = fopen(ime1, "r");
23   if(ulaz == NULL)
       greska();
25
     izlaz = fopen(ime2, "w");
27   if(izlaz == NULL)
       greska();
```

```
29
   char karakter;
31 if(c == 'u')
   {
33   while((karakter = fgetc(ulaz)) != EOF)
       fputc(toupper(karakter), izlaz);
35 }else{
     while((karakter = fgetc(ulaz)) != EOF)
37     fputc(tolower(karakter), izlaz);
   }
39 fclose(ulaz);
   fclose(izlaz);
41 return 0;
}
```

Rešenje ??

```
   #include <stdio.h>
2  #include <stdlib.h>
   #include <ctype.h>
4
   #define DUZINA 21
6
   int main()
8  {
     /* Otvaranje datoteka za citanje i pisanje. */
10   FILE *ulaz, *izlaz;
     char c;
12   char ime_datoteke1[DUZINA], ime_datoteke2[DUZINA];

14   printf("Unesi ime datoteke: ");
     scanf("%s", ime_datoteke1);
16
     printf("Unesi ime datoteke: ");
18   scanf("%s", ime_datoteke2);

20   ulaz = fopen(ime_datoteke1, "r");
     izlaz = fopen(ime_datoteke2, "w");
22
     if (ulaz == NULL) {
24     printf("Ne postoji datoteka plain.txt\n");
       exit(EXIT_FAILURE);
26   }

28   if (izlaz == NULL) {
       printf("Nije moguce kreirati datoteku sifra.txt\n");
30     exit(EXIT_FAILURE);
     }
32
     while((c = fgetc(ulaz)) != EOF) {
34     if (isdigit(c))
```

```
      fprintf(izlaz, "0");
    else if (isalpha(c))
      fprintf(izlaz, "1");
    else
      fprintf(izlaz, "2");
  }


  fclose(ulaz);
  fclose(izlaz);
  return 0;
}
```

Rešenje 4.1.6

```
/* Napisati program koji prebrojava mala slova u datoteci test.txt */

#include<stdio.h>

int main(){

  FILE* in;
  int c, broj_malih=0;

  /*Otvaramo datoteku test.txt za citanje i proveravamo da li smo je
    uspesno otvorili*/
  in = fopen("test.txt", "r");
  if(in == NULL){
  printf("Greska!");
  return 0;
  }

  /*Citamo karakter po karakter, i ukoliko je procitani
   *karakter  malo slovo, uvecevamo brojac*/
  while((c=fgetc(in))!=EOF){
  if(islower(c))
    broj_malih++;
  }

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

  /*Zatvaramo datoteku*/
  fclose(in);

  return 0;
}
```

Rešenje 4.1.7

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

#define DUZINA 21

int main()
{
  /* Otvaranje datoteka za citanje i pisanje. */
  FILE *ulaz;
  char c;
  char ime_datoteke[DUZINA];
  int niz[10];
  int i, max;

  printf("Unesi ime datoteke: ");
  scanf("%s", ime_datoteke);

  ulaz = fopen(ime_datoteke, "r");

  if (ulaz == NULL) {
    printf("Ne postoji datoteka plain.txt\n");
    exit(EXIT_FAILURE);
  }

  for(i=0; i<10; i++)
    niz[i] = 0;

  while((c = fgetc(ulaz)) != EOF) {
    if (isdigit(c))
      niz[c - '0']++;
  }

  max = 0;

  for(i=1; i<10; i++)
    if (niz[max] < niz[i])
      max = i;

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

  fclose(ulaz);
  return 0;
}
```

Rešenje 4.1.8

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

```

```c
int main(int argc, char** argv)
{
  /* Otvaranje datoteka za citanje i pisanje. */
  FILE *ulaz;
  char c;
  int broj_zagrada = 0;
  /* Na pocetku se pretpostavlja da zagrade jesu pravilno uparene. */
  int nisu_uparene = 0;

  if (argc != 2) {
    printf("Nedovoljno argumenata komandne linije\n");
    exit(EXIT_FAILURE);
  }

  ulaz = fopen(argv[1], "r");

  if (ulaz == NULL) {
    printf("Ne postoji datoteka plain.txt\n");
    exit(EXIT_FAILURE);
  }

  while((c = fgetc(ulaz)) != EOF) {
    if (c == '(')
      broj_zagrada++;
    else if (c == ')')
      broj_zagrada--;

    if (broj_zagrada < 0) {
      nisu_uparene = 1;
      break;
    }
  }

  if (broj_zagrada > 0 || broj_zagrada < 0 || nisu_uparene)
    printf("Zagrade nisu pravilno uparene\n");
  else
    printf("Zagrade jesu pravilno uparene\n");


  fclose(ulaz);
  return 0;
}
```

Rešenje 4.1.9

Rešenje 4.1.10

```c
/* Napisati program koji za rec s maksimalne duzine 20 karaktera koja
     se zadaje sa standardnog ulaza u datoteku
```

```c
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);
  }

  /*Zatvaramo datoteku rotacije.txt*/
  fclose(izlaz);
```

```
52    return 0;

54  }
```

Rešenje 4.1.11

```
#include <stdio.h>
2  #include <stdlib.h>
   #include <string.h>
4
   #define DUZINA_RECI 101
6  #define IME_DAT   21

8  void rotiraj(char* rec, unsigned int k, char* rezultat)
   {
10   unsigned int i;
     unsigned int n = strlen(rec);
12   k = k % n;

14   for(i=0; i<k; i++)
       rezultat[n-k+i] = rec[i];
16
     for(i=k; i<n; i++)
18     rezultat[i-k] = rec[i];


20
     rezultat[n] = '\0';
22 }

24 int main()
   {
26   /* Otvaranje datoteka za citanje i pisanje. */
     FILE *ulaz, *izlaz;
28   char ime_datoteke[IME_DAT];
     char rec[DUZINA_RECI];
30   char rezultat[DUZINA_RECI];
     unsigned int k;
32
     printf("Unesi ime datoteke: ");
34   scanf("%s", ime_datoteke);

36   ulaz = fopen(ime_datoteke, "r");

38   if (ulaz == NULL) {
       printf("Ne postoji datoteka plain.txt\n");
40     exit(EXIT_FAILURE);
     }
42
     izlaz = fopen("rotirano.txt", "w");
44   if (izlaz == NULL) {
       printf("Neuspesno kreiranje datoteke rotirano.txt\n");
```

```
46      exit(EXIT_FAILURE);
    }
48
    printf("Unesi prirodan broj: ");
50  scanf("%u", &k);

52  /* Rec po rec se ucitava iz datoteke. */
    while(fscanf(ulaz, "%s", rec) != EOF) {
54    rotiraj(rec, k, rezultat);
      fprintf(izlaz, "%s ", rezultat);
56  }


58


60  fclose(ulaz);
    fclose(izlaz);
62  return 0;
}
```

Rešenje 4.1.12

```
1  #include<stdio.h>
   #include<stdlib.h>
3  #include<string.h>

5  #define MAX_REC 51
   #define MAX_NAZIV 21
7
   void greska(){
9    fprintf(stderr, "-1\n");
     exit(EXIT_FAILURE);
11 }

13 int main(){
     char ime[MAX_NAZIV];
15   char rec[MAX_REC], prva_rec[MAX_REC];

17   scanf("%s", ime);
     FILE* ulaz, *izlaz;
19   ulaz = fopen(ime, "r");
     if(ulaz == NULL)
21     greska();

23   izlaz = fopen("rez.txt", "w");
     if(izlaz == NULL)
25     greska();

27   int n, i;
     fscanf(ulaz, "%d", &n);
29   fscanf(ulaz, "%s", prva_rec);
```

```
31    for(i = 1; i<n; i++)
      {
33      fscanf(ulaz, "%s", rec);
        if(strstr(rec, prva_rec) != NULL && strchr(rec, '_') != NULL)
35        fprintf(izlaz, "%s\n", rec);
      }
37
      fclose(ulaz);
39    fclose(izlaz);
      return 0;
41  }
```

Rešenje 4.1.13

```
    #include <stdio.h>
2   #include <stdlib.h>
    #include <string.h>
4   #include <ctype.h>

6   #define DUZINA 201

8   int palindrom(char* rec)
    {
10    int n = strlen(rec);
      int i;
12
      for(i =0; i<n/2; i++)
14      if (tolower(rec[i]) != tolower(rec[n-i-1]))
          return 0;
16
      return 1;
18  }

20  int main()
    {
22    /* Otvaranje datoteka za citanje i pisanje. */
      FILE *f = fopen("razno.txt", "r");
24    FILE *g = fopen("palindromi.txt", "w");
      char rec[DUZINA];
26
      if (f == NULL) {
28      printf("Ne postoji datoteka razno.txt\n");
        exit(EXIT_FAILURE);
30    }

32    if (g == NULL) {
        printf("Nije moguce kreirati datoteku palindromi.txt\n");
34      exit(EXIT_FAILURE);
      }
36
      /* Rec po rec se ucitava iz datoteke. */
```

```
38    while(fscanf(f, "%s", rec) != EOF) {
        if (palindrom(rec))
40        fprintf(g, "%s ", rec);
      }
42


44
      fclose(f);
46    fclose(g);
      return 0;
48  }
```

Rešenje 4.1.14

```
1   /* ovo je resenje gde se sve reci iz datoteke ucitavaju u niz, a onda
         se iz niza uklanjaju duplikati, 5_2.c je resenje gde se vec
       prilikom citanja reci iz datoteke uklanjaju duplikati, tako da u
       nizu nema ponavljanja reci */

3   #include<stdio.h>
    #include<stdlib.h>
5   #include<string.h>

7   #define MAX_RECI 256
    #define MAX_DUZINA 51
9   #define MAX_IME 21

11  void greska(){
      fprintf(stderr, "-1\n");
13    exit(EXIT_FAILURE);
    }
15
    int main(){
17
      char ime[MAX_IME];
19    char niz_reci[MAX_RECI][MAX_DUZINA];

21    ///Ucitavamno ime datoteke sa standardnog ulaza
      scanf("%s", ime);
23    FILE* ulaz;

25    //Otvaramo fajl za citanje
      ulaz = fopen(ime, "r");
27    if(ulaz == NULL)
        greska();
29
      //Citamo broj reci iz fajla
31    int n;
      fscanf(ulaz, "%d", &n);
33    if(n < 0 || n > MAX_RECI)
        greska();
```

```
35
   //Ucitavamo n reci u niz reci
37 int i;
   for(i=0; i<n; i++)
39 {
     fscanf(ulaz, "%s", niz_reci[i]);
41   printf("%s ", niz_reci[i]);
   }
43
   //U rez.txt ispisujemo niz bez duplikata
45 FILE* izlaz = fopen("rez.txt", "w");
   if(izlaz == NULL)
47   greska();

49 int k, ind = 0;
   for(i=0; i<n; i++)
51 {
     ind = 0;
53   for(k=0; k<i; k++)
     {
55     if(strcmp(niz_reci[k], niz_reci[i]) == 0)
       {
57       ind = 1;
         break;
59     }
     }
61   if(!ind)
       fprintf(izlaz, "%s\n", niz_reci[i]);
63 }

65 fclose(ulaz);
   fclose(izlaz);
67 return 0;
}
```

Rešenje 4.1.15

```
#include <stdio.h>
2 #include <stdlib.h>
#include <ctype.h>
4
#define DUZINA  201
6 #define ZBIR  1000

8 int zbir(char* rec)
{
10  int i = 0;
  int suma = 0;
12
  while(rec[i]) {
14   suma += rec[i];
```

```
    i++;
  }

  return suma;
}

int main()
{
  /* Otvaranje datoteka za citanje i pisanje. */
  FILE *f = fopen("ulaz.txt", "r");
  FILE *g = fopen("izlaz.txt", "w");
  char rec[DUZINA];

  if (f == NULL) {
    printf("Ne postoji datoteka ulaz.txt\n");
    exit(EXIT_FAILURE);
  }

  if (g == NULL) {
    printf("Nije moguce kreirati datoteku izlaz.txt\n");
    exit(EXIT_FAILURE);
  }

  /* Rec po rec se ucitava iz datoteke. */
  while(fscanf(f, "%s", rec) != EOF) {
    if (zbir(rec) > ZBIR)
      fprintf(g, "%s ", rec);
  }



  fclose(f);
  fclose(g);
  return 0;
}
```

Rešenje 4.1.16

```
/*
   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.
*/

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

```
11
   /*
13    Funkcija ucitaj_i_prebroj ucitava brojeve
      iz datoteke na koju pokazuje f i prebrojava
15    koliko je medju njima k-tocifrenih brojeva
   */
17 int ucitaj_i_prebroj(FILE* f, int k)
   {
19    int n;
      int x;
21    int i;
      int br;
23
      /* U datoteci je prvo naveden ukupan broj brojeva. */
25    fscanf(f,"%d",&n);

27    /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
       */
      if(n<=0)
29    {
         fprintf(stderr, "Greska: broj n mora biti prirodan\n");
31       exit(EXIT_FAILURE);
      }
33
      br=0;
35    for(i=0;i<n;i++)
      {
37       fscanf(f,"%d",&x);
         if(broj_cifara(x)==k)
39          br++;
      }
41
      return br;
43 }

45 int broj_cifara(int x)
   {
47    int br_c;

49    br_c=0;

51    /*
         Do while petlja je pogodnija od petlji sa preduslovom
53       jer tacno racuna broj cifara i za broj 0.
      */
55    do
      {
57      br_c++;
        x/=10;
59    } while(x);

61    return br_c;
```

```
}

int main(int argc, char* argv[])
{
   int n;
   int k;
   FILE* f;
   int br;

   if(argc!=3)
   {
      fprintf(stderr,"Greska: program se pokrece sa: %s
     naziv_datoteke k \n", argv[0]);
      exit(EXIT_FAILURE);
   }

   f=fopen(argv[1], "r");

   if(f==NULL)
   {
      fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
     .\n", argv[1]);
      exit(EXIT_FAILURE);
   }

   /* Argumenti komandne linije su stringovi. Da bismo konvertovali
     string
      u ceo broj koristimo ugradjenu funkciju atoi. */
   k = atoi(argv[2]);

   if (k<=0)
   {
      fprintf(stderr, "Greska: broj k mora biti prirodan\n");
      exit(EXIT_FAILURE);
   }

   printf("Broj %d-cifrenih brojeva u datoteci: %d\n", k,
     ucitaj_i_prebroj(f,k));

   fclose(f);
   return 0;
}
```

Rešenje 4.1.17

```
/* Napisati program koji pronalazi maksimum brojeva zapisanih u
     datoteci brojevi.txt */

#include<stdio.h>

int main(){
```

```
6
   FILE* in;
8  float broj, max_broj;

10 /*Otvaramo datoteku brojevi.txt za citanje i proveravamo da li smo
     je uspesno otvorili*/
   in = fopen("brojevi.txt", "r");
12 if(in == NULL){
   printf("Greska pri otvaranju datoteke!");
14 return 0;
   }
16
   /*
18  Kako bismo inicijalizovali promenljivu max_broj,
    citamo jedan broj iz datoteke i smestamo ga u
20  ovu promenljivu. */
   fscanf(in, "%f", &max_broj);
22
   /*U petlji citamo sve ostale brojeve i poredimo ih sa trenutnim
     maksimumom.*/
24 while(fscanf(in, "%f", &broj) != EOF){
   if(broj > max_broj)
26   max_broj = broj;
   }
28
   /*Ispisujemo rezultat*/
30 printf("Najveci broj je: %.2f\n", max_broj);

32 /*Zatvaramo datoteku brojevi.txt*/
   fclose(in);
34
   return 0;
36 }
```

Rešenje 4.1.18

```
1  #include<stdio.h>
   #include<stdlib.h>
3
   #define MAX_DIM 50
5  void greska(){
     fprintf(stderr, "-1\n");
7    exit(EXIT_FAILURE);
   }
9
   int main(){
11
   FILE* ulaz;
13 float A[MAX_DIM][MAX_DIM];

15 ulaz = fopen("matrica.txt", "r");
```

```
if(ulaz == NULL)
  greska();

int n, m;
fscanf(ulaz, "%d%d", &n, &m);
if(n <= 0 || n > MAX_DIM || m <= 0 || m > MAX_DIM)
  greska();

//Ucitavanje matrice
int i, j;
for(i=0; i<n; i++){
  for(j=0; j<m; j++){
    fscanf(ulaz, "%f", &A[i][j]);
  }
}

//Provera
int k, l;
float suma = 0;
for(i=0; i<n; i++){
  for(j=0; j<m; j++)
  {
    //Provera za poziciju (i,j):
    suma = 0;
    for(k=i-1; k<=i+1; k++){
      for(l=j-1; l<=j+1; l++){
        if(k>=0 && k<n && l>=0 && l<m)
          suma += A[k][l];
      }
    }
    suma -= A[i][j];
    if(A[i][j] == suma)
      printf("(%d, %d, %g)\n", i, j, A[i][j]);
  }
}

fclose(ulaz);
return 0;
}
```

Rešenje 4.1.19

Rešenje 4.1.20

Rešenje 4.1.21

Rešenje 4.1.22

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

#define IME 5

typedef struct{
  unsigned int a,b;
  char ime[IME];
}PRAVOUGAONIK;

void greska()
{
  fprintf(stderr, "-1\n");
  exit(EXIT_FAILURE);
}

int main(int argc, char* argv[]){

  //Proveravamo da li su navedeni potrebni argumenti
  if(argc != 2)
    greska();

  /*NAPOMENA:
   Najispravnije je koristiti unsigned kao tip za max_pov jer su a i
    b tog tipa,
   pa bi u slucaju velikih brojeva koji ne mogu stati u int doslo do
    greske pri racunanju
   povrsine. Ako bi ipak pretpostavili da program nece biti testiran
    za brojeve vece od
   MAX_INT i ako se koristi int za max_pov treba voditi racuna o
    sledecem:
   ako se max_pov inicijalizuje na -1,
   poredjenje if(p.a*p.b > max_pov) nece raditi ispravno
   iz razloga sto kada mesamo unsigned i signed int,
   svi operandi se tretiraju kao da su unsigned!
   Kako su p.a i p.b neoznaceni brojevi, a -1 oznacen,
   desava se da se u poredjenju -1 tretira kao neoznacen broj (i to
    jako
   veliki broj jer ima vodecu jedinicu).
   Ovo je greska koja se u kodu jako tesko pronalazi,
   kada se ubaci fleg -Wall, nece se ispisati nikakvo upozorenje,
   ali koriscenjem -Wextra dobijate upozorenje:
   warning: comparison between signed and unsigned integer
    expressions
   Probajte!
   */
  unsigned max_pov = 0;
  PRAVOUGAONIK p;

  //Otvaramo fajl za citanje
```

```
46    FILE* ulaz = fopen(argv[1], "r");
      if(ulaz == NULL)
48      greska();

50    //Citamo iz fajla sve dok ne dodjemo do kraja
      while(fscanf(ulaz, "%u%u%s", &p.a, &p.b, p.ime) == 3){
52      //Proveravamo da li su podaci ispravni
        if(p.a == 0 || p.b == 0)
54        greska();

56      //Obradjujemo trenutni pravougaonik
        if(p.a == p.b)
58        printf("%s ", p.ime);
        else{
60        if(p.a*p.b > max_pov)
            max_pov = p.a*p.b;
62      }
      }
64
      //Ispisujemo povrsinu najveceg pravougaonika
66    if(max_pov != 0)
        printf("%u\n", max_pov);
68    else
        printf("\n");
70
      //Zatvaramo datoteku
72    fclose(ulaz);
      return 0;
74  }
```

Rešenje 4.1.23

```
/* U datoteci studenti. txt se nalaze informacije o studentima: prvo
    broj studenata, a zatim u pojedinacnim linijama
2 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. */
4
  #include<stdio.h>
6
  #define MAXS 100
8
  /*Definisemo strukturu za cuvanje studenata*/
10 typedef struct st{
     char korisnicko_ime[8];
12   float prosek;
   }STUDENT;
14
   int main(){
16
```

```
  FILE *ulaz;
18  STUDENT studenti[MAXS];

20  int ocena1,ocena2,ocena3,ocena4,ocena5, i=0, i_max_prosek;
  float max_prosek = 0;
22
  /*Otvaramo datoteku studenti.txt za citanje*/
24  ulaz = fopen("studenti.txt", "r");
  if(ulaz == NULL){
26  printf("Greska pri otvaranju datoteke!\n");
  return 0;
28  }

30  /*Ucitavamo liniju po liniju iz datoteke, sve dok ne dodjemo do
      kraja.
     Korisnicko ime smestamo u niz, a ocene ucitavamo u pomocne
      promenljive ocena1,...ocena5.
32   Zatim, na osnovu ocena racunamo prosek.

34   Ovde paralelno sa ucitavanjem pronalazimo i studenta sa najvecim
      prosekom i
     pamtimo njegov prosek i njegovu poziciju u nizu studenata,
36   Nismo morali ovako. Mogli smo i prvu da ucitamo sve studente, a
      zatim da prodjemo
     jednom kroz niz i da nadjemo onog sa najvecim prosekom.
38
     */
40  while(fscanf(ulaz, "%s%d%d%d%d%d", studenti[i].korisnicko_ime, &
      ocena1, &ocena2, &ocena3, &ocena4, &ocena5) != EOF){
  studenti[i].prosek = (ocena1 + ocena2 + ocena3 + ocena4 + ocena5)
      /5.0;
42
  if(studenti[i].prosek > max_prosek){
44     max_prosek= studenti[i].prosek;
       i_max_prosek = i;
46  }
  i++;
48  }

50  /*Ispisujemo rezultat*/
  printf("korisnicko ime: %s, prosek ocena: %.2f\n", studenti[
      i_max_prosek].korisnicko_ime, studenti[i_max_prosek].prosek);
52
  /*Zatvaramo datoteku*/
54  fclose(ulaz);

56  return 0;
}
```

Rešenje 4.1.24

Rešenje 4.1.25

```
/* 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.
*/

/*
  Prilikom pokretanja programa se, pored naziva ulazne i izlazne
  datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
  obe opcije, sto znaci da je minimalni broj argumenata 3.

  Moguci nacini pokretanja:
  ./a.out ulaz.txt izlaz.txt -x
  ./a.out ulaz.txt izlaz.txt -y
  ./a.out ulaz.txt izlaz.txt -yx
  ./a.out ulaz.txt izlaz.txt -xy

*/

#include <stdio.h>
#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[])
{
   FILE *in, *out;
   char c;
   int i;
   int j;
   int xoption=0;
   int yoption=0;
   int dim;
   RAZLOMAK razlomci[MAX];
   RAZLOMAK r;

   /*
      Prilikom pokretanja programa se, pored naziva ulazne i izlazne
      datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
      obe opcije, sto znaci da je minimalni broj argumenata 3.

```

```
      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


109in= 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");
117if(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 '-'.*/

125if (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 4.1.26

```c
/*
  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.

*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX 31
#define MAXA 100

typedef struct automobil
{
   char marka[MAX];
   char model[MAX];
   float cena;
} AUTOMOBIL;

/*
   Struktura INFO sadrzi naziv
   marke automobila, prosek cena
   za tu marku i broj automobila
   te marke
*/
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.

    Za svaki automobil a[i] proveravamo da li se njegova marka vec
    nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
    vredost cene automobila a[i] i uvecavamo broj automobila sa
    tom markom. U suprotnom, dodajemo novi element u niz p. Posto
    ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
    na koju pokazuje pokazivac *pn.
*/
void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
{
    int i,j;
    int ind;
    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);

    stampaj_kandidate(kola, dim_kola, granica, infos, dim_infos);

    fclose(f);
    return 0;
}
```

Rešenje 4.1.27

```
/*
   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.
*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX_LEN 81

int main(int argc, char* argv[])
{
   char linija[MAX_LEN];
   char max_linija[MAX_LEN];
   int duzina;
   int max_duzina;

   FILE *ulaz, *izlaz;

   /*
     Proveravamo da li poziv programa ima dovoljan broj argumenata.
   */
   if(argc!=2)
   {
      fprintf(stderr,"Greska: program se pokrece sa: %s
    ime_ulazne_datoteke\n", argv[0]);
      exit(EXIT_FAILURE);
   }

   ulaz=fopen(argv[1],"r");
   if(ulaz==NULL)
   {
      fprintf(stderr,"error fopen(): Neuspelo otvoranje datoteke %s
    za citanje.\n", argv[1]);
      exit(EXIT_FAILURE);
   }

   /*
```

```
      Funkcija fgets ucitava jednu liniju teksta maksimalne duzine MAX_LEN
      iz datoteke ulaz u string linija. Ukoliko ucitavanje ne uspe (na primer,
      zato sto smo dosli do kraja datoteke), povratna vrednost ove funkcije
      bice prazan pokazivac (NULL).
*/

max_duzina=0;
while(fgets(linija, MAX_LEN, ulaz)!=NULL)
{
   duzina = strlen(linija);
   /*
      Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u petlju.
      Ovu promenljivu menjamo kada je duzina ucitana linije
      veca od max_duzina ili kada su jednake, ali je ucitana linija
      leksikografski ispred trenutne linije sa maksimalnom duzinom.

      Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii kodova prva dva
      razlicita karaktera stringova s1 i s2 na istim indeksima, ukoliko oni
      postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga osetljiva
      na mala i velika slova (npr 'D' je leksikografski ispred 'p').

   */

   if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija, max_linija)<0))
   {
      strcpy(max_linija, linija);
      max_duzina=duzina;
   }
}

/*
   Funkcija fputs ispisuje string koji je njen prvi argument u datoteku
   koja je njen drugi argument. Sve funkcije za ucitavanje iz datoteka i
   upis u datoteke (fgetc, fputc, fgets, fputs, ...) mozemo koristiti
   i kada radimo sa standardnim ulazom i standardnim izlazom. Kao nazive
   datoteka tada navodimo stdin i stdout.
*/
```

```
76      fputs(max_linija, stdout);

78      fclose(ulaz);
        return 0;
80  }
```

Rešenje 4.1.28

```
/* 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 4.1.29

```
/*
   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.
*/

#include <stdio.h>
#include <stdlib.h>
#define MAXL 81
#define MAXR 31

/*
   Funkcija broj_pojavljivanja broji koliko
   se puta pojavio string t u stringu s
*/
int broj_pojavljivanja(char s[], char t[])
{
   int br;
   int i,j;
   /*
      i - indeks karaktera u s
      j - indeks karaktera u t
      br - brojac koliko se puta t javlja u s
   */
   br=0;
   for(i=0;s[i];i++)
   {
      for(j=0;t[j];j++)
         if(s[i+j]!=t[j]) /* Ako naidjemo na razlicite karaktere, */
            break;        /* prekidamo petlju. */
      /*
         Do prekida petlje moze doci ili zbog toga sto su pronadjeni
         razliciti karakteri i usledio je break ili zbog toga sto
         je prestao da vazi uslov petlje, odnosno karakter t[j] je
         jednak '\0'. Ako vazi drugi slucaj, to znaci da se string
         t nalazi u stringu s pocev od indeksa i i potrebno je
    uvecati
         brojac br.
      */
      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 4.1.30

```
/* Napisati program koji prebrojava koliko se linija datoteke ulaz.
    txt zavrsava niskom s koja se ucitava sa stan-
2 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 */
4
  #include<stdio.h>
6 #include <string.h>
  #define MAXL 81
8 #define MAXS 21

10 /*Funkcija brojLinija proverava koliko linija u datoteci in se
    zavrsava niskom s.
   Funkcija radi tako sto cita jednu po jednu liniju iz datoteke,
12  i zatim kraj te linije poredi sa niskom s.*/
  int brojLinija(FILE* in, char* s){
14
    char linija[MAXL];
16  int broj_linija = 0;
    int duzina_s = strlen(s);
18  int duzina_linije;

20  while(fgets(linija, MAXL, in) != NULL){
    duzina_linije = strlen(linija);
22
    /* Ukoliko je znak za novi red bio indikacija kraja linije,
      uklanjamo ga kako bi mogli da izvrsimo
24   *ispravno poredjenje (jer niska s nema novi red na kraju) */
    if(linija[duzina_linije-1]=='\n'){
26    linija[duzina_linije-1] = '\0';
      duzina_linije--;
28  }

30  /*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
32    nam i treba. U primeru uspravna crta (|) oznacava pokazivac
        s              ab
34      duzina_s           2
        Linija:            aaabbbdfsssab
36                  |
        Linija + duzina linije   aaabbbdfsssab
38                       |
```

```
       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 4.1.31

```
/* 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>
```

```
10
   #define MAXL 81
12
   int main(int argc, char* argv[]){
14
     char linija[MAXL];
16   FILE* izlaz;

18   /*Indikatori koji oznacavaju koja opcija je navedena kao argument
       komandne linije
     vind - ispisuju se recenice koje pocinju velikim slovom
20   mind - ispisuju se recenice koje pocinju malim slovom
      */
22   int vind=0, mind = 0;

24   /*Proveravamo da li je program ispravno pozvan*/
     if(argc > 2){
26   printf("Greska pri pozivanju programa!\n");
     return 0;
28   }

30   /*Ako opcije nisu zadate, onda treba da se ispisuju sve recenice,
       pa postavljamo oba indikatora na 1*/
     if(argc == 1){
32   vind = mind = 1;
     }else{
34
     /*Proveravamo da li je postavljena neka od opcija -v,-V,-m, -M
36    Ako jeste, postavljamo odgovarajuci indikator
      Ako nije, onda ispisujemo poruku o gresci*/
38   if(strcmp(argv[1], "-v") == 0 || strcmp(argv[1], "-V") == 0)
       vind = 1;
40   else if(strcmp(argv[1], "-m") == 0 || strcmp(argv[1], "-M") == 0)
       mind = 1;
42   else{
       printf("Greska pri zadavanju opcije!\n");
44     return 0;
     }
46   }


48
     /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo da li smo je
        uspesno otvorili*/
50   izlaz = fopen("izlaz.txt", "w");
     if(izlaz == NULL){
52   printf("Greska pri otvaranju datoteke!\n");
     return 0;
54   }

56   /*Citamo liniju po liniju sa standardnog ulaza i ispisujemo je u
       datoteku.
      Liniju ispisujemo ukoliko je ispunjen neki od dva uslova:
```

```
58      1. Izabrana je opcija za ispis malih slova i linija pocinje malim
         slovom
        2. Izabrana je opcija za velika slova i linija pocinje velikim
        slovom
60     NAPOMENA: Kada dodje do kraja ulaza, funkcija fgets vraca NULL
       */
62    while(fgets(linija, MAXL, stdin) != NULL){
      if( mind && islower(linija[0]) || vind && isupper(linija[0]) ||
        mind && vind)
64      fputs(linija, izlaz);
      }
66
      /*Zatvaramo datoteku izlaz.txt*/
68    fclose(izlaz);

70    return 0;
  }
```

Rešenje 4.1.32

```
1  /*
   10_2.c je druga verzija resenja 10. zadatka
3  */

5  #include <stdio.h>
   #include <stdlib.h>
7  #include <string.h>

9  #define MAX 201

11 void greska();

13 int main(int argc, char** argv)
   {
15   if(argc != 3)
       greska();
17
     FILE* dat1 = fopen(argv[1], "r");
19   FILE* dat2 = fopen(argv[2], "r");

21   if(dat1 == NULL || dat2 == NULL)
       greska();
23
     char bafer1[MAX];
25   char bafer2[MAX];
     int i = 1;
27
     char* d1, *d2;
29
     d1 = fgets(bafer1, MAX, dat1);
31   d2 = fgets(bafer2, MAX, dat2);
```

```
33    while(d1 != NULL && d2 != NULL)
      {
35      if(strcmp(bafer1, bafer2) != 0)
          printf("%d ", i);
37
        d1 = fgets(bafer1, MAX, dat1);
39      d2 = fgets(bafer2, MAX, dat2);

41      i++;
      }
43
      while(d1 != NULL)
45    {
        printf("%d ", i);
47      d1 = fgets(bafer1, MAX, dat1);
        i++;
49    }

51    while(d2 != NULL)
      {
53      printf("%d ", i);
        d2 = fgets(bafer2, MAX, dat2);
55      i++;
      }
57


59    fclose(dat1);
      fclose(dat2);
61
      return 0;
63  }

65  void greska()
    {
67    fprintf(stderr, "-1");
      exit(EXIT_FAILURE);
69  }
```