PROGRAMIRANJE 1

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

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

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!
```

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
```

Zadatak 1.1.3 Napisati program koji za uneta dva cela broja 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 korektan, tj. da druga uneta vrednost nije 0.*

Primer 1

```
Interakcija sa programom:
 Unesite vrednost celobrojne promenljive x: 7
 Unesite vrednost celobrojne promenljive y: 2
 7 + 2 = 9
 7 - 2 = 5
 7 * 2 = 14
 7 / 2 = 3
 7 % 2 = 1
```

Primer 2

```
Interakcija sa programom:
 Unesite vrednost celobrojne promenljive x: -3
 Unesite vrednost celobrojne promenljive y: 8
 -3 + 8 = 5
 -3 - 8 = -11
 -3 * 8 = -24
 -3 / 8 = 0
 -3 % 8 = -3
```

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

Primer 1

```
Interakcija sa programom:
 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
```

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

Primer 1

```
Interakcija sa programom:
 Unesite kolicinu jabuka (u kg): 6
 Unesite cenu (u dinarima): 82
 Molimo platite 492 dinara.
```

Primer 1

```
Interakcija sa programom:
 Unesite kolicinu jabuka (u kg): 10
 Unesite cenu (u dinarima): 93
 Molimo platite 930 dinara.
```

Zadatak 1.1.6 Napisati program koji pomaže kasirki da obračuna kusur koji treba da vrati kupcu. Za unetu cenu artikla, količinu artikla i iznos koji je kupac dao, program treba da ispiše vrednost kusura. Napomena: *Pretpostaviti da su cene svih artikala pozitivni celi brojevi, kao i da su unete vrednosti ispravne, tj. da se može vratiti kusur.*

Primer 1

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

Primer 2

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

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
```

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

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dve celobrojne vrednosti: 237 -592
 Pre zamene: x=237, y=-592
 Posle zamene: x=-592, y=237
```

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

Primer 2

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

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
```

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

Primer 1

```
Interakcija sa programom:
 Unesite cenu proizvoda: 8367
 8367 = 1*5000 + 1*2000 + 1*1000 + 0*500 + 1*200 + 1*100 + 1*50 + 0*20 + 1*10 + 7*1
```

Primer 2

```
Interakcija sa programom:
 Unesite cenu proizvoda: 934
 934 = 0*5000 + 0*2000 + 0*1000 + 1*500 + 2*200 + 0*100 + 0*50 + 1*20 + 1*10 + 4*1
```

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
```

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
```

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
```

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
```

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
```

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
```

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
```

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
```

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
```

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
```

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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
```

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
```

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

Primer 1

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

Primer 2

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

Zadatak 1.1.27 Napisati program koji pomaže moleru da izračuna površinu zidova prostorije koju treba da okreči. Za unete dimenzije sobe u metrima (dužinu, širinu i visinu), program treba da ispiše površinu zidova za krečenje pod pretpostavkom da na vrata i prozore otpada oko 20%. Omogućiti i da na osnovu unete cene usluge po kvadratnom metru program izračuna ukupnu cenu krečenja. Sve realne vrednosti ispisati zaokružene na dve decimale. NAPOMENA: *Podrazumevati da su dimenzije sobe i cena krečenja po kvadratu pozitivni celi brojevi i 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
```

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

Primer 1

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

Primer 1

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

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
```

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
```

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
```

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
```

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 u koju ce biti upisan uneti
     broj. */
  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 celobrojne promenljive x: ");
  scanf("%d", &x);

  /* Ucitava se vrednost broja y. */
  printf("Unesite vrednost celobrojne 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);
```

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

32    /* Operator % izracunava ostatak pri celobrojnom deljenju dve
         celobrojne promenljive.
34       Da bi se odstampao karakter %, u naredbi printf se pise %%. */
      printf("%d %% %d = %d\n", x, y, x % y);
36
      return 0;
38  }
```

Rešenje 1.1.4

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

Rešenje 1.1.5

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

Rešenje 1.1.6

```
1   #include <stdio.h>

3   int main()
    {
5     /* Deklaracija promenljivih cija je vrednost neoznacen ceo broj. */
      unsigned int cena;
7     unsigned int kolicina;
      unsigned int iznos;
9     unsigned int kusur;

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

15    /* Izracunava se kusur. */
      kusur = iznos - kolicina * cena;
17
      /* Ispis vrednosti kusura. */
19    printf("Kusur je %u dinara.\n", kusur);

21    return 0;
```

```
}
```

Rešenje 1.1.7

```
#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

```c
#include<stdio.h>

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

  /* Ucitavaju se vrednosti x i y. */
  printf("Unesite dve celobrojne vrednosti:");
  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

```c
#include<stdio.h>

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

  /* Ucitavaju se vrednosti a i b. */
  printf("Unesite dve celobrojne vrednosti:");
  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);
```

```
21    return 0;
}
```

Rešenje 1.1.10

```
#include <stdio.h>

int main()
{
  /* Deklaracija promenljive u koju ce se upisati pozitivan trocifren
     broj. */
  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;
  char cifra_desetice;
  char 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
    cifra_jedinice, cifra_desetice i cifra_stotine:

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

  return 0;
}
```

Rešenje 1.1.11

```c
#include <stdio.h>

int main()
{
  /* Deklaracija i ucitavanje cene proizvoda. */
  unsigned int x;
  printf("Unesite cenu proizvoda:");
  scanf("%u", &x);

  /* Vrednost x/5000 predstavlja maksimalan broj novcanica od 5000
    dinara koje je moguce iskoristiti za placanje racuna.
    Na primer, neka je uneta cena 8367 dinara, vrednost izraza
    8367/5000 je jednaka 1. */
  printf("%u = %u*5000 + ", x, x / 5000);

  /* Da bi se isti postupak primenio i na ostale novcanice, potrebno
    je izracunati preostali iznos. Jedan nacin da se to uradi je
    racunanje ostatka pri deljenju unete vrednosti x (u primeru 8367)
     sa 5000. On iznosi 3367. Ovu vrednost dodeljujemo promeljivoj x.
     */
  x = x % 5000;

  /* Postupak se ponavlja i za za ostale novcanice. */
  printf("%u*2000 + ", x / 2000);
  x = x % 2000;
  printf("%u*1000 + ", x / 1000);
  x = x % 1000;
  printf("%u*500 + ", x / 500);
  x = x % 500;
  printf("%u*200 + ", x / 200);
  x = x % 200;
  printf("%u*100 + ", x / 100);
  x = x % 100;
  printf("%u*50 + ", x / 50);
  x = x % 50;
  printf("%u*20 + ", x / 20);
  x = x % 20;
  printf("%u*10 + ", x / 10);
  x = x % 10;
  printf("%u*1\n", x);

  return 0;
}
```

Rešenje 1.1.12

```c
#include <stdio.h>

int main()
{
```

```
  /* Deklaracija potrebnih promenljivih. */
  unsigned int x;
  unsigned int obrnuto_x;
  char cifra_jedinice;
  char cifra_desetice;
  char cifra_stotine;

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

  /* Izdvajaju se pojedinacne cifre broja. */
  cifra_jedinice = x % 10;
  cifra_desetice = (x / 10) % 10;
  cifra_stotine = x / 100;

  /* Formira se rezultujuci broj. */
  obrnuto_x = cifra_jedinice * 100 + cifra_desetice * 10 +
    cifra_stotine;

  /* Ispis rezultata. */
  printf("Obrnuto: %u\n", obrnuto_x);

  return 0;
}
```

Rešenje 1.1.13

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, broj_obrnuto, broj_zamena;
  char jedinice, desetice, stotine, hiljade;
  int proizvod_cifara, razlika_cifara, suma_kvadrata;

  /* Ucitava se jedan neoznacen broj. */
  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);

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

  /* Levi deo rezultata su sve cifre levo od cifre desetice.
     Na primer, za broj 1234, levi deo je broj 12 i dobija se
     deljenjem unetog broja sa 100. */
  levo = broj/100;

  /* Rezultat se dobija spajanjem levog i desnog dela.
     U datom primeru: 12*10 + 4 = 124. */
```

```
  novibroj = levo*10 + desno;

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

  return 0;
}
```

Rešenje 1.1.15

```
#include <stdio.h>

int main()
{
  /* Deklaracija potrebnih promenljivih. */
  unsigned int n, novibroj;
  unsigned int levi, desni, m;

  /* Ucitavaju se brojevi n i m. */
  printf("Unesite pozitivan ceo broj: ");
  scanf("%u", &n);
  printf("Unesite pozitivan dvocifreni broj:");
  scanf("%u", &m);

  /* Levi deo rezultata su sve cifre levo od cifre stotina.
     Na primer, ako je n=12345, levi deo rezultata je 12.
     On se dobija deljenjem unetog broja sa 1000. */
  levi = n / 1000;

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

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

  /* Ispis rezultata. */
  printf("Novi broj je %u\n", novibroj);

  return 0;
}
```

Rešenje 1.1.16

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

```c
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

```c
#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);
```

```
19    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, O, P;

  /* Ucitava se poluprecnik kruga. */
  printf("Unesite duzinu poluprecnika kruga:");
  scanf("%f", &r);

  /* Racunaju se obim i povrsina.
     M_PI je konstanta koja se nalazi u zaglavlju math.h
     i njena vrednost odgovara vrednosti broja pi. */
  O = 2 * r * M_PI;
  P = r * r * M_PI;
```

```
18
   /*Ispis rezultata na dve decimale. */
20 printf("Obim: %.2f\nPovrsina: %.2f\n", O, P);

22 return 0;
}
```

Rešenje 1.1.23

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

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

Rešenje 1.1.24

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

4 int main()
{
6   /* Deklaracija potrebnih promenljivih. */
   float a, b, c;
8  float obim, s, povrsina;

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

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

```
16
   /* Racuna se povrsina koriscenjem Heronovog obrasca. */
18 s = obim / 2;
   povrsina = sqrt(s * (s - a) * (s - b) * (s - c));
20
   /* Ispis rezultata. */
22 printf("Obim: %.2f\n", obim);
   printf("Povrsina: %.2f\n", povrsina);
24
   return 0;
26 }
```

Rešenje 1.1.25

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

Rešenje 1.1.26

```
1  #include<stdio.h>

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

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

13   /* Pogresan nacin: as = (a+b+c)/3;
        Kada se operacija / koristi nad celim brojevima,
15      deljenje je celobrojno.
        Na primer, (1+1+3)/3 ima vrednost 1.*/
17
     /* Ispravan nacin je da se bar jedan operand
19      pretvori u realan broj. */
     as = (a + b + c) / 3.0;
21
     /* Drugi ispravni nacini:
23      as=1.0*(a+b+c)/3;
        as=(0.0+a+b+c)/3;
25      as=((float)(a+b+c))/3; */

27   /* Ispis rezultata. */
     printf("Aritmeticka sredina unetih brojeva je %.2f\n", as);
29
     return 0;
31 }
```

Rešenje 1.1.27

```c
#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;
  char c;
  unsigned int levo, desno;
  unsigned int novo_x;

  /* Ucitavamo potrebne vrednosti. Sa unosom podataka tipa char
```

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

21      Ukoliko specifikatore promenljivih u format stringu pisemo
        spojeno, tada ih prilikom unosa mozemo razdvojiti bilo kojim
23      karakterom. Zbog toga blanko znakove u format stringu funkcije
        scanf treba izbegavati i ovo je redak slucaj kada je njihova
25      upotreba opravdana.Kada ucitavamo karaktersku promenljivu, njena
        numericka
        vrednost je jednaka ASCII kodu unetog karaktera. Na primer,
27      ako karakter '0' ucitamo u promenljivu c, njena numericka
        vrednost bice 48. Da bismo pretvorili ovu numericku vrednost u
29      numericku vrednost koja odgovara cifri, od nje oduzimamo ASCII
        kod karakterske konstante '0' koji iznosi upravo 48.
31
        Ako zelimo da odstampamo znak ", u format stringu funkcije
33      printf navodimo \". */
    printf("Unesite redom x, p i c: ");
35  scanf("%u %u %c", &x, &p, &c);

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

45  /* Racuna se deo broja koji se nalazi desno od pozicije p. */
    desno = x % (unsigned int) pow(10, p);
47
    /* Racuna se deo broja koji se nalazi levo od pozicije p. */
49  levo = x / (unsigned int) pow(10, p);

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

55  /* Ispisuje se dobijena vrednost. */
    printf("Rezultat je: %u\n", novo_x);
57
    return 0;
59
}
```

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);

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

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

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

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

  return 0;
}
```

Rešenje 1.1.30

```c
#include <stdio.h>

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

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

```

```
  /* Racuna se maksimum koriscenjem ternarnog operatora uslova. */
  max = (a > b) ? a : b;

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

  return 0;
}
```

Rešenje 1.1.31

Zadatak se rešava analogno zadatku 1.1.30

Rešenje 1.1.32

```
#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 za dva uneta cela broja ispisuje njihov minimum.

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

```
Interakcija sa programom:
Unesite cene tri artikla: 247 133 126
Cena sa popustom: 381
Usteda: 125
```

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

Primer 1

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

Primer 2

```
Interakcija sa programom:
Unesite broj: 238
Greska: Niste uneli cetvorocifren broj!
```

Primer 3

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

Primer 4

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

Zadatak 2.1.8 Napisati program koji za uneto vreme (broj sati iz intervala $[0, 24)$ i broj minuta iz intervala $[0, 60)$) ispisuje koliko je sati i minuta ostalo do ponoći.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

Zadatak 2.1.10 Napisati program koji za unetih pet karaktera ispisuje koliko je među njima malih slova.

Primer 1

```
Interakcija sa programom:
 Unesite karaktere: A u E f h
 Broj malih slova: 3
```

Primer 2

```
Interakcija sa programom:
 Unesite karaktere: k L M 9 o
 Broj malih slova: 2
```

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

Primer 1

```
Interakcija sa programom:
 Unesite karaktere: aBcAe
 2
```

Primer 2

```
Interakcija sa programom:
 Unesite karaktere: aa4A_
 3
```

Primer 3

```
Interakcija sa programom:
 Unesite karaktere: aAaAa
 5
```

Primer 4

```
Interakcija sa programom:
 Unesite karaktere: B6(vV
 0
```

Zadatak 2.1.12 Program učitava pet karaktera. Ispisati koliko puta su se pojavile cifre.

Primer 1

```
Interakcija sa programom:
 Unesite karaktere: A1cA3
 2
```

Primer 2

```
Interakcija sa programom:
 Unesite karaktere: 2a45_
 2
```

Primer 3

```
Interakcija sa programom:
 Unesite karaktere: 43986
 5
```

Primer 4

```
Interakcija sa programom:
 Unesite karaktere: B6(vV
 1
```

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

Zadatak 2.1.14 Broj je Armstrongov ako je jednak zbiru kubova svojih cifara. Napisati program koji za dati trocifren broj proverava da li je Armstrongov.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 153
 Broj je Amstrongov.
```

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

Zadatak 2.1.17 Napisati program koji ispituje da li se tačke $A(x_1, y_1)$ i $B(x_2, y_2)$ nalaze u istom kvadrantu i ispisuje odgovor `DA` ili `NE`.

Zadatak 2.1.18 Napisati program koji ispituje da li se tačke $A(x_1, y_1)$, $B(x_2, y_2)$ i $C(x_3, y_3)$ nalaze na istoj pravoj i ispisuje odgovor `DA` ili `NE`.

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

a) dužinu zajedničkog dela ta dva intervala

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

c) dužinu realne prave koju pokrivaju ta dva intervala

d) najmanji interval koji sadrži date intervale.

Primer 1

```
Interakcija sa programom:
 Unesite redom a1, b1, a2 i b2: 2 9 4 11
 Duzina zajednickog dela: 5
 Presek intervala: [4,9]
 Zajednicka duzina intervala: 9
 Najmanji interval: [2, 11]
```

Primer 2

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

Zadatak 2.1.20 Napisati program koji za uneti ceo broj x ispisuje njegov znak, tj da li je broj jednak nuli, manji od nule ili veći od nule.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

Zadatak 2.1.23 Napisati program koji učitava karakter i:

a) ako je c malo slovo, ispisuje odgovarajuće veliko

b) ako je c veliko slovo, ispisuje odgovarajuće malo

c) ako je c cifra, ispisuje poruku $cifra$

d) u ostalim slučajevima, ispisuje karakter c između dve zvezdice.

Primer 1

```
Interakcija sa programom:
 Unesite karakter: K
 k
```

Primer 2

```
Interakcija sa programom:
 Unesite karakter: 8
 cifra
```

Primer 3

```
Interakcija sa programom:
 Unesite karakter: >
 *>*
```

Primer 4

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

Zadatak 2.1.24 U nizu 12345678910111213....9899 ispisani su redom brojevi od 1 do 99. Napisati program koji za uneti ceo broj k ($1 \geq k \geq 189$) ispisuje cifru koja se nalazi na k-toj poziciji datog niza.

Primer 1

```
Interakcija sa programom:
 Unesite k: 13
 Na 13-toj poziciji je broj 1.
```

Primer 2

```
Interakcija sa programom:
 Unesite k: 105
 Na 105-toj poziciji je broj 7.
```

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

```
Interakcija sa programom:
 Unesite broj:  4
 U pitanju je: cetvrtak
```

Primer 2

```
Interakcija sa programom:
 Unesite broj:  7
 U pitanju je: nedelja
```

Primer 3

```
Interakcija sa programom:
 Unesite broj:  8
 Greska: nedozvoljeni unos!
```

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

Zadatak 2.1.28 Napisatiti program koji učitava dva cela broja i jedan od karaktera +, -, *, / ili % i ispisuje vrednost izraza dobijenog primenom date operacije na date argumente. U slučaju pogrešnog unosa ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite operator i dva cela broja: - 8 11
 Rezultat je: -3
```

Primer 2

```
Interakcija sa programom:
 Unesite operator i dva cela broja: / 14 0
 Greska: deljenje nulom nije dozvoljeno!
```

Primer 3

```
Interakcija sa programom:
 Unesite operator i dva cela broja: ? 5 7
 Greska: nepoznat operator!
```

Primer 4

```
Interakcija sa programom:
 Unesite operator i dva cela broja: / 19 5
 Rezultat je: 3
```

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

Primer 1

```
Interakcija sa programom:
 Unesite dan i mesec: 14 10
 jesen
```

Primer 2

```
Interakcija sa programom:
 Unesite dan i mesec: 2 8
 leto
```

Primer 3

```
Interakcija sa programom:
 Unesite dan i mesec: 27 2
 zima
```

Primer 4

```
Interakcija sa programom:
 Unesite dan i mesec: 19 5
 prolece
```

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

Zadatak 2.1.31 Napisati program koji za uneti datum u formatu *dan.mesec.godina.* proverava da li je korektan.

Primer 1

```
Interakcija sa programom:
 Unesite datum: 25.11.1983.
 Datum je korektan!
```

Primer 2

```
Interakcija sa programom:
 Unesite datum: 1.17.2004.
 Datum nije korektan!
```

Zadatak 2.1.32 Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum prethodnog dana.

Primer 1

```
Interakcija sa programom:
 Unesite datum: 30.4.2008.
 Prethodni datum: 29.4.2008.
```

Primer 2

```
Interakcija sa programom:
 Unesite datum: 1.12.2005.
 Prethodni datum: 30.11.2005.
```

Zadatak 2.1.33 Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum narednog dana.

Primer 1

```
Interakcija sa programom:
 Unesite datum: 30.4.2008.
 Naredni datum: 1.5.2008.
```

Primer 2

```
Interakcija sa programom:
 Unesite datum: 1.12.2005.
 Naredni datum: 2.12.2005.
```

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

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

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

Primer 1

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

Primer 2

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

*** Zadatak 2.1.35** Program učitava jedan karakter i osam realnih brojeva koji predstavljaju koordinate četiri tačke: $A(x_1, y_1), B(x_2, y_2), C(x_3, y_3), D(x_4, y_4)$. Na osnovu unetog karaktera ispisuje se odgovarajuća poruka na standardni izlaz:

- ukoliko je uneti karakter k - proverava da li su date tačke temena pravougaonika čije su stranice paralelne koordinatnim osama i u slučaju da jesu, ispisuje vrednost obima datog pravougaonika. Možemo podrazumevati da će korisnik koordinate tačaka unosi redom A, B, C, D, pri čemu $ABCD$ opisuje pravougaonik čije su stranice AB, BC, CD, DA, a dijagonale AC i BD. Na primer, tačke $(1, 1), (2, 1), (2, 2), (1, 2)$ čine pravougaonik čije su stranice paralelne koordinatnim osama i čiji je obim 4 a tačke $(1, 1), (2, 2), (3, 3), (4, 4)$ ne čine pravougaonik.
- ukoliko je uneti karakter h - proverava da li su unete tačke kolinearne i ukoliko jesu, ispisuje jednačinu prave kojoj pripadaju. Na primer, tačke $(1, 2), (2, 3), (3, 4), (4, 5)$ su kolinearne i pripadaju pravoj $y = x + 1$, tačke

$(1,1),(1,2),(1,3),(1,4)$ su kolinearne i pripadaju pravoj $x = 1$, a tačke $(1,1),(2,1),(2,2),(1,2)$ nisu kolinearne.

- ukoliko je uneti karakter j - Kramerovim pravilom proverava da li je sistem jednačina $x_1 * p + x_2 * q = x_4 - x_3, y_1 * p + y_2 * q = y_4 - y_3$ određen, neodređen ili nema rešenja, i u slučaju da je određen ispisuje rešenja.

Zadatak 2.1.36 Polje šahovske table se definiše parom prirodnih brojeva ne većih od 8: prvi se odnosi na red, drugi na kolonu. Ako su dati takvi parovi, napisati program koji proverava:

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

b) da li kraljica sa (k, l) ugrožava polje (m, n)

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

2.2 Rešenja

Rešenje 2.1.1

```c
#include <stdio.h>

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

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

  /* Ako je b<a, promenljiva min ce promeniti vrednost tj. bice joj
     dodeljena vrednost promenljive b. U suprotnom, vrednost ostaje
     ista. */

  if (b < a)
    min = b;

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

  return 0;
}
```

Rešenje 2.1.2 Rešenje je analogno rešenju broj 2.1.1.

Rešenje 2.1.3

```c
#include<stdio.h>

int main()
{
  float x;
  float apsolutno_x;

  printf("Unesite jedan realan broj:");
  scanf("%f", &x);

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

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

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

Rešenje 2.1.4

```c
#include <stdio.h>

int main()
{
  int x;
  float rx;

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

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

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

```
23
   /* Primenom operatora / na argumente 1 i x dobijamo rezultat
25    celobrojnog deljenja ovih argumenata. Da bismo dobili
      kolicnik, koji je realna vrednost, neophodno je da jedan od
27    argumenata zapisemo kao realnu vrednost, npr celobrojnu
      vrednost 1 zapisemo kao realnu vrednost 1.0. Ovakav postupak
29    se naziva implicitna konverzija. */

31   rx = 1.0 / x;
   printf("Reciprocna vrednost unetog broja:%.4f\n", rx);
33
   return 0;
35 }
```

Rešenje 2.1.5

```
1  #include<stdio.h>

3  int main()
   {
5    int a, b, c;
     int s;
7    printf("Unesite tri cela broja:");
     scanf("%d%d%d", &a, &b, &c);
9
     /* inicijalizujemo promenljivu s na nulu */
11   s = 0;

13   /* U naredbi dodele s=s+a vrednost izraza sa desne strane znaka
        jednakosti dodeljujemo promenljivoj sa leve strane znaka
15      jednakosti. Staru vrednost promenljive s saberemo sa vrednoscu
        promenljive a i dobijenu vrednost upisemo u promenljivu s. */
17
     if (a > 0)
19     s = s + a;

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

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

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

Rešenje 2.1.6

```c
#include <stdio.h>

int main()
{
  unsigned a, b, c;
  unsigned min;
  unsigned cena_bez_popusta, cena_sa_popustom;

  printf("Unesite cene tri artikla:");
  scanf("%u%u%u", &a, &b, &c);

  /*
     Racunamo minimum tri broja. Dodeljujemo promenljivoj min
     vrednost prvog broja. */
  min = a;

  /*
     Ako je drugi broj manji od minimuma, to znaci da promenljiva
     min ne sadrzi najmanji broj. Dodeljujemo joj vrednost drugog
     broja. */
  if (min > b)
    min = b;

  /*
     Ako je treci broj manji od minimuma, to znaci da promenljiva
     min ne sadrzi najmanji broj. Dodeljujemo joj vrednost treceg
     broja. */
  if (min > c)
    min = c;

  cena_bez_popusta = a + b + c;
  cena_sa_popustom = cena_bez_popusta - min + 1;

  printf("Cena sa popustom: %u\nUsteda: %u\n",
         cena_sa_popustom, cena_bez_popusta - cena_sa_popustom);

  return 0;
}
```

Rešenje 2.1.7

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

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

  /* Ucitavamo broj */
  printf("Unesite cetvorocifreni broj: ");
```

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

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

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

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

  /* Odredjujemo maksimalnu cifru */
  max = j;

  if (d > max)
    max = d;

  if (s > max)
    max = s;

  if (h > max)
    max = h;

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

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

  return 0;
}
```

Rešenje 2.1.8

```
#include<stdio.h>

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

```
  int preostali_minuti;

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

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

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

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

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

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

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

  /* Ukoliko naredbu printf zelimo da napisemo u dva reda i pritom
     ne prekidamo deo pod navodnicima, to mozemo uraditi bez
     navodjenja dodatnih karaktera: */
  printf("Do ponoci je ostalo %d sati i %d minuta\n",
         preostali_sati, preostali_minuti);

  return 0;
}
```

Rešenje 2.1.9

```
#include <stdio.h>

int main()
{
```

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

  printf("Uneti karakter: %c, njegov ASCII kod: %d\n", c, c);

  /* Razlika izmedju ASCII koda svakog malog i odgovarajuceg
     velikog slova je konstanta koja se moze sracunati izrazom
     'a'-'A' (i iznosi 32) */

  if (c >= 'A' && c <= 'Z')
    printf("odgovarajuce malo slovo: %c, njegov ASCII kod: ",
           "%d\n", c + ('a' - 'A'), c + ('a' - 'A'));

  if (c >= 'a' && c <= 'z')
    printf("odgovarajuce veliko slovo: %c, njegov ASCII kod: ",
           "%d\n", c - ('a' - 'A'), c - ('a' - 'A'));

  return 0;
}
```

Rešenje 2.1.10

```
#include <stdio.h>

int main()
{
  char c1, c2, c3, c4, c5;
  int broj_malih_slova = 0;

  /* Citamo karaktere */
  printf("Unesite karaktere: ");
  scanf("%c %c %c %c %c", &c1, &c2, &c3, &c4, &c5);

  /* Proveravamo da li je prvi karakter malo slovo */
  if (c1 >= 'a' && c1 <= 'z') {
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

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

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

```
29
  /* Proveravamo da li je cetvrti karakter malo slovo */
31  if (c4 >= 'a' && c4 <= 'z') {
    /* I ako jeste, uvecavamo broj malih slova */
33    broj_malih_slova++;
  }
35
  /* Proveravamo da li je peti karakter malo slovo */
37  if (c5 >= 'a' && c5 <= 'z') {
    /* I ako jeste, uvecavamo broj malih slova */
39    broj_malih_slova++;
  }
41
  /* Ispisujemo rezultat */
43  printf("Broj malih slova: %d\n", broj_malih_slova);

45  return 0;
}
```

Rešenje 2.1.11

```
1  #include <stdio.h>
   #include <ctype.h>
3
   int main()
5  {
     /* Broj pojavljivanja slova a i A se inicijalizuje na 0 */
7    int br_a = 0;

9    /* Funkcija getchar ucitava jedan karakter. Njena povratna
        vrednost je ASCII kod ucitanog karaktera.
11
        Funkcija tolower za dati karakter vraca: - odgovarajuce malo
13      slovo, ako je dati karakter veliko slovo - taj isti karakter,
        u suprotnom Ova funkcija je definisana u biblioteci ctype.h
15
        U slucaju da je uslov ispunjen, uvecavamo brojac br_a za jedan
17      pomocu operatora inkrementacije ++ */
     if (tolower(getchar()) == 'a')
19     br_a++;
     if (tolower(getchar()) == 'a')
21     br_a++;
     if (tolower(getchar()) == 'a')
23     br_a++;
     if (tolower(getchar()) == 'a')
25     br_a++;
     if (tolower(getchar()) == 'a')
27     br_a++;

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

```
31    return 0;
}
```

Rešenje 2.1.12

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

int main()
{
  int br_cif = 0;

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

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

  return 0;
}
```

Rešenje 2.1.13

```
#include <stdio.h>

int main()
{
  int x;
  printf("Unesite godinu:");
  scanf("%d", &x);

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

  return 0;
}
```

Rešenje 2.1.14

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

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

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

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

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

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

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

  return 0;
}
```

Rešenje 2.1.15

```c
#include <stdio.h>

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

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

```
11  n = abs(n);

13  if (n < 1000 || n > 9999) {
      printf("Broj nije cetvorocifren.\n");
15    return -1;
    }
17
    /* Izdvajamo cifre broja: j -jedinice, d - desetice, s - stotine
19     i h - hiljade */
    j = n % 10;
21  d = (n / 10) % 10;
    s = (n / 100) % 10;
23  h = n / 1000;

25  /* Inicijalizujemo broj parnih cifara na 0 */
    broj_parnih = 0;
27  /* Postavljamo proizvod parnih cifara na 1 (neutral za mnozenje) */
    proizvod_parnih = 1;
29
    /* Proveravamo da li je cifra jedinica parna */
31  if (j % 2 == 0) {
      proizvod_parnih = proizvod_parnih * j;
33    broj_parnih++;
    }
35
    /* Proveravamo da li je cifra desetica parna */
37  if (d % 2 == 0) {
      proizvod_parnih = proizvod_parnih * d;
39    broj_parnih++;
    }
41
    /* Proveravamo da li je cifra stotina parna */
43  if (s % 2 == 0) {
      proizvod_parnih = proizvod_parnih * s;
45    broj_parnih++;
    }
47
    /* Proveravamo da li je cifra hiljada parna */
49  if (h % 2 == 0) {
      proizvod_parnih = proizvod_parnih * h;
51    broj_parnih++;
    }
53
    /* Proveravamo da li u zapisu broja ima parnih cifara i
55     ispisujemo rezultat */
    if (broj_parnih == 0) {
57    printf("Nema parnih cifara.\n");
    } else {
59    printf("Proizvod parnih cifara: %d\n", proizvod_parnih);
    }
61
```

```
  return 0;
}
```

Rešenje 2.1.16

```
#include <stdio.h>

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

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

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

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

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

  if (c > max) {
    max = c;
    stepenMax = 100;
  }
  if (c < min) {
    min = c;
    stepenMin = 100;
  }

  if (d > max) {
    max = d;
    stepenMax = 1000;
```

```
47    }
      if (d < min) {
49      min = d;
        stepenMin = 1000;
51    }

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

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

65    return 0;
  }
```

Rešenje 2.1.20

```
1  #include <stdio.h>

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

9    if (x == 0)
       printf("Broj je jednak nuli\n");
11   else if (x < 0)
       printf("Broj je manji od nule\n");
13   else
       printf("Broj je veci od nule\n");
15
     return 0;
17 }
```

Rešenje 2.1.21

```
1  #include <stdio.h>
   #include <math.h>
3
   int main()
5  {
     float a, b, c;
```

```
  float D;
  float x1, x2;
  printf("Unesite koeficijente A, B i C:");
  scanf("%f%f%f", &a, &b, &c);

  /* Proveravamo da li je kvadratna jednacina korektno zadata. */
  if (a == 0)
    if (b == 0)
      /* slucaj a==0 && b==0 && c==0 */
      if (c == 0)
        printf("Jednacina ima beskonacno mnogo resenja\n");
  /* slucaj a==0 && b==0 && c!=0 */
      else
        printf("Jednacina nema resenja\n");
  /* slucaj a==0 && b!=0 */
    else {
      x1 = -c / b;
      printf("Jednacina ima jedinstveno realno resenje %.2f\n", x1);
    }
  /* slucaj a!=0 */
  else {
    D = b * b - 4 * a * c;
    if (D < 0)
      printf("Jednacina nema realnih resenja\n");
    else if (D > 0) {
      /* funkcija sqrt nalazi se u biblioteci math.h (prevodjenje
         sa -lm opcijom) */
      x1 = (-b + sqrt(D)) / (2 * a);
      x2 = (-b - sqrt(D)) / (2 * a);
      printf("Jednacina ima dva razlicita realna resenja %.2f ",
             "i %.2f\n", x1, x2);
    } else {
      x1 = (-b) / (2 * a);
      printf("Jednacina ima jedinstveno realno resenje %.2f\n", x1);
    }
  }

  return 0;
}
```

Rešenje 2.1.22

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

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

```
  char c1000;

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

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

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

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

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

  return 0;
}
```

Rešenje 2.1.23

```
#include <stdio.h>

int main()
{
  char c;

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

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

```
    printf("*%c*\n", c);

  return 0;
}
```

Rešenje 2.1.24

```
#include <stdio.h>

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

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

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

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

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

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

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

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

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

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

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

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

55  return 0;
}
```

Rešenje 2.1.25

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

13   /* Proveravaju se vrednosti za k */
     if (k < 1 || k > 3) {
15     printf("Greska: nedozvoljena vrednost za k!\n");
       return 0;
17   }
     printf("F(%f,%d)=", x, k);
19
     /* Analiziraju se moguci slucajevi */
21   if (k == 1) {
       F = 2 * cos(x) - x * x * x;
23   } else {
       if (k == 2) {
25       x = 2 * cos(x) - x * x * x;
         F = 2 * cos(x) - x * x * x;
27     } else {
         x = 2 * cos(x) - x * x * x;
29       x = 2 * cos(x) - x * x * x;
         F = 2 * cos(x) - x * x * x;
31     }
     }
33
     /* Ispisuje se rezultat */
35   printf("%f\n", F);
```

```
37    return 0;
}
```

Rešenje 2.1.26

```
#include <stdio.h>

int main()
{
  int dan;

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

  switch (dan) {
  case 1:
    printf("ponedeljak\n");
    break;
  case 2:
    printf("utorak\n");
    break;
  case 3:
    printf("sreda\n");
    break;
  case 4:
    printf("cetvrtak\n");
    break;
  case 5:
    printf("petak\n");
    break;
  case 6:
    printf("subota\n");
    break;
  case 7:
    printf("nedelja\n");
    break;
  default:
    printf("Greska: nedozvoljeni unos!\n");
  }

  return 0;
}
```

Rešenje 2.1.27

```
#include <stdio.h>

int main()
{
```

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

  /* Da bi se utvrdilo da li je karakter samoglasnik, neophodno je
     proveriti 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 2.1.28

```
#include <stdio.h>

int main()
{
  char op;
  int x, y;

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

  switch (op) {
  case '+':
    printf("Rezultat je: %d\n", x + y);
    break;
  case '-':
    printf("Rezultat je: %d\n", x - y);
    break;
  case '*':
    printf("Rezultat je: %d\n", x * y);
    break;
  case '/':
```

```
    if (y == 0)
      printf("Greska: deljenje nulom nije dozvoljeno!\n");
    else
      printf("Rezultat je: %f\n", x * 1.0 / y);
    break;
  case '%':
    printf("Rezultat je: %d\n", x % y);
    break;
  default:
    printf("Greska: nepoznat operator!\n");
  }

  return 0;
}
```

Rešenje 2.1.29

```
#include <stdio.h>

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

  /* Argument u naredbi switch mora biti celobrojna promenljiva,
     dok argument u naredbi case mora biti celobrojna konstanta. */
  switch (m) {
    /* Ispitujemo da li vazi m==1 ili m==2 */
  case 1:
  case 2:
    printf("zima\n");
    break;
  case 3:
    if (d < 21)
      printf("zima\n");
    else
      printf("prolece\n");
    break;
  case 4:
  case 5:
    printf("prolece\n");
    break;
  case 6:
    if (d < 21)
      printf("prolece\n");
    else
      printf("leto\n");
    break;
  case 7:
  case 8:
```

```
35      printf("leto\n");
        break;
37    case 9:
        if (d < 23)
39        printf("leto\n");
        else
41        printf("jesen\n");
        break;
43    case 10:
    case 11:
45      printf("jesen\n");
        break;
47    case 12:
        if (d < 22)
49        printf("jesen\n");
        else
51        printf("zima\n");
    }
53
    return 0;
55  }
```

Rešenje 2.1.30

```
1   #include <stdio.h>

3   int main()
    {
5     int godina;
      int mesec;
7     int prestupna;

9     printf("Unesite godinu: ");
      scanf("%d", &godina);
11
      if (godina < 0) {
13      printf("Lose uneta godina!\n");
        return -1;
15    }

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

27    switch (mesec) {
```

```
  case 1:
    printf("Januar, 31 dan\n");
    break;
  case 2:
    if (prestupna)
      printf("Februar, 29 dana\n");
    else
      printf("Februar, 28 dana\n");
    break;
  case 3:
    printf("Mart, 31 dan\n");
    break;
  case 4:
    printf("April, 30 dana\n");
    break;
  case 5:
    printf("Maj, 31 dan\n");
    break;
  case 6:
    printf("Jun, 30 dana\n");
    break;
  case 7:
    printf("Jul, 31 dan\n");
    break;
  case 8:
    printf("Avgust, 31 dan\n");
    break;
  case 9:
    printf("Septembar, 30 dana\n");
    break;
  case 10:
    printf("Oktobar, 31 dan\n");
    break;
  case 11:
    printf("Novembar, 30 dana\n");
    break;
  case 12:
    printf("Decembar, 31 dan\n");
    break;
  default:
    printf("Lose unet redni broj meseca!\n");
  }

  return 0;
}
```

Rešenje 2.1.31

```
#include <stdio.h>

int main()
```

```
{
  int dan, mesec, godina, dozvoljen_broj_dana;

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

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

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

  /* Ako je mesec korektan, proveravamo broj dana */
  switch (mesec) {
  case 1:
  case 3:
  case 5:
  case 7:
  case 8:
  case 10:
  case 12:
    /* Dozvoljeni broj dana za januar, mart, maj, jul, avgust,
       oktobar i decembar je 31 */
    dozvoljen_broj_dana = 31;
    break;
  case 2:
    /* Proveravamo da li je godina prestupna */
    if (godina % 4 == 0 && godina % 100 != 0 || godina % 400 == 0)
      /* Ako jeste, dozvoljeni broj dana za februar je 29 */
      dozvoljen_broj_dana = 29;
    else
      /* Ako nije, dozvoljeni broj dana za februar je 28 */
      dozvoljen_broj_dana = 28;
    break;
  case 4:
  case 6:
  case 9:
  case 11:
    /* Dozvoljeni broj dana za april, jun, septembar i novembar je
       30 */
    dozvoljen_broj_dana = 30;
    break;
  }
  /* Proveravamo dan */
  if (dan < 0 || dan > dozvoljen_broj_dana) {
```

```
    printf("Datum nije korektan (neispravan dan)!\n");
    return 0;
  }

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

  return 0;
}
```

Rešenje 2.1.32

```
#include <stdio.h>

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

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

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

  /* I po potrebi vrsimo korekcije */

  /* Ako je u pitanju prvi u mesecu */
  if (prethodni_dan == 0) {
    /* Treba korigovati mesec */
    prethodni_mesec = mesec - 1;
    /* Ako je u pitanju januar */
    if (prethodni_mesec == 0) {
      /* Treba korigovati i godinu */
      prethodni_mesec = 12;
      prethodni_godina = godina - 1;
    }

    /* Analiziramo redni broj meseca kako bi odredili tacan dan */
    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;
    }
  }

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

  return 0;
}
```

Rešenje 2.1.33 Rešenje je analogno rešenju zadatka 2.1.32.

Rešenje 2.1.35

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

int main()
{
  char c;
  float x1, y1, x2, y2, x3, y3, x4, y4;
  float kab, kbc, kad;
  float dab, dad;
  float delta, deltap, deltaq;
  float O;
  float k, n;

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

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

  switch (c) {
  case 'k':
    if (y1 == y2 && y3 == y4 && x1 == x4 && x2 == x3) {
      dab = sqrt(pow(x1 - x2, 2) + pow(y1 - y2, 2));
```

```
24        dad = sqrt(pow(x1 - x4, 2) + pow(y1 - y4, 2));
          O = 2 * dab + 2 * dad;
26        printf("Obim pravougaonika je %f\n", O);
        } else
28        printf("Tacke ne cine pravougaonik sa stranicama ",
                 "koje su paralelne koordinatnim osama\n");
30      break;
    case 'h':
32      /*
           Ukoliko se tacke A(x1,y1) i B(x2,y2) ne nalaze na pravoj
34         koja je paralelna x osi, izracunamo k,n za pravu odredjenu
           tackama A(x1,y1) i B(x2,y2) */
36      if ((x1 - x2) != 0) {
          k = (y1 - y2) / (x1 - x2);
38        n = y1 - k * x1;
          /*
40           Proverimo da li tacke C(x3,y3) i D(x4,y4) nalaze na toj
             pravoj */
42        if (y3 == x3 * k + n && y4 == x4 * k + n)
            printf("Tacke su kolinearne, pripadaju pravoj ",
44                 "y=%f*x+%f\n", k, n);
          else
46          printf("Tacke nisu kolinearne\n");
        }
48      /*
           Ukoliko se A i B nalaze na pravoj koja je paralelna x osi,
50         proverimo da li tacke C(x3,y3) i D(x4,y4) nalaze na toj
           pravoj */
52      else if (x3 == x1 && x4 == x1)
          printf("Tacke su kolinearne, pripadaju pravoj ",
54                 "x=%f\n", x1);
        else
56        printf("Tacke nisu kolinearne\n");
        break;
58    case 'j':
        delta = x1 * y2 - x2 * y1;
60      deltap = x2 * (y4 - y3) - y2 * (x4 - x3);
        deltaq = x1 * (y4 - y3) - y1 * (x4 - x3);
62      if (delta != 0)
          printf("Sistem ima jedinstveno resenje %.2f, %.2f\n",
64               deltap / delta, deltaq / delta);
        else if (deltap == 0 && deltaq == 0)
66        printf("Sistem je neodredjen ili nema resenja.\n");
        else
68        printf("Sistem nema resenja\n");
        break;
70    default:
        printf("Nekorektan unos\n");
72    }

74    return 0;
}
```

2.3 Petlje

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

Primer 1

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

(a) Koristiti `while` petlju.

(b) Koristiti `for` petlju.

(c) Koristiti `do-while` petlju.

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

Primer 3

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

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

Primer 1

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

Primer 2

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

Primer 3

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

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

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite redom brojeve x i n: -3 2
 9.000
```

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite broj: 856
 856
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 140
 14
```

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
```

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Zadatak 2.3.13 Program unosi ceo pozitivan broj n, a potom i n celih brojeva. Izračunati i ispisati zbir onih brojeva koji su neparni i negativni.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 Unesite n brojeva:
 5 8 13 17
 0
```

Primer 3

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

Zadatak 2.3.14 Program učitava ceo pozitivan broj n, a potom n celih brojeva. Naći sumu brojeva koji su deljivi sa 5, a nisu deljivi sa 7. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

```
Interakcija sa programom:
 Unesite broj m: 12.37
 Unesite broj n: 5
 Unesite n brojeva: 11 54.13 6 13 8
 3
```

Primer 2

```
Interakcija sa programom:
 Unesite broj m: 2
 Unesite broj n: 4
 Unesite n brojeva: 1 11 4.32 3
 1
```

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 2

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

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

Primer 1

```
Interakcija sa programom:
 Unesite broj: 1857
 Cifra 5 se nalazi u zapisu!
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 84
 Cifra 5 se ne nalazi u
     zapisu!
```

Primer 3

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Zadatak 2.3.19 U prodavnici se nalaze artikala čije cene su realni pozitivni

brojevi. Program unosi cene artikala sve do unosa broja nula 0. Napisati program koji izračunava i ispisuje prosečnu vrednost cena u radnji.

Primer 1

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

Primer 2

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

Primer 3

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Zadatak 2.3.22 Program učitava ceo pozitivan broj n, a zatim i n celih brojeva. Napisati program koji ispisuje broj sa najvećom cifrom desetica. Ukoliko

ima više takvih, ispisati prvi.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite brojeve:
 18 365 25 1 78
 Broj sa najvecom cifrom desetica je 78.
```

Primer 2

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

Primer 3

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

Zadatak 2.3.23 Program učitava ceo pozitivan broj n, a zatim i n celih brojeva. Napisati program koji ispisuje broj sa najvećim brojem cifara. Ukoliko ima više takvih, ispisati prvi.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite n brojeva: 18 365 25 1 78
 Najvise cifara ima broj 365.
```

Primer 2

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

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

Primer 1

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

Primer 1

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

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

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 8 6 5 2 11 7 0
 Razlika: 9
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: 8 -1 8 6 0
 Razlika: 9
```

Zadatak 2.3.26 Program učitava cele pozitivane brojeve n ($n > 1$) i d, a zatim i n celih brojeva. Napisati program koji izračunava koliko ima parova uzastopnih brojeva među unetim brojevima koji se nalaze na rastojanju d. Rastojanje između brojeva je definisano sa $d(x, y) = |y - x|$. Ispisati rezultat.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve n i d: 5 2
 Unesite n brojeva: 2 3 5 1 -1
 Broj parova: 2
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve n i d: 10 5
 Unesite n brojeva:
 -3 6 11 -20 -25 -8 42 37 1 6
 Broj parova: 4
```

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

Primer 1

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

Primer 2

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

Primer 3

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

*** Zadatak 2.3.29** Napisati program koji na osnovu unetog prirodnog broja

formira i ispisuje broj koji se dobija izbacivanjem cifara koje su jednake zbiru svojih suseda.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 28631
 2631
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 440
 40
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -5
 Neispravan unos.
```

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

Primer 1

```
Interakcija sa programom:
 Unesite broj: 25452
 Broj je palindrom!
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 895
 Broj nije palindrom!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 5
 Broj je palindrom!
```

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

Zadatak 2.3.37 Program učitava ceo broj n, a zatim i n karaktera. Napisati program koji proverava da li se od unetih karaktera može napisati reč *Zima*.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 4
 Unestite 1. karakter:  +
 Unestite 2. karakter:  o
 Unestite 3. karakter:  Z
 Unestite 4. karakter:  j
 Ne moze se napisati rec Zima.
```

Primer 2

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

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

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite pozitivan ceo broj: 14
 Suma kubova od 1 do 14: 11025
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite pozitivan ceo broj 25
 Suma kubova od 1 do 25: 105625
```

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

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite pozitivan ceo broj: 5
 i=1, s=1
 i=2, s=9
 i=3, s=36
 i=4, s=100
 i=5, s=225
```

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

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

Primer 1

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

Primer 2

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

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

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite x: 3
 Unesite tacnost eps: 0.000001
 S=0.049787
```

Primer 2

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

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

Primer 1

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

Primer 2

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

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

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

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite prirodan broj: 4
 Razlomak = 0.697674
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite prirodan broj: 20
 Razlomak = 0.697775
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite prirodan broj: 0
 Neispravan unos.
```

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

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

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

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite x i n: 5.6 8
 S=0.735084
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite x i n: 14.32 11
 S=17273.136719
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite prirodan broj: -6
 Neispravan unos.
```

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

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

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

Primer 1

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

Primer 2

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

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj n: 0
 Neispravan unos.
```

Primer 4

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

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

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

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

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 9
 855
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 11
 -9540
```

Primer 3

```
Interakcija sa programom:
 Unesite broj n: 20
 Neispravan unos
```

Primer 4

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

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

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

U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Unesite dva relana broja x i a:: 3.2 0.2
 Unesite prirodan broj: 5
 Izraz = 135380494030332048.000000
```

Primer 2

```
Interakcija sa programom:
 Unesite dva relana broja x i a:: 2 1
 Unesite prirodan broj: 3
 Izraz = 10201.000000
```

Primer 3

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

Primer 4

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

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

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

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

Primer 1

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

Primer 2

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

(b) Slika sadrži rub kvadrata dimenzije n.

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

Primer 3

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

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

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

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

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

Primer 1

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

Primer 2

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

*** Zadatak 2.3.59** Program učitava prirodan broj n. Napisati program koji

štampa romb sastavljen od minusa u pravougaoniku sastavljenom od zvezdica. Podrazumevati da je unos korektan.

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 1

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

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

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 1 2 3 4 5 2 3 4 3
```

Primer 2

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

Primer 3

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

Primer 4

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

*** Zadatak 2.3.62** Napisati program koji učitava ceo pozitivan broj n i ispisuje sve brojeve od 1 do n, zatim svaki drugi broj od 1 do n, zatim svaki treći broj od 1 do n itd., završavajući sa svakim n-tim (tj. samo sa 1). U slučaju greške pri unosu podataka odštampati ogovarajuću poruku.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

2.4 Rešenja

Rešenje 2.3.1

```c
#include <stdio.h>

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

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

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

```
    i++;

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

  return 0;
}
```

Rešenje 2.3.2

```
#include<stdio.h>

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

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

  /* Pre ulaska u telo petlje proverava se da li je ispunjen uslov
     ulaska u petlju. */
  while (i < n) {
    printf("Mi volimo da programiramo.\n");
    i++;
  }

  return 0;
}
```

Rešenje 2.3.3

```
#include <stdio.h>

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

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

```
15    /* U slucaju neispravnih podataka ispisujemo odgovarajucu poruku
         i izlazimo iz programa. */
17    if (n < 0) {
        printf("Neispravan unos. Promenljiva mora biti pozitivna!\n");
19      return -1;
      }
21
      /* Ispis pocinjemo od 0, zato promenljivu x postavljamo na 0. */
23    x = 0;
      while (x <= n) {
25      /* Ispisujemo broj. */
        printf("%d\n", x);
27      /* Uvecavamo promenljivu za jedan jer smo broj ispisali i sada
           zelimo da ispisemo sledeci broj. */
29      x++;
      }
31
      return 0;
33  }
```

Rešenje 2.3.4

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

3   #include <stdio.h>

5   int main()
    {
7     /* Promenljive koje oznacavaju granice intervala. */
      int n, m;
9     /* Promenljiva koja oznacava trenutno ispisani broj intervala. */
      int i;
11
      printf("Unesi dva cela broja: ");
13    scanf("%d%d", &n, &m);

15    if (m < n) {
        printf
17          ("Neispravan unos. Nisu dobro zadate granice intervala!\n");
        return -1;
19    }

21    /* Na pocetku ispisujemo prvi broj intervala, a to je n. */
      i = n;
23    /* uslov petlje se proverava pre ulaska u telo petlje */
      while (i <= m) {
25      printf("%d ", i);
        i++;
27    }
```

```
29    printf("\n");


31
      return 0;
33  }
```

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

3   #include <stdio.h>

5   int main()
    {
7
      /* Promenljive koje oznacavaju granice intervala. */
9     int n, m;
      /* Promenljiva koja oznacava trenutno ispisani broj intervala. */
11    int i;

13    printf("Unesi dva cela broja: ");
      scanf("%d%d", &n, &m);
15
      if (m < n) {
17      printf
            ("Neispravan unos. Nisu dobro zadate granice intervala!\n");
19      return -1;
      }
21


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

29    printf("\n");


31
      return 0;
33  }
```

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

11    printf("Unesi dva cela broja: ");
```

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

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

  /* 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. */
  i = n;
  do {                          /* Petlja se zapocinje bez provere
                                   uslova. */
    printf("%d ", i);           /* Stampa se vrednost promenljive
                                   i. */
    i++;                        /* Uvecava se vrednost promenljive
                                   i. */
  }
  while (i <= m);               /* Proverava se uslov i ukoliko je
                                   ispunjen, nastavlja se sa
                                   sledecom iteracijom. */
  /* U suprotnom, petlja se zavrsava i program se nastavlja od prve
     naredbe koja sledi za petljom. */
  printf("\n");

  return 0;
}
```

Rešenje 2.3.5

```
#include<stdio.h>

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

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

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

```
21    if (x >= 22) {
        printf("Broj je veliki, dolazi do prekoracenja.\n");
23      return -1;
      }
25
      original = x;
27    f = 1;

29    while (x > 1) {
        f = f * x;
31      x--;
      }
33
      printf("Faktorijel = %lu\n", f);
35
      return 0;
37  }
```

Rešenje 2.3.6

```
1   #include <stdio.h>

3   int main()
    {
5     int n;
      float x;
7     float vrednost;
      unsigned exp;
9
      printf("Unesite redom brojeve x i n: ");
11    scanf("%f %d", &x, &n);

13    if (n < 0) {
        printf("Neispravan unos.\n");
15      return -1;
      }
17
      /* Pocetna vrednost stepena koji se racuna.  */
19    vrednost = 1;

21    for (exp = 1; exp <= n; exp++)
        vrednost = vrednost * x;
23
      printf("%f\n", vrednost);
25
      return 0;
27  }
```

Rešenje 2.3.7

```c
#include <stdio.h>

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

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

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

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

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

  return 0;
}
```

Rešenje 2.3.8

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

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

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

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

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

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

  return 0;
}
```

Rešenje 2.3.9

```
#include <stdio.h>

int main()
{
  int n;

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

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

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

```
  }

  return 0;
}
```

Rešenje 2.3.10

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

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

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

  /* Kako uklanjamo cifre broja (pogledati telo petlje) u nekom
     trenutku broj ce postati 0 jer smo uklonili sve njegove cifre.
     Tada prekidamo rad petlje. */
  while (x > 0) {
    /* Izdvajamo poslednju cifru broja x. */
    cifra = x % 10;
    printf("%d\n", cifra);
    /* Uklanjamo poslednju cifru broja x. */
    x /= 10;
  }

  return 0;
}
```

Rešenje 2.3.11

```
#include <stdio.h>

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

```

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

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

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

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

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

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

  return 0;
}
```

Rešenje 2.3.12

```
#include<stdio.h>

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

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

  if (n < 0) {
    printf("Neispravan unos.\n");
```

```
    return -1;
  }

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

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

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

    i++;
  }

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

Rešenje 2.3.13

```
#include <stdio.h>

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

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

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

  printf("Unesite n brojeva: ");

  /* Inicijalizuje se brojac sa kojim se kontrolise broj *
     ucitavanja - treba da ih bude tacno n. */
  i = 0;
```

```
  while (i < n) {
    /* Ucitava se broj. */
    scanf("%d", &x);

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

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

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

  return 0;
}
```

Rešenje 2.3.14

```
#include <stdio.h>

int main()
{
  int n, broj;
  int suma = 0;
  /* Brojac. */
  int i;

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

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

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

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

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

  return 0;
}
```

Rešenje 2.3.15

```c
#include <stdio.h>

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

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

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

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

  if (n < 0) {
    printf("Broj artikala ne moze biti negativan.\n");
    return -1;
  }

  printf("Unesite cene artikala: ");

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

    if (cena <= 0) {
      printf("Cena ne moze biti negativna.\n");
      return -1;
    }

    /* Provera da li je cena manji od zadatog broja m. */
    if (cena < m) {
      /* Ako jeste, uvecava se brojac brojeva za 1. */
      broj_brojeva++;
    }

    i++;
  }

```

```
  printf("%d\n", broj_brojeva);

  return 0;
}
```

Rešenje 2.3.16

```
#include <stdio.h>
int main()
{
  int x;

  /* U promenljivoj p se cuva prozivod. */
  int p;

  /* Promenljiva u sluzi za proveru da li su brojevi uopste
     uneseni. Na pocetku se pretpostavlja da nisu i postavlja se na
     0. */
  int u = 0;

  /* Promenljiva unesen_pozitivan sluzi za proveru da li su
     pozitivni brojevi uopste uneseni. Na pocetku se pretpostavlja
     da nisu i postavlja se na 0. */
  int unesen_pozitivan = 0;

  p = 1;

  /* Izraz 1 je konstantan, razlicit je od nule sto znaci da je to
     tacan izraz. Uslov petlje je uvek tacan! */
  printf("Unesite brojeve:");

  while (1) {
    scanf("%d", &x);

    /* Proveravanje da li je uneta nula. */
    if (x == 0)

      /* Naredba break prekida petlju. Izvrsavanje se nastavlja od
         prve naredbe nakon petlje. */
      break;

    /* Ako je makar 1 broj razlicit od 0 promenljiva u ce biti
       postavljena na 1. */
    u = 1;

    /* Ako je unet negativan broj, taj broj se ne mnozi sa ukupnim
       proizvodom p; zato se nastavlja dalje. */
    if (x < 0)
      /* Naredba continue prekida trenutnu iteraciju petlje tako
         sto preskace sve naredbe koje nakon njega slede.
         Izvrsavanje se nastavlja od provere uslova petlje. */
```

```
45        continue;

47      /* Ako je makar jedan broj pozitivan, promenljiva
           unesen_pozitivan se postavja na 1. */
49      unesen_pozitivan = 1;
        p = p * x;
51    }

53    if (u == 0)
        printf("Nisu uneseni brojevi.\n");
55    else if (unesen_pozitivan == 0)
        printf("Nisu uneseni pozitivni brojevi. \n");
57    else
        printf("Proizvod pozitivnih unetih brojevi je %d.\n", p);
59
    return 0;
61  }
```

Rešenje 2.3.17

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    int main()
5   {
      int n, cifra;
7     int indikator = 0;

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

13    if (n < 0)
        n = abs(n);
15
      /* Sve dok imamo cifara u zapisu broja. */
17    while (n > 0) {

19      /* Izdvajamo posledjnju cifru broja. */
        cifra = n % 10;
21
        /* Proveravamo da li je bas ona jednaka broju 5 */
23      if (cifra == 5) {
          /* Ako jeste postavljamo indikator na vrednost 1 tako da
25           znamo da smo pronasli peticu i prekidamo sa izvrsavanjem
             petlje. */
27        indikator = 1;
          break;
29      }
        /* Ako izvdvojena cifra nije jednaka broju 5, broj delimo sa 10
31       * kako bi mogli da izdvojimo i preostale cifre broja na isti
```

```
     * nacin.
     */
    n = n / 10;
  }

  /* Ispisujemo rezultat */
  if (indikator == 0) {
    printf("Cifra 5 se ne nalazi u zapisu!\n");
  } else {
    printf("Cifra 5 se nalazi u zapisu!\n");
  }

  return 0;
}
```

Rešenje 2.3.18

```
#include <stdio.h>

int main()
{
  int x;
  int broj_brojeva;
  int suma;

  broj_brojeva = 0;
  suma = 0;

  printf("Unesite brojeve: ");

  while (1) {
    /* Ucitavanje broja. */
    scanf("%d", &x);

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

    /* Procitani broj dodaje se na sumu. */
    suma += x;

    /* I uvecava se broj ucitanih brojeva. */
    broj_brojeva++;
  }

  if (broj_brojeva == 0)
    printf("Nisu uneseni brojevi.\n");
  else
    /* Prilikom deljenja celih brojeva kao rezultat se dobija ceo
       broj. Kako je aritmeticka sredina realan broj, potrebno je
       izvrsiti konverziju prilikom deljenja da bi se dobio
```

```
35        ispravan rezultat. */
      printf("Aritmeticka sredina: %.4f\n",
37            (double) suma / broj_brojeva);

39    return 0;
}
```

Rešenje 2.3.19

```
1  #include <stdio.h>

3  int main()
   {
5    float cena;
     int broj_artikla;
7    float suma;

9    broj_artikla = 0;
     suma = 0;
11
     printf("Unesite cene: ");
13
     while (1) {
15     scanf("%f", &cena);

17     if (cena == 0)
         break;
19
       if (cena < 0) {
21       printf("Cena ne moze biti negativna.\n");
         return -1;
23     }

25     suma += cena;

27     /* I uvecava se broj ucitanih brojeva. */
       broj_artikla++;
29   }

31   if (broj_artikla == 0)
       printf("Nisu unesene cene.\n");
33   else
       printf("Aritmeticka sredina: %.4f\n", suma / broj_brojeva);
35
     return 0;
37 }
```

Rešenje 2.3.20

```c
#include <stdio.h>

int main()
{
  int n;
  /* Ucitavaju se dva broja, broj i sledbenik, i proverava se da li
     su razlicitog znaka. */
  double broj, sledbenik;
  /* Brojac. */
  int i;
  int broj_promena = 0;

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

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

  /* Prvo se proveara da li uopste ima unosa, i ako unosa nema,
     ispisuje se odgovarajuca poruka i izlazi iz programa. */
  if (n == 0) {
    printf("Broj promena je 0.\n");
    return 0;
  }

  printf("Unesite brojeve: ");
  /* Pre petlje ucitava se jedan broj, a u petlji se ucitava njegov
     sledbenik i proverava se da li su razlicitog znaka. */
  scanf("%lf", &broj);

  /* Kako je vec jedan broj unesen, brojac se postavlja na 1, a ne
     na 0. */
  for (i = 1; i < n; i++) {
    /* Ucitava se sledbenik. */
    scanf("%lf", &sledbenik);

    /* Ako su razlicitog znaka proizvod je manji od 0. */
    if (sledbenik * broj < 0)
      broj_promena++;
    /* Problem je ako je proizvod jednak 0. Tada mora provera da li
       je jedan od brojeva negativan jer tada postoji promena
       znaka. */
    else if (sledbenik * broj == 0 && (sledbenik < 0 || broj < 0))
      broj_promena++;

    /* Tekuci sledbenik postaje tekuci broj, a u sledecoj iteraciji
       petlje se ucitava sledeci sledbenik. */
    broj = sledbenik;
  }
```

```
53    printf("Broj promena je %d.\n", broj_promena);

55    return 0;
}
```

Rešenje 2.3.21

```
#include <stdio.h>

int main()
{
  /* Broj artikala. */
  int n;
  /* Brojac. */
  int i;
  /* Cena trenutno unetnog artikla. */
  float cena;
  /* Minimalna cena. */
  float min_cena;

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

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

  /* Prva cena se unosi iznad petlje kako bi bio njegova vrednost
     bila dodeljena promenljivoj min_cena. Neophodno je da
     promenljiva min bude inicijalizovana pre ulaska u petlju da bi
     uslov x<min mogao da bude ispitan u prvoj iteraciji. */
  printf("Unesite cenu artikala:");
  scanf("%f", &cena);
  /* Proveravamo da li je cena isprano uneta vrednost. */
  if (cena <= 0) {
    printf("Cena ne moze biti negativna.\n");
    return -1;
  }

  min_cena = cena;
  i = 0;
  while (i < n - 1) {
    scanf("%f", &cena);

    if (cena <= 0) {
      printf("Cena ne moze biti negativna.\n");
      return -1;
    }

```

```
    /* Provera da li je uneta cena manja od tekuce minimalne cene. */
    if (cena < min_cena)
      min_cena = cena;
    i++;
  }

  printf("Minimalna cena je: %f\n", min_cena);

  return 0;
}
```

Rešenje 2.3.22

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

int main()
{
  int n;
  int x, x_desetica;
  int max_desetica, broj;
  int i;

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

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

  if (n == 0) {
    printf("Nisu uneseni brojevi.\n");
    return 0;
  }

  /* Maksimalna cifra desetice se postavlja na na 0 jer 0 je
     svakako najmanja cifra pa je pocetna vrednost neutralna tj. ne
     moze da utice na izracunavanje maksimuma. Ipak, treba biti
     pazljiv jer nije uvek zgodno pretpostaviti da je maksimalna
     vrednost 0. Na primer, ako je zadatak naci maksimum celih
     brojeva, a korisnik unese -32 -7 i -22, maksimalni je broj -7,
     sto je manje od 0. */
  max_desetica = 0;

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

    /* Izdvajanje cifre desetica procitanog broja. */
    x_desetica = (abs(x) / 10) % 10;
```

```
39
      /* Proverava da li je izdvojena cifra veca od trenutne
41       maksimalne cifre desetica. */
      if (x_desetica > max_desetica) {
43      /* Ako jeste vece, pamti se nova najveca cifra, kao i broj u
           kom se pojavila. */
45      max_desetica = x_desetica;
        broj = x;
47    }
  }
49
  printf("Broj sa najvecom cifrom desetica je %d\n", broj);
51
  return 0;
53 }
```

Rešenje 2.3.23

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

7   int n;
    int x, x_kopija, br_cifara;
9   int max_br_cifara, broj;
    int i;
11
    printf("Unesite broj n: ");
13  scanf("%d", &n);

15  if (n < 0) {
      printf("Neispravan unos.\n");
17    return -1;
    }
19
    if (n == 0) {
21    printf("Nisu uneseni brojevi.\n");
      return 0;
23  }

25  /* Maksimalan broj cifara se postavlja na 0, svaki broj ima vise
       od 0 cifara pa je ova vrednost neutralna. */
27  max_br_cifara = 0;

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

33    /* Odredjivanje broja cifara unetog broja x. */
```

```
    x_kopija = abs(x);
    br_cifara = 0;
    while (x_kopija != 0) {
      x_kopija = x_kopija / 10;
      br_cifara++;
    }

    /* Ako je broj cifara unetog broja veci od maksimalnog */
    if (br_cifara > max_br_cifara) {
      /* Cuvamo ga */
      max_br_cifara = br_cifara;
      /* I zbog ispisa rezultata, cuvamo i originalni broj */
      /* Zbog ovoga smo morali i da racunamo broj cifara nad
         kopijom broja x kako ne bismo promenili njegovu vrednost */
      broj = x;
    }
  }

  printf("Najvise cifara ima broj %d\n", broj);

  return 0;
}
```

Rešenje 2.3.24

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

int main()
{

  int n;
  int x, x_kopija;
  int broj;
  int vodeca_cifra, max_vodeca_cifra;
  int i;

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

  /* Postavljamo maksimalnu vodecu cifru na 0 - cifre broja su vece
     ili jednake od 0 pa je ova vrednost neutralna */
  max_vodeca_cifra = 0;

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

    /* Odredjujemo vodecu cifru broja */
```

```
    x_kopija = abs(x);
    while (x_kopija > 10) {
      x_kopija = x_kopija / 10;
    }
    vodeca_cifra = x_kopija;

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

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

  return 0;

}
```

Rešenje 2.3.25

```
#include <stdio.h>

int main()
{
  int x;
  int min, max;

  printf("Unesite brojeve: ");

  /* Prvi broj se ucitava izvan petlje zbog inicijalizacije
     maksimuma i minimuma. */
  scanf("%d", &x);
  max = x;
  min = x;

  /* Sve dok se ne unese 0, ucitavaju se brojevi u petlji. */
  while (x != 0) {

    /* Provera da li je procitani broj veci od aktuelnog maksimuma.
     */
    if (x > max)
      max = x;
    /* Provera da li je procitani broj manji od aktuelnog minimuma.
     */
    if (x < min)
```

```
      min = x;

    /* Ucitavanje narednog broja. */
    scanf("%d", &x);
  }

  printf("Razlika: %d\n", max - min);

  return 0;
}
```

Rešenje 2.3.26

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

int main()
{
  int n;
  int d;
  /* Uzastopni brojevi za koje se racuna rastojanje. */
  int x, y;
  int broj_parova;
  int i;


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

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

  broj_parova = 0;

  printf("Unesite n brojeva: ");

  /* Prvi broj se ucitava pre petlje. */
  scanf("%d", &x);

  for (i = 1; i < n; i++) {
    scanf("%d", &y);

    /* Provera da li su brojevi na rastojanju d. */
    if (abs(y - x) == d)
      broj_parova++;

    /* Broj iz tekuce iteracije se cuva kako bi mogao da se
       upotrebljava u narednoj iteraciji. */
    x = y;
```

```
39    }

41    printf("Broj parova: %d\n", broj_parova);

43    return 0;
}
```

Rešenje 2.3.27

```
#include <stdio.h>

int main()
{
  int x;
  /* Tezina trenutne pozicije u broju. Moze biti 1, 10, 100, 1000
     itd. */
  int pozicija;
  /* Trenutna izdvojena cifra iz broja x. */
  int cifra;
  /* Broj dobijen nakon transformacije. */
  unsigned int y;

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


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

  /* Posto pocinjemo sa izdvajanjem cifara od cifre jedinica,
     postavlja se tezinu (stepen) pozicije na 1. */
  pozicija = 1;
  y = 0;

  /* Provera da li ima cifara u zapisu broja. */
  while (x > 0) {

    /* Izdvaja se poslednja cifra iz zapisa. */
    cifra = x % 10;

    /* Provera da li je cifra parna. */
    if (cifra % 2 == 0) {
      /* I ako jeste, uvecava se. */
      cifra++;

    }

    /* Novi broj se formira tako sto se izdvojena cifra pomnozi
       odgovarajucom tezinom (stepenom) pozicije. */
```

```
43      y += cifra * pozicija;
45      /* Priprema se broj za izdvajanje naredne cifre, uklanja se
           poslednja cifra broja. */
47      x /= 10;
49      /* Uvecava se tezinu (stepen) pozicije. */
        pozicija *= 10;
51    }
53    /* Ispisuje se izracunatu vrednost. */
      printf("%d\n", y);
55
      return 0;
57  }
```

Rešenje 2.3.28

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

5   int main()
    {
7     int x;
      /* Tezina trenutne pozicije u broju. Moze biti 1, 10, 100, 1000
9        itd. */
      int stepen_deset;
11    /* Trenutna izdvojena cifra iz broja x. */
      int cifra;
13    /* Redni broj cifre koja se trenutno obradjuje, gledano s desna
         na levo. */
15    int rbr;
      /* Broj dobijen nakon transformacije. */
17    int y;
      /* Promenljiva znak cuva znak unesenog broja. Moze biti -1 za
19       negativnu vrednost ili 1 za poziivnu vrednost. */
      int znak = 1;
21
      /* Ucitavanje broja. */
23    printf("Unesite broj: ");
      scanf("%d", &x);
25
      if (x <= 0) {
27      x = abs(x);
        znak = -1;
29    }
      /* Postavlja se vrednost stepena na 0 - to znaci da se prvo mnozi
31       sa 10^0=1. */
      stepen_deset = 0;
33
```

```
  /* Postavlja se vrednost broja koji se formira na 0. */
  y = 0;
  /* Postavlja se redni broj pozicije na 0. */
  rbr = 0;

  /* Provera da li ima cifara u zapisu broja. */
  while (x > 0) {

    /* Izdvajanje cifre. */
    cifra = x % 10;

    /* Proverava se da li je pozicija izdvojene cifre parna - cifre
       na parnim pozicijama se zadrzavaju. */
    if (rbr % 2 == 0) {
      /* Ako jeste parna izdvojena cifra se dodaje novom broju.
         Neophodno je izvrsiti promenu tipova, jer je double
         povratni tip funkcije pow. */
      y += cifra * ((int) pow(10, stepen_deset));

      /* Uvecava se stepen zbog naredne cifre. */
      stepen_deset++;
    }

    /* Azurira se redni broj cifre. */
    rbr++;
    /* I priprema se broj za naredno izdvajanje. */
    x /= 10;
  }

  y = znak * y;

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

  return 0;
}
```

Rešenje 2.3.29

```
#include <stdio.h>

int main()
{
  int n, novo_n;
  int stepen;
  int cifra_levo, cifra_sredina, cifra_desno;

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

```
13    if (n <= 0) {
        printf("Neispravan unos.\n");
15      return -1;
      }
17
      /* Stepen broja 10 sa kojim se mnoze cifre izdvojenog broja. */
19    stepen = 1;

21    /* Nova vrednost broja. */
      novo_n = 0;
23
      /* Provera da li u zapisu broja postoje barem tri cifre.  */
25    while (n > 99) {
        /* Izdvaja se srednja cifra, cifra desno od nje i cifra levo od
27         nje: npr. za trojku 583 8 je srednja cifra, 3 je cifra
           desno, a 5 cifra levo. */
29      cifra_desno = n % 10;
        cifra_sredina = (n / 10) % 10;
31      cifra_levo = (n / 100) % 10;

33      /* U novi broj se smesta desna cifra. */
        novo_n += cifra_desno * stepen;
35
        /* Azurira se vrednost stepena. */
37      stepen = stepen * 10;

39      /* Provera da li je srednja cifra jednaka zbiru leve i desne
           cifre. */
41      if (cifra_levo + cifra_desno == cifra_sredina) {

43        /* Treba izbaciti srednju cifru, pa broj n se azurira tako
             sto se podeli sa 100. */
45        n = n / 100;
        } else {
47
          /* Inace, zadrzava se srednja cifra i odbacuje se samo
49           poslednja. */
          n = n / 10;
51      }
      }
53
      /* Na novi broj se dodaje preostali dvocifreni ili jednocifreni
55       broj. */
      novo_n = n * stepen + novo_n;
57
      /* Ispisivanje rezultata. */
59    printf("%d\n", novo_n);


61
      return 0;
63
}
```

Rešenje 2.3.30

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

int main()
{
  int x;
  int broj_cifara;
  int min_stepen, max_stepen;
  int pom;
  int leva_cifra, desna_cifra;
  int indikator;

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

  /* Ako je korisnik uneo negativan broj, analizira se njegova
     apsolutna vrednost. */
  if (x < 0)
    x = -x;

  /* Odredjuje se broj cifara u zapisu broja x da bi moglo da se
     izdvajaju istovremeno cifre i sa leve i sa desne strane. */
  broj_cifara = 0;
  pom = x;
  while (pom > 0) {
    pom /= 10;
    broj_cifara++;
  }

  /* Odredjuje se stepen koji stoji uz krajnju levu cifru broja. */
  max_stepen = (int) pow(10, broj_cifara - 1);

  /* Indikator je promenljiva koja ukazuje da li je broj palindrom
     ili ne. */
  indikator = 1;
  while (x != 0 && indikator == 1) {
    /* Izdvaja se leva cifra. */
    leva_cifra = x / max_stepen;
    /* Izdvaja se desna cifra. */
    desna_cifra = x % 10;
    /* Ako su cifre razlicite, odmah se moze zakljuciti da broj
       nije palindrom i prekida se izvrsavanje petlje. */
    if (leva_cifra != desna_cifra) {
      indikator = 0;
      break;
    }
    /* Formira se nova vrednost broja x tako sto se odbacuje
       krajnja leva i krajnja desna cifra. */
    x = (x % max_stepen - x % 10) / 10;
    /* Koriguje se maksimalan stepen tako dobijenog broja - deli se
```

```
51        sa 100 jer su odbacene 2 cifre. */
      max_stepen = max_stepen / 100;
53    }

55    /* Ispisuje se rezultat. */
      if (indikator == 1)
57      printf("Broj je palindrom!\n");
      else
59      printf("Broj nije palindrom!\n");

61    return 0;
  }
```

Rešenje 2.3.31

```
1  #include <stdio.h>

3  int main()
   {
5    /* Pamtimo uzastopna dva Fibonacijeva broja i na osnovu njih
        racunamo sledeci. */
7    /* Promenljive prvi i drugi su brojevi koje pamtimo i na osnovu
        njih racunamo treci. */
9    /* Na osnovu teksta zadatka, promenljive prvi i drugi postavljamo
        na 1. */
11   int prvi = 1;
     int drugi = 1;
13   int treci;
     /* Promenljiva pozicija je podatak koji ucitavamo i odnosi se na
15      poziciju u Fibonacijevom nizu za koju treba izracunati
        vrednost. */
17   int pozicija;
     /* Promenljiva i oznacava do koje pozicije smo izracunali
19      vrednosti. Kako imamo prve dve vrednosti, ovu promenljivo
        postavljamo na 2. */
21   int i = 2;

23   printf("Unesite poziciju u Fibonacijevom nizu: ");
     scanf("%d", &pozicija);
25
     /* Pozicija ne moze biti 0 i ne moze biti negativan broj. */
27   if (pozicija < 1) {
       printf
29         ("Neispravan unos. Pozicija u Fibonacijevom nizu mora biti
       pozitivan broj koji nije 0!\n");
       return -1;
31   }

33   while (i < pozicija) {
       /* Na osnovu dva uzastopna racunamo treci. */
35     treci = prvi + drugi;
```

```
37      /* Potom razmenjujemo vrednosti. Uzastopna dva koja pamtimo
           postaju sledeca uzastopna dva broja Fibonacijevog niza. */
39      prvi = drugi;
        drugi = treci;
41
        /* Prelazimo na racunanje sledeceg broja na sledecoj poziciji. */
43      i++;
    }
45
    printf("Trazeni broj je: %d\n", drugi);
47
    return 0;
49  }
```

Rešenje 2.3.32

```
#include<stdio.h>
2
int main()
4  {
    int a0;
6   int an, an1;
    printf("Unesi pocetni clan niza brojeva:");
8   scanf("%d", &a0);
    if (a0 <= 0) {
10    printf("Nekorektan unos. Broj mora biti pozitivan.\n");
      return -1;
12  }
    printf("%d\n", a0);
14  an = a0;
    while (an != 1) {
16    if (an % 2) {
        /* Ukoliko je vrednost izraza an%2 razlicita od nule, izraz
18         se tumaci kao tacan i izvrsavaju se naredbe iz if grane. */
        an1 = (3 * an + 1) / 2;
20    } else {
        /* U suprotnom, ukoliko je vrednost izraza an%2 jednaka nuli,
22         izraz se tumaci kao netacan i izvrsavaju se naredbe iz
           else grane. */
24      an1 = an / 2;
      }
26    printf("%d\n", an1);
      an = an1;
28  }
    return 0;
30  }
```

Rešenje 2.3.33

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

int main()
{
  int format;
  /* Pomocna promenljiva koja sluzi kao brojac u petlji. */
  int i;
  /* Trenutne vrednosti za sirinu i visinu i pomocna promenljiva za
     promene u petlji. */
  double sirina, duzina, nova_duzina;
  unsigned int konacna_sirina, konacna_duzina;

  printf("Uneti format papira: ");
  scanf("%d", &format);

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

  /* duzina/sirina = 1 : sqrt(2) duzina*sirina = 1000x1000mm^2 Na
     osnovu ovih odnosa dobijamo pocetnu vrednost za sirinu i
     duzinu, odnosno vrednosti za papir A0. */
  duzina = sqrt(1000 * 1000 / sqrt(2));
  sirina = sqrt(2) * duzina;

  /* Kako vec imamo odredjenu sirinu i duzinu za papir A0, petlju
     krecemo od izracunavanja za papir A1, pa brojac i postavljamo
     na 1. */
  for (i = 1; i <= format; i++) {
    nova_duzina = sirina / 2;
    sirina = duzina;
    duzina = nova_duzina;
  }

  /* Duzina i sirina celi brojevi. */
  konacna_sirina = (unsigned int) sirina;
  konacna_duzina = (unsigned int) duzina;

  printf("%u %u\n", konacna_duzina, konacna_sirina);

  return 0;
}
```

Rešenje 2.3.34

```c
#include <stdio.h>

int main()
```

```
{
  char c;

  /* Funkcija getchar ucitava jedan karakter. Naredbom dodele
     (c=getchar()) promenljivoj c bice dodeljena vrednost ascii
     koda unetog karaktera. Obratiti posebnu paznju na zagrade. */

  while ((c = getchar()) != '.') {
    if (c >= 'A' && c <= 'Z')
      /* Razlika izmedju ascii koda svakog malog i odgovarajuceg
         velikog slova je konstanta koja se moze sracunati izrazom
         'a'-'A' (i iznosi 32). */
      putchar(c + 'a' - 'A');
    else if (c >= 'a' && c <= 'z')
      putchar(c - 'a' + 'A');
    else
      putchar(c);
  }

  return 0;
}
```

Rešenje 2.3.35

```
#include <stdio.h>

int main()
{
  char c;

  /* Inicijalizacija brojaca na 0. */
  int br_v = 0;
  int br_m = 0;
  int br_c = 0;
  int br_b = 0;
  int br_k = 0;
  int suma = 0;

  /* Petlja se zavrsava kada korisnik ne unese karakter, vec zada
     konstantu EOF . Ova konstanta se zadaje kombinacijom tastera
     CTRL+D. U tom slucaju, getchar() vraca -1. */
  while ((c = getchar()) != EOF) {
    if (c >= 'A' && c <= 'Z')
      br_v++;
    else if (c >= 'a' && c <= 'z')
      br_m++;
    else if (c >= '0' && c <= '9') {
      br_c++;
      /* Kada od promenljive tipa char oduzimamo karakter (ili neku
         drugu promenljivu tipa char), zapravo se vrsi oduzimanje
         njihovih ascii vrednosti i dobija se broj. */
```

```
28        suma = suma + c - '0';
      } else if (c == '\t' || c == '\n' || c == ' ')
30        br_b++;

32      br_k++;
    }
34
    printf("velika: %d, mala: %d, cifre: %d, beline: %d \n", br_v,
36          br_m, br_c, br_b);
    printf("suma cifara: %d\n", suma);
38
    return 0;
40  }
```

Rešenje 2.3.36

```
#include <stdio.h>
2
int main()
4  {
    /* Promenljiva i je brojac. */
6   int n, i;
    /* Brojaci za svaki od samoglasnika. */
8   int broj_a = 0, broj_e = 0, broj_i = 0, broj_o = 0, broj_u = 0;
    /* Promenljiva c je tekuci ucitani karakter. */
10  char c, belina;

12  printf("Unesite broj n: ");
    scanf("%d", &n);
14
    if (n < 0) {
16    printf("Neispravan unos.\n");
      return -1;
18  }

20  for (i = 0; i < n; i++) {
      /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
22       pa tek posle procitane beline se cita uneseni karakter. */
      scanf("%c%c", &belina, &c);
24
      /* Provera da li je ucitani karakter samoglasnik. */
26    switch (c) {
      case 'a':
28    case 'A':
        broj_a++;
30      break;
      case 'e':
32    case 'E':
        broj_e++;
34      break;
      case 'i':
```

```
    case 'I':
      broj_i++;
      break;
    case 'o':
    case 'O':
      broj_o++;
      break;
    case 'u':
    case 'U':
      broj_u++;
      break;
    }
  }

  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 2.3.37

```
/* Sa standardnog ulaza se unosi ceo broj n, a zatim i n karaktera.
   Napisati program koji proverava da li se od unetih karaktera
   moze napisati rec Zima. */

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

int main()
{
  int n;
  int broj_Z, broj_i, broj_m, broj_a;
  char novi_red, c;
  int i;

  broj_Z = 0;
  broj_i = 0;
  broj_m = 0;
  broj_a = 0;

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

  /* Ucitavanje karakter po karakter. */
  for (i = 0; i < n; i++) {
    printf("Unestite %d. karakter: ", i + 1);
    /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
```

```
27        pa tek posle procitane beline se cita uneseni karakter. */
      scanf("%c%c", &novi_red, &c);
29
      /* Analiziramo karakter */
31    switch (c) {
      case 'Z':
33      broj_Z++;
        break;
35    case 'i':
        broj_i++;
37      break;
      case 'm':
39      broj_m++;
        break;
41    case 'a':
        broj_a++;
43      break;
      }
45  }

47  /* Ako u unosu ima barem jedno veliko slovo z i barem po jedno
       malo slovo i, m i a, rec se moze napisati. A u suprotnom ne
49     moze. */
    if (broj_Z && broj_i && broj_m && broj_a) {
51    printf("Moze se napisati rec Zima.\n");
    } else {
53    printf("Ne moze se napisati rec Zima.\n");
    }
55
    return 0;
57 }
```

Rešenje 2.3.38

```
1  #include <stdio.h>
   int main()
3  {
     int n;
5    /* Brojac. */
     int i;
7    /* Promenljiva u kojoj se cuva suma kubova. */
     int s;
9
     printf("Unesite pozitivan ceo broj:");
11   scanf("%d", &n);
     if (n < 0) {
13     printf("Neispravan unos.\n");
       return -1;
15   }

17   for (s = 0, i = 1; i <= n; i++)
```

```
    s += i * i * i;

  printf("Suma kubova od 1 do %d: %d\n", n, s);

  return 0;
}
```

Rešenje 2.3.39

```
#include <stdio.h>

int main()
{
  int n;
  /* Brojac. */
  int i;
  /* Promenljiva u kojoj se cuva suma kubova. */
  int s;

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

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

  i = 1;
  s = 0;

  for (i = 1; i <= n; i++) {
    s += i * i * i;
    printf("i=%d, s=%d\n", i, s);
  }

  return 0;
}
```

Rešenje 2.3.40

```
#include <stdio.h>

int main()
{
  int n, i;
  float x, S, stepen;

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

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

  /* Inicijalizacija sume. */
  S = 0;

  /* Stepen promenljiva ce sadrzati vrednosti stepena x^n. Pocetna
     vrednost joj je 1 jer je x^0 = 1. */
  stepen = 1;

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

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

  return 0;
}
```

Rešenje 2.3.41

```
#include <stdio.h>

int main()
{
  unsigned n, i;
  float x, S, stepen;

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

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

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

  return 0;
}
```

Rešenje 2.3.42

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

int main()
{
  /* Promenljiva i je brojac, promenljiva S cuva izracunatu sumu, a
     promenljiva clan je tekuci clan niza. */
  int i;
  float S;
  float x, eps;
  float clan;

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

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

  S = 0;
  /* Prvi clan sume je 1. */
  clan = 1;
  i = 1;
  while (clan > eps) {
    S = S + clan;
    clan = clan * x / i;
    i++;
  }

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

  return 0;
}
```

Rešenje 2.3.43

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

int main()
{
  /* Promenljiva i je brojac, promenljiva S cuva izracunatu sumu,
     promenljiva znak moze bito 1 ili -1 i odredjuje znak trenutnog
     clana sume, a promenljiva clan je tekuci clan niza. */
  int i, znak;
  float S;
  float x, eps, clan;

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

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


  S = 0;
  clan = 1;
  i = 1;
  znak = -1;

  /* Kako clanovi sume mogu biti negativni, potrebno je posmatrati
     apsolutnu vrednost clana. */
  while (fabs(clan) > eps) {
    S = S + clan;

    /* Promena znaka. */
    clan = clan * x / i;
    clan *= znak;

    i++;
  }

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

  return 0;
}
```

Rešenje 2.3.44

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

int main()
{
  int n, i;
  double x;
  double stepen = 1;
  double proizvod = 1;

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

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

  for (i = 0; i < n; i++) {
    stepen *= x;
    proizvod *= 1 + cos(stepen);
  }

```

```
  printf("Proizvod = %lf\n", proizvod);

  return 0;
}
```

Rešenje 2.3.45

```
#include <stdio.h>

int main()
{
  int n, i;
  double Razlomak;

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

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

  Razlomak = n;

  /* Razlomak se izracunava "od nazad", odnosno, krece se od
     najnizeg razlomka 1/n i od njega se nadalje formira sledeci,
     "visi" razlomak itd. Zavrsava se kada se stigne do koraka 0 +
     1/R. */
  for (i = n - 1; i >= 0; i--)
    Razlomak = i + 1 / Razlomak;

  printf("Razlomak = %lf\n", Razlomak);

  return 0;
}
```

Rešenje 2.3.46

```
#include <stdio.h>

int main () {
    /* Promenljiva i je brojac , promenljiva S cuva izracunatu sumu ,
    promenljiva znak moze bito 1 ili -1 i odredjuje znak trenutnog
    clana sume , a promenljiva clan je tekuci clan niza. */

    int i, znak , n;
    float S;
    float x, clan;
    printf(" Unesite x i n: ");
```

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

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

    S = 1;
    clan = 1;
    i = 1;
    znak = -1;

    /* Kako clanovi sume mogu biti negativni , potrebno je posmatrati
    apsolutnu vrednost clana. */
    while (i <= 2 * n - 1) {

        /* Promena znaka. */
        /* Svaki clan suma se od prethodnog clana razlikuje za
        x^2/(i*(i+1)). */
        clan = clan * x * x / (i * (i + 1));
        clan *= znak;

        S = S + clan;

        i += 2;
    }

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

    return 0;
}
```

Rešenje 2.3.47

```
#include <stdio.h>

int main()
{
  int n, i;
  /* Promenljiva clan je deo proizvoda i predstavlja 1/i!. */
  double clan;
  double S = 1;

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

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

```

```
  clan = 1;
  for (i = 2; i <= n; i++) {
    clan = clan / i;
    S *= 1 + clan;
  }

  printf("S = %lf\n", S);

  return 0;
}
```

Rešenje 2.3.48

```
#include <stdio.h>

int main()
{
  int n, i, znak = -1;
  /* Promenljiva clan je deo proizvoda i predstavlja 1*3*5*...*i. */
  long int clan;
  long int S = 0;

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

  if (n < 5 || n % 2 == 0) {
    printf("Neispravan unos.\n");
    return -1;
  }

  clan = 1 * 3;
  for (i = 5; i <= n; i += 2) {
    clan = znak * clan * i;
    S += clan;
  }

  printf("S = %ld\n", S);

  return 0;
}
```

Rešenje 2.3.49

```
#include <stdio.h>

int main()
{
  int n, i;
  double P;
```

```
  double x, a;

  printf("Unesite dva relana broja x i a: ");
  scanf("%lf%lf", &x, &a);

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

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

  P = x;
  for (i = 0; i < n; i++)
    P = (P + a) * (P + a);

  printf("Izraz = %lf\n", P);

  return 0;
}
```

Rešenje 2.3.50

Rešenje (a)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= n; i++) {
    for (j = 1; j <= n; j++)
      /* U tablici mnozenja vrednost svakog polja je proizvod vrste
         i kolone u kojoj se nalazi. */
      printf("%3d ", i * j);

    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Brojac koji broji koliko brojeva je ispisano u jednom redu. */
  j = 0;
  for (i = 1; i <= n * n; i++) {
    printf("%3d ", i);
    /* Uvecavamo brojac */
    j++;

    /* Kada je ispisano n brojeva u jednom redu, ispisuje se znak
       za novi red, da bi ispis krenuo u novom redu i vrednost
       brojaca j se postavlja na 0 jer u novom redu jos ni jedan
       broj nije ispisan. */
    if (j == n) {
      j = 0;
      printf("\n");
    }
  }

  return 0;
}
```

Rešenje (c)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= n; i++) {
    for (j = 0; j < n; j++)
      if ((j + i) % n == 0)
        printf("%3d", n);
      else
        printf("%3d", (j + i) % n);

    printf("\n");
  }
```

```
  return 0;
}
```

Rešenje (d)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    for (j = 0; j < n - i; j++)
      printf("(%d, %d)", i, j);

    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.51

Rešenje (a)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Kvadrat predstavlja tabelu sa n vrsta gde svaka vrsta sadrzi
       n polja, a svako polje je isto i predstavlja karakter *. */
    for (j = 0; j < n; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (b)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter
       *, a unutrasnjost kvadrata je karakter blanko. */
    for (j = 0; j < n; j++)
      /* Provera da li je ivica. */
      if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje (c)

```c
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter
       *, a unutrasnjost kvadrata je karakter blanko osim na
       mestima na kojima je glavna dijagonala. */
    for (j = 0; j < n; j++)
      /* Provera da li je ivica ili glavna dijagonala. */
      if (j == 0 || j == n - 1 || i == 0 || i == n - 1 || i == j)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }
```

```
23    return 0;
}
```

Rešenje 2.3.52

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 0; i < n; i++) {
    /* Veliko slovo X se moze posmatrati kao dijagonale kvadrata
       (glavna i sporedna). Zato, treba ispisivati blanko na
       mestima gde nije dijagonala, a karakter * na mestima gde je
       neka od dijagonala. */
    for (j = 0; j < n; j++)
      /* Provera da li je mesto glavne ili sporedne dijagonale. */
      if (i == j || i + j == n - 1)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.53

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  if (n % 2 == 0) {
    printf("Pogresan unos.\n");
    return -1;
  }

  for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++)
```

```
17        if (i == n / 2 || j == n / 2)
            printf("+");
19        else
            printf(" ");
21      printf("\n");
    }
23
    return 0;
25  }
```

Rešenje 2.3.54

Rešenje (a)

```
#include <stdio.h>
2
int main()
4  {
    unsigned int n, i, j;
6
    printf("Unesite broj n: ");
8   scanf("%u", &n);

10  for (i = 0; i < n; i++) {
      for (j = 0; j < n - i; j++)
12      printf("*");
      printf("\n");
14  }

16  return 0;
}
```

Rešenje (b)

```
1  #include <stdio.h>

3  int main()
   {
5    unsigned int n, i, j;

7    printf("Unesite broj n: ");
     scanf("%u", &n);
9
     for (i = 0; i < n; i++) {
11     for (j = 0; j <= i; j++)
         printf("*");
13     printf("\n");
     }
```

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

Rešenje (c)

```
1  #include <stdio.h>

3  int main()
   {
5    unsigned int n, i, j;

7    printf("Unesite broj n: ");
     scanf("%u", &n);
9
     for (i = 0; i < n; i++) {
11     /* Prvo se ispisuju beline koje prethode karakterima *. */
       for (j = 0; j < i; j++)
13       printf(" ");
       /* Posle belina se ispisuje potreban broj karaktera *. */
15     for (j = 0; j < n - i; j++)
         printf("*");
17     printf("\n");
     }
19
     return 0;
21 }
```

Rešenje (d)

```
1  #include <stdio.h>

3  int main()
   {
5    unsigned int n, i, j;

7    printf("Unesite broj n: ");
     scanf("%u", &n);
9
     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
11   for (i = 0; i < n; i++) {
       /* Prvo se ispisuju beline koje prethode karakterima *. */
13     for (j = 0; j < n - i - 1; j++)
         printf(" ");
15     /* Posle belina se ispisuje potreban broj karaktera *. */
       for (j = 0; j <= i; j++)
17       printf("*");
       printf("\n");
19   }
```

```
  return 0;
}
```

Rešenje (e)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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


  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j <= i; j++)
      printf("*");
    printf("\n");
  }

  /* Potrebno je iscrtati i donji deo slike, odnosno donji trougao.
     Brojac i odredjuje koji red donjeg trougla se trentno iscrtava.
     * Kako je prvi red dodnjeg trougla vec iscrtan (to je poslednji
     red gornjeg trougla), brojac se postavlja na 1. */
  for (i = 1; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < i; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < n - i; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje (f)

```
#include <stdio.h>

int main()
```

```
{
  unsigned int n, i, j;
  char c, blanko;

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

  printf("Unesite karakter c: ");
  /* Zbog pritiskanja tastera ENTER nakon unosa promenljive broj
     potrebno je ucitati karakter za novi red u promenljivu blanko
     pre ucitavanja karaktera kojim se iscrtava trougao. */
  scanf("%c%c", &blanko, &c);

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Iscrtavaju se samo ivice trougla, ostalo se popunjava
       belinama. */
    for (j = 0; j <= i; j++)
      if (i == n - 1 || j == 0 || j == i)
        printf("%c", c);
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.55

Rešenje (a)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < 2 * i + 1; j++)
      printf("*");
    printf("\n");
```

```
19    }

21    return 0;
}
```

Rešenje (b)

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
     izracunavanja koliko zvezdica i praznina je potrebno ispisati
     u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
     iteraciji petlje. */
  for (i = n - 1;; i--) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i + 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < 2 * i + 1; j++)
      printf("*");
    printf("\n");

    /* Posebna paznja mora da se obrati na cinjenicu da su brojaci
       tipa unsigned int. Problem nastaje kada je i==0 i pokusa se
       oduzimanje (i--). Posto su brojevi unsigned, nova vrednost
       nece biti -1, vec pozitivan ceo broj. Imajuci to na umu,
       uslov i>=0 ne moze da se stavi u uslov za for petlju. Mnogo
       sigurnije je brojace deklarisati da budu tipa int i izbeci
       ovakvu vrstu problema. */
    if (i == 0)
      break;
  }

  return 0;
}
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
```

```
  unsigned int n;
  int i, j;

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

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < 2 * i + 1; j++)
      printf("*");
    printf("\n");
  }

  /* Sada se ispisuje donji trougao. Kako je prvi red donjeg
     trougla vec ispisan (poslednji red gornjeg trougla), potrebno
     je naciniti jednu iteraciju manje. */

  /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
     izracunavanja koliko zvezdica i praznina je potrebno ispisati
     u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
     iteraciji petlje. */
  for (i = n - 2; i >= 0; i--) {
    /* Prvo se ispisuju 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()
{
  unsigned int n, i, j;

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

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
```

```
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje sam troougao. Ako je brojac na
       ivici onda se ispisuje karakter *, a inace praznina.
       Takodje, proverava se da li se ispisuje poslednji red (i==n)
       i u njemu se ispisuje svaki drugi put *, a inace praznina.
       Kako se ispisuje svaki drugi put vrsi se provera j%2 == 0. */
    for (j = 0; j < 2 * i + 1; j++)
      if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  return 0;
}
```

Rešenje (c)

```
#include <stdio.h>

int main()
{
  unsigned int n;
  int i, j;

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

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    /* Posle belina se ispisuje sam troougao. Ako je brojac na
       ivici onda se ispisuje karakter *, a inace praznina.
       Takodje, proverava se da li se ispisuje poslednji red (i==n)
       i u njemu se ispisuje svaki drugi put *, a inace praznina.
       Kako se ispisuje svaki drugi put vrsi se provera j%2 == 0. */
    for (j = 0; j < 2 * i + 1; j++)
      if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  /* Sada se ispisuje donji trougao. Kako je prvi red donjeg
     trougla vec ispisan (poslednji red gornjeg trougla), potrebno
```

```
31      je naciniti jednu iteraciju manje. */

33    for (i = n - 2; i >= 0; i--) {
        /* Prvo se ispisuju beline koje prethode karakterima *. */
35      for (j = 0; j < n - i - 1; j++)
          printf(" ");
37      /* Posle belina se ispisuje potreban broj karaktera *. */
        for (j = 0; j < 2 * i + 1; j++)
39        if (j == 0 || j == 2 * i)
            printf("*");
41        else
            printf(" ");
43      printf("\n");
    }
45
    return 0;
47  }
```

Rešenje 2.3.56

```
1   #include <stdio.h>

3   int main()
    {
5     unsigned int n, i, j;

7     printf("Unesite broj n: ");
      scanf("%u", &n);
9
      /* Strelica se moze posmatrati kao spojena dva pravougla trougla
11       kojima se ispisuje hipotenuza i jedna, donja kateta. */

13    /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
      for (i = 0; i < n; i++) {
15      for (j = 0; j <= i; j++)
          /* Proverava se da li se ispisuje karakter na hipotenuzi (j
17           == i-1) ili da se ispisuje poslednji red (i == n-1). */
          if (j == i || i == n - 1)
19          printf("*");
          else
21          printf(" ");
        printf("\n");
23    }

25    /* Potrebno je iscrtati i donji deo slike, odnosno donji trougao.
         Brojac i odredjuje koji red donjeg trougla se trentno iscrtava.
27       * Kako je prvi red dodnjeg trougla vec iscrtan (to je poslednji
         red gornjeg trougla), brojac se postavlja na 1. */
29    for (i = 1; i < n; i++) {
        for (j = 0; j < n - i; j++)
31        /* Provera da li se ispisuje hipotenuza. */
```

```
      if (j == n - i - 1)
33      printf("*");
      else
35      printf(" ");
    printf("\n");
37  }

39  return 0;
}
```

Rešenje 2.3.57

```
1  #include <stdio.h>

3  int main()
   {
5    unsigned int n;
     int i, j, k;
7
     printf("Unesite broj n: ");
9    scanf("%u", &n);

11   /* Brojac j odredjuje koliko ukupno karaktera (praznina i
        karaktera *) u svakom redu se ispisuje. U svakom drugom redu
13      ovaj broj se povecava za 2. Na pocetku je 1 (jer se ispisuje
        samo jedna zvezda). */
15   j = 1;

17   /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
     for (i = 1; i <= n; i++) {
19     /* U svakom drugom redu broj ispisanih karaktera se uvecava za
          2. */
21     if (i % 2 == 0)
         j += 2;
23     for (k = 0; k < j; k++)
         /* U svakom drugom redu se naiazmenicno ispisuje * ili
25          praznina. */
         if (i % 2 == 0) {
27         if (k % 2 == 0)
             printf("*");
29         else
             printf(" ");
31       } else
           printf("*");
33
       printf("\n");
35   }

37   return 0;
}
```

Rešenje 2.3.58

```c
#include <stdio.h>

int main()
{
  unsigned int n, m;
  int i, j, k;

  printf("Unesite brojeve n i m: ");
  scanf("%u%u", &n, &m);

  for (i = 1; i <= m; i++) {
    /* Za svaki kvadrat se racuna duzina bez poslednje ivice.
       Kvadrat je sastavljen od (m-1) zvezdice i (m-1) praznine
       (praznine se nalaze izmedju zvezdica). Znaci ukupna duzina
       je 2*(m-1) karakter, a kako ima n kvadrata, duzina je
       n*2*(m-1). */
    for (j = 0; j <= n * 2 * (m - 1); j++)
      /* Provera da li se ispisuje prvi ili poslednji red. */
      if (i == 1 || i == m)
        /* Naizmenicno se ispisuje * i praznina. */
        if (j % 2 == 0)
          printf("*");
        else
          printf(" ");
      else
        /* Na kraju svakog kvadrata (nakon svake (m-1) zvezdice i
           (m-1) praznine se ispisuje ivica kvadrata. */
      if (j % (2 * (m - 1)) == 0)
        printf("*");
      else
        printf(" ");

    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.59

```c
#include <stdio.h>

int main()
{
  unsigned int n;
  int i, j;

  printf("Unesite broj n: ");
```

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

  /* Potrebno je spojiti sve slike u jednu, sliku gornjeg dela
     romba i sliku donjeg dela romba. */

  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
  for (i = 0; i < n; i++) {
    /* Prvo se ispisuju * koje prethode karakterima -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    /* Posle * se ispisuje potreban karakter -. */
    for (j = 0; j < 2 * i; j++)
      printf("-");
    /* Potom se ispisuju * koje su nakon karaktera -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    printf("\n");
  }

  /* Sada se ispisuje donji trougao. Kako je prvi red donjeg
     trougla vec ispisan (poslednji red gornjeg trougla), potrebno
     je naciniti jednu iteraciju manje. */

  /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
     izracunavanja koliko zvezdica i praznina je potrebno ispisati
     u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
     iteraciji petlje. */
  for (i = n - 2; i >= 0; i--) {
    /* Prvo se ispisuju * koje prethode karakterima -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    /* Posle * se ispisuje potreban karakter -. */
    for (j = 0; j < 2 * i; j++)
      printf("-");
    /* Potom se ispisuju * koje su nakon karaktera -. */
    for (j = 0; j < n - i; j++)
      printf("*");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.60

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;
```

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

  /* Slika se sastoji iz dva dela, trougla i kvadrata i svaki deo
     se nezavisno iscrtava. */

  /* Prvo se iscrtava krov, odnosno trougao. */
  for (i = 0; i < n - 1; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
      printf(" ");
    for (j = 0; j < 2 * i + 1; j++)
      if (j == 0 || j == 2 * i)
        printf("*");
      else
        printf(" ");
    printf("\n");
  }

  /* Potom se iscrtava kvadat. Da bi iscrtavanje bilo lakse
     istovremeno se ispisuju dva karaktera. */
  for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++)
      /* Provera da li je ivica. */
      if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
        printf("* ");
      else
        printf("  ");
    printf("\n");
  }

  return 0;
}
```

Rešenje 2.3.61

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= (n + 1) / 2; i++) {
    for (j = i; j <= n + 1 - i; j++)
      printf("%d ", j);
  }

  return 0;
```

```
}
```

Rešenje 2.3.62

```
#include <stdio.h>

int main()
{
  unsigned int n, i, j;

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

  for (i = 1; i <= n; i++) {
    for (j = 1; j <= n; j++)
      if (j % i == 1 || i == 1)
        printf("%d ", j);

    printf("\n");
  }

  return 0;
}
```

2.5 Funkcije

Zadatak 2.5.1 Napisati funkciju `int kvadrat(int x)` koja računa kvadrat datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 15
 Kvadrat broja 15 je 225
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: -89
 kvadrat broja -89 je 7921
```

Zadatak 2.5.2 Napisati funkciju `int kub(int x)` koja računa kub datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 15
 Kub broja 15 je 3375
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: -89
 Kub broja -89 je -704969
```

Zadatak 2.5.3 Napisati funkciju `unsigned int apsolutna_vrednost(int x)` koja izračunava apsolutnu vrednost broja x. Napisati program koji učitava jedan ceo broj i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj: -34
 Apsolutna vrednost: 34
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 5
 Apsolutna vrednost: 5
```

Zadatak 2.5.4 Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji učitava tri cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 19 8 14
 Minimum je: 8
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: -6 11 -12
 Minimum je: -12
```

Zadatak 2.5.5 Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja x. Napisati program koji učitava jedan realan broj i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 8.235
 Razlomljeni deo: 0.235000
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: -5.11
 Razlomljeni deo: 0.110000
```

Zadatak 2.5.6 Napisati funkciju `float stepen(float x, int n)` koja računa vrednost n-tog stepena realnog broja x. Napisati program koji učitava relan broj x i ceo broj n i ispisuje rezultat rada funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite jedan realan i jedan ceo broj:
 4.5 3
 4.5000003=91.125000
```

Primer 2

```
Interakcija sa programom:
 Unesite jedan realan i jedan ceo broj:
 -33.2 5
 -33.2000015=-40335776.000000
```

Zadatak 2.5.7 Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva broja x i n utvrđuje da li je x neki stepen broja n. Ukoliko jeste, funkcija vraća izložilac stepena, a u suprotnom vraća -1. Napisati program koji učitava dva cela pozitivna broja i ispisuje rezultat poziva funkcije. Napomena:

Pretpostaviti da je unos korektan.

Primer 1

```
Interakcija sa programom:
 Unesite dva broja: 81 3
 Jeste: 81 = 34
```

Primer 2

```
Interakcija sa programom:
 Unesite dva broja: 162 11
 Broj 162 nije stepen broja 11.
```

Zadatak 2.5.8 Napisati funkciju `int faktorijel(int n)` koja računa faktorijel broja n. Napisati i program koji učitava dva cela broja x i y iz intervala $[0, 12]$ i ispisuje vrednost zbira $x! + y!$.

Primer 1

```
Interakcija sa programom:
 Unesite dva broja: 4 5
 144
```

Primer 2

```
Interakcija sa programom:
 Unesite dva broja: 18 -5
 Greska: pogresan unos!
```

Primer 3

```
Interakcija sa programom:
 Unesite dva broja: 6 0
 721
```

Zadatak 2.5.9 Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najveći zajednički delilac primenom Euklidovog algoritma. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite dva cela broja: 1024 832
 Najveci zajednicki delilac je 64
```

Primer 2

```
Interakcija sa programom:
 Unesite dva cela broja: -900 112
 Najveci zajednicki delilac je -4
```

Zadatak 2.5.10 Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato n vraća zbir recipročnih vrednosti brojeva od 1 do n. Napisati program koji učitava ceo broj i ispisuje rezultat rada funkcje zaokružen na dve decimale.

Primer 1

```
Interakcija sa programom:
 Unesi jedan pozitivan ceo broj: 10
 Zbir reciprocnih je 2.93
```

Primer 2

```
Interakcija sa programom:
 Unesi jedan pozitivan ceo broj: 100
 Zbir reciprocnih je 5.19
```

Zadatak 2.5.11 Napisati funkciju `void ispis(float x, float y, unsigned n)` koja za dva realna broja x i y i jedan pozitivan ceo broj n ispisuje

vrednosti sinusne funkcije u n ravnomerno raspoređenih tačaka intervala $[x, y]$. Napisati program koji učitava odgovarajuće vrednosti i testira rad ove funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite dva realna broja: 7 32
 Unesite jedan prirodan broj: 10
 0.6570 -0.3457 -0.0108 0.3659 -0.6731
 0.8922 -0.9945 0.9666 -0.8122
```

Primer 2

```
Interakcija sa programom:
 Unesite dva realna broja: 20.5 -8.32
 Unesite jedan prirodan broj: 5
 -0.8934 -0.8979 -0.1920 0.6658 0.9968
```

Primer 3

```
Interakcija sa programom:
 Unesite dva realna broja: 8 8
 Nekorektan unos.
```

Primer 1

```
Interakcija sa programom:
 Unesite dva realna broja: 7 32
 Unesite jedan prirodan broj: -10
 Nekorektan unos.
```

Zadatak 2.5.12 Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji učitava ceo broj i ispisuje rezultat na tri decimale.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 461
 3.667
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 1001
 0.500
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -84723
 4.800
```

Zadatak 2.5.13 Napisati funkciju `int sadrzi(int x, int c)` koja ispituje da li se cifra c nalazi u zapisu celog broja x. Funkcija treba da vrati 1 ako se cifra nalazi u broju, a 0 inače. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj i cifru: 17890 7
 Cifra se nalazi u broju.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj i cifru: 1982 6
 Cifra se ne nalazi u broju.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj i cifru: 17890 26
 Neispravan unos.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj i cifru: -1982 9
 Cifra se nalazi u broju.
```

Zadatak 2.5.14 Napisati funkciju `int broj_neparnih_cifara(int x)` koja određuje broj neparnih cifre u zapisu datog celog broja. Testirati rad ove funkcije u programu koji učitava cele brojeve dok se ne unese nula i ispisuje broj neparnih cifara svakog unetog broja.

Primer 1

```
Interakcija sa programom:
 Unesite cele brojeve:
 2341
 Broj neparnih cifara je 2
 78
 Broj neparnih cifara je 1
 800
 Broj neparnih cifara je 0
 -99761
 Broj neparnih cifara je 4
 0
```

Primer 2

```
Interakcija sa programom:
 Unesite cele brojeve:
 987611
 Broj neparnih cifara je 4
 135
 Broj neparnih cifara je 3
 -701
 Broj neparnih cifara je 2
 602
 Broj neparnih cifara je 0
 -884
 Broj neparnih cifara je 0
 79901
 Broj neparnih cifara je 4
 0
```

Zadatak 2.5.15 Napisati program za ispitivanje svojstava cifara datog celog broja.

(a) Napisati funkciju `sve_parne_cifre` koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.

(b) Napisati funkciju `sve_cifre_jednake` koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

Napisati program koji učitava ceo broj i ispisuje da li su sve cifre parne i da li su sve cifre jednake.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 86422
 Sve cifre broja su parne.
 Cifre broja nisu jednake.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 55555
 Sve cifre broja nisu parne.
 Cifre broja su jednake.
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -88
 Sve cifre broja su parne.
 Cifre broja su jednake.
```

Primer 4

```
Interakcija sa programom:
 Unesite broj i cifru: -342
 Sve cifre broja nisu parne.
 Cifre broja nisu jednake.
```

Zadatak 2.5.16 Napisati funkciju `int zapis(int x, int y)` koja proverava da li se brojevi x i y zapisuju pomoću istih cifara. Funkcija treba da vrati vrednost 1 ako je uslov ispunjen, a 0 ako nije. Napisati i program koji učitava dva cela broja i ispisuje rezultat primene funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite dva broja: 251 125
 Uslov je ispunjen!
```

Primer 2

```
Interakcija sa programom:
 Unesite dva broja: 8898 9988
 Uslov nije ispunjen!
```

Primer 3

```
Interakcija sa programom:
 Unesite dva broja: -7391 1397
 Uslov je ispunjen!
```

Zadatak 2.5.17 Napisati funkciju `int rastuce(int n)` koja ispituje da li su cifre datog celog broja u rastućem poretku. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i ispisuje poruku da li su cifre unetog broja u rastućem poretku.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 2689
 Cifre su u rastucem poretku!
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 559
 Cifre su u rastucem poretku!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 628
 Cifre nisu u rastucem poretku!
```

Zadatak 2.5.18 Napisati funkciju `int par_nepar(int n)` koja ispituje da li su cifre datog celog broja naizmenično parne i neparne. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i testira rad funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 2749
 Broj ispunjava uslov!
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: -963
 Broj ispunjava uslov!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 27449
 Broj ne ispunjava uslov!
```

Zadatak 2.5.19 Napisati funkciju `int ukloni_stotine(int n)` koja modifikuje zadati broj tako što iz njegovog zapisa uklanja cifru stotina (ako postoji). Napisati program koji za brojeve koji se unose sve do pojave broja 0 ispisuje odgovarajuće brojeve kojima je uklonjena cifra stotine.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 1210
 110
 Unesite broj: 18
 18
 Unesite broj: 3856
 356
 Unesite broj: 0
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: -9632
 -932
 Unesite broj: 246
 46
 Unesite broj: -52
 -52
 Unesite broj: 0
```

Zadatak 2.5.20 Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sve do pojave broja 0 ispisuje odgovarajuće rotirane brojeve.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 146
 461
 Unesite broj: 18
 81
 Unesite broj: 3856
 8563
 Unesite broj: 7
 7
 Unesite broj: 0
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 89
 98
 Unesite broj: -369
 -693
 Unesite broj: -55281
 -52815
 Unesite broj: 0
```

Zadatak 2.5.21 Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja n. Napisati program koji učitava ceo broj k i ispisuje zbir delilaca svakog broja od 1 do k.

Primer 1

```
Interakcija sa programom:
 Unesite broj k: 6
 1 3 4 7 6 12
```

Primer 2

```
Interakcija sa programom:
 Unesite broj k: -2
 Greska: pogresan unos!
```

Zadatak 2.5.22 Broj je prost ako je deljiv samo sa 1 i sa samim sobom. Napisati funkciju `int prost (int x)` koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati 1 ako je broj prost i 0 u suprotnom. Napisati program koji za uneti ceo broj n ispisuje prvih n prostih brojeva.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 17
 Broj je prost!
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 24
 Broj nije prost!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: -11
 Broj nije prost!
```

Zadatak 2.5.23 Napisati funkciju `void prosti_brojevi(int m)` koja ispisuje sve proste brojeve manje od broja m. Napisati program koji učitava ceo broj veći od 1 i ispisati rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite broj: 15
 2 3 5 7 11 13
```

Primer 2

```
Interakcija sa programom:
 Unesite broj: 9
 2 3 5 7
```

Primer 3

```
Interakcija sa programom:
 Unesite broj: 1
 Greska: pogresan unos!
```

Zadatak 2.5.24 Napisati funkciju `double e_na_x(double x, double eps)` koja računa vrednost e^x kao parcijalnu sumu reda $\sum_{n=0}^{\infty} \frac{x^n}{n!}$, pri čemu se sumiranje vrši dok je razlika sabiraka u redu po apsolutnoj vrednosti manja od ε. Napisati program koji učitava dva realna broja x i *eps* i ispisuje izračunatu vrednost e^x.

Primer 1

```
Interakcija sa programom:
 Unesite broj x: 5
 Unesite eps: 0.001
 Rezultat: 148.412951
```

Primer 2

```
Interakcija sa programom:
 Unesite broj x: -3
 Unesite eps: 0.0001
 Rezultat: 0.049796
```

Zadatak 2.5.25 Za dati broj može se formirati niz tako da je svaki sledeći član niza dobijen kao suma cifara prethodnog člana niza. Broj je *srećan* ako se dati niz završava jedinicom. Napisati funkciju `int srecan(int x)` koja vraća 1 ako je broj srećan, a 0 u suprotnom. Napisati program koji za uneti prirodan broj n ispisuje sve srećne brojeve od 1 do n. NAPOMENA: *Pretpostaviti da je unos korektan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 100
 Srecni brojevi:
 1 10 19 28 37 46 55 64 73 82 91 100
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 0
 Nema srecnih brojeva.
```

Zadatak 2.5.26 Broj a je Armstrongov ako je jednak sumi n-tih stepena svojih cifara, pri čemu je n broj cifara broja a. Napisati funkciju `int armstrong(int x)` koja vraća 1 ako je broj Armstrongov, odnosno 0 ako nije. Napisati program koji za učitani ceo broj proverava da li je Armstrongov.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 153
 Broj je Armstrongov!
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 1634
 Broj je Armstrongov!
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj: 118
 Broj nije Armstrongov!
```

Zadatak 2.5.27 Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj x pojavljuje u nizu brojeva koji se unose sve do pojave broja 0. Napisati program koji učitava vrednost broja x i testira rad napisane funkcije.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj x: 2.84
 Unesite brojeve: 8.13 2.84 5 21.6 2.84 11.5 0
 Broj pojavljivanja broja 2.84 je: 2
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj x: -1.17
 Unesite brojeve: -128.35 8.965 8.968 89.36 0
 Broj pojavljivanja broja -1.17 je: 0
```

Zadatak 2.5.28 Napisati funkciju `long unsigned fibonaci(int n)` koja računa n-ti element Fibonačijevog niza. Napisati i program koji učitava ceo broj n ($0 \leq n \leq 50$) i ispisuje traženi Fibonačijev broj.

Primer 1

```
Interakcija sa programom:
Unesite broj n: 7
21
```

Primer 2

```
Interakcija sa programom:
Unesite broj n: 65
Greska: nedozvoljena vrednost!
```

Zadatak 2.5.29 Napisati funkciju `int konverzija (int c)` koja prebacuje veliko slovo u ekvivalentno malo i obrnuto. Napisati program koji testira ovu funkciju na karakterima koji se unose do pojave znaka `EOF`.

Primer 1

```
Interakcija sa programom:
Unesite karaktere: ZDRAVO
zdravo
```

Primer 2

```
Interakcija sa programom:
Unesite karaktere: Dobro jutro, R2D2!
dOBRO JUTRO, r2d2!
```

Zadatak 2.5.30 Napisati funkciju `char sifra(char c, int k)` koja za dati karakter c određuje šifru na sledeći način: ukoliko je c slovo, šifra je karakter koji se nalazi k pozicija 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 karakter po karakter do kraja ulaza i ispisuje šifrovani tekst.

Primer 1

```
Interakcija sa programom:
Unesite broj k: 2
Unesite tekst (CTRL+D za prekid):
c
a
8
8
+
+
Z
X
```

Zadatak 2.5.31 Napisati funkciju `int prestupna(int godina)` koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati 1 ako je godina prestupna ili 0 ako nije. Napisati program koji učitava dva cela broja $g1$ i $g2$ i ispisuje sve godine iz intervala $[g1, g2]$ koje su prestupne.

Primer 1

```
Interakcija sa programom:
Unesite dve godine: 2001 2010
Prestupne godine su: 2004 2008
```

Primer 2

```
Interakcija sa programom:
Unesite dve godine: 2005 2015
Prestupne godine su: 2008 2012
```

Primer 3

```
Interakcija sa programom:
Unesite dve godine: 2010 2001
Greska: pogresan unos!
```

Primer 4

```
Interakcija sa programom:
Unesite dve godine: 2001 2002
Nema prestupnih godina u ovom intervalu!
```

Zadatak 2.5.32 Napisati funkciju `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu. Napisati program koji učitava dva cela broja (mesec i godinu) i ispisuje broj dana u datom mesecu. U slučaju nekorektnog unosa ispisati odgovarajuću poruku o grešci. Napomena: *Pogledati rešenje zadataka 2.5.34*.

Primer 1

```
Interakcija sa programom:
Unesite mesec i godinu: 8
1998
Broj dana je: 31
```

Primer 2

```
Interakcija sa programom:
Unesite mesec i godinu: 2
2004
Broj dana je: 29
```

Primer 3

```
Interakcija sa programom:
Unesite mesec i godinu: 24
2004
Neispravan unos.
```

Zadatak 2.5.33 Napisati funkciju `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan. Napisati program koji učitava tri cela broja (dan, mesec, godinu) i ispisuje da li je datum ispravan ili ne. Napomena: *Pogledati rešenje zadataka 2.5.34*.

Primer 1

```
Interakcija sa programom:
Unesite datum: 24.8.1998.
Datum je ispravan.
```

Primer 2

```
Interakcija sa programom:
Unesite datum: 31.4.1789.
Datum nije ispravan.
```

Primer 3

```
Interakcija sa programom:
Unesite datum: 29.2.2004.
Datum je ispravan.
```

Primer 4

```
Interakcija sa programom:
Unesite datum: 29.14.2004.
Datum nije ispravan.
```

Zadatak 2.5.34 Napisati funkciju `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum ispisuje datum sledećeg dana. Napisati program koji učitava tri cela broja i ispisuje datum sledećeg dana.

Primer 1

```
Interakcija sa programom:
Unesite datum: 24.8.1998.
Datum sledeceg dana je: 25.8.1998.
```

Primer 2

```
Interakcija sa programom:
Unesite datum: 31.12.1789.
Datum sledeceg dana je: 1.1.1790.
```

Primer 3

```
Interakcija sa programom:
Unesite datum: 28.2.2003.
Datum sledeceg dana je: 1.3.2004.
```

Primer 4

```
Interakcija sa programom:
Unesite datum: 31.4.2004.
Uneti datum nije ispravan.
```

Zadatak 2.5.35 Napisati funkciju `int od_nove_godine(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do datog datuma. Napisati program koji učitava tri cela broja i ispisuje koliko dana je proteklo od Nove godine. Napomena: *Pogledati rešenje zadataka 2.5.37.*

Primer 1

```
Interakcija sa programom:
Unesite datum: 24.8.1998.
Broj dana od Nove godine je: 235
```

Primer 2

```
Interakcija sa programom:
Unesite datum: 31.12.1680.
Broj dana od Nove godine je: 366
```

Primer 3

```
Interakcija sa programom:
Unesite datum: 28.2.2003.
Broj dana od Nove godine je: 58
```

Primer 4

```
Interakcija sa programom:
Unesite datum: 31.4.2004.
Datum nije ispravan.
```

Zadatak 2.5.36 Napisati funkciju `int do_kraja_godine(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati program koji učitava tri cela broja i ispisuje broj dana do krja godine. Napomena: *Pogledati rešenje zadataka 2.5.37.*

Primer 1

```
Interakcija sa programom:
Unesite datum: 24.8.1998.
Broj dana do Nove godine je: 129
```

Primer 2

```
Interakcija sa programom:
Unesite datum: 31.12.1680.
Broj dana od Nove godine je: 0
```

Primer 3

```
Interakcija sa programom:
Unesite datum: 28.2.2004.
Broj dana od Nove godine je: 307
```

Primer 4

```
Interakcija sa programom:
Unesite datum: 31.4.2004.
Datum nije ispravan.
```

Zadatak 2.5.37 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 šest celih brojeva, koji označavaju dva datuma, i na standarni izlaz ispisuje broj dana između ta dva datuma. U slučaju pogrešnog unosa ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite prvi datum: 12.3.2008.
 Unesite drugi datum: 5.12.2008.
 Broj dana izmedju dva datuma je: 268
```

Primer 2

```
Interakcija sa programom:
 Unesite prvi datum: 26.9.1986.
 Unesite drugi datum: 2.2.1701.
 Broj dana izmedju dva datuma je: 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.
 Uneti datum nije ispravan.
```

Zadatak 2.5.38 Napisati funkciju `void romb(int n)` koja iscrtava romb čija je stranica dužine n. Napisati program koji učitava ceo pozitivan broj i ispisuje rezultat poziva funkcije.

Primer 1

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

Primer 2

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

Primer 3

```
Interakcija sa programom:
 Unesite broj n: -5
 Greska: pogresna dimenzija!
```

Zadatak 2.5.39 Napisati funkciju `void grafikon_h(int a, int b, int c, int d)` koja iscrtava horizontalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i prikazuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite vrednosti: 4 1 7 5
 ****
 *
 *******
 *****
```

Primer 2

```
Interakcija sa programom:
 Unesite vrednosti: 8 -2 5 4
 Greska: pogresan unos!
```

Primer 3

```
Interakcija sa programom:
 Unesite vrednosti: 5 2 2 10
 *****
 **
 **
 **********
```

Zadatak 2.5.40 Napisati funkciju `void grafikon_v(int a, int b, int c, int d)` koja iscrtava vertikalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i ispisuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite vrednosti: 4 1 7 5
   *
   *
   **
 * **
 * **
 * **
 ****
```

Primer 2

```
Interakcija sa programom:
 Unesite vrednosti: 8 -2 5 4
 Greska: pogresan unos!
```

Primer 3

```
Interakcija sa programom:
 Unesite vrednosti: 5 2 2 4
 *
 *  *
 *  *
 ****
 ****
```

2.6 Rešenja

Rešenje 2.5.1

```
1 #include <stdio.h>
2
3 int kvadrat(int x)
4 {
5    /* Promenljive u listi argumenata funkcije, kao i one deklarisane
6     u samoj funkciji, lokalne su za tu funkciju sto znaci da se
7     promenljivama x i y ne moze pristupiti nigde izvan funkcije
8     kvadrat. */
```

```
    int y;
    y = x*x;
    return y;
}

int main()
{
    int a,kv, kb;
    printf("Unesi ceo broj:");
    scanf("%d",&a);

    /* Promenljiva kv dobija povratnu vrednost funkcije kvadrat. */
    kv = kvadrat(a);

    printf("Kvadrat broja %d je %d\n", a, kv);
    return 0;
}
```

Rešenje 2.5.2

```
#include <stdio.h>

int kub(int a)
{
    /* U listi argumenata funkcije moze, a ne mora, postojati
     promenljiva istog naziva kao promenljiva koja je deklarisana u
     main funkciji (u ovom slucaju promenljiva a). Ova promenljiva se
     razlikuje od promenljive a deklarisane u main funkciji i vidljiva
      je samo unutar funkcije kub. */
    return a*a*a;
}

int main()
{
    int a, kb;
    printf("Unesi ceo broj:");
    scanf("%d",&a);

    /* Promenljivoj kb se dodeljuje povratna vrednost funkcije kub. */
    kb = kub(a);

    printf("Kub broja %d je %d\n", a, kb);
    return 0;
}
```

Rešenje 2.5.3

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

```c

/* Funkcija koja racuna apsolutnu vrednost. */
unsigned int apsolutna_vrednost(int x){
  /* Kako funkcija vraca unsigned, a x je tipa int, potrebno je
     kastovati rezultat u tip unsigned. */
  return (unsigned)(x<0?-x:x);
}

int main(){
  int n;

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

  /* Ispis njegove apsolutne vrednosti. */
  printf("Apsolutna vrednost: %u\n", apsolutna_vrednost(n));

  return 0;
}
```

Rešenje 2.5.4

```c
#include <stdio.h>

/*  Funkcija koja racuna minimum tri cela broja. */
int min(int x, int y, int z){
  int min;

  /* Pretpostvka je da je minimum broj x. Potom se minimum poredi sa
     druga dva broja i ukoliko pretpostavka nije dobra, vrednost
     minimuma se menja. */
  min=x;

  if(min>y)
    min=y;

  if(min>z)
    min=z;

  return min;
}

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

  /* Unose se tri broja. */
  printf("Unesite brojeve: ");
  scanf("%d%d%d", &x, &y, &z);

  /* Poziv funkcije i ispis rezultata. */
```

```
27    printf("Minimum je: %d\n", min(x,y,z));

29    return 0;
}
```

Rešenje 2.5.5

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

/* Funkcija koja vraca razlomljeni deo prosledjenog broja. */
float razlomljeni_deo(float x){

  /* Funkcija fabs vraca apsolutnu vrednost realnog broja.
   * NAPOMENA: funkcija fabs se nalazi u zaglavlju math.h.
   * NAPOMENA2: funkcija abs se nalazi u zaglavlju stdlib.h, ali se
     koristi samo za cele brojeve!
  */
  x = fabs(x);

  /* Razlomljeni deo broja se dobija tako sto od samog broja se
     oduzme njegov ceo deo. */
  return x - (int)x;
}

int main(){
  float n;

  /* Unos broja. */
  printf("Unesite broj:");
  scanf("%f", &n);

  /* Ispis rezultata. */
  printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));

  return 0;
}
```

Rešenje 2.5.6

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

/* Funkcija za realan broj x i ceo broj n odredjuje vrednost stepena
    x^n. */
float stepen(float a, int b)
{
  float s=1;
  int i, abs_b = abs(b);
```

```
  for(i=0;i<abs_b;i++)
    s=s*a;

/* Ukoliko je izlozilac b negativan, prvo se izracuna a^|b|, a vraca
    se reciprocnu vrednost izracunatog stepena. */
  return b>0 ? s : 1/s;
}

int main()
{
  int n;
  float x;
  float s;

  printf("Unesi jedan realan i jedan ceo broj:");
  scanf("%f%d",&x,&n);

  s = stepen(x,n);

  printf("%f^%d=%f\n",x,n,s);

  return 0;
}
```

Rešenje 2.5.7

```
#include <stdio.h>

/* Funkcija za dva uneta neoznacena broja x i n utvrdjuje da li je x
    neki stepen
broja n. Ukoliko jeste, funkcija vraca izlozilac stepena, a u
    suprotnom vraca -1. */
int je_stepen(unsigned x, unsigned n)
{
   int i=1;
   int s=n;

   /* Racuna se stepen broja n sve dok je on manji od x. */
   while(s<x)
   {
      s=s*n;
      i++;
   }

   /* Ukoliko je izracunati stepen jednak broju x onda funkcija vraca
     izlozilac stepena. */
   if (s==x)
      return i;

   /* U suprotnom vraca -1. */
```

```
    return -1;
}

int main()
{
  unsigned x;
  unsigned n;
  int st;

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

  st = je_stepen(x,n);

  if (st!=-1)
    printf("%u=%u^%d\n",x,n,st);
  else
    printf("%u nije stepen broja %u\n",x,n);

  return 0;
}
```

Rešenje 2.5.8

```
#include <stdio.h>

/* Funkcija racuna faktorijel broja. */
int faktorijel(int x) {

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

int main() {

  int x, y;

  printf("Unesite dva broja: ");
  scanf("%d%d", &x, &y);

  /* Faktorijel je funkcija koja veoma brzo raste, tj. s povecanjem
     broja x, drasticno brzo uvecava se i vrednost x!. Tip podatka int
      ima ogranicenje u velicini broja koji moze da sadrzi. Za 13! i
     vece, int ne bi mogao da sacuva sve cifre potrebne za zapis tako
     velikog broja, te bi doslo do prekoracenja. Faktorijel nije
     definisan nad skupom negativnih celih brojeva. */
  if(x < 0 || y < 0 || x > 12 || y > 12) {
    printf("Greska: pogresan unos!\n");
```

```
  }
  else{
  printf("%d\n", faktorijel(x) + faktorijel(y));
  }
  return 0;
}
```

Rešenje 2.5.9

```
#include <stdio.h>
.
int euklid(int x, int y)
{
  int r;
  /* Euklidov algoritam: http://elib.mi.sanu.ac.rs/files/journals/nm
    /245/nm581202.pdf */
  /* Kada promenljiva y postane 0, algoritam se zaustavlja. */
  while(y)
  {
    r=x%y;
    x=y;
    y=r;
  }

  /* U promenljivoj x se nalazi nzd. */
  return x;
}

int main()
{
  int a,b;
  int nzd;

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

  nzd = euklid(a,b);

  printf("Najveci zajednicki delilac je %d\n", nzd);

  return 0;
}
```

Rešenje 2.5.10

```
#include <stdio.h>

float zbir_reciprocnih(int n)
{
```

```
  float z=0;
  int i;
  for(i=1;i<=n;i++)
    /* Reciprocna vrednost broja je jednaka deljenju broja 1 sa tim
    brojem. Ipak, zbog specificnosti jezika C, 1/broj bi bilo
    celobrojeno deljenje (jer je 1 ceo broj i broj je ceo broj), pa
    bi rezultat bio ceo broj, a ne realan. Da bi izbegli takvo
    ponasanje deljenik ce biti 1.0 umesto 1, cime se obezbedjuje da
    se deljenik posmatra kao realan broj i da vrednost kolicnika bude
     realan broj. */
    z+=1.0/i;
  return z;
}

int main()
{
  int n;
  printf("Unesi jedan pozitivan ceo broj:\n");
  scanf("%d", &n);
  /* Povratna vrednost funkcije zbir_reciprocnih je float. Funkcija
    moze biti pozvana u okviru poziva druge funkcije, pa tako se
    funkcija zbir_reciprocnih poziva u okviru funkcije printf i
    umesto specifikatora %.2f bice ispisana povratna vrednost
    funkcije zbir_reciprocnih zaokruzena na dve decimale. */
  printf("Zbir je %.2f\n", zbir_reciprocnih(n));

  return 0;
}
```

Rešenje 2.5.11

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

/* Funkcija nema povratnu vrednost, zbog toga je povratni tip void.
    */
void ispis(float x, float y, int n)
{
  float i;
  float korak=(y-x)/(n-1);

  for(i=x;i<=y;i+=korak)
    printf("%.4f ", sin(i));

  printf("\n");

}

int main()
{
  float a,b;
```

```
20    int n;
      float t;
22    printf("Unesi dva realna broja:");
      scanf("%f%f",&a,&b);
24    printf("Unesi jedan prirodan broj:");
      scanf("%u",&n);
26
      if (n<=1 || a==b)
28    {
          printf("Nekorektan unos\n");
30        return -1;
      }
32    /* Kada je desni kraj intervala manji od levog, zamene im se mesta.
         */
      if (b<a)
34    {
         t=a;
36       a=b;
         b=t;
38    }


40


42    ispis(a,b,n);


44


      return 0;
46  }
```

Rešenje 2.5.12

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    /* Funkcija racuna aritmeticku sredinu cifara datog celog broja. */
5   float aritmeticka_sredina(int x) {

7     /* Aritmeticka sredina broja 0 je 0. */
      if(x==0)
9       return 0;

11    /* Brojaci sa svojim pocetnim vrednostima. */
      int zbir_cifara = 0;
13    int broj_cifara = 0;

15    /* U slucaju da je broj negativan, ostaci pri deljenju sa 10 bi
         takodje bili negativni. Stoga je zgodnije posmatrati samo
         apsolutnu vrednost broja. */
      x = abs(x);
17
```

```
  /* Izdvaja se cifra po cifra s kraja zapisa broja x. Uslov while(x
     > 0) ekvivalentan je uslovu while(x). Broj x se deli sa 10 i
     cifre se izdvajaju sve dok x ne postane 0. */
  while(x) {

    zbir_cifara += x % 10;
    broj_cifara++;

    x /= 10;
  }

  /* Da bi rezultat bio realan broj potrebno je izvrsiti kastovanje
     barem jednog od parametara jer su oba parametra celi brojevi. */
  return (float) zbir_cifara / broj_cifara;
}

int main() {

  int x;

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

  printf("Arimeticka sredina cifara broja %d je %.3f\n", x,
    aritmeticka_sredina(x));

  /* Iako u funkciji aritmeticka_sredina broj x se menja,
     prosledjivanjem argumenata funkciji pravi se lokalna kopija
     promenljive x za tu funkciju, tako da iako je ona menjana unutar
     funkcije aritmeticka_sredina, promenljiva x iz funkcije main()
     ostaje nepromenjena. */

  return 0;
}
```

Rešenje 2.5.13

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

/* Funkcija ispituje da li se cifra c nalazi u zapisu celog broja x.
    */
int sadrzi(int x, int c)
{
   char cifra;
   /* Posmatra se pozitivna vrednost broja. */
   x=abs(x);

   while(x)
   {
     /* Odredjuje se poslednja cifra broja. */
```

```
      cifra = x%10;
      /* Proverava se da li je poslednja cifra jednaka trazenoj cifri
       i ako jeste prekida se izvrsavanje funkcije i vraca se 1, kao
      oznaka da je broj nadjen. */
      if (cifra==c)
         return 1;
      /* Inace, broj se umanjuje za poslednju cifru i prelazi se na
      sledecu iteraciju. */
      x/=10;
   }
   /* Ukoliko se cifra ne nalazi u broju uslov u petlji nikad nece
    biti ispunjen. */
   return 0;
}
int main()
{
  int x;
  int c;
  printf("Unesi jedan ceo broj i jednu cifru:");
  scanf("%d%d",&x,&c);
  if (sadrzi(x,c))
     printf("Cifra %d se nalazi u zapisu broja %d\n",c,x);
  else
     printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
  return 0;
}
```

Rešenje 2.5.14

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

/* Funkcija odredjuje broj neparnih cifara u zapisu datog celog broja
    . */
int broj_neparnih_cifara(int x)
{
   int broj_neaprnih_cifara=0;
   /* Pomocna promenljiva u koju se smesta izdvojena cifra. */
   char cifra;
   /* Posmatra se pozitivna vrednost broja. */
   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. */
```

```
        broj_neparnih_cifara+=(cifra%2);
        x/=10;
    }

    return broj_neparnih_cifara;
}

int main()
{
  int x;
  do
  {
     scanf("%d",&x);
     printf("%d\n", broj_ncifara(x));
  } while(x!=0);

  return 0;
}
```

Rešenje 2.5.15

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

int sve_parne_cifre(int x) /* funkcija vraca 1 ako su sve cifre broja
     parne i 0 u suprotnom*/
{
  char d;
  /* Za slucaj da je broj negativan posmatra se apsolutna vrednost.
    */
  x=abs(x);

  while (x>0)
  {
    /* Izdvaja se poslednja cifra broja. */
    d=x%10;

    /* Ako je izdvojena cifra neparna oznacava da nisu sve cifre
    broja parne i funkcija vraca 0. */
    if (d%2==1)
      return 0;

    /* Poslednja cifra broja se ukklanja celobrojnim deljenjem sa 10.
     */
    x/=10;
  }

  return 1;
}
```

```c
/* Funkcija vraca 1 ako su sve cifre broja jednake i 0 u suprotnom. */
int sve_cifre_jednake(int x)
{
  char d;
  char prva_cifra;
  /* Posmatra se apsolutna vrednost broja. */
  x=abs(x);
  /* Izdvaja se prva cifra broja. */
  prva_cifra = x%10;
  /* Uklanja se poslednja cifra broja tako sto se broj podeli sa 10. */
  x/=10;

  while(x)
  {
    /* Izdvaja se poslednja cifra u broju. */
     d = x%10;

     /* Ukoliko izdvojena cifra nije jednaka prvoj izdvojenoj cifri
    onda broj nema sve jednake cifre. */
     if (d!=prva_cifra)
        return 0;

     x/=10;
  }

  return 1;
}

main()
{
  int x;
  int d;

  printf("unesi ceo broj:");
  scanf("%d", &x);

  if (sve_parne_cifre(x))
    printf("Sve cifre broja su parne\n");
  else
    printf("Broj sadrzi bar jednu neparnu cifru\n");

  if (sve_cifre_jednake(x))
    printf("Sve cifre broja su jednake\n");
  else
    printf("Broj sadrzi razlicite cifre \n");

}
```

Rešenje 2.5.16

```c
#include <stdio.h>

/* U broju y (ukoliko postoji) izbacuje se cifra 'cifra'. Poredak
     cifara nije bitan, dobijena vrednost ima obrnut redosled  cifara.
      */
int izbaci_cifru(int y, int cifra) {
  /* Vrednost broja y nakon eliminisanja cifre. Poredak cifara nije
     bitan pa ce vrednost novo_y imati obrnut poredak u odnosu na broj
      y. Na pocetku se postavlja na 0 i generise se koriscenjem
     Heronovog obrasca. */
  int novo_y = 0;
  /* Indikator da li je cifra uklonjena iz broja y. */
  int indikator = 0;
  /* Pomocni parametar u kome se cuva trenutno izdvojena cifra u
     petlji. */
  int izdvojena_cifra;

  while(y) {

    izdvojena_cifra = y % 10;
    /* Tekuca cifra se ukljucuje prilikom formiranja novo_y u slucaju
     da se cifra razlikuje od one koja se eliminise ili ukoliko je
     jedna cifra vec elimisana. */
    if(izdvojena_cifra != cifra || indikator)

      /* Heronov obrazac. Menja poredak cifara, ali on u ovom slucaju
     i nije bitan. */
      novo_y = novo_y*10 + izdvojena_cifra;
    else

      /* U slucaju da je cifra vec eliminisana, ne treba je opet
     eliminisati. Zato se menja vrednost indikatora. */
      indikator = 1;

    y /= 10;
  }

  return novo_y;
}

/* Funkcija proverava da li su dva cela broja napisana pomocu istih
     cifara, kao i da li se te cifre pojavljuju isti broj puta. */
int zapis(int x, int y) {

  /* Cifra koja se izdvaja iz x, a onda eliminise iz y. */
  int cifra;

  /* Radi lakseg rada, posmatraju se pozitivne vrednosti datih
     brojeva. */
  x = abs(x);
  y = abs(y);
```

```
  /* Iz broja x izdvajaju se redom cifra po cifra s kraja, a zatim se
     svaka takva cifra trazi i u broju y. Ukoliko postoji, eliminise
    se prvi put kada se pojavi (dakle, samo jednom). Ukoliko su sve
    cifre iste (***redosled nije bitan***), na kraju ce i iz x i iz y
     biti sve cifre eliminisane, te ostaju nule u oba broja. */
  while(x) {

    cifra = x % 10;
    x /= 10;

    y = izbaci_cifru(y, cifra);
  }

  return (x == 0 && y == 0);
}

int main() {

  int x, y;
  printf("Unesite dva cela broja: ");
  scanf("%d%d", &x, &y);

  if(zapis(x, y))
    printf("Uslov je ispunjen!\n");
  else
    printf("Uslov nije ispunjen!\n");

  return 0;
}
```

Rešenje 2.5.17

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

/* Funkcija proverava da li se cifre u zapisu broja nalaze u rastucem
     poretku. */
int rastuce(int n) {

  int tekuca_cifra;
  int prethodna_cifra;

  n = abs(n);

  /* Izvan petlje se izdvaja poslednja cifra u zapisu broja da bi u
    petlji mogla da se poredi sa sledecom. */
  tekuca_cifra = n % 10;
  n /= 10;

  while(n) {
```

```
17
    /* Izdvaja se nova cifra i ona postaje tekuca, a prethodna cifra
    se pamti. */
19  prethodna_cifra = tekuca_cifra;
    tekuca_cifra = n % 10;
21
    /* Ukoliko dve uzastopne izdvojene cifre ne ispunjavaju rastuci
    poredak prekida se izvrsavanje funkcije sa odgovarajucom
    povratnom vrednoscu 0. */
23   if(prethodna_cifra < tekuca_cifra)
    return 0;
25
    /* Inace, nastavlja se izdvajanje cifara. */
27  n /= 10;
  }
29
  /* 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. */
31 return 1;
}
33
int main() {
35
  int x;
37 printf("Unesite broj: ");
  scanf("%d", &x);
39
  if(rastuce(x))
41  printf("Cifre su u rastucem poretku!\n");
  else
43  printf("Cifre nisu u rastucem poretku!\n");

45 return 0;
}
```

Rešenje 2.5.18

```
1  #include <stdio.h>
   #include <stdlib.h>
3
    *
5   * Interesantna situacija je ukoliko su dve uzastopne cifre
    * obe parne, odnosno obe neparne.
7   * Ovaj uslov svodimo na poredjenje njihovih ostataka pri deljenju sa
        2:
    * ukoliko su ostaci isti, cifre su iste parnosti,
9   * te ne treba dalje proverati da li je uslov zadovoljen,
    * vec odmah prekinuti sa izvrsavanjem funkcije.
11  *
```

```
 * Ukoliko dve uzastopne cifre ni u jednom slucaju nisu bile iste parnosti,
 * a izdvojene su sve cifre iz broja x,
 * uslov je ispunjen, pa funkcija vraca 1.
 */

/* Funkcija proverava da li su dve uzastopne cifre razlicite parnosti. */
int par_nepar(int x) {
  /* Dve uzastopne cifre broja se pamte u promenljivama prethodna_cifra i tekuca_cifra. */
  int prethodna_cifra;
  int tekuca_cifra;

  /* U slucaju da je uneti broj negativan posmatra se samo apsolutna vrednost. */
  x = abs(x);

  /* Poslednja cifra broja se izdvaja van petlje da bi u petlji moglo da se vrsi poredjenje. */
  prethodna_cifra = x % 10;
  x /= 10;

  while(x) {
    /* Izdvaja se cifra broja. */
    tekuca_cifra = x % 10;

   /* Ukoliko su uzastopne cifre iste parnosti, uslov nije ispunjen, rad petlje i funkcije se prekida i vraca se 0. */
    if(tekuca_cifra % 2 == prethodna_cifra % 2)
      return 0;

    /* Tekuca cifra postaje prethodna cifra za narednu iteraciju. */
    prethodna_cifra = tekuca_cifra;
    x /= 10;
  }

  /* Sve uzastopne cifre su razlicite parnosti jer ni jednom u petlji uslov da su cifre iste parnosti nije bio ispunjen. */
  return 1;
}

int main() {

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

  if(par_nepar(x))
    printf("Broj ispunjava uslov!\n");
  else
    printf("Broj ne ispunjava uslov!\n");
```

```
57
  return 0;
59 }
```

Rešenje 2.5.19

```
#include <stdio.h>
2
/* Funkcija koja uklanja broj stotina iz broja n. */
4 int ukloni_stotine(int n){

6   /* Broj izmedju -100 i 100 nema cifru stotina, pa se vraca taj isti
       broj. */
    if(n>-100 && n<100)
8   return n;
    else
10  {
    /* U suprotnom se vraca broj sa uklonjenom cifrom stotina. */
12
    /* Odredjivanje znaka broja. */
14  int znak=(n<0)? -1 : 1;

16  /* Da bi se izbegle greske menja se apsolutna vrednost broja, a
      znak se naknadno dodaje pomocu mnozenja. */
    n=abs(n);
18
    return znak*((n/1000)*100 + n%100);
20  }
}
22
/* Funkcija koja vraca znak broja. */
24 int znak(int broj){
    return broj<0?-1:1;
26 }

28 int main(){

30  int broj;

32  while(1){

34  /* Unos broja. */
    printf("Unesite broj: ");
36  scanf("%d", &broj);

38  /* Broj 0 oznacava kraj unosa. */
    if(broj == 0)
40    break;

42  /* Poziv funkcije i ispis rezultata. */
    printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
```

```
44    }

46
      return 0;
48  }
```

Rešenje 2.5.20

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

4 int rotacija(int n){

6   int broj, br = 0, znak;

8   /* Ako je broj jednocifren, nema potrebe da se rotira. */
    if(n>-10 && n < 10)
10   return n;

12  /* Odredjivanje znaka broja. */
    znak=(n<0) ? -1: 1;
14
    /* Nadalje se radi sa apsolutnom vrednoscu broja. */
16  n=abs(n);

18  /* U promenljivoj broj se cuva kopija broja n. */
    broj=n;
20
    /* Izdvaja se cifra po cifra, sve do krajnje leve cifre broja (one
       koja treba da postane krajnje desna), npr za n = 1234, treba da
       se izdvoji prvi broj = 1 i ostatak = 234, a zatim na kraj
       ostataka treba dodati prvu cifru = 2341. */
22
    /* Nakon zavrsetka ove petlje, u parametru n se nalazi najlevlja
       cifra broja (koja treba da postane krajnje desna), dok se u
       parametru br nalazi broj cifara unetog broja. */
24  while(n >=10){
      n/=10;
26    br++;
    }
28
    /* Ostatak (234) se dobija kao n%(10^broj_cifara). Zatim se ostatak
       pomnozi sa 10, da bi se broj uvecao za jednu, poslednju cifru
      (2340). Na kraju, na dobijeni broj se doda (sabre) prvi broj koji
       se nalazi u parametru n. */
30
    return znak* ((broj%(int)pow(10, br))*10 + n);
32  }

34  int main(){
```

```
36    int n;
      while(1){
38
      /* Unos broja. */
40    printf("Unesite broj: ");
      scanf("%d", &n);
42
      /* Broj 0 oznacava kraj unosa. */
44    if(n == 0)
        break;
46
      /* Poziv funkcije i stampanje broja rotiranog za jedno mesto u levo
         . */
48    printf("%d\n", rotacija(n));
      }
50


52    return 0;
}
```

Rešenje 2.5.21

```
1  #include <stdio.h>

3  /* Funkcija koja racuna zbir delilaca broja x. */
   int zbir_delilaca(int x){
5    int i=0;

7    /* Na pocetku zbir se inicijalizuje na 0. */
     int zbir = 0;
9
     /* Svaki broj izmedju 1 i x koji deli broj x se dodaje u zbir. */
11   for(i=1; i<=x; i++){
     if(x % i == 0)
13     zbir += i;
     }
15
     /* Povratna vrednost funkcije je dobijeni zbir. */
17   return zbir;
   }
19
   int main(){
21
     int k, i;
23
     /* Ucitava se broj k. */
25   printf("Unesite broj k:");
     scanf("%d", &k);
27
     /* Provera korektnosti ulaza. */
29   if(k <= 0)
```

```
    printf("Greska: pogresan unos!\n");
  else{

    /*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 2.5.22

```
#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;

  /* Dovoljno je proveravati da li delioc datog broja postoji u
     intervalu od 3 do korena iz tog broja. */
  for (i=3; i<=sqrt(x);i+=2)
    /* Ako neki broj i deli broj x, onda on nije prost i izlazi se iz
     funkcije. */
    if (x%i==0)
      return 0;

  /* Ako ni jedan od prethodnih uslova nije bio ispunjen, to znaci da
     ni jedan broj ne deli x, pa je on prost. */
  return 1;
}

int main()
{
  int n;
  scanf("%d",&n);
  int i,j;
```

```
33    /* Tekuci broj za koji se ispituje da li je prost. */
      i=1;
35    /* Brojac prostih brojeva. */
      j=0;
37    while(j<n)
      {
39       /* Poziva se funkcija da se odredi da li je tekuci broj i prost.
         */
         if (prost(i))
41       {
            /* Ako jeste, stampa se i uvecava se brojac prostih brojeva.
        */
43          printf("%d\n", i);
            j++;
45       }
         /* U narednoj iteraciji ispituje se sledeci broj. */
47       i++;
      }
49
      return 0;
51 }
```

Rešenje 2.5.24

```
   #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 se sumiranje
      vrsi dok je razlika sabiraka u redu po apsolutnoj vrednosti manja
       od eps. */
   double e_na_x(double x, double eps)
6  {
     double s=1;
8    double clan=1;
     int n=1;
10
     /* Parcijalnu suma se formira tako sto se u svakoj iteraciji petlje
        promenljivoj s doda jedan sabirak sume oblika (x^n)/n! koji se
      cuva u promenljivoj clan.
12
        Svaki sabirak se dobija na osnovu prethodnog tako sto se
      prethodni pomnozi sa x i podeli sa n (n predstavlja redni broj
      sabirka u sumi).
14
        Prvi sabirak (kome odgovara n=0) iznosi 1; zbog toga promenljive
        s i clan se inicijalizuju na vrednost 1.
16
        Sumiranje se sprovodi sve dok je sabirak po apsolutnoj vrednosti
        veci od trazene tacnosti eps. */
18
```

```
  do
  {
     clan = (clan*x)/n;
     s += clan;
     n++;
  } while(fabs(clan)>eps);

  return s;
}

int main()
{
  double x,eps;
  printf("x=");
  scanf("%lf", &x);
  printf("eps=");
  scanf("%lf", &eps);

  printf("e^%f=%f\n", x, e_na_x(x,eps));
  return 0;
}
```

Rešenje 2.5.25

```
#include<stdio.h>

/* Funkcija koja vraca zbir cifara datog broja x. */
int zbir_cifara(int x)
{
   int s=0;
   char cifra;
   while(x)
   {
      cifra = x%10;
      s+=cifra;
      x/=10;
   }
   return s;
}

int srecan(int x)
{
   int suma_cifara;

   do
   {
     /* Za svaki tekuci clan niza odredjuje se suma njegovih cifara,
    a onda ta izracunata suma upravo i postaje novi tekuci clan niza.
     */
     suma_cifara = zbir_cifara(x);
     x = suma_cifara;
```

```
    } while(x>=10);
    /* Petlja se izvrsava sve dok clan niza ne postane jednocifren.
     Zbir cifara jednocifrenog broja je upravo sam taj broj. */

    return (x==1);

}

int main()
{
    unsigned n;
    int i;
    printf("Unesi jedan neoznacen broj:");
    scanf("%u",&n);

  /* U petlji se ispisuju svi srecni brojevi od 1 do n. Za svaki broj
     od 1 do n se poziva funkcija koja vraca vrednost 1 ako je broj
     srecan. */
    for(i=1;i<=n;i++)
       if (srecan(i))
          printf("%d je srecan\n", i);

  return 0;
}
```

Rešenje 2.5.26

```
#include <stdio.h>

/* Funkcija racuna broj x na n-ti stepen. */
int stepen(int x, int n) {

  int i;
  int st = 1;

  for(i = 1; i <= n; i++)
    st *= x;

  return st;
}

/* Funkcija odredjuje broj cifara broja. */
int broj_cifara(int x)
{
  int i=0;

  while(x)
  {
    i++;
    x /= 10;
  }
```

```
25
  return i;
27 }

29 /* Funkcija proverava da li je broj Armstrongov. */
int armstrong(int x) {
31
  /* U y se cuva zbir i-tih stepena cifara. */
33  int suma = 0;
  /* Stepen za koji se proverava. */
35  int i = broj_cifara(x);
  /* Prilikom izdvajanja cifara, broj x se menja i zato se koristi
     dodatna promenljiva u kojoj se cuva vrednost x pre promena. */
37  int original = x;

39  while(x) {
      /* Za svaku izdvojenu cifru racuna se stepen i dodaje na ukupnu
      sumu. */
41    suma += stepen(x % 10, i);
    x /= 10;
43  }

45   /* Nakon rada petlje x je postao 0 pa se zato poredjenje vrsi sa
     pomocnom promenljivom. Ukoliko je broj Armstrongov izracunata
     suma i originalan broj su jednaki. */
  return original == suma;
47 }

49 int main() {

51  int x;
  printf("Unesite broj: ");
53  scanf("%d", &x);

55  if(armstrong(x))
    printf("Broj je Armstrongov!\n");
57  else
    printf("Broj nije Armstrongov!\n");
59
  return 0;
61 }
```

Rešenje 2.5.27

```
#include <stdio.h>
2
/* Funkcija broji koliko puta se realan broj x javlja u nizu unetih
    brojeva. */
4 int prebrojavanje(float x) {

6  float y;
```

```
  /* Promenljiva br_pojavljivanja je brojac koji broji koliko puta se
     broj x javlja u unetom nizu brojeva. */
  int br_pojavljivanja = 0;

  printf("Unesite brojeve: ");
  do {

    /* Unos broja. */
    scanf("%f", &y);

    /* Poredjenje unetog broja sa datim brojem. Ukoliko je unet
    jednak trazenom broju, brojac se uvecava. */
    if(x == y)
      br_pojavljivanja++;

  } while(y);
  /* Brojevi se unose sve dok se ne unese 0. */

  return br_pojavljivanja;
}

int main() {

  float x;
  int br_pojavljivanja;

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

  br_pojavljivanja = prebrojavanje(x);
  printf("Broj pojavljivanja broja %.2f je: %d\n", x,
    br_pojavljivanja);

  return 0;
}
```

Rešenje 2.5.28

```
#include <stdio.h>

/* Funkcija racuna n-ti clan Fibonacijevog niza. */
long unsigned fibonaci(int n) {

  int i;

  /* Prva dva clana niza su jednaki 1. */
  long unsigned f0 = 1;
  long unsigned f1 = 1;
  /* Promenljive u kojima se cuvaju tekuci clanovi niza: n+2, n+1. i
    n-ti clan. */
  long int fn2, fn1, fn;
```

```
13
    /* Nulti i prvi clan Fibonacijevog niza su poznati pa ih ne treba
       racunati u petlji. */
15  if(n == 0 || n == 1)
      return 1;
17
    /* Tekuci clanovi niza se postavljaju  na pocetne vrednosti. */
19  fn = f0;
    fn1 = f1;
21  /* Elementi niza se racunaju od drugog clana do n-tog. */
    for(i = 2; i <= n; i++) {
23
      /* Naredni clan niza (element n+2-i) se dobija sabiranjem
      prethodna dva clana. */
25    fn2 = fn1 + fn;
      /* Menjaju se vrednosti tekucih clanova niza zbog naredne
      iteracije. */
27    fn =  fn1;
      fn1 = fn2;
29  }

31  return fn2;
}
33
int main() {
35
    int n;
37  printf("Unesite broj n: ");
    scanf("%d", &n);
39
    /* Provera vrednosti za broj n. */
41  if(n <0 || n > 50) {
      printf("Greska: nedozvoljena vrednost!\n");
43  }
    else{
45    printf("%lu\n", fibonaci(n));
    }
47
    return 0;
49  }
```

Rešenje 2.5.29

```
1  #include <stdio.h>

3  /* Funkcija konvertuje malo slovo u veliko ili veliko slovo u malo,
       ostali karakteri su nepromenjeni. U resenju zadatka se mogu
       koristiti i ugradjene funkcije islower, isupper, tolower i
       toupper (procitati vise o njima na man stranama). Funkcije
       pripadaju zaglavlju ctype.h. Za vezbu napisati resenje
       koriscenjem ugradjenih funkcija. */
```

```c
int konverzija(int c)
{
  if (c>='A' && c<='Z')
    return c+'a'-'A';

  if (c>='a' && c<='z')
    return c-'a'+'A';

  return c;
}

int main()
{
  int c;

  /* Korisnik unosi karakter po karakter do konstante EOF. */
  while((c=getchar())!=EOF)
    /* Poziva se funkcija koja menja karakter i novodobijeni karakter
     se ispisuje sa funkcijom putchar. */
    putchar(konverzija(c));

  return 0;
}
```

Rešenje 2.5.30

```c
#include <stdio.h>

/* Funkcija vraca karakter koji se u abecedi nalazi k mesta pre datog
     karaktera c. */
char sifra(char c, int k) {

  /* Provera da li je karakter malo slovo. */
  if(c >= 'a' && c <= 'z')
    /* Ako karakter koji je k pozicija pre datog karaktera ispada iz
    opsega malih slova. */
    if(c-k < 'a')
      /* Od k se oduzima rastojanje izmedju c i 'a' (jer je za toliko
     karaktera vec vraceno u nazad), kako bi se odredilo koliko
    preostali broj karaktera koji treba preskociti od karaktera 'z'.
       */
      return 'z' - (k - (c -  'a') - 1);
    else
      /* U suprotnom, karakter c-k ne ispada iz opsega malih slova,
    te je dovoljno njega vratiti. */
      return c-k;

  /* Provera da li je karakter veliko slovo. */
  else if(c >= 'A' && c <= 'Z')
    if(c-k < 'A')
      return 'Z' - (k - (c -  'A') - 1);
```

```
21      else
          return c-k;
23
25    return c;
}
27
int main() {
29
  int k;
31  char c;

33  printf("Unesite broj k: ");
  scanf("%d", &k);
35
  printf("Unesite tekst (CTRL + D za prekid): ");
37  while((c = getchar()) != EOF)
      putchar(sifra(c, k));
39
  return 0;
41 }
```

Rešenje 2.5.31

```
1  #include<stdio.h>

3  /* Funkcija koja proverava da li je godina prestupna. */
  int prestupna(int godina){
5    if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
    return 1;
7    else
    return 0;
9  }

11 /* Funkcija koja proverava da li postoji prestupna godina u datom
       intervalu. */
  int postoji_prestupna(int g1, int g2){
13   for(; g1<=g2; g1++){
    if(prestupna(g1))
15      return 1;
    }
17   return 0;
  }
19
  int main(){
21
    int g1, g2;
23
    /* Unos dve godine. */
25  printf("Unesite dve godine: ");
    scanf("%d%d", &g1, &g2);
```

```
27
   /* Proverava se korektnost ulaza. */
29 if(g1 < 0 || g2 < 0 || g1>g2){
   printf("Greska: pogresan unos!\n");
31 }
   else{
33
   /* Proverava se da li postoji prestupna godina u datom intervalu.
     */
35 if(postoji_prestupna(g1,g2)){
     /* Ako postoje, prestupne godine se ispisuju. */
37   printf("Prestupne godine su: ");
     for(; g1<=g2; g1++){
39   if(prestupna(g1))
       printf("%d ", g1);
41   }
     printf("\n");
43 }else{
     /* A ako ne postoje, stampa se odgovarajuca poruka. */
45   printf("Nema prestupnih godina u ovom intervalu!\n");
   }
47 }
   return 0;
49 }
```

Rešenje 2.5.34

```
1  #include<stdio.h>

3  /* Funkcija koja proverava da li je godina prestupna. */
   int prestupna(int godina){
5    if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
     return 1;
7    else
     return 0;
9  }

11 /* Funkcija odredjuje broj dana u datom mesecu. */
   int broj_dana(int mesec, int godina)
13 {
     switch(mesec)
15   {
       case 1: case 3: case 5: case 7: case 8: case 10: case 12: return
       31;
17     case 4: case 6: case 9: case 11: return 30;
       case 2: if (prestupna(godina))
19               return 29;
             else
21               return 28;
     }
23 }
```

```
25
/* Funkcija proverava da li je datum ispravan. Ako je datum  ispravan
     funkcija vraca 1, inace vraca 0. */
27 int ispravan(int dan, int mesec, int godina)
{
29   /* Ako je godina negativna, datum nije ispravan. */
  if (godina < 0)
31    return 0;

33   /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */
  if (mesec < 1 || mesec > 12)
35    return 0;

37  /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,
     datum nije ispravan. */
  if (dan < 1 || dan > broj_dana(mesec, godina))
39    return 0;

41  return 1;
}
43

45 void sledeci_dan(int dan, int mesec, int godina)
{
47  /* Za kraj godine, odnosno za datum 31.12. sledeci datum je 1.1. i
     godina se uvecava za jedan. */
  if (mesec == 12 && dan == 31)
49    printf("1.1.%d.\n", godina+1);
  /* Ukoliko je dan jednak poslednjem danu u tom mesecu, odnosno ako
     je jednak broju dana u tom mesecu, onda je sledeci datum kada se
     mesec uveca za 1, a dan postane 1. Bitan je redosled ovih naredbi
     . Ako bi ovo ispitivanje bilo prvo, onda bi se mesec mogao
     uvecati na 13. sto ne bi bio ispravan datu. Zato se prvo
     proverava da li je kraj godine, pa tek onda da li je kraj meseca.
      */
51  else if (dan == broj_dana(mesec, godina))
    printf("1.%d.%d.\n", mesec+1, godina);
53   /* Ako nije ni jedan od prethodna dva slucaja, onda se dan moze
     uvecati na 1, bez bojazni da ce se prekoraciti broj dana u datom
     mesecu. */
  else
55    printf("%d.%d.%d.\n", dan+1, mesec, godina);
}
57

59 int main()
{
  int dan, mesec, godina;
61

   /* Unos podataka. */
63  printf("Unesite datum:");
  scanf("%d.%d.%d.", &dan, &mesec, &godina);
```

```
65
   /* Provera ispravnosti. Ako datum nije ispravan prekida se rad
      programa. */
67   if (!ispravan(dan, mesec, godina))
   {
69     printf("Uneti datum nije ispravan.\n");
     return -1;
71   }

73    /* Poziva se funkcija za ispis sledeceg dana. */
   printf("Datum sledeceg dana je:");
75   sledeci_dan(dan, mesec, godina);

77   return 0;
}
```

Rešenje 2.5.37

```
   #include<stdio.h>
2
   /* Funkcija koja proverava da li je godina prestupna. */
4  int prestupna(int godina){
     if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
6    return 1;
     else
8    return 0;
}
10
   /* Funkcija odredjuje broj dana u datom mesecu. */
12 int broj_dana(int mesec, int godina)
{
14   switch(mesec)
   {
16     case 1: case 3: case 5: case 7: case 8: case 10: case 12: return
       31;
       case 4: case 6: case 9: case 11: return 30;
18     case 2: if (prestupna(godina))
               return 29;
20           else
               return 28;
22   }
}
24

26 /* Funkcija proverava da li je datum ispravan. Ako je datum  ispravan
        funkcija vraca 1, inace vraca 0. */
   int ispravan(int dan, int mesec, int godina)
28 {
     /* Ako je godina negativna, datum nije ispravan. */
30   if (godina < 0)
       return 0;
```

```
32
    /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */
34  if (mesec < 1 || mesec > 12)
      return 0;
36
    /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,
       datum nije ispravan. */
38  if (dan < 1 || dan > broj_dana(mesec, godina))
      return 0;
40
    return 1;
42 }

44 /* Funkcija odredjuje koliko dana je proteklo od pocetka godine. */
   int od_nove_godine(int dan, int mesec, int godina)
46 {
    int suma_dana = 0, i;
48
    /* Za sve mesece pre datog datuma dodaje se broj dana za dati mesec
       . */
50  for(i=1; i<mesec; i++)
      suma_dana += broj_dana(mesec, godina);
52
    /* Na kraju se dodaje koliko je dana proteklo u datom mesecu, a to
       je zadato sa promenljivom dan. */
54  return suma_dana+dan;
   }
56
   /* Funkcija odredjuje koliko dana ima do kraja godine. */
58 int do_kraja_godine(int dan, int mesec, int godina)
   {
60  int suma_dana = 0, i;

62  /* Za sve mesece posle datog datuma dodaje se broj dana za dati
       mesec. */
    for(i=mesec+1; i<=12; i++)
64    suma_dana += broj_dana(mesec, godina);

66  /* Na kraju se dodaje koliko je dana je ostalo u datom mesecu. */
    return suma_dana + broj_dana(mesec, godina) - dan;
68 }

70 /* 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)
72 {
    if (godina1 < godina2)
74    return 1;
    else if (godina1 > godina2)
76    return 0;
    else if (mesec1 < mesec2)
```

```
78      return 1;
    else if (mesec1 > mesec2)
80      return 0;
    else if (dan1 < dan2)
82      return 1;
    else return 0;
84  }

86  /* Funkcija vraca broj dana u datoj godini. */
    int broj_dana_u_godini(int godina)
88  {
    if (prestupna(godina))
90      return 366;
    else
92      return 365;
    }
94
    int broj_dana_izmedju(int dan1, int mesec1, int godina1, int dan2,
        int mesec2, int godina2)
96  {
    int pom, i;
98  int suma_dana = 0;

100   /* Provera koji od datuma je ranije. Da bi osigurali da manji datum
         bude onaj uz koji ide broj 1, datumi se razmenjuju. Obratiti
      paznju da se u okviru ove if naredbe vise puta javlja jedan te
      isti deo koda i da bi bilo lepo to zapisati u okviru funkcije.
      Ipak, u ovom trenutku to nije moguce jer je potrebno koristiti
      pokazivace o kojima ce biti reci kasnije. Razmisliti kako bi se
      ovo resenje moglo popraviti kada se budu presli pokazivaci. */
    if (!prethodi(dan1, mesec1, godina1, dan2, mesec2, godina2))
102 {
      pom = dan1;
104   dan1 = dan2;
      dan2 = pom;
106
      pom = mesec1;
108   mesec1 = mesec2;
      mesec2 = pom;
110
      pom = godina1;
112   godina1 = godina2;
      godina2 = pom;
114 }

116 /* Ako su godine razlicite. */
     if (godina1 != godina2)
118 {
      /* Za manji datum dodaje se broj dana do kraja godine. */
120   suma_dana = do_kraja_godine(dan1, mesec1, godina1);

122   /* Za sve godine koje su izmedju dve date godine dodaje se broj
```

```
    dana u tim godinama. */
    for(i=godina1+1; i<godina2; i++)
      suma_dana += broj_dana_u_godini(i);

    /* Za veci datum dodaje se broj dana od pocetka godine. */
      suma_dana += od_nove_godine(dan2, mesec2, godina2);
  }
  /* Ako su godine iste, ali meseci razliciti. */
  else if (mesec1 != mesec2)
  {
    /* Dodaje se broj dana do kraja prvog meseca. */
    suma_dana = broj_dana(mesec1, godina1) - dan1;

    /* Dodaje se broj dana za svaki mesec koji je izmedju dva data
    meseca. Kako su godina1 i godina2 jednake svejedno je koja od ove
     dve promenljive se koristi u pozivu funkcije. */
    for(i= mesec1+1; i<mesec2; i++)
      suma_dana += broj_dana(i, godina1);

    /* Dodaje se broj dana od pocetka meseca. */
    suma_dana += dan2;
  }
  /* Ako su i godine i meseci jednaki. */
  else
    suma_dana = dan2 - dan1;

  return suma_dana;
}

int main()
{
  int dan1, mesec1, godina1, dan2, mesec2, godina2;

   /* Unos podataka. */
  printf("Unesite prvi datum:");
  scanf("%d.%d.%d.", &dan1, &mesec1, &godina1);

   printf("Unesite drugi datum:");
  scanf("%d.%d.%d.", &dan2, &mesec2, &godina2);

  /* Provera ispravnosti. Ako datum nije ispravan prekida se rad
    programa. */
  if (!ispravan(dan1, mesec1, godina1) || !ispravan(dan2, mesec2,
    godina2))
  {
    printf("Uneti datum nije ispravan.\n");
    return -1;
  }

   /* Poziva se funkcija za ispis sledeceg dana. */
  printf("Broj dana izmedju dva datuma je:%d\n", broj_dana_izmedju(
    dan1, mesec1, godina1, dan2, mesec2, godina2));
```

```
170   return 0;
}
```

Rešenje 2.5.38

```
#include<stdio.h>

/* Funkcija koja iscrtava romb. */
void romb(int n){
  int i, j;

  /* Petlja iscrtava liniju po liniju romba. */
  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(" ");

  /* A zatim se ispisuje n zvezdica. */
  for(j=0; j<n; j++)
    printf("*");

  /* Na kraju svake linije stoji oznaka za novi red. */
  printf("\n");
  }

}

int main(){
  int n;

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

  /* Proverava se korektnost ulaza i poziva funkcija za iscrtavanje
    romba. */
  if(n<=0)
  printf("Greska: pogresna dimenzija!\n");
  else
  romb(n);

  return 0;
}
```

Rešenje 2.5.39

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

```
2
/* Funkcija koja stampa n zvezdica za kojima sledi znak za novi red.
    */
4 void stampaj_zvezdice(int n){
  int i;
6   for(i=0; i<n; i++)
    printf("*");
8
  printf("\n");
10 }

12 /* Funkcija koja crta grafikon. */
void grafikon_h(int a, int b, int c, int d)
14 {
  int i;
16
  /* Prvo se ispisuje a zvezdica. */
18   stampaj_zvezdice(a);

20   /* Zatim u sledecem redu b zvezdica. */
  stampaj_zvezdice(b);
22
  /* Zatim u sledecem redu c zvezdica. */
24   stampaj_zvezdice(c);

26   /* Zatim u poslednjem redu d zvezdica. */
  stampaj_zvezdice(d);
28
}
30
int main(){
32   int a,b,c,d;

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

38   /* Proverava se korektnost ulaza i ispisuje se rezultat. */
  if(a <0 || b<0 || c<0 || d<0){
40   printf("Greska: pogresan unos!\n");
  }else{
42   grafikon_h(a,b,c,d);
  }
44
  return 0;
46 }
```

Rešenje 2.5.40

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

```
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;
}

void stampaj_znak(int polje, int granica) {
  if(polje<granica)
    printf(" ");
  else
    printf("*");
}

/* Funkcija koja iscrtava vertikalni grafikon. */
void grafikon_v(int a, int b, int c, int d){
  int i, max;

  /* Na pocetku je potrebno pronaci najvecu od ove cetiri vrednosti.
    */
  max=maksimum(a, b, c, d);

  /* Grafikon ukupno ima max horizontalnih linija. */
  for(i=0; i<max; i++){

    /* U svakoj od horizontalnih linija se nalazi po 4 polja: polje
    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 koja linija se
     trenutno ispisuje. */

  /* Stampa se znak za polje a. */
    stampaj_znak(i, max-a);

  /*  Stampa se znak za polje b. */
    stampaj_znak(i, max-b);

  /* Stampa se znak za polje c. */
    stampaj_znak(i, max-c);

  /* Stampa se znak za polje d. */
    stampaj_znak(i, max-d);

  /* Na kraju svake horizontalne linije stampa se novi red. */
  printf("\n");
  }
```

```
51 }

53 int main(){
     int a,b,c,d;
55
     /* Ucitavanje vrednosti cetiri broja. */
57   printf("Unesite vrednosti: ");
     scanf("%d%d%d%d", &a, &b, &c, &d);
59
     /* Proverava se korektnost ulaza i poziva se funkcija za ispis
        grafikona. */
61   if(a <0 || b<0 || c<0 || d<0)
     printf("Greska: pogresan unos!\n");
63   else
     grafikon_v(a,b,c,d);
65
     return 0;
67 }
```

3

Predstavljanje podataka

3.1 Nizovi

Zadatak 3.1.1 Ako su $a = (a_1, \ldots, a_n)$ i $b = (b_1, \ldots, b_n)$ vektori dimenzije n, njihov skalarni proizvod je $a \cdot b = a_1 \cdot b_1 + \ldots + a_n \cdot b_n$. Napisati program koji računa skalarni proizvod dva vektora. Vektori se zadaju kao celobrojni nizovi sa najviše 100 elemenata. Program učitava dimenziju i elemente nizova, a na izlaz ispisuje vrednost skalarnog proizvoda.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 5
 Unesite koordinate vektora a:
 8 -2 0 2 4
 Unesite koordinate vektora b:
 35 12 5 -6 -1
 Skalarni proizvod vektora a i b:
 240
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 3
 Unesite koordinate vektora a:
 -1 0 1
 Unesite koordinate vektora b:
 5 5 5
 Skalarni proizvod vektora a i b:
 0
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju vektora:
 120
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.2 Napisati program koji za učitani niz ispisuje:

(a) elemente niza koji se nalaze na parnim pozicijama.

(b) parne elemente niza.

Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 6
 Unesite elemente niza:
 1 8 2 -5 -13 75
 Elementi niza na parnim pozicijama:
 1 2 -13
 Parni elementi niza:
 8 2
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 3
 Unesite elemente niza:
 11 81 -63
 Elementi niza na parnim pozicijama:
 11 -63
 Parni elementi niza:
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza:
 -4
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.3 Takmičari na Beogradskom maratonu su označeni rednim brojevima počevši od 0, a vreme za koje su istrčali maraton je dato minutima. Ovi podaci zadati su nizom celih brojeva, pri čemu indeks niza označava redni broj takmičara, a vrednosti u nizu označavaju vreme trčanja. Napisati funkcije za rad sa ovim nizom.

(a) Napisati funkciju `void ucitaj(int a[], int n)` koja učitava elemente niza a dimenzije n.

(b) Napisati funkciju `void stampaj(int a[], int n)` koja štampa elemente niza a dimenzije n.

(c) Napisati funkciju `int suma(int a[], int n)` koja računa i vraća ukupno vreme trčanja svih takmičara.

(d) Napisati funkciju `float prosek(int a[], int n)` koja računa i vraća prosečno vreme (aritmetičku sredinu) trčanja takmičara.

(e) Napisati funkciju `int maksimum(int a[], int n)` koja izračunava i vraća najduže vreme trčanja takmičara.

(f) Napisati funkciju `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 testira rad zadatih funkcija. Maksimalan broj takmičara je 1000.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 5
 19 47 27 34 16
 Vreme trcanja takmicara: 19 47 27 34 16
 Ukupno vreme: 143
 Prosecno vreme trcanja: 28.6
 Maksimalno vreme trcanja: 47
 Indeks pobednika: 4
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 -5
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.4 Napisati funkciju `int min_max(int a[], int n)` koja pronalazi indekse najmanjeg i najvećeg elementa u nizu a dimenzije n koristeći samo jedan prolaz kroz niz. Funkcija kao povratnu vrednost vraća manji od ta dva indeksa. Napisati program koji testira ovu funkciju za učitane nizove celih brojeva maksimalne dužine 100 elemenata.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza:
 7
 Unesite elemente niza:
 5 8 -4 11 17 89 1
 2
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza:
 3
 Unesite elemente niza:
 9 11 6
 1
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza:
 -45
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.5 Napisati funkciju `int prebrojavanje(int a[], int n)` koja izračunava broj elemenata celobrojnog niza a dužine n koji su manji od poslednjeg elementa niza. Napisati i program koji testira rad funkcije. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 4
 Unesite elemente niza: 11 2 4 9
 2
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza: 7
 Unesite elemente niza: 7 2 1 14 65 2 8
 4
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza: 5
 Unesite elemente niza: 25 18 29 30 14
 0
```

Zadatak 3.1.6 Napisati funkciju `int prebrojavanje(int a[], int n)` koja izračunava broj parnih elemenata niza celih brojeva a dužine n koji prethode maksimalnom elementu niza. Napisati i program koji testira rad funkcije. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 4
 Unesite elemente niza: 11 2 4 9
 0
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza: 7
 Unesite elemente niza: 7 2 1 14 65 2 8
 2
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza: 5
 Unesite elemente niza: 25 18 29 30 14
 1
```

Zadatak 3.1.7 Elementi niza celih brojeva su podaci o nadmorskim visinama u nekom području sveta. Na kartografskoj mapi su indeksima označene različite tačke, a u nizu je svakom indeksu pridružen neki ceo broj (odnosno nadmorska visina). Napisati funkcije za rad sa ovim nizom.

(a) Napisati funkciju koja proverava da li niz sadrži zadatu nardmorsku 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.

Napisati i program koji testira rad napisanih funkcija za uneti broj m. Maksimalan broj elemenata niza je 100.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 7
 800 1100 -200 1400 -200 1100 800
 Ucitani niz: 800 1100 -200 1400 -200 1100 800
 Unesite jedan ceo broj:
 1100
 Niz sadrzi element cija je vrednost 1100.
 Indeks njegovog prvog pojavljivanja u nizu je 1.
 Indeks njegovog poslednjeg pojavljivanja u nizu je 5.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 -5
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.8 Marko skuplja sličice za Svetsko prvenstvo u fudbalu. U celobrojnom nizu a se nalaze brojevi onih sličica koje je već sakupio. Marko je primetio da mu se neke sličice ponavljaju i rešio je da ih razmeni sa drugarima. Napisati program koji od datog niza a formira niz b sličica koje se u nizu a pojavljuju više od dva puta. Maksimalan broj elemenata niza je 100.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 8
 Unesite elemente niza a:
 4 11 4 6 8 4 6 6
 Elementi niza b: 4 6
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 13
 Unesite elemente niza a:
 8 26 7 2 1 1 7 2 2 2 7 5 1
 Elementi niza b: 7 2 1
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 2
 Unesite elemente niza a:
 9 5
 Elementi niza b:
```

Zadatak 3.1.9 Sa standardnog ulaza se učitava dimenzija niza, elementi niza i jedan ceo broj k. Napisati program koji štampa indekse elemenata koji su deljivi sa k. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 10 14 86 20
 Unesite broj k: 5
 0 3
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 4
 Unesite elemente niza: 6 14 8 9
 Unesite broj k: 5
 U nizu nema elemenata koji su
 deljivi brojem 5!
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 6
 Unesite elemente niza: 8 9 11 -4 8 11
 Unesite broj k: 2
 0 3 4
```

Zadatak 3.1.10 Autobusi su označeni rednim brojevima (počevši od 1) i u nizu se čuva vreme putovanja svakog autobusa u minutima. Međutim, zbog radova na putu između Požege i Užica, svi autobusi koji saobraćaju na tom potezu (autobusi označeni rednim brojevima od k do t) saobračaju m minuta duže. Uneti potrebne izmene u niz i ispisati elemente niza. Maksimalan broj autobusa je 200.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 8
 Unesite vreme putovanja:
 24 78 13 124 56 90 205 45
 Unesite redne brojeve autobusa koji
 putuju na potezu Pozega, Uzice:
 3 6
 Unesite novo vreme:
 23
 Vreme putovanja nakon izmena:
 24 78 36 147 79 113 205 45
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 8
 Unesite vreme putovanja:
 24 78 13 124 56 90 205 45
 Unesite redne brojeve autobusa koji
 putuju na potezu Pozega, Uzice:
 3 15
 Redni brojevi autobusa nisu
 u dozvoljenom opsegu.
```

Zadatak 3.1.11 Napisati funkciju `int zbir(int a[], int n, int i, int j)` koja računa zbir elemenata niza celih brojeva a dužine n od pozicije i do pozicije j. Napisati i program koji testira rad funkcije. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 5
 Unesite elemente niza: 11 5 6 48 8
 Unesite vrednosti za i i j: 0 2
 Zbir je: 22
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza: 3
 Unesite elemente niza: -2 8 1
 Unesite vrednosti za i i j: 8 12
 Greska: Nekorektne vrednosti granica!
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza: 7
 Unesite elemente niza: -2 5 9 11 6 -3 -4
 Unesite vrednosti za i i j: 2 5
 Zbir: 23
```

Zadatak 3.1.12 Napisati program koji transformiše uneti niz tako što kvadrira sve negativne elemente niza. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 6
 Unesite elemente niza:
 12.34 -6 1 8 32.4 -16
 12.34 36 1 8 32.4 256
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza: 9
 Unesite elemente niza:
 -8.25 6 17 2 -1.5 1 -7 2.65 -125.2
 68.0625 6 17 2 2.25 1 49 2.65 15675.04
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza: 4
 Unesite elemente niza:
 9.53 5 1 4.89
 9.53 5 1 4.89
```

Zadatak 3.1.13 Napisati funkciju `void kvadriranje(float a[], int n)` koja kvadrira elemente realnog niza a dužine n koji se nalaze na parnim pozicijama. Napisati program koji tranformiše na ovaj način uneti niz. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 8
 Unesite elemente niza:
 2.34 1 -12.7 5.2 -8 -6.2 7 14.2
 5.4756 1 161.29 5.2 64 -6.2 49 14.2
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza: 3
 Unesite elemente niza:
 -6 -8.14 -15
 36 -8.14 225
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza: 1
 Unesite elemente niza:
 -35.11
 1232.71
```

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 i program koji testira rad funkcije. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 8
 Unesite elemente niza:
 2.34 1 -12.7 5.2 -8 -6.2 7 14.2
 Unesite vrednost za k: 3
 Zbir je: 8.54
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza: 3
 Unesite elemente niza:
 -6.598 -8.14 -15
 Unesite vrednost za k: 4
 Zbir je: 0.00
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza: 7
 Unesite elemente niza:
 -35.11 5.29 -1.98 12.1 12.2 -3.33 -4.17
 Unesite vrednost za k: 15
 Zbir: 29.59
```

Zadatak 3.1.15 Napisati funkciju `int blizu_3(int a[],int n)` koja pronalazi i vraća indeks elementa niza koji je po vrednosti najbliži aritmetičkoj sredini onih elemenata niza koji su deljivi brojem tri. Napisati program koji testira rad funkcije. Maksimalan broj elemenata niza je 200.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza: 5
 Unesite elemente niza:
 1 2 3 4 5
 3
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza: 5
 Unesite elemente niza:
 3 6 2 4 7
 4
```

Zadatak 3.1.16 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 pojaviljuje 1 puta
 U zapisu broja -39902, cifra 2 se pojaviljuje 1 puta
 U zapisu broja -39902, cifra 3 se pojaviljuje 1 puta
 U zapisu broja -39902, cifra 9 se pojaviljuje 2 puta
```

Zadatak 3.1.17 Napisati funkciju `int cifre(char s[], int n)` koja izračunava broj cifara u nizu karaktera a dužine n. Napisati program koji za karaktere koji se unose u zasebnim redovima ispisuje broj unetih cifara. Maksimalan broj elemenata niza je 100.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 5
 Unesite elemente niza:
 4
 +
 A
 u
 8
 Broj cifara je: 2
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 7
 Unesite elemente niza:
 J
 M
 a
 5
 5
 -
 2
 Broj cifara je: 3
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 3
 Unesite elemente niza:
 e
 k
 F
 Broj cifara je: 0
```

Zadatak 3.1.18 Napisati program koji učitava karaktere sa standardnog ulaza sve do kraja ulaza i izračunava koliko se puta u unetom tekstu pojavila svaka od cifara, svako malo slovo i svako veliko slovo. Ispisati broj pojavljivanja samo za karaktere koji su se u unetom tekstu pojavili barem jednom. UPUTSTVO: *Za evidenciju broja pojavljivanja cifara, malih i velih slova 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
```

Zadatak 3.1.19 Sa standardnog ulaza se unosi jedna linija teksta. Napisati program koji izračunava i ispisuje koliko puta se pojavilo svako od slova engleskog alfabeta u unetom tekstu. Ne praviti razliku između malih i velikih slova.

Primer 1

```
Interakcija sa programom:
 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
```

Zadatak 3.1.20 Napisati program koji učitane karaktere (najviše njih 100, učitavaju se sve do pojave karaktera *) ispisuje u redosledu suprotnom od redosleda čitanja.

Primer 1

```
Interakcija sa programom:
 Unesite karakter: a
 Unesite karakter: 8
 Unesite karakter: 5
 Unesite karakter: Y
 Unesite karakter: I
 Unesite karakter: o
 Unesite karakter: ?
 Unesite karakter: *
 ? o I Y 5 8 a
```

Primer 2

```
Interakcija sa programom:
 Unesite karakter: g
 Unesite karakter: g
 Unesite karakter: 2
 Unesite karakter: 2
 Unesite karakter: )
 Unesite karakter: )
 Unesite karakter: *
 ) ) 2 2 g g
```

Primer 3

```
Interakcija sa programom:
 Unesite karakter: U
 Unesite karakter: 4
 Unesite karakter: a
 Unesite karakter: u
 Unesite karakter: *
 u a 4 U
```

Zadatak 3.1.21 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). Maksimalan broj elemenata niza je 200.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza:
 15
 Unesite elemente niza:
 AnaVoliMilovana
 Jeste palindrom!
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza:
 26
 Unesite elemente niza:
 Zanimljivo je programirati!
 Nije palindrom.
```

Zadatak 3.1.22 Napisati program koji učitava dimenziju i elemente niza i štampa niz u kojem su najveći i najmanji element niza razmenili mesta. Ukoliko se najmanji ili najveći element više puta pojavljuju u nizu, uzeti u obzir njihova prva pojavljivanja. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju niza: 5
 Unesite elemente niza: 8 -2 11 19 4
 8 19 11 -2 4
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju niza: 10
 Unesite elemente niza:
 46 -2 51 8 -5 66 2 8 3 14
 46 -2 51 8 66 -5 2 8 3 14
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 145
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.23 Korišćenjem nizova moguće je predstaviti skupove podataka. Napisati program koji demonstrira osnovne operacije nad skupovima — unija, presek i razlika. Pomoću dva niza predstaviti dva skupa celih brojeva. Maksimalan broj elemenata niza je 500.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza a: 5
 Unesite elemente niza a: 2 8 1 5 2
 Unesite broj elemenata niza b: 3
 Unesite elemente niza b: 5 7 8
 Unija: 2 8 1 5 2 5 7 8
 Presek: 5
 Razlika: 2 1 2
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza a: 3
 Unesite elemente niza a: 11 4 4
 Unesite broj elemenata niza b: 2
 Unesite elemente niza b: 18 9
 Unija: 11 4 4 18 9
 Presek:
 Razlika: 11 4 4
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza a: 6
 Unesite elemente niza a: 12 7 9 12 5 1
 Unesite broj elemenata niza b: 4
 Unesite elemente niza b: 1 12 22 12
 Unija: 12 7 9 12 5 1 1 12 22 12
 Presek: 12 12 1
 Razlika: 7 9 5
```

Zadatak 3.1.24 Napisati program koji za dva učitana niza a i b dimenzije n formira i na izlaz ispisuje niz c koji se dobija naizmeničnim raspoređivanjem elemenata nizova a i b, tj. $c = [a_0, b_0, a_1, b_1, \ldots, a_{n-1}, b_{n-1}]$. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite dimenziju nizova:
 5
 Unesite elemente niza a:
 2 -5 11 4 8
 Unesite elemente niza b:
 3 3 9 -1 17
 Rezultujuci niz:
 2 3 -5 3 11 9 4 -1 8 17
```

Primer 2

```
Interakcija sa programom:
 Unesite dimenziju nizova:
 105
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.25 Napisati program koji za dva učitana niza a i b dimenzije n formira i na izlaz ispisuje niz c čija prva polovina odgovara elemetima niza b, a druga polovina elementima niza a, tj. $c = [b_0, b_1, \ldots, b_{n-1}, a_0, a_1, \ldots, a_{n-1}]$. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 Unesite elemente niza a: 4 -8 32
 Unesite elemente niza b: 5 2 11
 5 2 11 4 -8 32
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 Unesite elemente niza a: 1 0 -1 0
 Unesite elemente niza b: 5 5 5 3
 5 5 5 3 1 0 -1 0
```

Primer 3

```
Interakcija sa programom:
 Unesite dimenziju niza: 145
 Greska: Nedozvoljena vrednost!
```

*** Zadatak 3.1.26** Sa standardnog ulaza se učitava ceo broj n manji od 100 i elementi dvaju celobrojnih, sortiranih neopadajuće nizova a i b dimenzije n. Napisati program koji formira i ispisuje niz c koji se dobija spajanjem nizova a i b u treći, takođe sortiran neopadajuće, niz.

Primer 1

```
Interakcija sa programom:
 Uneti broj elemenata niza: 5
 Uneti elemente sortiranog niza:
 2 11 28 40 63
 Uneti elemente sortiranog niza:
 -19 -5 5 11 52
 Niz c:
 -19 -5 2 5 11 11 28 40 52 63
```

Primer 2

```
Interakcija sa programom:
 Uneti broj elemenata niza: 3
 Uneti elemente sortiranog niza:
 -2 4 8
 Uneti elemente sortiranog niza:
 6 15 19
 Niz c:
 -2 4 6 8 15 19
```

Primer 3

```
Interakcija sa programom:
 Uneti broj elemenata niza: 145
 Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.27 Napisati program koji sa standardnog ulaza učitava 10 celih brojeva i razdvaja ih na parne i neparne tako što parne brojeve upisuje na početak niza, a neparne brojeve na kraj niza. Ispisati niz dobijen na ovaj način. Napomena: *Nije dozvoljeno koristiti pomoćne nizove.*

Primer 1

```
Interakcija sa programom:
Unesite 10 brojeva:
-2 8 11 53 59 20 17 -8 3 14
Rezultujuci niz:
14 142 -6 -278 28 34 33 -69 -9 9
```

Primer 2

```
Interakcija sa programom:
Unesite 10 brojeva:
9 142 -9 -278 -69 33 34 28 -6 14
Rezultujuci niz:
-2 8 14 -8 20 59 17 53 3 11
```

Zadatak 3.1.28 Napisati funkcije za rad sa nizovima celih brojeva.

(a) Napisati funkciju koja obrće elemente niza.

(b) Napisati funkciju koja rotira niz ciklično za jedno mesto u levo.

(c) Napisati funkciju koja rotira niz ciklično za k mesta u levo.

Napisati i program koji testira rad napisanih funkcija za uneti broj m. Maksimalan broj elemenata niza je 100.

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: Nedozvoljena vrednost!
```

Zadatak 3.1.29 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 dosao 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 plaćaju platni promet ili 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 lako 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.

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
```

Zadatak 3.1.30 Napisati program koji za učitani niz formira i ispisuje niz koji se dobija izbacivanjem svih neparnih elemenata niza. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Maksimalan broj elemenata niza je 100.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 4
 Unesite elemente niza: 8 9 15 12
 8 12
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 6
 Unesite elemente niza: 21 5 3 22 19 188
 22 188
```

Primer 3

```
Interakcija sa programom:
Unesite broj elemenata niza: 4
Unesite elemente niza: 133 129 121 101
```

Primer 4

```
Interakcija sa programom:
Unesite broj elemenata niza: 8
Unesite elemente niza: 15 -22 -23 13 18 46 14 -31
-22 18 46 14
```

Zadatak 3.1.31 Napisati program koji učitava dimenziju n celobrojnog niza a i njegove elemente, i iz niza a izbacuje sve elemente koji nisu deljivi svojom poslednjom cifrom. Izuzetak su elementi čija je poslednja cifra 0 koje treba zadržati. Program treba da ispiše izmenjeni niz na standardni izlaz. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
Unesite dimenziju niza:
9
Unesite elemente niza a:
173 -25 23 7 17 25 34 61 -4612
Niz a nakon izmene:
-25 7 25 61 -4612
```

Primer 2

```
Interakcija sa programom:
Unesite dimenziju niza:
0
Greska: Nedozvoljena vrednost!
```

Zadatak 3.1.32 Napisati program koji u nizu dužine n čiji se elementi učitavaju sa ulaza eliminiše sve brojeve koji nisu deljivi svojim indeksom. Niz reorganizovati tako da nema *rupa* koje su nastale eliminacijom elemenata i ispisati na standardni izlaz. Maksimalan broj elemenata niza je 700. Ne razmatrati da li je u novom nizu, nakom brisanja i pomeranja, element deljiv svojim indeksom. Napomena: *Nulti element niza treba zadržati jer nije dozvoljeno deljenje nulom.*

Primer 1

```
Interakcija sa programom:
Unesite broj elemenata niza: 10
Unesite elemente niza:
4 2 1 6 7 8 10 2 16 3
Novi niz:
4 2 6 16
```

Primer 2

```
Interakcija sa programom:
Unesite broj elemenata niza: 10
Unesite elemente niza:
-8 5 10 6 7 10 8 2 16 27
Novi niz:
-8 5 10 6 10 16 27
```

Zadatak 3.1.33 Napisati program koji za učitani niz ispisuje niz koji se

dobija izbacivanjem svih elemenata koji su prosti brojevi. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Maksimalan broj elemenata niza je 100. NAPOMENA: *Broj* 1 *nije prost.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 5
 Unesite elemente niza: 11 5 6 48 8
 6 48 8
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 4
 Unesite elemente niza: 11 5 19 21
 21
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 5
 Unesite elemente niza: 12 18 9 31 7
 12 18 9
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 3
 Unesite elemente niza: -31 11 -19
```

Primer 5

```
INTERAKCIJA SA PROGRAMOM:
 Unesite broj elemenata niza: 5
 Unesite elemente niza: -2 15 -11 8 7
 15 8
```

Zadatak 3.1.34 Napisati funkciju koja proverava da li su elementi celebrojnog niza uređeni neopadajuće. Maksimalan broj elemenata niza je 300.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 8
 Unesite elemente niza:
 -40 -8 -8 2 30 30 46 200
 Jeste uredjen neopadajuce!
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite dimenziju niza:
 4
 Unesite elemente niza:
 4 23 15 30
 Nije uredjen neopadajuce.
```

Zadatak 3.1.35 Saki indeks niza označava jedan dan u mesecu, a elementi celobrojnog niza predstavljaju broj artikala koji se prodao tog dana. Naći koliko najduže je iz dana u dan broj prodatih artikala rastao.

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.
```

Zadatak 3.1.36 Napisati funkciju koja određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva. Maksimalan broj elemenata niza je 100.

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.
```

Zadatak 3.1.37 Napisati funkciju koja određuje da li se jedan niz javlja kao podniz uzastopnih elemenata drugog niza. Zadatak rešiti na dva načina:

(a) Elementi moraju da budu uzastopni.

(b) Elementi ne moraju da budu uzastopni, ali je redosled pojavljivanja isti.

Maksimalan broj elemenata niza je 100.

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.
```

Zadatak 3.1.38 Za celobrojni niz a dimenzije n kažemo da je *permutacija* ako sadrži sve brojeve od 1 do n.

(a) Napisati funkciju `void brojanje(int a[], int b[], int n)` koja na osnovu celobrojnog niza a dimenzije n formira niz b tako što i-ti element niza b odgovara broju pojavljivanja vrednosti i u nizu a.

(b) Napisati funkciju `int permutacija(int a[], int n)` koja proverava da li je zadati niz permutacija. Funkcija vraća vrednost 1 ako je svojstvo ispunjeno, odnosno 0 ako nije. Uputstvo: *Koristiti funkciju brojanje iz tačke (a).*

Napisati program koji sa standardnog ulaza učitava dimenziju niza i elemente niza i ispisuje da li je uneti niz permutacija ili ne. Maksimalan broj elemenata niza je 100.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza:  5
 Unesite elemente niza:
 1 5 4 3 2
 Uneti niz je permutacija.
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza:  6
 Unesite elemente niza:
 2 3 3 1 1 5
 Uneti niz nije permutacija.
```

Zadatak 3.1.39 Napisati program koji za dva cela broja x i y koja se učitavaju sa standardnog ulaza proverava da li se zapisuju pomoću istih cifara. Uputstvo: *Rešiti korišćenjem nizova. Pogledati zadatak 2.5.16.*

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!

3.2 Rešenja

Rešenje 3.1.1

```
#include <stdio.h>

/* Predprocesorska direktiva kojom se definise maksimalni broj
    elemenata niza. */
#define MAX 100

int main()
{
  int a[MAX];
  int b[MAX];
  int n;
  int i;
  int skalarni_proizvod;


  /* Ucitava se dimenzija vektora i proverava njena ispravnost. */
  printf("Unesite dimenziju vektora: ");
  scanf("%d", &n);
  if (n<1 || n>100)
  {
    printf("Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se koordinate vektora. */
  printf("Unesite koordinate vektora a: ");
  for (i=0; i<n; i++)
  {
    scanf("%d", &a[i]);
  }

  printf("Unesite koordinate vektora b: ");
  for (i=0; i<n; i++)
  {
    scanf("%d", &b[i]);
  }

```

```
  /* Izracunava se skalarni proizvod po zadataj formuli. */
  skalarni_proizvod=0;

  for (i=0; i<n; i++)
    skalarni_proizvod = skalarni_proizvod + a[i]*b[i];

  /* I ispisuje se njegova vrednost. */
  printf("Skalarni proizvod vektora a i b: %d\n",skalarni_proizvod);

  return 0;
}
```

Rešenje 3.1.2

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX];
  int n;
  int i;

  /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
  printf("Unesi dimenziju niza:\n");
  scanf("%d", &n);
  if (n<1 || n>MAX)
  {
    printf("Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza. */
  printf("Unesi elemente niza:\n");
  for (i=0; i<n; i++)
  {
    scanf("%d", &a[i]);
  }

  /* Ispisuju se elementi niza na parnim pozicijama. */
  printf("Elementi niza na parnim pozicijama:\n");
  for (i=0; i<n; i+=2)
  {
    printf("%d ", a[i]);
  }
  printf("\n");

  /* Ispisuju se parni elementi niza. */
  printf("Parni elementi niza:\n");
  for (i=0; i<n; i++){
```

```
38      if (a[i]%2==0){
          printf("%d ", a[i]);
40      }
    }
42  printf("\n");


44
    return 0;
46  }
```

Rešenje 3.1.3

```
#include <stdio.h>
2
#define MAX 1000
4
/* Funkcija koja ucitava elemente niza. */
6 void ucitaj(int a[], int n)
{
8    int i;
    for(i=0;i<n;i++)
10   {
        scanf("%d",&a[i]);
12   }
}
14
/* Funkcija koja stampa elemente niza. */
16 void stampaj(int a[], int n)
{
18   int i;
    for(i=0;i<n;i++)
20      printf("%d ",a[i]);
    printf("\n");
22 }

24 /* Funkciju racuna sumu elemenata niza. */
int suma(int a[], int n)
26 {
    int i;
28   int s=0;
    for(i=0;i<n;i++)
30      s+=a[i];
    return s;
32 }


34
/* Funkciju koja racuna prosecnu vrednost elemenata niza. */
36 float prosek(int a[], int n)
{
38   int i;
    int s = suma(a,n);
```

```
40     return (float) s/n;
}
42

44 /* Funkciju koja izracunava maksimum elemenata niza.*/
int maksimum (int a[],int n)
46 {
     int m;
48   int i;

50   /* Maksimum se inicijalizuje prvim elementom niza (a[0]), a zatim
      se prolazi kroz ostatak niza. U svakom koraku se poredi vrednost
      maksimuma sa tekucim elementom niza. */
     m = a[0];
52   for(i=1;i<n;i++)
        if (a[i] > m)
54         m = a[i];

56   /* Vraca se izracunata vrednost maksimuma. */
     return m;
58 }

60

   /* Funkciju koja izracunava poziciju maksimalnog elementa u nizu. */
62 int pozicija_maksimuma (int a[],int n)
   {
64   int m;
     int m_pozicija;
66   int i;

68   /* Minimum se inicijalizuje prvim elementom niza (a[0]) i pamti se
        njegova pozicija (0), a zatim se prolazi kroz ostatak niza. U
      svakom koraku se poredi vrednost minimuma sa tekucim elementom
      niza i ukoliko je potrebno menjaju se vrednosti minimuma i
      njegove pozicije. */

70   m = a[0];
     m_pozicija=0;
72   for(i=1;i<n;i++)
        if (a[i] < m)
74      {
           m = a[i];
76         m_pozicija=i;
        }
78

     /* Vraca se izracunata pozicija. */
80   return m_pozicija;
}
82


84
   int main()
```

```
{
    int a[MAX];
    int n;

    /* Ucitava se dimenzija niza i proverava njena ispravnost. */
    printf("Unesite dimenziju niza:");
    scanf("%d",&n);
    if (n<1 || n>MAX)
    {
        printf("Nedozvoljena vrednost!\n");
        return -1;
    }


    /* Testira se funkcija kojom se ucitavaju elementi niza. */
    ucitaj(a,n);

    /* Testira se funkcija kojom se ispisuju elementi niza. */
    printf("Vreme trcanja takmicara: ");
    stampaj(a,n);

    /* Testira se funkcija kojom se izracunava suma elemenata niza. */
    printf("Ukupno vreme: %d\n", suma(a,n));

    /* Testira se funkcija kojom se racuna prosek elemenata niza. */
    printf("Prosecno vreme trcanja: %.2f\n", prosek(a,n));

    /* Testira se funkcija kojom se izracunava minimum niza. */
    printf("Maksimalno vreme trcanja: %d\n", minimum(a,n));

    /* Testira se funkcija kojom se izracunava pozicija maksimalnog
     elementa. */
    printf("Indeks pobednika: %d\n", pozicija_maksimuma(a,n));

    return 0;
}
```

Rešenje 3.1.4

Rešenje 3.1.5

```
#include <stdio.h>

#define MAX 100

/* Funkcija prebrojavanje vraca broj elemenata niza koji su manji od
    poslednjeg elementa. */
int prebrojavanje(int a[], int n)
{
  int i;
```

```
  /* Brojac elemenata koji su manji od poslednjeg. */
  int broj_manjih=0;

  /* Obilazi se element po element niza, */
  for(i=0;i<n-1;i++){
    /* Ako je tekuci element manji od poslednjeg (on se nalazi na
    poziciji n-1) uvecava se brojac. */
    if(a[i]<a[n-1])
      broj_manjih++;
  }

  /* Vraca se izracunata vrednost. */
  return broj_manjih;
}

int main()
{
  int a[MAX];
  int n;
  int i;

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
    . */
  printf("Unesite broj elemenata niza:");
  scanf("%d", &n);
  if(n<=0 || n>MAX)
  {
     printf("Greska: Nedozvoljena vrednost!\n");
     return 0;
  }

  /* Ucitavaju se elementi niza. */
  printf("Unesite elemente niza:");
  for(i=0;i<n;i++)
   scanf("%d",&a[i]);

  /* Ispisuje se rezultat poziva funkcije. */
  printf("%d\n", prebrojavanje(a,n));

  return 0;
}
```

Rešenje 3.1.6

```
#include <stdio.h>

#define MAX 100

/* Funkcija vraca broj parnih elemenata niza koji prethode
    maksimalnom elementu niza. */
```

```
6  int prebrojavanje(int a[], int n)
   {
8    int i;

10   int maksimum;
     int pozicija_maksimuma;
12
     /* Brojac elemenata koji su parni i prethode maksimalnom. */
14   int broj_parnih;

16   /* Pronalazi se maksimalni element niza i njegova pozicija. */
     maksimum = a[0];
18   pozicija_maksimuma = 0;

20   for(i=1;i<n-1;i++)
       if(a[i]> maksimum)
22     {
         maksimum = a[i];
24       pozicija_maksimuma = i;
       }
26
     /* Prebrojavaju se parni elementi koji prethode maksimalnom. */
28   broj_parnih = 0;
     for(i=0; i < pozicija_maksimuma; i++){
30     if(a[i]%2==0){
         broj_parnih++;
32     }
     }
34
     /* Vraca se izracunata vrednost. */
36   return broj_parnih;
   }
38
   int main()
40 {
     int a[MAX];
42   int n;
     int i;
44
     /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
        . */
46   printf("Unesite broj elemenata niza:");
     scanf("%d", &n);
48   if(n<=0 || n>MAX)
     {
50      printf("Greska: Nedozvoljena vrednost!\n");
        return 0;
52   }

54   /* Ucitavaju se elementi niza. */
     printf("Unesite elemente niza:");
56   for(i=0;i<n;i++){
```

```
    scanf("%d",&a[i]);
  }

  /* Ispisuje se rezultat poziva funkcije. */
  printf("%d\n", prebrojavanje(a,n));

  return 0;
}
```

Rešenje 3.1.7

```
#include <stdio.h>

#define MAX 100

/* Funkcija koja ucitava elemente niza. */
void ucitaj(int a[], int n)
{
   int i;
   for(i=0;i<n;i++)
   {
      scanf("%d",&a[i]);
   }
}

/* Funkcija koja ispisuje elemente niza. */
void stampaj(int a[], int n)
{
   int i;
   for(i=0;i<n;i++)
      printf("%d ",a[i]);
   printf("\n");
}


/* Funkcija koja proverava da li niz sadrzi zadatu vrednost m. */
int sadrzi(int a[], int n, int m)
{
   int i;

   /* Poredi se element po element niza a sa zadatim brojem m. */

   for(i=0;i<n;i++){
     /* Ukoliko je tekuci element niza jednak trazenom broju funkcija
     vraca vrednost 1. */
     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 koja vraca vrednost prve pozicije na kojoj se nalazi
     element koji ima vrednost m, ili -1 ukoliko element nije u nizu.
     */
int prvo_pojavljivanje(int a[], int n, int m)
{
   int i;

   /* Poredi se element po element niza a sa zadatim brojem m. */

   for(i=0;i<n;i++){
    /* Ukoliko je tekuci element niza jednak trazenom broju vraca se
     njegov indeks. */
      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 koja vraca vrednost poslednje pozicije na kojoj se nalazi
      element koji ima vrednost m, ili -1 ukoliko element nije u nizu.
      */
int poslednje_pojavljivanje(int a[], int n, int m)
{
   int i;

   /* Polazi se od kraja niza i poredi se element po element sa
     zadatim brojem m. */

   for(i=n-1;i>=0;i--){
     /* Ukoliko je tekuci element niza jednak trazenom broju. */
      if (a[i]==m){
       /* Vraca se njegov indeks. */
        return i;
      }
   }

   /* Ako se stigne do pocetka niza i ne naidje na vrednost koja je
     jednaka broju m, to znaci da se broj ne nalazi u nizu i da
     funkcija treba da vrati -1. */
   return -1;
}


int main()
```

```
{
    int a[MAX];
    int n;
    int m;
    int i;

    /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
    printf("Unesite dimenziju niza:");
    scanf("%d",&n);
    if (n<1 || n>MAX)
    {
        printf("Nedozvoljena vrednost!\n");
        return -1;
    }

    /* Ucitavaju se i ispisuju elementi niza. */
    ucitaj(a,n);
    printf("Ucitani niz:");
    stampaj(a,n);

    /* Ucitava se vrednost za pretragu. */
    printf("Unesi jedan ceo broj:");
    scanf("%d",&m);

    /* Proverava se rad napisanih funkcija. */
    if(sadrzi(a,n,m))
        printf("Niz sadrzi element cija je vrednost %d\n", m);
    else
        printf("Niz ne sadrzi element cija je vrednost %d\n", m);

    i = prvo_pojavljivanje(a,n,m);
    if(i!=-1)
        printf("Indeks njegovog prvog pojavljivanja u nizu je %d\n", m,
     i);


    i = poslednje_pojavljivanje(a,n,m);
    if(i!=-1)
        printf("Indeks njegovog poslednjeg pojavljivanja u nizu je %d\n
      ", m,i);

    return 0;
}
```

Rešenje 3.1.8

```
#include <stdio.h>

#define MAX 100

/* Funkcija koja vraca broj pojavljivanja broja x u nizu */
```

```
int broj_pojavljivanja(int niz[], int n, int x)
{
  int i;

  /* Broj pojavljivanja broja x */
  int brojac = 0;

  /* Obilazi se element po element niza */
  for(i=0;i<n;i++){
    /* Ukoliko je tekuci element jednak trazenom broju */
    if(niz[i] == x){
      /* Uvecava se broj pojavljivanja */
      brojac++;
    }
  }

  /* Vraca se izracunata vrednost */
  return rezultat;
}

int main()
{
  /* Niz elemenata koje zadaje korisnik */
  int a[MAX];

  /* Niz elemenata koji se pojavljuju tri puta */
  int b[MAX];

  int i, j, n, n_b;

  /* Ucitava se broj elemenata korisnickog niza i proverava se
    njegova ispravnost. */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<1 || n>MAX)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi korisnickog niza. */
  printf("Unesite elemente niza a: ");
  for(i=0;i<n;i++)
    scanf("%d", &a[i]);

  /* Parametar j je brojac elemenata rezultujuceg niza b. */
  j = 0;

  /* Obilazi se element po element niza a. */
  for(i=0;i<n;i++)
  {
    /* Ukoliko se tekuci element pojavljuje vise od dva puta u nizu a
```

```
      i nije upisan u niz b koji trenutno ima j elemenata, dodaje se u
      niz b na poziciju j i uvecava se broj elemenata niza b. */
57    if(broj_pojavljivanja(a, n, a[i])>=3 && broj_pojavljivanja(b, j,
      a[i])==0)
      {
59      b[j] = a[i];
        j++;
61    }
  }
63
  /* Ispisuje se rezultujuci niz b, broj elemenata u nizu b je j. */
65 n_b = j;
  for(i=0;i<n_b;i++)
67   printf("%d ", b[i]);
  printf("\n");
69
  return 0;
71 }
```

Rešenje 3.1.9

```
1  #include <stdio.h>

3  #define MAX 100

5  int main()
   {
7    int brojevi[MAX];
     int n, i, k, indikator;
9
     /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
11   printf("Unesite dimenziju niza: ");
     scanf("%d", &n);
13   if(n<1 || n>MAX)
     {
15     printf("Greska: Nedozvoljena vrednost!\n");
       return -1;
17   }

19   /* Ucitavaju se elementi niza. */
     printf("Unesite elemente niza: ");
21   for(i=0;i<n;i++)
       scanf("%d", &brojevi[i]);
23

25   /* Ucitava se broj k i proverava se njegova ispravnost. */
     printf("Unesite broj k: ");
27   scanf("%d", &k);
     if(k == 0)
29   {
       printf("Greska: Pogresan unos!\n");
```

```
31      return -1;
    }
33
    /* Promenljiva koja cuva informaciju o tome da li je u nizu
       postojao element koji je deljiv brojem k. Inicijalna vrednost je
       0. */
35
    indikator = 0;
37
    /* Ukoliko je element niza deljiv brojem k, indikator se postavlja
       na 1 i ispisuje se indeks tog elementa. */
39
    for(i=0;i<n;i++){
41    if(brojevi[i]%k == 0)
        {
43        indikator = 1;
          printf("%d ",i);
45      }
    }
47
    /* Ukoliko je indikator jednak nuli to znaci da ne postoji element
       u nizu koji je deljiv brojem k. */
49
    if(indikator == 0){
51    printf("U nizu nema elemenata koji su deljivi brojem %d!\n",k);
    }
53

55  return 0;
}
```

Rešenje 3.1.10

```
#include <stdio.h>
2
/* Indeksiranje pocinje od 1, pa zato maksimalna dimenzija niza mora
    biti 201, a ne 200. */
4 #define DIM 201

6 int main()
{
8   int n, niz[DIM], i;
    int k, t;
10  int m;

12  /* Ucitavanje dimenzije i elemenata niza. */
    printf("Unesite dimenziju niza: ");
14  scanf("%d", &n);

16  if (n<=0 || n>DIM)
    {
```

```
18      printf("Nedozvoljena dimenzija niza.\n");
        return 0;
20    }

22    printf("Unesite vreme putovanja:\n");
      for(i=1; i<=n; i++)
24      scanf("%d", &niz[i]);

26    /* Unos rednih brojeva autobusa. */
      printf("Unesite redne brojeve autobusa koji putuju na potezu Pozega
          , Uzice:\n");
28    scanf("%d%d", &k, &t);

30    if (k<=0 || k>n || t<=0 || t>n)
      {
32      printf("Redni brojevi autobusa nisu u dozvoljenom opsegu.\n");
        return -1;
34    }

36    /* Unos vremena. */
      printf("Unesite novo vreme:\n");
38    scanf("%d", &m);

40    if (m < 0)
      {
42      printf("Vreme ne moze biti negativno.\n");
        return -1;
44    }

46    /* Unos izmena u niz. */
      for(i=k; i<=t; i++)
48      niz[i] += m;

50    /* Ispis niza. */
      printf("Vreme putovanja nakon izmena:");
52    for(i=1; i<=n; i++)
        printf("%d ", niz[i]);
54    printf("\n");

56    return 0;
  }
```

Rešenje 3.1.11

```
1  #include<stdio.h>

3  #define MAX 100

5  /* Funkcija racuna zbir elemenata niza od pozicije i do pozicije j.
       */
   int zbir(int a[], int n, int i, int j){
```

```
7   int k;

9   /* Zbir elemenata niza iz zadatog opsega. */
    int z = 0;
11
    /* Obilaze se elementi niza. */
13  for(k=i; k<=j; k++){
    z+=a[k];
15  }

17  /* Vraca se izracunata vrednost. */
    return z;
19 }

21 int main(){

23  int n, i, j;
    int a[MAX];
25
    /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
       . */
27  printf("Unesite broj elemenata niza: ");
    scanf("%d", &n);
29  if(n <= 0 || n > MAX)
    {
31  printf("Greska: Nedozvoljena vrednost!\n");
    return 0;
33  }

35  /* Ucitavaju se elementi niza. */
    printf("Unesite elemente niza:");
37  for(i=0; i<n; i++)
    scanf("%d", &a[i]);
39
    /* Ucitavaju se vrednosti granica. */
41  printf("Unesite vrednosti za i i j: ");
    scanf("%d%d", &i, &j);
43
    /* Proverava se korektnost zadatog intervala. */
45  if(i < 0 || j < 0 || i > n-1 || j > n-1 || i > j){
    printf("Greska: Nekorektne vrednosti granica!\n");
47  return 0;
    }
49
    /* Ispisuje se rezultat poziva funkcije. */
51  printf("Zbir je: %d", zbir(a,n,i,j));

53  return 0;
}
```

Rešenje 3.1.12

```c
#include <stdio.h>

#define MAX 100

int main()
{
  float brojevi[MAX];
  int n, i;

  /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);
  if(n<1 || n>MAX)
  {
    printf("Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza. */
  printf("Unesite elemente niza:\n");
  for(i=0;i<n;i++){
    scanf("%f", &brojevi[i]);
  }

  /* Ukoliko je i-ti element niza brojevi[i] negativan broj, 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");

  return 0;
}
```

Rešenje 3.1.13

```c
#include<stdio.h>

#define MAX 100

/* Funkcija koja kvadrira elemente niza koji se nalaze na parnim
   pozicijama. */
void kvadriranje(float a[], int n){

```

```
  int i;

  /* Obilaze se elementi na parnim pozicijama i kvadriraju se: a[i] =
     a[i]*a[i]. */
  for(i=0; i<n; i+=2)
    a[i]*=a[i];
}

int main(){

  int n, i, j;
  float a[MAX];

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     . */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);
  if(n <=0 || n>MAX)
  {
  printf("Greska: Nedozvoljena vrednost!\n");
  return 0;
  }

  /* Ucitavaju se elementi niza. */
  printf("Unesite elemente niza:");
  for(i=0; i<n; i++)
  scanf("%f", &a[i]);

  /* Poziva se funkcija. */
  kvadriranje(a,n);

  /* Ispisuju se elementi novodobijenog niza. */
  for(i=0; i<n; i++)
  printf("%g ", a[i]);
  printf("\n");

  return 0;
}
```

Rešenje 3.1.14

```
#include<stdio.h>

#define MAX 100

/* Funckija racuna zbir prvih k pozitivnih elemenata niza. */
float zbir_pozitivnih(float a[], int n, int k){

  int i;

  /* Zbir pozitivnih elemenata. */
```

```
  float zbir=0;

  /* Obilazi se element po element niza. Postupk se zavrsava ukoliko
    se dodje do kraja niza ili ukoliko se sabere k pozitivnih
    elemenata. */
  for(i=0; i<n && k>0; i++){
    /* Ako je tekuci element pozitivan broj. */
  if(a[i] >= 0){
    /* Dodaje se zbiru. */
    zbir+=a[i];
     /* Umanjuje se brojac pozitivnih elemenata. */
    k--;
  }
  }

  /* Vraca se izracunata vrednost. */
  return zbir;
}

int main(){
  int n, i, k;
  float a[MAX];

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
    . */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);
  if(n<=0 || n> MAX){
  printf("Greska: Nedozvoljena vrednost!\n");
  return 0;
  }

  /* Ucitavaju se elementi niza. */
  printf("Unesite elemente niza: ");
  for(i=0; i<n; i++)
  scanf("%f", &a[i]);

  /* Ucitava se broj k i proverava se njegova ispravnost. */
  printf("Unesite vrednost za k: ");
  scanf("%d", &k);
  if(k<0 || k>n){
  printf("Greska: Nedozvoljena vrednost!");
  return 0;
  }

  /* Ispisuje se rezultat poziva funkcije. */
  printf("Zbir je: %.2f\n", zbir_pozitivnih(a,n,k));

  return 0;
}
```

Rešenje 3.1.15

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

#define DIM 200

/* Funkcija koja racuna aritmeticku sredinu onih elemenata niza koji
    su deljivi sa 3. */
float artmeticka_sredina_3(int niz[], int n)
{
  /* Brojac u petlji. */
  int i;
  /* Parametar koji predstavlja zbir svih elemenata deljivih sa 3. */
  int suma = 0;
  /* Parametar koji predstavlja brojih elemenata niza koji su deljivi
     sa 3. */
  int br_3 = 0;

  /* U petlji se proverava svaki element niza i za one koji su
    deljivi sa 3 uvecava se suma i odgovarajuci brojac. */
  for(i=0; i<n; i++)
    if (niz[i]%3 == 0)
    {
      suma += niz[i];
      br_3++;
    }

  return (float)suma/br_3;
}


int blizu_3(int a[], int n)
{
  /* Parametar koji predstavlja aritmeticku sredinu onih elemenata
    niza koji su deljivi sa 3. */
  float art = artmeticka_sredina_3(a, n);
  /* Pretpostavka je da je prvi element niza najblizi izracunatoj
    aritmetickoj sredini. */
  int element = a[0];
  /* Radi brzine izracunavanja pamti se razdaljina izmedju trenutno
    najblizeg elementa i aritmeticke sredine. Izracunava se apsolutna
     vrednost razdaljine jer ona moze biti i negativna. Moze se
    koristiti ugradjena funkcija fabs za racunanje apsolutne
    vrednosti realnih brojeva. */
  float razdaljina = fabs(art - element);
  /* Pomocni parametar koji sluzi da se u petlji pamti tekuca
    razdaljina. */
  float nova_razdaljina;
  /* Brojac u petlji. */
  int i;
```

```
  /* U petlji se ispituje svaki element niza i proverava se da li
     postoji neki koji je po vrednosti blizi aritmetickoj sredini. */
  for(i=1; i<n; i++)
  {
    nova_razdaljina = fabs(a[i] - art);
    /* Izracunava se razdaljina izmedju tekuceg elementa niza i
    poredi sa trenutnom razdaljinom. Ako je novodobijena vrednost
    manja vrsi se zamena zapamcenih parametara. */
    if (nova_razdaljina < razdaljina)
    {
      razdaljina = nova_razdaljina;
      element = a[i];
    }
  }

  return element;
}

int main()
{
  int niz[DIM], n, i;

  /* Ucitavanje dimenzije i elemenata niza. */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);

  if (n<=0 || n>DIM)
  {
    printf("Nedozvoljena dimenzija niza.\n");
    return 0;
  }

  printf("Unesite elemente niza: ");
  for(i=0; i<n; i++)
    scanf("%d", &niz[i]);

  /* Ispisivanje rezultata. */
  printf("Trazeni broj je %d.\n", blizu_3(niz, n));

  return 0;
}
```

Rešenje 3.1.16

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

int main()
{
   int x;
   int brojaci[10];
```

```
    char cifra;
    int original;
    int i;

    /* Ucitava se ceo broj sa standardnog ulaza. */
    printf("Unesite ceo broj:\n");
    scanf("%d",&x);

    /* Cuva se njegova originalna vrednost zbog finalnog ispisa. */
    original = x;

    /* Nadalje posmatra apsolutna vrednost. */
    x = abs(x);

    /* Svaki element niza brojaci predstavljace brojac za jednu od
     cifara:
       brojac[0] predstavljace broj nula u zapisu broja x
       brojac[1] predstavljace broj jedinica u zapisu broja x
       ...
       brojac[9] predstavljace broj devetki u zapisu broja x. */


    /* Brojaci se na pocetku inicijalizuju nulama, */
    for(i=0;i<10;i++){
       brojaci[i]=0;
    }

    /* Sve dok ima cifara u zapisu broja x */
    do
    {
      /* Izdvaja se krajnja desna cifara. */
      cifra = x%10;

      /* Uvecava se njen broj pojavljivanja. */
       brojaci[cifra]++;

       /* I prelazi se na analiziranje sledece cifre. */
       x/=10;

    } while(x);

    /* Ispisuju se informacije o ciframa koje se nalaze u zapisu broja
      x. */
    for(i=0; i<10; i++){
       if(brojaci[i]){
          printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n
     ", original, i, brojaci[i]);
       }
    }


  return 0;
```

```
}
```

Rešenje 3.1.17

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

#define MAX 100

/* Funkcija prebrojava cifre u datom nizu karaktera. */
int cifre(char a[], int n)
{
  int i;

  /* Brojac cifara. */
  int broj_cifara = 0;

  /* Obilazi se element po element niza. */
  for(i=0;i<n;i++){
    /* Ako je tekuci element cifra uvecava se broj cifara. */
    if(isdigit(a[i]))
      broj_cifara++;
  }

  /* Vraca se izracunata vrednost. */
  return broj_cifara;
}

int main()
{
  char a[MAX];
  int n;
  int i;

  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
     . */
  printf("Unesite broj elemenata niza:");
  scanf("%d", &n);
  if(n<=0 || n>MAX)
  {
     printf("Greska: Nedozvoljena vrednost!\n");
     return 0;
  }

  /* Ucitavaju se elementi niza. */
  printf("Unesite elemente niza:");
  for(i=0;i<n;i++) {
   /* Preskace se prethodno uneti znak za novi red. */
   getchar();

   /* A zatim se ucitava sam karakter i smesta u niz. */
```

```
47    scanf("%c",&a[i]);
    }
49
    /* Ispisuje se rezultat poziva funkcije. */
51  printf("Broj cifara je: %d\n", cifre(a,n));

53  return 0;
}
```

Rešenje 3.1.18

```
#include <stdio.h>
2
/* Funkcija za ispis elemenata niza. */
4 void ispis(int niz[], int n, char c)
{
6   int i;

8   for(i = 0; i < n; i++){
       if (niz[i]!=0)
10      printf("Karakter %c se pojavljuje %d puta\n", c + i, niz[i]);
    }
12 }

14 int main()
{
16  /* Niz u kojem ce se cuvati informacije o broju pojavljivanja
      cifara. */
    int cifre[10];
18
    /* Niz u kojem ce se cuvati informacije o broju pojavljivanja malih
       slova. */
20  int mala_slova[26];

22  /* Niz u kojem ce se cuvati informacije o broju pojavljivanja
      velikih slova. */
    int velika_slova[26];
24
    int c, i;
26
    /* Brojaci karaktera se na pocetku inicijalizuju nulama. */
28  for(i=0;i<10;i++){
      cifre[i]=0;
30  }

32  for(i=0;i<26;i++)
    {
34    mala_slova[i]=0;
      velika_slova[i]=0;
36  }
```

```
  /* Ucitavaju se karakteri sve do kraja ulaza. */
  while((c = getchar()) != EOF)
  {
    /* Ako je procitani karakter veliko slovo uvecava se broj
    pojavljivanja odgovarajuceg velikog slova. */
    if (c>='A' && c<='Z'){
      velika_slova[c-'A']++;
    }
    else{
      /* Ako je procitani karakter malo slovo uvecava se broj
    pojavljivanja odgovarajuceg malog slova. */
      if (c>='a' && c<='z'){
        mala_slova[c-'a']++;
      }
      else{
        /* Ako je procitani karakter cifra uvecava se broj
    pojavljivanja odgovarajuce cifre. */
        if(c >='0' && c <= '9'){
          cifre[c-'0']++;
        }
      }
    }
  }

  /* Ispisuju se trazene informacije. */
  ispis(cifre, 10, '0');
  ispis(mala_slova, 26, 'a');
  ispis(velika_slova, 26, 'A');

  return 0;
}
```

Rešenje 3.1.19

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

/* Funkcija za ispis elemenata niza. */
void ispis(int niz[], int n)
{
  int i;

  for(i = 0; i < n; i++)
    printf("%c:%d  ", 'a' + i, niz[i]);
  putchar('\n');
}

int main()
{
  /* Niz u kojem ce se cuvati informacije o broju pojavljivanja malih
     slova. */
```

```
  int mala_slova[26];
  int c, i;

  for(i=0;i<26;i++)
    mala_slova[i]=0;

  /* Ucitavaju se karakteri sve do kraja ulaza. */
  while((c = getchar()) != EOF)
  {
    /* Ako je procitani karakter slovo broj pojavljivanja slova se
    uvecava. Kako se zanemaruje velicina slova, svako slovo se
    pretvori u malo i potom se element na odgovarajucoj poziciji u
    nizu uveca. */
    if (isalpha(c))
      mala_slova[tolower(c)-'a']++;
  }

  /* Ispisuju se trazene informacije. */
  ispis(mala_slova, 26);

  return 0;
}
```

Rešenje 3.1.20

```
#include <stdio.h>

#define MAX 100

int main()
{
  /* Niz karaktera. */
  char karakteri[MAX];
  char c;
  int i, n;


  for(i=0;i<MAX;i++)
  {
    /* Ucitava se karakter po karakter sa standardnog ulaza sve dok
    se ne unese * ili se ne prekoraci maksimalni broj karaktera.s */
    printf("Unesite karakter: ");
    scanf("%c", &c);

    /* Cita se znak za novi red nakon unesenog karaktera. */
    getchar();

    /* Ukoliko je unet karakter * prekida se dalje citanje i izlazi
    se iz petlje. */
    if(c == '*')
      break;
```

```
25
    /* Inace, procitani karakter se smesta u niz. */
27    karakteri[i] = c;
  }
29
  /* Broj unetih karaktera je nakon izlaska iz petlje i-1. */
31  n = i-1;

33  /* Ispisuju se karakteri u obrnutom redosledu. */
  for(i=n;i>=0;i--)
35  {
    printf("%c ", karakteri[i]);
37  }
  printf("\n");
39

41  return 0;
}
```

Rešenje 3.1.21

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

4 #define DIM 200

6 int main()
{
8   int n, i;
  char niz[DIM], blanko;
10
  /* Ucitavanje dimenzije i elemenata niza. */
12  printf("Unesite trenutni broj klijenata: ");
  scanf("%d", &n);
14
  if (n<=0 || n>DIM)
16  {
    printf("Nedozvoljena dimenzija niza.\n");
18    return 0;
  }
20
   /* Nakon unosa dimenzije, korisnik unosi blanko ili znak za novi
    red. Ovaj karakter je potrebno ucitati pre unosa elemenata niza,
    inace ce ovaj karakter biti prvi element niza, sto nije zeljeno
    ponasanje. */
22   scanf("%c", &blanko);

24  printf("Unesite elemete niza: ");
  for(i=0; i<n; i++)
26    scanf("%c", &niz[i]);
```

```
28    /* Provera da li je niz palindrom. Uporedjuju se prvi i poslednji,
         drugi i pretposlednji... odnosno i-ti i ((n-1)-i)-ti element niza
         . Ako nisu jednaki, niz nije palindrom. Kako se istovremeno
         posmatraju dva elementa niza dovoljno je da brojac u petlji ide
         do polovine dimenzije niza.*/
      for(i=0; i<n/2; i++)
30      /* Kako se zanemaruje velicina slova, svaki element niza se
         pretvara u malo slovo i potom se vrsi uporedjivanje. Ako je
         element niza veliko slovo, on ce biti pretvoren u malo; ako nije
         veliko slovo, element nece biti promenjen. Moglo je se i
         drugacije resiti, recimo pretvaranjem svih slova u velika, sa
         funkcijom toupper. */
        if (tolower(niz[i]) != tolower(niz[n-1-i]))
32      {
          printf("Niz nije palindrom.\n");
34        /* Ukoliko niz nije palindrom, ispitivanje se moze prekinuti i
        izaci iz programa, dalje ispitivanje nije potrebno. */
          return 0;
36      }

38    /* U sluacju kada je petlja zavrsena, a nije izaslo iz programa,
         niz jeste palindrom jer uslov nikada nije bio ispunjen. */
      printf("Niz jeste palindrom!\n");
40
      return 0;
42  }
```

Rešenje 3.1.22

```
1   #include <stdio.h>

3   #define MAX 100

5   int main()
    {
7     int brojevi[MAX];
      int n, i, poz_max, poz_min, max, min, tmp;
9
      /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
11    printf("Unesite dimenziju niza: ");
      scanf("%d", &n);
13    if(n<1 || n>MAX)
      {
15      printf("Greska: Nedozvoljena vrednost!\n");
        return -1;
17    }

19    /* Ucitavaju se elementi niza. */
      printf("Unesite elemente niza:\n");
21    for(i=0;i<n;i++)
        scanf("%d", &brojevi[i]);
```

```
23
   /* Maksimalnim tj. minimalnim elementom niza proglasava se nulti
      element niza. Pozicije maksimalnog tj. minimalnog elementa se
      postavljaju na 0. */
25 max = brojevi[0];
   min = brojevi[0];
27 poz_max = 0;
   poz_min = 0;
29
   /* U prolazu kroz niz trazi se maksimalni i minimalni element i
      pamte se njihove pozicije. */
31 for(i=1;i<n;i++)
   {
33   if(brojevi[i] > max)
     {
35     max = brojevi[i];
       poz_max = i;
37   }

39   if(brojevi[i] < min)
     {
41     min = brojevi[i];
       poz_min = i;
43   }
   }
45
   /* Zamenjuju se elementi na pozicijama poz_min i poz_max. */
47 tmp = max;
   brojevi[poz_max] = min;
49 brojevi[poz_min] = tmp;

51 /* Ispisuje se rezultujuci niz. */
   for(i=0;i<n;i++)
53   printf("%d ", brojevi[i]);
   printf("\n");
55


57 return 0;
}
```

Rešenje 3.1.23

```
#include <stdio.h>
2
#define MAX 500
4
/* Ucitavanje dimenzije i elemenata niza. */
6 int ucitavanje(int niz[])
{
8   int i, n;
```

```
10    printf("Unesite dimenziju niza: ");
      scanf("%d", &n);
12
      if (n<=0 || n>DIM)
14    {
        printf("Nedozvoljena dimenzija niza.\n");
16      exit(EXIT_FAILURE);
      }
18
      printf("Unesite elemente niza: ");
20    for(i=0; i<n; i++)
        scanf("%d", &niz[i]);
22
      return n;
24  }

26  /* Funkcija koja vraca 1 ukoliko broj x postoji u nizu, 0 inace. */
    int postoji(int niz[], int n, int x)
28  {
      int i;
30
      for(i=0;i<n;i++)
32      if(niz[i] == x)
          return 1;
34
      return 0;
36  }

38  int main()
    {
40    int a[MAX], b[MAX], unija[2*MAX], presek[MAX], razlika[MAX];
      int i, j, n_a, n_b, n_u, n_p, n_r, indikator;
42
      /* Unose se dva niza. */
44    n_a = ucitavanje(a);
      n_b = ucitavanje(b);
46
      /* Brojaci elemenata u nizovima unija, presek i razlika. */
48    n_u = 0;
      n_p = 0;
50    n_r = 0;

52    for(i=0;i<n_a;i++)
      {
54      /*  Ukoliko se element a[i] ne nalazi u uniji, dodaje se u uniju
        i povecaca se brojac elemenata u nizu unija. */
        if(postoji(unija,n_u,a[i]) == 0)
56      {
          unija[n_u] = a[i];
58        n_u++;
        }
60
```

```
    /* Ukoliko se element a[i] nalazi u nizu b i ne postoji u nizu
    presek, dodaje se presek i povecava se brojac elemenata u nizu
    presek. */
    if(postoji(b, n_b, a[i])==1 && postoji(presek, n_p, a[i])==0)
    {
      presek[n_p] = a[i];
      n_p++;
    }

    /* Ukoliko element a[i] ne postoji u nizu b i ne postoji u nizu
    razlika, dodaje se u razliku i povecava se brojac elemenata u
    nizu razlika. */
    if(postoji(b, n_b, a[i])==0 && postoji(razlika, n_r, a[i])==0)
    {
      razlika[n_r] = a[i];
      n_r++;
    }
  }

  /* Elemente niza b koji nisu uneti u uniju dodaju se u uniju. */
  for(i=0;i<n_b;i++)
    if(postoji(unija, n_u, b[i]))
    {
      unija[n_u] = b[i];
      n_u++;
    }

  printf("Unija: ");
  for(i=0;i<n_u;i++)
    printf("%d ", unija[i]);

  printf("\nPresek: ");
  for(i=0;i<n_p;i++)
    printf("%d ", presek[i]);

  printf("\nRazlika: ");
  for(i=0;i<n_r;i++)
    printf("%d ", razlika[i]);

  return 0;
}
```

Rešenje 3.1.24

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX];
```

```
8   int b[MAX];

10  /* Rezultujuci niz ima najvise 2*MAX elemenata. */
    int c[2*MAX];
12
    int n;
14  int i,j;


16
    /* Ucitava se dimenzija nizova i proverava njena ispravnost. */
18  printf("Unesite dimenziju nizova:\n");
    scanf("%d", &n);
20  if (n<1 || n>MAX)
    {
22    printf("Nedozvoljena vrednost!\n");
      return -1;
24  }

26  /* Ucitavaju se elementi prvog niza. */
    printf("Unesite elemente niza a:\n");
28  for(i=0;i<n;i++)
    {
30    scanf("%d", &a[i]);
    }
32
    /* Ucitavaju se elementi drugog niza. */
34  printf("Unesite elemente niza b:\n");
    for(i=0;i<n;i++)
36  {
      scanf("%d", &b[i]);
38  }

40  /* Formira se treci niz.
       Koriste se dva indeksa:
42
        - indeks i pomocu kojeg se pristupa elementima nizova a i b i
      koji treba uvecati za 1 nakon svake iteracije
44
        - 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. */
46  for(i=0,j=0;i<n;i++,j+=2)
    {
48    c[j]=a[i];
      c[j+1]=b[i];
50  }

52  /* Ispisuju se elementi rezultujuceg niza. */
    printf("Rezultujuci niz:\n");
54  for(i=0;i<2*n;i++)
       printf("%d ",c[i]);
```

```
56    printf("\n");

58    return 0;
}
```

Rešenje 3.1.25

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX], b[MAX], c[2*MAX];
  int i, n;

  /* Ucitava se broj elemenata nizova i proverava se njegova
     ispravnost. */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<1 || n>MAX)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi nizova */
  printf("Unesite elemente niza a: ");
  for(i=0;i<n;i++)
    scanf("%d", &a[i]);

  printf("Unesite elemente niza b: ");
  for(i=0;i<n;i++)
    scanf("%d", &b[i]);

  /* Niz c ima 2*n elemenata: prvih n elemenata su elementi niza b, a
     narednih n elemenata elementi niza a. Elementi niza b se nalaze
     na pozicijama 0,1,2,...n-1, a elementi niza a na pozicijama n,n
     +1,...2*n-1. Jednim prolaskom kroz petlju na poziciju i u nizu c
     se postavlja element b[i] niza b, a na poziciju n+i element a[i]
     niza a. */
  for(i=0;i<n;i++)
  {
    c[i] = b[i];
    c[n+i] = a[i];
  }

  /* Ispisuju se elementi niza c. */
  for(i=0;i<2*n;i++)
    printf("%d ", c[i]);
  printf("\n");
```

```
39
   return 0;
41 }
```

Rešenje 3.1.26

```
1  #include <stdio.h>

3  #define DIMENZIJA 100

5  /* Funkcija za ucitavanje niza. */
   void unos(int niz[], int n)
7  {
     int i;
9
     printf("Uneti elemente sortiranog niza:\n");
11   for(i=0; i<n; i++)
       scanf("%d", &niz[i]);
13 }

15 /* Funkcija za ispis niza. */
   void ispis(int niz[], int n)
17 {
     int i;
19
     for(i=0; i<n; i++)
21     printf("%d ", niz[i]);

23   printf("\n");
   }
25
   int main()
27 {
     int a[DIMENZIJA], b[DIMENZIJA];
29   /* Niz c moze biti duplo veci od nizova a i b, pa za njega treba
       odvojiti dupli prostor. */
     int c[2*DIMENZIJA];
31   int m, n;
       /* Brojac u petlji za elemente niza a. */
33   int i = 0;
       /* Brojac u petlji za elemente niza b. */
35   int j = 0;
     /* Brojac u petlji za elemente niza c. */
37   int k = 0;

39   printf("Uneti broj elemenata niza: ");
     scanf("%d", &n);
41
     unos(a, n);
43   unos(b, n);
```

```
45    while(i<n && j<n)
      {
47      /* Porede se elementi nizova a i b i u niz c upisuje se samo onaj
         koji je manji. */
          if (a[i] < b[j])
49      {
          c[k] = a[i];
51        /* Kako element niza a je upisan u niz c, uvecava se brojac
        niza a. Element niza b nije upisan u niz c, te brojac za niz b ne
         treba uvecavati. */
          i++;
53      }
        else
55      {
          c[k] = b[j];
57        j++;
        }
59
        /* U nizu c na poziciju k je upisan ili a[i] ili b[j]. Brojac k
        se uvecava. */
61      k++;
      }
63
      /* Ukoliko je ostalo elemenata u nizu a, upisuju se u niz c. */
65    while(i < n)
      {
67      c[k] = a[i];
        k++;
69      i++;
      }
71
        /* Ukoliko je ostalo elemenata u nizu b, upisuju se u niz c. */
73    while(j < n)
      {
75      c[k] = b[j];
        k++;
77      j++;
      }
79
      /* Ispis elemenata niza c cija dimenzija je zbir dimenzija nizova a
         i b. */
81    ispis(c, 2*n);

83    return 0;
}
```

Rešenje 3.1.27

```
1  #include <stdio.h>

3  #define DIMENZIJA 10
```

```c

/* Funkcija za ucitavanje niza. */
void unos(int niz[])
{
  int i;

  printf("Unesite %d brojeva:\n", DIMENZIJA);
  for(i=0; i<DIMENZIJA; i++)
    scanf("%d", &niz[i]);
}

/* Funkcija za ispis niza. */
void ispis(int niz[], int n)
{
  int i;

  for(i=0; i<DIMENZIJA; i++)
    printf("%d ", niz[i]);

  printf("\n");
}

int main()
{
  int niz[DIMENZIJA];
  /* Brojaci i i j. Brojac i krece od pocetka niza, a brojac j od
    kraja. */
  int i=0, j = DIMENZIJA-1;
  /* Pomocna promenljiva za razmenu elemenata niza. */
  int pom;

  unos(niz);

    /* Ideja u resenju je da se krene sa dva kraja niza -- sa pocetka
     niza i sa kraja i svaki put kada se naidje na elemente koji po
    parnosti ne odgovaraju delu niza u kome treba da budu, ti
    elementi se zamene. */
  while(i < j && i < DIMENZIJA && j >= 0)
  {
    /* Ukoliko elementi na pozicijama i i j su razlicite parnosti,
    vrsi se razmena tih elemenata niza. */
    if (niz[i] % 2 != 0 && niz[j] % 2 == 0)
    {
      pom = niz[i];
      niz[i] = niz[j];
      niz[j] = pom;
    }

    /* Ukoliko je element na poziciji i paran, prelazi se na sledeci
    element niza, brojac i se uvecava. */
    if (niz[i] % 2 == 0)
      i++;
```

```
    /* Ukoliko je element na poziciji j neparan, prelazi se na
    sledeci element niza, brojac j se smanjuje. */
    if (niz[j] % 2 != 0)
      j--;
  }

  printf("Rezultujuci niz:\n");
  ispis(niz, DIMENZIJA);

  return 0;
}
```

Rešenje 3.1.28

```
#include<stdio.h>

#define MAX 100

/* Funkcija kojom se ucitavaju elementi niza a dimenzije n. */
void ucitaj(int a[], int n)
{
   int i;
   for(i=0;i<n;i++)
   {
        scanf("%d",&a[i]);
   }
}

/* Funkcija kojom se ispisuju elementi niza a dimenzije n. */
void stampaj(int a[], int n)
{
   int i;
   for(i=0;i<n;i++)
        printf("%d ",a[i]);
   printf("\n");
}



/* Funkcija koja obrce elemente niza. */
void obrni(int a[], int n)
{

   int t;
   int 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. */

36    for(i=0,j=n-1;i<j;i++, j--)
      {
38          t = a[i];
            a[i] = a[j];
40          a[j] = t;
      }
42
}
44
/* Funkcija koja rotira niz ciklicno za jedno mesto u levo. */
46 void rotiraj1(int a[], int n)
{
48    int i;
      int tmp;
50
      /* Izdvaja se prvi element niza. */
52    tmp=a[0];

54    /* Pomeraju se preostali elementi niza. */
      for(i=0;i<n-1;i++){
56          a[i]=a[i+1];
      }
58
      /* Poslednjem elementu se dodeljuje sacuvana vrednost prvog
       elementa. */
60    a[n-1] = tmp;
}
62
/* Funkcija koja rotira niz ciklicno za k mesta u levo. */
64 void rotirajk(int a[], int n, int k)
{
66    int i;

68    /* Odredjuje se vrednost broja k koja je u opsegu od 0 do n-1 kako
        bi se izbegla suvisna pomeranja. */
      k=k%n;
70
      /* Niz se rotira za jednu poziciju ulevo k puta. */
72    for(i=0;i<k;i++)
            rotiraj1(a,n);
74 }

76 int main()
{
78   int a[MAX];
     int n;
80   int i;
     int k;
82   int m;
```

```
84
   /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
86 printf("Unesite dimenziju niza:");
   scanf("%d",&n);
88 if (n<1 || n>MAX)
   {
90       printf("Nedozvoljena vrednost!\n");
         return -1;
92 }

94 /* Ucitavaju se elementi niza. */
   ucitaj(a,n);
96
   /* Testira se rad napisanih funkcija */
98
   /* Obrtanje niza. */
100 printf("Elementi niza nakon obrtanja:\n");
   obrni(a,n);
102 stampaj(a,n);

104 /* Rotiranje za jedno mesto u levo. */
   printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
106 rotiraj1(a,n);
   stampaj(a,n);
108
   /* Rotiranje za k mesta u levo. */
110 printf("Unesite jedan pozitivan ceo broj:");
   scanf("%d",&k);
112 if (k<=0)
   {
114       printf("Nekorektan unos\n");
         return -1;
116 }
   rotirajk(a,n,k);
118 printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n",k);
   stampaj(a,n);
120

122 return 0;
}
```

Rešenje 3.1.29

```
1 #include <stdio.h>

3 #define DIM 2000

5 /* Funkcija za ispis elemenata niza. */
  void ispis(int niz[], int n)
7 {
```

```
  int i;

  for(i=0; i<n; i++)
    printf("%d ", niz[i]);
  printf("\n");
}

/* Funkcija za ubacivanje na kraj niza. Vraca novu dimenziju niza. */
int ubaci_na_kraj(int niz[], int n, int x)
{
  niz[n] = x;
  return n+1;
}

/* Funkcija za ubacivanje na pocetak niza. Vraca novu dimenziju niza.
     */
int ubaci_na_pocetak(int niz[], int n, int x)
{
  int i;

  /* Prvo se svi elementi niza pomere za jednu poziciju u desno da bi
     se oslobodio prostor za prvi element niza. Poslednji element
    niza se pomera sa pozicije (n-1) na poziciju (n). Slicno se
    pomeraju i ostali elementi. */
  for(i=n; i>0; i--)
    niz[i] = niz[i-1];

  /* Na prvu poziciju se upisuje novi element. Bitan je redosled
    naredbi: ako bi prvo bio upisan novi element, a tek onda izvrseno
     pomeranje, element na poziciji niz[0] bi bio obrisan i ne bi
    mogao biti upisan na poziciju niz[1]. */
  niz[0] = x;

  return n+1;
}


/* Funkcija za ubacivanje elementa na neku poziciju u nizu. Vraca
    novu dimenziju niza. */
int ubaci_na_poziciju(int niz[], int n, int x, int pozicija)
{
  int i;

  /* Prvo se svi elementi niza od pozicije do kraja pomere za jedno
    mesto u desno da bi se oslobodio prostor za novi element niza.
    */
  for(i=n; i>pozicija; i--)
    niz[i] = niz[i-1];

  /* Na poziciju se upisuje novi element. */
  niz[pozicija] = x;

```

```
  return n+1;
}

/* Funkcija za brisanje prvog elementa niza. Vraca novu dimenziju
    niza. */
int brisi_prvog(int niz[], int n)
{
  int i;

   /* Svi elementi niza pomeraju se za jedno mesto u levo. */
  for(i=0; i<n-1; i++)
    niz[i] = niz[i+1];

  return n-1;
}

/* Funkcija za brisanje poslednjeg elementa niza. Vraca novu
    dimenziju niza. */
int brisi_poslednjeg(int niz[], int n)
{
  /* Dovoljno je smanjiti dimenziju niza, elemente niza nije potrebno
     brisati. */
  return n-1;
}

/* Funkcija za brisanje elementa niza. Pretpostavlja se da element
    ima samo jedno pojavljivanje (za vezbu napisati funkciju koja
    brise sva pojavljivanja, ako ih ima vise). Vraca novu dimenziju
    niza. */
int brisi_element(int niz[], int n, int x)
{
  int i, j;

   /* Prvo treba pronaci poziciju elementa u nizu. */
  for(i=0; i<n; i++)
    if (niz[i] == x)
      break;

  /* Provera da li element postoji u nizu. Ako je brojac stigao do
    kraja niza, onda element ne postoji u nizu. */
  if (i == n)
  {
    printf("Klijent sa rednim brojem %d ne postoji u nizu.\n", x);
    return n;
  }

  /* Ukoliko element postoji u nizu, svi elementi niza nakon njega se
     pomeraju za jedno mesto u levo. */
  for(j = i; j<n-1; j++)
    niz[j] = niz[j+1];

  return n-1;
```

```
}

int main()
{
  int n, niz[DIM], i, klijent, pozicija;

  /* Ucitavanje dimenzije i elemenata niza. */
  printf("Unesite trenutni broj klijenata: ");
  scanf("%d", &n);

  if (n<=0 || n>DIM)
  {
    printf("Nedozvoljena dimenzija niza.\n");
    return 0;
  }

  printf("Unesite niz sa rednim brojevima klijenata: ");
  for(i=0; i<n; i++)
    scanf("%d", &niz[i]);


   /* Ubacivanje klijenta na kraj. */
  printf("Unesite klijenta kojeg treba ubaciti u niz: ");
  scanf("%d", &klijent);
  n = ubaci_na_kraj(niz, n, klijent);
  printf("Niz nakon ubacivanja klijenta:\n");
  ispis(niz, n);

  /* Ubacivanje klijenta na pocetak. */
   printf("Unesite prioritetnog klijenta kojeg treba ubaciti u niz: "
    );
  scanf("%d", &klijent);
  n = ubaci_na_pocetak(niz, n, klijent);
  printf("Niz nakon ubacivanja klijenta:\n");
  ispis(niz, n);

  /* Ubacivanje klijenta na zadatu poziciju. */
   printf("Unesite prioritetnog klijenta kojeg treba ubaciti u niz i
    njegovu poziciju:");
  scanf("%d%d", &klijent, &pozicija);
   if (pozicija < 0 || pozicija > n)
  {
    printf("Neispravna pozicija.\n");
  }
  else
   {
    n = ubaci_na_poziciju(niz, n, klijent, pozicija);
    printf("Niz nakon ubacivanja klijenta:\n");
    ispis(niz, n);
  }

  /* Brisanje prvog klijenta. */
```

```
    n = brisi_prvog(niz, n);
145   printf("Niz nakon odlaska klijenta:\n");
    ispis(niz, n);
147
    /* Brisanje poslednjeg klijenta. */
149   n = brisi_poslednjeg(niz, n);
    printf("Niz nakon odlaska klijenta:\n");
151   ispis(niz, n);

153   /* Brisanje klijenta sa datim rednim brojem. */
    printf("Unesite redni broj klijenta koji je napustio red: ");
155   scanf("%d", &klijent);
    n = brisi_element(niz, n, klijent);
157   printf("Niz nakon odlaska klijenta:\n");
    ispis(niz, n);
159
    return 0;
161 }
```

Rešenje 3.1.30

```
#include <stdio.h>
2
#define MAX 100
4
int main()
6 {
  int a[MAX];
8   int i, j, n_a;

10   /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
       . */
    printf("Unesite broj elemenata niza: ");
12   scanf("%d", &n_a);
    if(n_a<1 || n_a>100)
14   {
      printf("Greska: Nedozvoljena vrednost!\n");
16     return -1;
    }
18
    /* Ucitavaju se elementi niza. */
20   printf("Unesite elemente niza: ");
    for(i=0;i<n_a;i++)
22     scanf("%d", &a[i]);

24   /* Parametar 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_a;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. */
  }

  /* U nizu a se sada na pozicijama od 0,...,j-1 nalaze elementi koji
     su parni, te je njegova nova dimenzija j. */
  n_a=j;

  /* Ispisuju se elementi modifikovanog niza a. */
  for(i=0;i<n_a;i++){
    printf("%d ", a[i]);
  }
  printf("\n");

  return 0;
}
```

Rešenje 3.1.31

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX];
  int n;
  int i,j;
  char poslednja_cifra;
  int novo_n;

  /* Ucitava se dimenzija niza i proverava njena ispravnost. */
  printf("Unesite dimenziju niza:\n");
  scanf("%d", &n);
  if (n<1 || n>MAX)
  {
    printf("Nedozvoljena vrednost!\n");
    return -1;
  }
```

```
22    /* Ucitavaju se elementi niza a. */
      printf("Unesite elemente niza a:\n");
24    for(i=0;i<n;i++)
        scanf("%d", &a[i]);
26
      /* Obilaze se svi elementi niza a. */
28    for(i=0, j=0; i<n; i++)
      {
30       /* Izdvaja se poslednja cifra tekuceg elementa. */
         poslednja_cifra = a[i]%10;
32
         /* Ako je poslednja cifra 0 ili je element deljiv svojom
        poslednjom cifrom, taj element se zadrzava i smesta na poziciju j
        . */
34       if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
         {
36          a[j]=a[i];
            j++;
38       }
      }
40
      /* Dimenzija novog niza odgovara posledjoj vrednosti brojaca j. */
42    novo_n=j;

44    /* Ispisuje se rezultujuci niz. */
      printf("Niz a nakon izmena:\n");
46    for(i=0; i<novo_n;i++)
        printf("%d ", a[i]);
48    printf("\n");

50    return 0;
}
```

Rešenje 3.1.32

```
      #include <stdio.h>
2
      #define DIM 700
4
      /* 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. */
6     void pomeri_za_jedno_mesto(int a[], int n, int j)
      {
8       int i;

10      for(i=j; i<n; i++)
          a[i] = a[i+1];
12    }
```

```c
int main()
{
  int n, niz[DIM], i;

  /* Ucitavanje dimenzije i elemenata niza. */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n);

  if (n<=0 || n>DIM)
  {
    printf("Nedozvoljena dimenzija niza.\n");
    return 0;
  }

  printf("Unesite elemente niza: ");
  for(i=0; i<n; i++)
    scanf("%d", &niz[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--;
    }


  /* Stampanje novog niza. */
  printf("Novi niz:\n");
  for(i=0; i<n; i++)
    printf("%d ", niz[i]);
  printf("\n");

  return 0;
}
```

Rešenje 3.1.33

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

```c
#define MAX 100

/* Funkcija koja proverava da li je zadati broj prost broj. Povratna
    vrednost funkcije je 1 ukoliko broj jeste prost, inace je 0. */
int prost(int x)
{
  int i;

  /* Posmatra se apsolutna vrednost broja kako bi se pokrio i slucaj
    negativnih brojeva. */
  x=abs(x);

  /* Brojevi 2 i 3 su prosti. */
  if(x == 2 || x == 3)
    return 1;

  /* Ako je broj paran nije prost. */
  if(x%2 == 0)
    return 0;

  /* Ako broj ima delioce u skupu [3, koren_broja(x)] takodje nije
    prost. */
  for(i=3;i<=sqrt(x);i+=2){
    if(x%i == 0)
      return 0;
  }

  /* Ako su svi uslovi ispunjeni, broj je prost. */
  return 1;
}

int main()
{
  int a[MAX];
  int i, j, n_a, n_b;
  /* Rezultujuci niz. */
  int b[MAX];


  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
    . */
  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n_a);
  if(n_a<1 || n_a>MAX)
  {
    printf("Greska: Nedozvoljena vrednost!\n");
    return -1;
  }

  /* Ucitavaju se elementi niza a. */
  printf("Unesite elemente niza: ");
  for(i=0;i<n_a;i++)
```

```
52      scanf("%d", &a[i]);

54    /* 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_a;i++){
56      if(prost(a[i]) == 0)
        {
58        b[j] = a[i];
          j++;
60      }
      }
62
      /* Broj elemenata novodobijenog niza b je j. */
64    n_b = j;

66    /* Ispisuju se elementi niza b. */
      for(i=0;i<n_b;i++)
68      printf("%d ", b[i]);
      printf("\n");
70
      return 0;
72  }
```

Rešenje 3.1.34

```
#include <stdio.h>
2
#define DIM 300
4
int main()
6 {
    int n, niz[DIM], i;
8
    /* Ucitavanje dimenzije i elemenata niza. */
10  printf("Unesite dimenziju niza: ");
    scanf("%d", &n);
12
    if (n<=0 || n>DIM)
14  {
      printf("Nedozvoljena dimenzija niza.\n");
16    return 0;
    }
18
    printf("Unesite elemente niza: ");
20  for(i=0; i<n; i++)
      scanf("%d", &niz[i]);
22

24  /* Provera da li su elementi niza uredjeni neopadajuce. */
    for(i=0; i<n-1; i++)
```

```
26      /* Porede se svaka dva uzastopna elementa niza. Ukoliko za bilo
        koja takva dva elementa uslov nije ispunjen, prekida se dalje
        ispitivanje i vraca se poruka da niz nije uredjen neopadajuce. U
        suprotnom se nakon izlaska iz petlje ispisuje poruka da je niz
        uredjen neopadajuce. */
        if (niz[i] > niz[i+1])
28      {
          printf("Nije uredjen neopadajuce.\n");
30        return 0;
        }
32
    printf("Jeste uredjen neopadajuce!\n");
34
    return 0;
36  }
```

Rešenje 3.1.35

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    /* 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. */
5   #define DIM 32

7   /* Ucitavanje dimenzije i elemenata niza. */
    int ucitavanje(int niz[])
9   {
      int i, n;
11
      printf("Unesite dimenziju niza: ");
13    scanf("%d", &n);

15    if (n<=0 || n>DIM)
      {
17      printf("Nedozvoljena dimenzija niza.\n");
        exit(EXIT_FAILURE);
19    }

21    printf("Unesite broj prodatih artikala: ");
      for(i=0; i<n; i++)
23      scanf("%d", &niz[i]);

25    return n;
    }
27
    int najduzi_neopadajuci(int a[], int n)
29  {
      int i;
31    /* Parametar u kome se cuva duzina serije. Na pocetku je duzina
        serije jednaka 1, odnosno, samo jedan element je u seriji.
```

```
    Kasnije, u petlji se ovaj broj moze uvecati. */
  int ts = 1;
  /* Parametar ns predstavlja duzinu najduze serije. Na pocetku se
    postavlja na 1, odnosno pretpostavlja se da je duzina najduze
    serije 1, a u petlji se ova vrednost moze izmeniti. */
  int ns = 1;

  for (i = 1; i < n; i++)
  {
    /* Proverava se da li uzastopni elementi ispunjavaju neopadajuci
    uslov. Ako je to slucaj uvecava se duzina serije. */
    if (a[i] >= a[i-1])
      ts++;
    else
    /* 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. */
      ts = 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 (ts > ns)
      ns = ts;
  }

  return ns;
}

int main()
{
  int n, a[DIM];
  int i;

  n = ucitavanje(a);

  printf("Duzina najduzeg neopadajuceg prodavanja je %d.\n",
    najduzi_neopadajuci(a, n));

  return 0;
}
```

Rešenje 3.1.36

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

#define DIM 100

/* Ucitavanje dimenzije i elemenata niza. */
int ucitavanje(int niz[])
```

```
{
  int i, n;

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

  if (n<=0 || n>DIM)
  {
    printf("Nedozvoljena dimenzija niza.\n");
    exit(EXIT_FAILURE);
  }

  printf("Unesite elemente niza: ");
  for(i=0; i<n; i++)
    scanf("%d", &niz[i]);

  return n;
}

int najduza_serija(int a[], int n)
{
  int i;
  /* Parametar u kome se cuva duzina serije. Na pocetku je duzina
     serije jednaka 1, odnosno, samo jedan element je u seriji.
     Kasnije, u petlji se ovaj broj moze uvecati. */
  int ts = 1;
  /* Parametar ns predstavlja duzinu najduze serije. Na pocetku se
     postavlja na 1, odnosno pretpostavlja se da je duzina najduze
     serije 1, a u petlji se ova vrednost moze izmeniti. */
  int ns = 1;

  for (i = 1; i < n; i++)
  {
    /* Proverava se da li su uzastopni elementi jednaki. Ako je to
    slucaj uvecava se duzina serije. */
    if (a[i] == a[i-1])
      ts++;
    else
    /* 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. */
      ts = 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 (ts > ns)
      ns = ts;
  }

  return ns;
}
```

```
51
int main()
53 {
   int n, a[DIM];
55   int i;

57   n = ucitavanje(a);

59   printf("Duzina najduze serije je %d.\n", najduza_serija(a, n));

61   return 0;
}
```

Rešenje 3.1.37

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

4 #define DIM 100

6 /* Ucitavanje dimenzije i elemenata niza. */
int ucitavanje(int niz[])
8 {
   int i, n;
10
   printf("Unesite dimenziju niza: ");
12   scanf("%d", &n);

14   if (n<=0 || n>DIM)
   {
16     printf("Nedozvoljena dimenzija niza.\n");
     exit(EXIT_FAILURE);
18   }

20   printf("Unesite elemente niza: ");
   for(i=0; i<n; i++)
22     scanf("%d", &niz[i]);

24   return n;
}
26
/* Resenje pod a. */
28 int podniz_uzastopnih(int a[], int n, int b[], int m)
{
30   int i, j;

32   /* 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++)
34   {
```

```
    /* Prolaze se elementi drugog niza. Za svaki element niza b
    proverava se da li je jednak odgovarajucem elementu niza a.
36        Za niz a razmatra se da li podniz pocinje od pozicije i.
    Tako 0-ti element niza b je na poziciji i, 1-vi element je na
    poziciji i+1, 2-gi 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++)
38    if (a[i + j] != b[j])
        break;
40  /* 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)
42    return 1;
  }
44
  /* 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. */
46 return 0;
}
48
/* Resenje pod b. */
50 int podniz(int a[], int n, int b[], int m)
{
52 int i, j;

54 /* Petljom se prolaze elementi niza a. */
  for (i = 0, j = 0; i < n && j < m; i++)
56  /* 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])
58    j++;

60 /* 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;
62 }

64 int main()
{
66 int n, a[DIM];
  int m, b[DIM];
68 int i;

70 n = ucitavanje(a);
  m = ucitavanje(b);
72
```

```
  if (podniz_uzastopnih(a, n, b, m))
74    printf("Elementi drugog niza cine uzastopni podniz prvog niza.\n"
      );
  else
76    printf("Elementi drugog niza ne cine uzastopni podniz prvog niza
      .\n");

78

  if (podniz(a, n, b, m))
80    printf("Elementi drugog niza cine podniz prvog niza.\n");
  else
82    printf("Elementi drugog niza ne cine podniz prvog niza.\n");

84  return 0;
}
```

Rešenje 3.1.38

```
1 #include <stdio.h>

3 #define DIM 100

5 void brojanje(int a[], int b[], int n)
{
7   int i;

9   /* Niz b se inicijalizuje nulama jer se za svaki element postavi da
       se poljavuljuje 0 puta u nizu a. */
    for(i=1; i<=n; i++)
11    b[i] = 0;

13   /* Peljom se prolazi kroz niz a i za svaki element a[i] uvecava se
       broj njegovog pojavljivanja u nizu b.
       Na primer, ako je a[3] = 7, onda treba uvecati broj
     pojavljivanja broja 7, a to je b[7]++, sto se
15     krace moze zapisati kao b[a[3]]++.
       Pretpostavlja se da je niz a dobro zadat, odnosno da su sve
     njegove vrednosti u intervalu od 1 do n.
17   */
   for(i=0; i<n; i++)
19    b[a[i]]++;
}
21
int main()
23 {
   int a[DIM];
25   /* Niz b moze imati index DIM (jer niz b se posmatra od 1 do DIM),
       pa zato njegova dimenzija mora biti za jedan veca. */
    int b[DIM+1];
27  int i, n;
```

```
29    /* Unos dimenzije i elemenata niza. */
      printf("Unesite broj elemenata niza: ");
31    scanf("%d", &n);

33    if (n <=0 || n > DIM)
      {
35      printf("Neispravna dimenzija niza.\n");
        return -1;
37    }

39    printf("Unesite elemente niza:\n");
      for(i=0; i<n; i++)
41    {
        scanf("%d", &a[i]);
43      /* Niz a moze sadrzati elemente koji nisu u opsegu od 1 do n. U
        tom slucaju taj niz nije permutacija. */
        if (a[i] <=0 || a[i] > n)
45      {
          printf("Uneti niz nije permutacija.\n");
47        return 0;
        }
49    }

51    brojanje(a, b, n);

53    /* Ukoliko se svaki element niza a javlja tacno jednom u nizu a,
        onda niz a jeste permutacija. Ovo svojstvo se proverava
        koriscenjem dobijenog niza b. */
      for(i=1; i<=n; i++)
55      if (b[i] != 1)
        {
57        printf("Uneti niz nije permutacija.\n");
          return 0;
59      }

61    /* Ukoliko prethodna petlja nije ranije zavrsena (i nije se izaslo
        iz programa), onda svaki element niza b je jednak 1, odnosno
        svaki element se pojavljuje tacno jednom, pa niz a jeste
        permutacija. */
      printf("Uneti niz je permutacija.\n");
63
      return 0;
65  }
```

Rešenje 3.1.39

```
1   #include <stdio.h>

3   #define BROJ_CIFARA 10

5   /* Analiziraju se cifre broja i smestaju u odgovarajuci niz. */
```

```
void analiza_cifara(int broj, int niz[])
{
  int c;

  /* Niz predstavlja brojace za cifre broja.
     Na pocetku se ovi nizovi inicijalizuju nulama. */
  for(i=0;i<BROJ_CIFARA;i++)
      niz[i] = 0;

  do
  {
    c = broj%10;
    niz[c]++;
    broj /= 10;
  }
  while(broj);
}

int main()
{
  char c;
  /* Niz cifrex predstavlja brojace za cifre broja x.
     Niz cifrey predstavlja brojace za cifre broja y. */
  int cifrex[BROJ_CIFARA], cifrey[BROJ_CIFARA];
  int x, y, i, indikator;

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

  /* Za slucaj da su unete vrednosti negativne, posmatra se njihova
    apsolutna vrednost. Ovo je opravdano iz razloga sto se brojevi x
    i -x zapisuju istim ciframa. */
  x=abs(x);
  y=abs(y);

  analiza_cifara(x, cifrex);
  analiza_cifara(y, cifrey);


  /* Promenljiva indikator sluzi za pracenje da li su oba broja
    sastavljena od istih cifara. */
  indikator = 1;

  for(i=0;i<BROJ_CIFARA;i++){
    /* Ako se broj pojavljivanja cifre i u zapisu broja x razlikuje
    od broja pojavljivanja cifre i u zapisu broja y, brojevi se ne
    zapisuju istim ciframa. Zato se vrednost indikatora moze
    postaviti na 0 i prekinuti dalje uporedjivanje broja
    pojavljivanja. */
    if(cifrey[i] != cifrex[i])
    {
```

```
51        indikator = 0;
          break;
53      }
    }
55
      /* Ako je vrednost promenljive indikator ostala 1, to znaci da u
      petlji nije pronadjena cifra koja se ne pojavljuje isti broj puta
       u zapisima brojeva x i y. Zato se moze zakljuciti da se brojevi
      zapisuju istim ciframa. */
57    if(indikator)
        printf("Brojevi se zapisuju istim ciframa!\n");
59    else
        printf("Brojevi se ne zapisuju istim ciframa!\n");
61
63    return 0;
}
```

3.3 Pokazivači

Zadatak 3.3.1 Napisati funkciju koja uređuje svoja dva celobrojna argumenta tako da se u prvom nalazi manji, a u drugom veći. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite vrednosti promenljivih x i y:  2 5
 Uredjene promenljive: x=2, y=5
```

Primer 2

```
Interakcija sa programom:
 Unesite vrednosti promenljivih x i y:  11 -4
 Uredjene promenljive: x=-4, y=11
```

Zadatak 3.3.2 Napisati funkciju koja za boju datu u *rgb* formatu računa *cmy* format po formulama:
$c = 1 - (r/255)$
$m = 1 - (g/255)$
$y = 1 - (b/255)$
Napisati program koji učitava tri cela broja broja (*rgb* format) i ispisuje rezultat poziva funkcije (*cmy* format). Napomena: *Vrednosti boja u rgb formatu su u opsegu* $[0, 255]$.

Primer 1

```
Interakcija sa programom:
 Unesite boju u rgb formatu:  56 111 24
 c=0.78, m=0.56, y=0.91
```

Primer 2

```
Interakcija sa programom:
 Unesite boju u rgb formatu:  156 -90 5
 Nekorektan unos.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite boju u rgb formatu:  9 0 237
 c=0.96, m=1.00, y=0.07
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Unesite boju u rgb formatu:  300 11 27
 Nekorektan unos.
```

Zadatak 3.3.3 Napisati funkciju koja za dve prave date svojim koeficijentima pravca i slobodnim članovima određuje njihovu tačku preseka. Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju tačku preseka (ako su paralelne). Napisati program koji učitava podatke o pravama, poziva napisanu funkciju i 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.
```

Zadatak 3.3.4 Napisati funkciju koja za dva cela broja izračunava njihov količnik i ostatak pri deljenju. Funkcija treba da vrati vrednost 1 ukoliko je uspešno izračunala vrednosti i 0 ukoliko deljenje nije moguće. Napisati program koji testira rad ove funkcije.

Zadatak 3.3.5 Napisati funkciju koja za dinarsku vrednost cene artikla izračunava odgovarajuću cenu u evrima i u dolarima. Napisati program koji testira rad ove funkcije.

Zadatak 3.3.6 Napisati funkciju koja za dužinu trajanja filma kaja je data u sekundama, određuje ukupno trajanje filma u satima, minutama i sekundama. Napisati program koji testira rad ove funkcije.

Zadatak 3.3.7 Napisati funkciju `void modifikacija(char* s, char* t, int* br_modifikacija)` koja na osnovu niske s formira nisku t tako što svako malo slovo zamanjuje velikim. Broj izvršenih modifikacija se čuva u okviru argumenta $br\_modifikacija$. Pretpostaviti da niska s neće biti duža od 20 karaktera. Napisati program koji testira rad napisane funkcije.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:  123abc789XY
 Modifikovana niska je: 123ABC789XY
 Broj modifikacija je: 3
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:  zimA
 Modifikovana niska je: ZIMA
 Broj modifikacija je: 3
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: SNEG
 Modifikovana niska je: SNEG
 Broj modifikacija je: 0
```

Primer 4

```
Interakcija sa programom:
 Unesite nisku: 1234
 Modifikovana niska je: 1234
 Broj modifikacija je: 0
```

Zadatak 3.3.8 Napisati funkciju `void interpunkcija(int* br_tacaka, int* br_zareza)` koja prebrojava tačke i zareze u tekstu koji se unosi sa standardnog ulaza. Napisati program koji testira napisanu funkciju.

Primer 1

```
Interakcija sa programom:
 Unesite tekst:
 a.b.c.d
 a,b,,c,d,e
 Broj tacaka: 3
 Broj zareza: 5
```

Primer 2

```
Interakcija sa programom:
 Unesite tekst:
 .....789.....
 Broj tacaka: 10
 Broj zareza: 0
```

Primer 3

```
Interakcija sa programom:
 Unesite tekst:
 sunce
 Broj tacaka: 0
 Broj zareza: 0
```

Zadatak 3.3.9 Napisati funkciju `void par_nepar(int a[], int n, int parni[], int* pn, int neparni[], int* nn)` koja razbija niz a na niz parnih i niz neparnih brojeva. Pokazivači pn i nn redom treba da sadrže broj elemenata niza parnih tj. niza neparnih elemenata. Pretpostaviti da dužina niza a neće biti veća od 50. Napisati program koji testira napisanu funkciju.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza:  8
 Unesite elemente niza:
 1 8 9 -7 -16 24 77 4
 Niz parnih brojeva: 8 -16 24 4
 Niz neparnih brojeva: 1 9 -7 77
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza:  5
 Unesite elemente niza:
 2 4 6 8 -11
 Niz parnih brojeva: 2 4 6 8
 Niz neparnih brojeva: -11
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza:  2
 Unesite elemente niza:
 -15 15
 Niz parnih brojeva:
 Niz neparnih brojeva: -15 15
```

Primer 4

```
Interakcija sa programom:
 Unesite broj elemenata niza:  1
 Unesite elemente niza:
 0
 Niz parnih brojeva: 0
 Niz neparnih brojeva:
```

Zadatak 3.3.10 Napisati funckiju `void min_max(float a[], int n, float* min, float* max)` koja izračunava minimalni i maksimalni element niza a dužine n. Napisati program koji učitava niz realnih brojeva maksimalne dužine 50 i ispisuje vrednosti minimuma i maksimuma zaokruženu na tri decimale.

Primer 1

```
Interakcija sa programom:
 Unesite broj elemenata niza:  5
 Unesite elemente niza:
 24.16 -32.11 999.25 14.25 11
 Minimum: -32.110
 Maksimum: 999.250
```

Primer 2

```
Interakcija sa programom:
 Unesite broj elemenata niza:  4
 Unesite elemente niza:
 -5.126 -18.29 44 29.268
 Minimum: -18.290
 Maksimum: 44.000
```

Primer 3

```
Interakcija sa programom:
 Unesite broj elemenata niza:  1
 Unesite elemente niza:
 4.16
 Minimum: 4.160
 Maksimum: 4.160
```

Primer 4

```
Interakcija sa programom:
 Unesite broj elemenata niza:  3
 Unesite elemente niza:
 7.82 18.989 7.82
 Minimum: 7.820
 Maksimum: 18.989
```

Zadatak 3.3.11 Napisati program koji ispisuje broj navedenih argumenata komandne linije, a zatim i same argumenate i njihove redne brojeve.

Primer 1

```
Pokretanje: ./a.out abcde 123 -5 3.7
Interakcija sa programom:
 Broj argumenata je 5:
 0: ./a.out
 1: abcde
 2: 123
 3: -5
 4: 3.7
```

Primer 2

```
Pokretanje: ./a.out
Interakcija sa programom:
 Broj argumenata je 1:
 0: ./a.out
```

Zadatak 3.3.12 Napisati program koji ispisuje zbir numeričkih argumenata komandne linije. Uputstvo: *Koristiti funkciju* `atoi`.

Primer 1

```
Pokretanje: ./a.out 5 mkp 9 -2 11 a 4 2
Interakcija sa programom:
 Zbir numerickih argumenata: 29
```

Primer 2

```
Pokretanje: ./a.out ab u f hj
Interakcija sa programom:
 Zbir numerickih argumenata: 0
```

Primer 3

```
Pokretanje: ./a.out 33 1 p 44
Interakcija sa programom:
 Zbir numerickih argumenata: 78
```

Primer 4

```
Pokretanje: ./a.out
Interakcija sa programom:
 Zbir numerickih argumenata: 0
```

Zadatak 3.3.13 Napisati program koji ispisuje argumente komandne linije koji počinju slovom *z*.

Primer 1

```
POKRETANJE: ./a.out zima jabuka zvezda Zrak
INTERAKCIJA SA PROGRAMOM:
 zima zvezda
```

Primer 2

```
POKRETANJE: ./a.out bundeva pomorandza
INTERAKCIJA SA PROGRAMOM:
```

Primer 3

```
POKRETANJE: ./a.out sanke zapad zujanje
INTERAKCIJA SA PROGRAMOM:
 zapad zujanje
```

Primer 4

```
POKRETANJE: ./a.out
INTERAKCIJA SA PROGRAMOM:
```

Zadatak 3.3.14 Napisati program koji ispisuje broj argumenata komandne linije koji sadrže slovo z.

Primer 1

```
POKRETANJE: ./a.out zvezda grozd jesen kisa
INTERAKCIJA SA PROGRAMOM:
 2
```

Primer 2

```
POKRETANJE: ./a.out AZBUKA deda mraz
INTERAKCIJA SA PROGRAMOM:
 2
```

Primer 3

```
POKRETANJE: ./a.out japan caj
INTERAKCIJA SA PROGRAMOM:
 0
```

Primer 4

```
POKRETANJE: ./a.out
INTERAKCIJA SA PROGRAMOM:
 0
```

Zadatak 3.3.15 Napisati program koji na osnovu broja n koji se zadaje kao argument komandne linije ispisuje cele brojeve iz intervala $[-n,\ n]$.

Primer 1

```
POKRETANJE: ./a.out 2
INTERAKCIJA SA PROGRAMOM:
 -2 -1 0 1 2
```

Primer 2

```
POKRETANJE: ./a.out 4
INTERAKCIJA SA PROGRAMOM:
 -4 -3 -2 -1 0 1 2 3 4
```

Primer 3

```
POKRETANJE: ./a.out 0
INTERAKCIJA SA PROGRAMOM:
 0
```

Primer 4

```
POKRETANJE: ./a.out
INTERAKCIJA SA PROGRAMOM:
 Greska: nedostaje argument komandne linije!
```

Zadatak 3.3.16 Napisati program koji proverava da li se među zadatim argumentima komandne linije nalaze barem dva ista.

Primer 1

```
Pokretanje: ./a.out pec zima deda mraz pec
Interakcija sa programom:
 Medju argumentima ima istih.
```

Primer 2

```
Pokretanje: ./a.out xyz abc abc abc efgh
Interakcija sa programom:
 Medju argumentima ima istih.
```

Primer 3

```
Pokretanje: ./a.out 11 15 abc 888
Interakcija sa programom:
 Medju argumentima nema istih.
```

Primer 4

```
Pokretanje: ./a.out
Interakcija sa programom:
 Medju argumentima nema istih.
```

Zadatak 3.3.17 Napisati funkciju koja za dva data stringa određuje koliko se uzastopnih karaktera prvog stringa nalazi u drugom stringu počev od početka. Napisati program koji testira napisanu funkciju za dva stringa koji se unose kao argumenti komandne linije.

Primer 1

```
Pokretanje: ./a.out aladin bal
Interakcija sa programom:
 3
```

Primer 2

```
Pokretanje: ./a.out aladin lad
Interakcija sa programom:
 4
```

Primer 3

```
Pokretanje: ./a.out Aladin ala
Interakcija sa programom:
 0
```

Primer 4

```
Pokretanje: ./a.out aladin
Interakcija sa programom:
 Nekorektan poziv
 Program treba pozvati sa ./a.out arg1 arg2
```

Zadatak 3.3.18 Napisati program koji ispisuje sve opcije koje su navedene u komandnoj liniji.

Primer 1

```
Pokretanje: ./a.out -abc input.txt -d -Fg output
Interakcija sa programom:
 a b c d F g
```

Primer 2

```
Pokretanje: ./a.out
Interakcija sa programom:
```

Primer 3

```
Pokretanje: ./a.out ulaz.txt
Interakcija sa programom:
```

Primer 4

```
Pokretanje: ./a.out file.txt -x -yZ -g output
Interakcija sa programom:
 x yZ g
```

Zadatak 3.3.19 Napisati funkciju `void sifruj(char s[], char c, int k)` koja šifruje string s na sledeći način: svako malo i veliko slovo stringa s konvertuje u slovo koje je u abecedi od njega udaljeno k pozicija, i to k pozicija ulevo,

ako je karakter c jednak karakteru 'L' ili udesno ako je karakter c jednak karakteru 'D'. Šifrovanje treba da bude kružno. Ako string s sadrži karakter koji nije alfanumerički, ostaviti ga nešifriranog. Napisati program koji testira napisanu funkciju za string i prirodan broj koji se unose kao argumenti komandne linije dok se pravac šifrovanja unosi kao opcija -p koja može imati vrednosti 'L' ili 'D'. Ukoliko opcija -p nije navedena, podrazumevani pravac je udesno. NAPOMENA: *Možemo podrazumevati da string sadrži najviše 30 karaktera.*

Primer 1

```
POKRETANJE: ./a.out abcd 2
INTERAKCIJA SA PROGRAMOM:
 cdef
```

Primer 2

```
POKRETANJE: ./a.out abcd 2 -p D
INTERAKCIJA SA PROGRAMOM:
 cdef
```

Primer 3

```
POKRETANJE: ./a.out abcd 2 -p L
INTERAKCIJA SA PROGRAMOM:
 yzab
```

Primer 4

```
POKRETANJE: ./a.out abcd -3 -p L
INTERAKCIJA SA PROGRAMOM:
 Nekorektan unos
```

Primer 5

```
POKRETANJE: ./a.out abcd 3 -p X
INTERAKCIJA SA PROGRAMOM:
 Nekorektan unos
```

Primer 6

```
POKRETANJE: ./a.out ab12cd 2 -p D
INTERAKCIJA SA PROGRAMOM:
 Nekorektan unos
```

Zadatak 3.3.20 Parametri komandne linije su n, a i b $(a < b)$. Treba popuniti prvih `n` elemenata niza `A` celim slučajnim brojevima koji su između `a` i `b`. Ištampati niz `A` na standarni izlaz. Maksimalan broj elemenata niza `A` je 200. Ukoliko nisu zadati svi argumenti komandne linije ili ne zadovoljavaju potrebna svojstva ispisati poruku o grešci.

3.4 Rešenja

Rešenje 3.3.20

```
#include <stdio.h>

/*
   Argumenti funkcije uredi_pogresno, promenljive a i b,
   predstavljaju lokalne promenljive za ovu funkciju
   i prestaju da postoje po zavrsetku funkcije. Zbog toga
   se efekti razmene vrednosti promenljivih a i b u slucaju
   da je a>b vide u funkciji, ali se ne vide u glavnom programu.
*/
void uredi_pogresno(int a, int b)
```

```
{
  int t;

  if (a>b)
  {
     t = a;
     a = b;
     b = t;
  }
  printf("uredi_pogresno :: a=%d, b=%d\n", a, b);
  printf("uredi_pogresno :: &a=%p, &b=%p\n", &a, &b);
}

/*
   Argumenti funkcije uredi_tacno, promenljive pa i pb,
   takodje su lokalne promenljive za ovu funkciju i
   prestaju da postoje kada se funkcija zavrsi.
   Njima prosledjujemo adrese promenljivih a i b koje zelimo
   da razmenimo u slucaju da je a>b.

   Promenljivoj a pristupamo preko pokazivacke promenljive
   pa sa *pa i slicno, promenljivoj pb pristupamo sa *pb.

   Vrednosti promenljivih *pa i *pb razmenjujemo kao
   i vrednosti bilo koje dve celobrojne promenljive.

*/
void uredi_tacno(int * pa, int * pb)
{
  int t;
  if (*pa>*pb)
  {
     t = *pa;
     *pa = *pb;
     *pb = t;
  }
  printf("uredi_tacno :: *pa=%d, *pb=%d\n ", *pa, *pb);
  printf("uredi_tacno :: pa=%p, pb=%p\n ", pa, pb);

}
int main()
{
  int a,b;

  printf("Unesi dve celobrojne promenljive:");
  scanf("%d%d",&a,&b);

  printf("main :: a=%d, b=%d\n", a,b);
  printf("main :: &a=%p, &b=%p\n", &a, &b);
  uredi_pogresno(a,b);
  printf("main :: nakon uredi_pogresno, a=%d, b=%d\n", a, b);
```

```
63    /*
          Funkcija uredi_tacno kao argument ima dve pokazivacke
        promenljive
65       (int*,int*). Zbog toga joj je u pozivu funkcije neophodno
        proslediti
          adrese promenljivih koje zelimo da uredimo rastuce, &a i &b.
67    */

69    uredi_tacno(&a, &b);
      printf("main :: nakon uredi_tacno, a=%d, b=%d\n", a, b);
71
      return 0;
73  }
```

Rešenje 3.3.2

```
1   /*
      Napisati funkciju koja za boju datu u rgb formatu
3     racuna cmy format po formulama:
      C = 1 - ( R / 255 )
5     M = 1 - ( G / 255 )
      Y = 1 - ( B / 255 )
7
      Napisati program koji ucitava boju u rgb formatu,
9     primenjuje odgovarajucu funkciju i ispisuje boju u cmy formatu.

11  */

13  #include <stdio.h>
    #include <math.h>
15
    void rgb_to_cmy(float* a, float* b, float* c)
17  {
      /* Zagrade su neophodne jer aritmeticke operacije
19       imaju veci prioritet od operatora dereferenciranja (*).
      */
21    *a=1-(*a)/255;
      *b=1-(*b)/255;
23    *c=1-(*c)/255;

25    /*
      Pomocu return ne mozemo vratiti vise od jedne vrednosti.
27
      Ceste greske:
29    return a,b,c;          return vraca samo jednu vrednost
      return a; return b; return c; return ce vratiti samo a
31
      Zato je neophodno da promenljive ciju vrednost
33    zelimo da promenimo prenesemo preko pokazivaca.
      */
35  }
```

```
37
int rgb_korektno(float a)
39 {
   if(a<0 || a>255)
41    return 0;
   return 1;
43 }

45
int main()
47 {
  float a,b,c;
49
  /*
51     Argumenti funkcije rgb_to_cmy su
       pokazivaci na float. Njima prosledjujemo
53     adrese promenljivih a, b i c.
  */
55
  printf("Unesi boju u rgb formatu (vrednosti izmedju 0 i 255:");
57 scanf("%f%f%f",&a,&b,&c);

59 if(rgb_korektno(a) && rgb_korektno(b) && rgb_korektno(c))
     rgb_to_cmy(&a,&b,&c);
61 else
   {
63    printf("Nekorektan unos\n");
      return -1;
65 }

67 printf("Nakon konverzije: %.2f,%.2f,%.2f\n", a,b,c);

69 return 0;
}
```

Rešenje 3.3.3

```
/*
2    Napisati funkciju koja za dve prave date svojim koeficijentima
     pravca i slobodnim clanovima odredjuje njihovu tacku preseka.
4    Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju
     tacku preseka (ako su paralelne). Napisati glavni program
6    koji ucitava podatke o pravama, poziva napisanu funkciju i
     ispisuje odgovarajucu poruku.
8 */

10 #include<stdio.h>

12 /*
    Funkcija presek treba da izracuna tri vrednosti:
```

```
   1. indikator da li su koeficijenti pravca jednaki ili ne
   2. prvu koordinatu presecne tacke (ukoliko prave nisu paralelne)
   3. drugu koordinatu presecne tacke (ukoliko prave nisu paralelne)

   Indikator funkcija vraca kao povratnu vrednost, preko kljucne reci
   return.

   Koordinate presecne tacke (ako postoji) funkcija vraca preko
   liste argumenata, zbog cega promenljive kojima ce koordinate
   biti dodeljene prenosimo preko pokazivaca (promenljive px i py)

   Promenljive koje sadrze podatke o pravama (k1,n1,k2,n2) se ne
   menjaju u funkciji i zbog toga ih ne moramo prenositi preko
   pokazivaca.
*/

int presek(float k1, float n1, float k2, float n2, float* px, float*
    py)
{
   if (k1==k2)
     return 0;

   *px = -(n1-n2)/(k1-k2);
   *py = k1*(*px)+n1;
   return 1;
}

int main()
{
   float k1,k2,n1,n2;
   float x,y;

   printf("Unesi k i n za prvu pravu:");
   scanf("%f%f",&k1,&n1);

   printf("Unesi k i n za drugu pravu:");
   scanf("%f%f",&k2,&n2);

   if(presek(k1,n1,k2,n2,&x,&y))
      printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
   else
      printf("Prave su paralelne\n");

   return 0;
}
```

Rešenje 3.3.7

```
#include <stdio.h>

#define MAX 21
```

```
4
void modifikacija(char *s, char *t, int *br_modifikacija)
6 {
    int i;
8   for(i=0;s[i];i++)
      if(s[i]>='a' && s[i]<='z')
10    {
        t[i] = toupper(s[i]);
12      (*br_modifikacija)++;
      }
14    else
        t[i] = s[i];
16 }

18 int main()
{
20   char s[MAX], t[MAX];
    int br_modifikacija = 0;
22
    printf("Unesite nisku: ");
24  scanf("%s", s);

26  modifikacija(s, t, &br_modifikacija);

28  printf("Modifikovana niska je: %s\nBroj modifikacija je: %d\n", t,
      br_modifikacija);

30  return 0;
}
```

Rešenje 3.3.8

```
1 /*
Napisati funkciju
3   void interpunkcija(int * br_tacaka, int * br_zareza)
koja za tekst koji se unosi sa standardnog ulaza sve do kraja ulaza
    prebrojava
5 broj tacaka i zareza. Napisati zatim program koji testira napisanu
    funkciju.
*/
7

9 #include <stdio.h>

11 void interpunkcija(int* br_tacaka, int* br_zareza){

13   int tacke=0, zarezi=0;
    char c;
15
    while((c=getchar())!=EOF){
17    if(c=='.')
```

```
      tacke++;

    if(c==',')
      zarezi++;
  }

  *br_tacaka=tacke;
  *br_zareza=zarezi;

}

int main(){
  int br_tacaka, br_zareza;

  printf("Unesite tekst: \n");

  interpunkcija(&br_tacaka, &br_zareza);

  printf("Broj tacaka: %d\n", br_tacaka);
  printf("Broj zareza: %d\n", br_zareza);

  return 0;
}
```

Rešenje 3.3.9

```
/*
Napisati funkciju
  void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
     int* nn)
koja razbija niz a na niz parnih i niz neparnih brojeva. Pokazivaci
    pn i nn
redom treba da sadrze broj elemenata niza parnih tj. niza neparnih
    elemenata.
Pretpostaviti da duzina niza a nece biti veca od 50. Napisati program
     koji
testira napisanu funkciju.
*/

#include <stdio.h>
#define MAX 50

void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
    int* nn){

  int i, j, k;

  /* i - brojac niza a */
  /* j - brojac niza parnih brojeva */
  /* k - brojac niza neparnih brojeva */

```

```
  for(i=0, j=0, k=0; i<n; i++){
22      /* Ako je element niza paran */
        if(a[i]%2==0){
24          /* Smestamo ga u niz parnih brojeva i uvecavamo indeks niza
      j */
            parni[j]=a[i];
26          j++;
        }
28      else{
            /* Inace, smestamo ga u niz neparnih brojeva i uvecavamo
     indeks niza k */
30          neparni[k]=a[i];
            k++;
32      }
  }
34

  *pn=j;
36 *nn=k;

38}

40int main(){
  int n, i, j, pn, nn;
42 int a[MAX], parni[MAX], neparni[MAX];

44 /* Ucitavamo dimenziju niza */
  printf("Unesite broj elemenata niza: ");
46 scanf("%d", &n);

48 if(n<0 || n>MAX){
      printf("Greska: pogresna dimenzija niza!\n");
50    return 0;
  }
52

  /* Ucitavamo elemente niza */
54 printf("Unesite elemente niza: ");
  for(i=0; i<n; i++){
56   scanf("%d", &a[i]);
  }
58

  /* Pozivamo funkciju koja razbija zadati niz na niz parnih i niz
     neparnih */
60 par_nepar(a, n, parni, &pn, neparni, &nn);


62

  /* Ispisujemo dobijene nizove */
64 printf("Niz parnih brojeva: ");
  for(i=0; i<pn; i++)
66   printf("%d ", parni[i]);
  printf("\n");
68

  printf("Niz neparnih brojeva: ");
```

```
  for(i=0; i<nn; i++)
    printf("%d ", neparni[i]);
  printf("\n");

  return 0;
}
```

Rešenje 3.3.10

```
/*
  Napisati funckiju
    void min_max(float a[], int n, float* min, float* max)
  koja izracunava minimalni i maksimalni element niza a duzine n.
  Napisati zatim i program koji ucitava niz realnih brojeva
    maksimalne
  duzine 50 i ispisuje vrednosti minimuma i maksimuma na tri decimale
    .

*/

#include<stdio.h>
#define MAX 50

void min_max(float a[], int n, float* min, float* max){

  int i;

  /* Inicijalizujemo vrednosti minimuma i maksimuma */
  *min=a[0];
  *max=a[0];

  /* Obilazimo preostale elemente niza */
  for(i=1; i<n; i++){

    /* Ako je tekuca vrednost veca od maksimalne, azuriramo maksimum
    */
    if(a[i]>*max){
      *max=a[i];
    }

    /* Ako je tekuca vrednost manja od minimalne, azuriramo minimum
    */
    if(a[i]<*min){
      *min=a[i];
    }
  }
}

int main(){
  int i, n;
  float a[MAX], min, max;
```

```
40    /* Ucitavamo dimenziju niza */
      printf("Unesite broj elemenata niza: ");
42    scanf("%d", &n);

44    if(n<0 || n>MAX){
        printf("Greska: pogresna dimenzija niza!\n");
46      return 0;
      }
48
      /* Ucitavamo elemente niza */
50    printf("Unesite elemente niza:\n");
      for(i=0; i<n; i++){
52      scanf("%f", &a[i]);
      }
54
      /* Pozivamo funkciju za racunanje maksimuma i minimuma */
56    min_max(a, n, &min, &max);

58    /* Ispisujemo rezultat */
      printf("Minimum: %.3f\n", min);
60    printf("Maksimum: %.3f\n", max);

62    return 0;

64  }
```

Rešenje 3.3.11

```
/*
2       Napisati program koji ispisuje broj navedenih argumenata komandne
         linije,
        a zatim i same argumenate i njihove redne brojeve.
4   */

6   #include <stdio.h>

8   /*
       Argumenti komandne linije cuvaju se u nizu niski pod nazivom
10     argv. Svaki element tog niza odgovara jednom argumentu komandne
       linije pri cemu prvi element predstavlja naziv programa koji
12     pokrecemo. Celobrojna promenljiva argc predstavlja ukupan
       broj argumenata komandne linije ukljucujuci i argument koji
14     odgovara nazivu programa.
    */
16
    int main(int argc, char *argv[])
18  {
       int i;
20
       printf("Broj argumenata je: %d\n",argc);
```

```
22
    for(i=0; i<argc; i++)
24      printf("%d: %s\n",i,argv[i]);

26  return 0;
}
```

Rešenje 3.3.12

```
#include <stdio.h>

int main(int argc, char* argv[]) {

  int i;
  int s = 0;

  /* char *argv[] <--- niz niski koje predstavljaju argumente
     navedene iza poziva programa
     int argc      <--- ukupan broj niski (sa sve nazivom programa)
     navedenih prilikom pozivanja

     Ukoliko je program pozvan sa ./a.out 12 abc 6 5 3ab

     argv[0] = "./a.out".
     argv[1] = "12"
     argv[2] = "abc"
     argv[3] = "6"
     argv[4] = "5"
     argv[5] = "3ab"

     argc iznosi 6


     Kako je argv[] po prirodi niz,
     koristimo tzv. brojacku odnosno
     for petlju
     i obradjujemo svaki od argumenata.
  */

  /* Funkcija atoi() prihvata nisku,
     i racuna dekadnu vrednost prosledjene niske,
     dokle god se ona moze racunati.
     Na primer, ukoliko je niska "-123",
     atoi() vraca broj -123.
     Ako je, pak, niska "123abc",
     atoi() ce vratiti 123
     (prilikom prve pojave karaktera koji nije cifra, funkcija prekida
      izracunavanje).

     To za posledicu ima da, ukoliko je funkciji
     prosledjeno nesto
```

```
   sto se ne moze pretvoriti u broj,
   na primer niska "abcd",
   funkcija atoi() vraca 0.
  */

  for(i = 1; i < argc; i++)
    s += atoi(argv[i]); /* Zbog nacina rada funkcije atoi(), mozemo
    je pozvati nad svim argumentima
              komandne linije, i sabrati odgovarajuce dekadne
    vrednosti.
              Ukoliko neki argument i nije broj, to ne predstavlja
    problem
              jer ce u tom slucaju odgovarajuci sabirak biti 0
            */


    printf("Zbir numerickih argumenata: %d\n", s);

  return 0;
}
```

Rešenje 3.3.13

```
#include <stdio.h>

int main(int argc, char* argv[]) {

  int i;

  /* Prolazimo for petljom kroz niz argumenata,
   i trazimo one niske ciji je prvi karakter bas 'z'.
   Ukoliko je trenutni argument koji se ispituje
   argv[i],
   kako je on sam po sebi niska,
   do prvog karaktera dolazimo kao i pri dosadasnjem
   radu sa niskama --> argv[i][0]
                       ^
                       |
                     index prvog karaktera u niski argv[i]
  */

  for(i = 1; i < argc; i++)
    if(argv[i][0] == 'z')
      printf("%s ", argv[i]);

  putchar('\n');

  return 0;
}
```

Rešenje 3.3.14

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

int main(int argc, char* argv[]) {

  int i;
  int br = 0;

  /* Da bismo proverili da li se karakter 'z' (tj. 'Z')
   nalazi u niski argv[i],
   to mozemo uciniti koriscenjem funkcije
   strchr() koja se nalazi u string.h.

   Ukoliko je karakter sadrzan u okviru niske,
   strchr() vraca pokazivac na taj karakter
   unutar same niske.
   Inace, ukoliko se karakter ne nalazi u niski,
   funkcija vraca NULL.
  */

  for(i = 1; i < argc; i++)
    if(strchr(argv[i], 'z') != NULL || strchr(argv[i], 'Z') != NULL)
      br++;

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

  return 0;
}
```

Rešenje 3.3.15

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

int main(int argc, char *argv[])
{
  int n,i;

  /*
    Ispisujemo gresku ukoliko nema dovoljno argumenata komandne
    linije.
  */
  if(argc != 2)
  {
    printf("Greska: nedostaje argument komandne linije!\n");
    return -1;
  }
```

```
  /*
    Pretvaramo argument komandne linije koji je string u ceo broj
    koriscenjem funkcije atoi
  */
  n = atoi(argv[1]);
  n = abs(n);

  for(i=(-1)*n;i<=n;i++)
    printf("%d ",i);

  return 0;
}
```

Rešenje 3.3.16

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

int main(int argc, char *argv[])
{
  int indikator = 0;
  int i,j;
  /*
    Ukoliko imamo samo jedan argument komandne linije,
    ispisujemo da nema istih i zavrsavamo program.
  */
  if(argc < 2)
  {
    printf("Medju argumentima nema istih.\n");
    return -1;
  }

  /*
    Prolazimo kroz niz argumenata i za svaki posebno proverimo
    da li medju ostalima postoji neki koji mu je jednak i ako postoji
    ispisujemo poruku i zavrsavamo program.
    Ako smo izasli iz prve petlje to znaci da nismo pronasli dva ista
     elementa
    i ispisujemo odgovarajucu poruku.
  */
  for(i=0;i<argc;i++)
  {
    for(j=0;j != i && j<argc; j++)
      if(strcmp(argv[i], argv[j]) == 0)
      {
        printf("Medju argumentima ima istih.\n");
        return 0;
      }
  }

  printf("Medju argumentima nema istih.\n");
```

```
  return 0;
}
```

Rešenje 3.3.17

```
/*
   Napisati funkciju koja za dva data stringa str i
   accept odredjuje koliko se uzastopnih karaktera stringa str
   nalazi u stringu accept pocev od pocetka niza str. Napisati
   potom program koji testira napisanu funkciju  za dva stringa
   koji se unose kao argumenti komandne linije. Primeri upotrebe:

   1:
   ./a.out aladin bal
   3

   2:
   ./a.out aladin lad
   4

   3:
   ./a.out Aladin ala
   0

*/

#include <stdio.h>
#include <string.h>

/*
   Funkcija strspn(str,accept) je ugradjena funkcija koja vraca broj
    karaktera
   stringa str koji se nalaze u stringu accept, pocev od pocetka
    stringa str.

   Funkcija strspn se nalazi u zaglavlju string.h.

   Funkcija strspn_klon je jedna implementacija funkcije strspn.

   U zadacima cemo uvek koristiti ugradjenu funkciju strspn osim ako
    u tekstu zadatka
   nije naglaseno da se ona ne sme koristiti. Funkcija strspn_klon
    sluzi da pokaze na koji
   nacin radi ugradjena funkcija strspn.

   Ugradjena funkcija strspn poziva se na isti nacin kao funkcija
    strspn_klon:
   strspn(s1,s2)

*/
```

```c
int strspn_klon(char str[], char accept[])
{
   int br=0;
   int i;

   for(i=0; str[i];i++)
      if(strchr(accept, str[i])!=NULL)
         br++;
      else      /* ako pronadjemo karakter u stringu str koji nije */
         break; /* u stringu accept, prekidamo petlju */

   return br;
}

int main(int argc, char* argv[])
{

   int br;

   if(argc<3)
   {
      printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
     arg2\n");
      return -1;
   }

   br = strspn_klon(argv[1],argv[2]);
   printf("Broj karaktera stringa %s koji se nalaze u stringu %s,
    pocev od pocetka stringa %s: %d\n", argv[1],argv[2],argv[1],br);
   return 0;
}
```

Rešenje 3.3.20

```c
#include <stdio.h>

void suma(int a, int b, int *s);


int main()
{
    int a,b,s;

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

    suma(a,b,&s);

    printf("suma: %d\n",s);

    return 0;
}
```

```
19 void suma(int a, int b, int *s)
   {
21     *s = a + b;
   }
```

Rešenje 3.3.19

```
1  /*
     Napisati funkciju void sifruj(char s[], char c, int k) koja
       sifruje
3    string s na sledeci nacin: svako malo i veliko slovo stringa s
       konvertuje u
     slovo koje je u abecedi od njega udaljeno k pozicija, i to
5    k pozicija ulevo, ako je karakter c jednak karakteru 'L' ili
       udesno
     ako je karakter c jednak karakteru 'D'. Sifrovanje treba da bude
       kruzno. Ako string
7    s sadrzi karakter koji nije alfanumericki, ostaviti ga
       nesifriranog.

9    Napisati potom glavni program koji testira napisanu funkciju za
       string i prirodan
     broj koji se unose kao argumenti komandne linije dok se pravac
       sifrovanja unosi
11   kao opcija -p koja moze imati vrednosti 'L' ili 'D'. Ukoliko
       opcija -p nije
     navedena, podrazumevani pravac je udesno.
13
     Mozemo podrazumevati da string sadrzi najvise 30 karaktera.
15
     Primeri upotrebe:
17
     1:
19   ./a.out abcd 2
     cdef
21
     2:
23   ./a.out abcd 2 -p D
     cdef
25
     3:
27   ./a.out abcd 2 -p L
     yzab
29
     4:
31   ./a.out abcd -3 -p L
     Nekorektan unos
33
     5:
35   ./a.out abcd 3 -p X
```

```
   Nekorektan unos

   6:
   ./a.out ab12cd 2 -p D
   cd12ef

*/

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAX 31

void sifruj(char s[], char c, int k)
{
   int i;
   int znak;
   char t;

   /*
      S obzirom da ce korektnost unosa podataka
      biti ispitana pre poziva funkcije, promenljiva
      c ce imati vrednost 'L' ili 'D'.

      Promenljiva znak ima vrednost 1 ili -1
      i sluzi kao pomocna promenljiva u slucaju
      da prilikom sifriranja konvertovani
      karakter izadje iz opsega malih ili velikih slova.

   */
   znak=1;
   if (c=='L')
      znak = -1;


   for(i=0; s[i];i++)
      if(isalpha(s[i]))
      {
         /*
            Promenljiva t predstavlja sifrirani karakter s[i].
            Ako je promenljiva t izvan opsega malih ili velikih slova
    ,
            dodajemo joj ili oduzimamo ukupan broj slova u abecedi
    (26),
            u zavisnosti od pravca sifriranja, kako bismo omogucili
            kruzno sifriranje.
         */
         t = s[i]+znak*k;
         if((islower(s[i]) && (t<'a' || t>'z')) || (isupper(s[i]) &&
    (t<'A' || t>'Z')))
            s[i]=t-znak*26;
         else
```

```
85              s[i]=t;
        }
87  }

89  int main(int argc, char* argv[])
    {
91
        int k;
93      char pravac;
        char rec[MAX];
95
        /*
97          Program mozemo pozivati na dva nacina:
            ./a.out abcd 2
99          ili
            ./a.out abcd 2 -p D
101
            Zbog toga, broj argumenata moze biti 3 ili 5.
103     */

105     if (argc!=3 && argc!=5)
        {
107         printf("Nekorektan unos: broj argumenata moze biti 3 ili 5\n");
            return -1;
109     }

111     /*
            Argumenti komandne linije su stringovi. Ako program pokrecemo
113         na sledeci nacin:
            ./a.out abcd 2 -p D
115         to znaci da je argument koji odgovara dvojci u stvari
            string "2". Da bismo string konvertovali u ceo broj,
117         koristimo ugradjenu funkciju atoi iz biblioteke stdlib.h.
        */
119
        k = atoi(argv[2]);
121
        /*
123         Ispitujemo korektnost datih podataka:
        */
125     if (k<=0)
        {
127         printf("Nekorektan unos: broj pozicija mora biti pozitivan ceo
          broj\n");
            return -1;
129     }

131     /* Korektnost unosa je ispitana, sto znaci da
        argc moze biti 3 ili 5 */
133
        if (argc==3) /* Ako je argc 3: */
135         pravac='D';
```

```
    else        /* Ako argc nije 3, tada je sigurno 5, jer je */
    {           /* korektnost unosa ispitana, a unos je korektan
     jedino za argc==3 ili argc==5 */
       /*
          Ispitujemo korektnost pretposlednjeg argumenta koji mora da
     bude u formatu "-p".
          Ovaj argument je string argv[3]. Njegovom prvom karakteru (
     koji treba
          da bude '-') pristupamo sa argv[3][0] a drugom sa argv
     [3][1].
       */
       if (argv[3][0] != '-')
       {
          printf("Nekorektan unos: pri zadavanju opcija prvi karakter
     mora biti '-' \n");
          return -1;
       }

       if (argv[3][1]!='p')
       {
          printf("Nekorektan unos: nedozvoljena opcija\n");
          return -1;
       }

       /*
          Nakon argumenta -p sledi argument koji zadaje vrednost ove
     opcije. To je
          poslednji argument kome pristupamo sa argv[4]. Ovaj argument
      treba
          da sadrzi samo jedan karakter - 'L' ili 'D' i njemu
     pristupamo sa
          argv[4][0].
       */
       if(argv[4][0]=='L' || argv[4][0]=='D')
          pravac=argv[4][0];
       else
       {
          printf("Nekorektan unos: pravac moze biti L ili D\n");
          return -1;
       }
    }

    strcpy(rec, argv[1]);
    sifruj(rec,pravac,k);

    printf("Sifrovana rec: %s\n", rec);

    return 0;
}
```

3.5 Niske

Zadatak 3.5.1 Napisati funkciju `void konvertuj(char s[])` koja menja datu nisku s tako što u njoj mala slova zamenjuje odgovarajućim velikim slovima, a velika slova zamenjuje odgovarajućim malim slovima. Napisati program koji testira ovu funkciju za učitanu nisku maksimalne dužine 10 karaktera.

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
```

Zadatak 3.5.2 Napisati funkciju `void modifikacija(char s[])` koja modifikuje nisku s tako što u njoj svaki drugi karakter zameni zvezdicom. Napisati program koji testira rad napisane funkcije za učitanu nisku maksimalne dužine 20 karaktera.

Primer 1

```
Interakcija sa programom:
 Unesite nisku: 123abc789XY
 Modifikovana niska je: 1*3*b*7*9*Y
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku: zimA
 Modifikovana niska je: z*m*
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: SNEG
 Modifikovana niska je: S*E*
```

Primer 4

```
Interakcija sa programom:
 Unesite nisku: *a*b*c*
 Modifikovana niska je: *******
```

Zadatak 3.5.3 Napisati program koji vrši poređenje niski. Napisati funkcije:

(a) `int poredjenje(char s1[], char s2[])` — vraća 1 ako su s_1 i s_2 jednake niske, a 0 u suprotnom;

(b) `void u_velika_slova(char s[])` — pretvara sva slova niske s u velika, a ostale znakove ne menja.

Napisati program koji za učitane dve reči dužine najviše 20 znakova ispituje da li su jednake, pri čemu se zanemaruje razlika između velikih i malih slova.

Primer 1

```
Interakcija sa programom:
 isPit2010
 IsPiT2010
 jesu jednake
```

Primer 2

```
Interakcija sa programom:
 Prog1
 prog2
 nisu jednake
```

Primer 3

```
Interakcija sa programom:
 junski
 septembarski
 nisu jednake
```

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!
```

Zadatak 3.5.5 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
```

Zadatak 3.5.6 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
 Niska ne sadrzi veliko slovo
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 DiopTrija0.75
 Niska sadrzi veliko slovo
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Unesite nisku:
 21.06.2017.
 Niska ne sadrzi veliko slovo
```

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 dve niske maksimalne dužine 10 karaktera i testira rad napisane funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite nisku s: abcde
 Unesite nisku t: bcd
 t je podniska niske s!
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku s: abcde
 Unesite nisku t: bCd
 t nije podniska niske s!
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku s: abcde
 Unesite nisku t: def
 t nije podniska niske s!
```

Zadatak 3.5.8 Napisati funkciju `void skrati(char s[])` koja uklanja beline sa kraja date niske. Napisati program koji testira ovu funkciju za učitanu liniju maksimalne dužine 100 karaktera. Prikazati 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*
```

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:
 abcd123ABCD
 123
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku:
 1+2=3
 1+2=3
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku:
 malaVELIKA
```

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. Testirati funkciju u programu za učitanu liniju od najviše 100 karaktera.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Danas je 10 stepeni.
 Dns j 10 stpn.
```

Primer 3

```
Interakcija sa programom:
 Ima vetra, kise i hladnoce.
 Im vt, ks i hdnoe.
```

Zadatak 3.5.11 Napisati program koji učitava nisku *src* i formira nisku *dst* trostrukim nadovezivanjem niske *src*. Možemo pretpostaviti da niska *src* sadrži najviše 30 karaktera.

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
```

Zadatak 3.5.12 Napisati program koji za unetu reč maksimalne dužine 20 karaktera formira rezultujuću reč tako što unetu reč kopira 4 puta, pri čemu se između svaka dva kopiranja umeće crtica.

Zadatak uraditi:

(a) pisanjem odgovarajuće funkcije koja vrši nadovezivanje reči,

(b) koristeći postojeću funkciju `strcat` iz biblioteke `string.h`.

Primer 1

```
Interakcija sa programom:
 Unesite nisku: ana
 ana-ana-ana-ana
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku: 123
 123-123-123-123
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: x*y
 x*y-x*y-x*y-x*y
```

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.

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
```

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 testira rad napisane funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite nisku: zima
 zziimmaa
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku: A+B+C
 AA++BB++CC
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: C
 CC
```

Zadatak 3.5.15 Napisati program koji učitava nisku cifara sa eventualnim vodećim znakom i pretvara je u ceo broj. Napomena: *Zadatak realizovati bez korišćenja ugrađene funkcije atoi iz biblioteke stdlib.h*

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
```

Zadatak 3.5.16 Napisati program koji učitava ceo broj i pretvara ga u 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
```

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!
```

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
```

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
```

Zadatak 3.5.20 Napisati funkcije za rad sa rečenicama:

(a) `int procitaj_recenicu(char s[], int max_len)` koja učitava rečenicu i smešta je u nisku s. Funkcija vraća dužinu učitane rečenice. Učitavanje se završava nakon učitanog karaktera `.` ili nakon $max\_len - 1$ učitanih karaktera.

(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 razlika broja malih i velikih slova najveća.

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. Napisati program koji za učitanu

nisku maksimalne dužine 20 karaktera i dodatni karakter testira rad napisane funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite nisku s: programiranje
 Unesite karakter c: a
 Karakter se nalazi u niski!
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku s: 123456789
 Unesite karakter c: y
 Karakter se ne nalazi u niski!
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku s: leto2017
 Unesite karakter c: 0
 Karakter se nalazi u niski!
```

Primer 4

```
Interakcija sa programom:
 Unesite nisku s: jedrilica
 Unesite karakter c: I
 Karakter se ne nalazi u niski!
```

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
```

Zadatak 3.5.23 Napisati funkciju `int strcspn_klon(char s[], char t[])` koja izračunava dužinu početnog dela niske s sastavljenog isključivo od karaktera sadržanih u niski t. Napisati program koji testira ovu funkciju za dve unete niske maksimalne dužine 100 karaktera.

Primer 1

```
Interakcija sa programom:
 Unesite prvu nisku:
 734a.bf62
 Unesite drugu nisku:
 0123456789
 3
```

Primer 2

```
Interakcija sa programom:
 Unesite prvu nisku:
 abrakadabra
 Unesite drugu nisku:
 brada
 4
```

Primer 3

```
Interakcija sa programom:
 Unesite prvu nisku:
 popokatepetl
 Unesite drugu nisku:
 opna
 2
```

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.
```

Zadatak 3.5.25 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
```

Zadatak 3.5.26 Napisati funkciju `void rotiraj(char s[], int k)` koja rotira nisku s za k mesta ulevo. Napisati program koji rotira učitanu nisku maksimalne dužine 20 karaktera i ispisuje rotiranu nisku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 sveska
 2
 eskasv
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 olovka
 6
 olovka
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 rezac
 8
 acrez
```

Zadatak 3.5.27 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
```

Zadatak 3.5.28 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
```

Zadatak 3.5.29 Napisati funkciju `void indel(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
```

Zadatak 3.5.30 Napisati funkciju `int prepis(char a[][21], int na, char b[][21])` koja iz niza reči a dužine na prepisuje u niz b one reči koje su sastavljene samo od malih ili samo od velikih slova i vraća dužinu niza b. Napisati program koji za učitani broj n ($0 < n \leq 50$) i n reči razdvojenih blanko znakom ispisuje sve unete reči sastavljene samo od malih ili samo od velikih slova. Pretpostaviti da su unete reči maksimalne dužine 20 karaktera. U slučaju da je n van dozvoljenog opsega, ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 3 abc ABC aBc
 abc ABC
```

Primer 2

```
Interakcija sa programom:
 2 mnB RGa
```

Primer 3

```
Interakcija sa programom:
 -3
 Nekorektan broj reci!
```

Zadatak 3.5.31 Napisati program za rad sa brojevima zapisanim u različitim brojevnim sistemima.

(a) Napisati funkciju `unsigned btoi(char s[], unsigned char b)` koja određuje dekadnu vrednost zapisa datog neoznačenog broja s u datoj osnovi b.

(b) Napisati funkciju `void itob(unsigned n, unsigned char b, char s[])` koja datu dekadnu vrednost n zapisuje u datoj osnovi b i smešta rezultat u nisku s. Pretpostaviti da je $0 < b \leq 16$.

Napisati program koji za svaku učitanu liniju koja sadrže po jedan dekadni, oktalni ili heksadekadni broj (zapisan kao što se zapisuju konstante u programskom jeziku C) ispisuje odgovarajući binarni zapis. Linije se unose sve do kraja ulaza. Pretpostaviti da će sve linije sadržati ispravne brojeve i da će ti brojevi biti u opsegu tipa `unsigned`.

Primer 1

```
Interakcija sa programom:
 0x49 0x1ABC
 1001001 1101010111100
```

Primer 2

```
Interakcija sa programom:
 012 435 0x64FE
 1010 110110011 110010011111110
```

Primer 3

```
Interakcija sa programom:
 123 0777
 1111011 111111111
```

Primer 4

```
Interakcija sa programom:
 981
 1111010101
```

3.6 Rešenja

Rešenje 3.5.1

```
/*
   Napisati funkciju koja konvertuje dati string tako sto
   mala slova menja u velika a velika u mala. Napisati
   potom glavni program koji ucitava string, poziva napisanu
   funkciju i ispisuje konvertovani string. Mozemo pretpostaviti
```

```
   da string ne sadrzi vise od 10 karaktera.
*/

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

/*
   Kada je niz argument funkcije, dodatni argument je obavezno
   njegova dimenzija. Kod stringova to nije slucaj jer svaki string
   ima isti poslednji element - terminirajucu nulu - i to je oznaka
   kraja stringa.
*/
void konvertuj(char s[])
{
   int i;

   for(i=0; s[i]!='\0'; i++)
      if (s[i]>='a' && s[i]<='z')
         s[i] = toupper(s[i]);  /* toupper - konvertuje malo slovo u
    odgovarajuce veliko */
      else if (s[i]>='A' && s[i]<='Z')
         s[i] = tolower(s[i]);  /* tolower - konvertuje veliko slovo
    u odgovarajuce malo */
   /*
      Funkcije toupper i tolower se nalaze u zaglavlju ctype.h.

      Konverzija malog slova u veliko bez upotrebe funkcije toupper:
          s[i] = s[i]-'a'+'A';
      Konverzija velikog slova u malo bez upotrebe funkcije tolower:
          s[i] = s[i]+'a'-'A';

   */
}

int main()
{
   /*
      Poslednji karakter svakog stringa je terminirajuca
      nula '\0', specijalni karakter ciji je ASCII kod 0.

      Ukoliko pretpostavljamo da string sadrzi najvise 30
      karaktera, neophodno je deklarisati niz od 31 karaktera,
      pri cemu se dodatni izdvaja za terminirajucu nulu.

   */
   char s[31];
   printf("Unesi string:");

   /*
      Za razliku od nizova koji se ucitavaju i stampaju
      element po element, stringovi se mogu ucitati i
      odstampati pomocu jedne scanf/printf naredbe koriscenjem
```

```
      specifikatora %s.

      Funkcija scanf ucitava string do prvog pojavljivanja razmaka.
   */
   scanf("%s", s);

   konvertuj(s);

   printf("Konvertovani string: %s\n", s);

   return 0;

}
```

Rešenje 3.5.2

Rešenje 3.5.31

Rešenje 3.5.4

```
/*
a) Napisati funkciju int samoglasnik(char c) koja proverava da li je
    zadati karakter samoglasnik. Funkcija
treba da vrati vrednost 1 ako karakter c jeste samoglasnik, odnosno 0
     ako nije.
b) Napisati funkciju int samoglasnik_na_kraju(char s[]) koja
    proverava da li se niska s zavrsava samoglasnikom
(koristiti funkciju iz tacke a)).
c) Napisati program koji ucitava nisku maksimalne duzine 20 karaktera
     i ispisuje da li zavrsava samoglasnikom ili ne.
*/

#include <stdio.h>
#include <ctype.h>
#include <string.h>
#define MAX_DUZINA 20

/* Funkcija proverava da li je karakter c samoglasnik */
int samoglasnik(char c){

  char C;

  /* Konvertujemo karakter u veliko slovo kako bismo smanjili broj
    provera */
  C=toupper(c);

  /* Samoglasnici su slova a, e, i, o i u */
  if(C=='A' || C=='E' || C=='I' || C=='O' || C=='U')
    return 1;
```

```
25
  return 0;
27 }

29 /* Funkcija koja proverava da li se niska s zavrsava samoglasnikom */
int samoglasnik_na_kraju(char s[]){
31   int duzina;

33   /* Odredjujemo duzinu niske */
  duzina=strlen(s);
35
  /* Proveravamo da li je niska prazna */
37   if(duzina==0)
    return 0;
39
  /* Ako niska nije prazna, proveravamo da li se samoglasnik nalazi
     na kraju */
41   /* Numeracija karaktera u niski pocinje nulom pa zato proveravamo
     poziciju duzina -1 */
  return samoglasnik(s[duzina-1]);
43 }


45
int main(){
47   char s[MAX_DUZINA+1];

49   /* Ucitavamo nisku */
  printf("Unesite nisku: ");
51   scanf("%s", s);

53   /* Proveravamo da li se zavrsava samoglasnikom i ispisujemo
     odgovarajucu poruku */
  if(samoglasnik_na_kraju(s))
55     printf("Niska se zavrsava samoglasnikom!\n");
  else
57     printf("Niska se ne zavrsava samoglasnikom!\n");

59   return 0;
}
```

Rešenje 3.5.5

```
/*
2    Napisati program koji za uneti string s i karakter c utvrdjuje
   da li se c pojavljuje u stringu s i ukoliko se pojavljuje,
4    ispisuje indeks prvog pojavljivanja a u suprotnom ispisuje
   odgovarajucu poruku. Mozemo pretpostaviti da string ima najvise
6    20 karaktera.
*/
8
#include <stdio.h>
```

```c
#include <string.h>

int main()
{
   char s[21];
   char c;
   char* p;

   printf("Unesi karakter:");
   c=getchar();
   printf("Unesi string:");
   scanf("%s", s);

   /*
     Da smo ucitavali obrnutim redom (prvo string pa karakter)
     to bismo realizovali na sledeci nacin:
     printf("Unesi string:");
     scanf("%s",s);
     getchar();
     printf("Unesi karakter:");
     c=getchar();

     Dodatni getchar() bi sluzio da "pokupi" karakter kojim
     razdvajamo unos stringa i karaktera (razmak, novi red ili
     slicno).

   */

   /*
      Funkcija strchr(s,c) je ugradjena funkcija koja vraca pokazivac
      na prvi karakter u stringu s koji je jednak karakteru c, ako
    takav
      postoji, a NULL u suprotnom.

      Indeks od pokazivaca dobijamo na isti nacin kao u prethodnom
    zadatku
      sa strstr.
   */

   p = strchr(s,c);
   if(p!=NULL)
      printf("%c se pojavljuje u %s na poziciji %d\n", c, s, p-s);
   else
      printf("%c se NE pojavljuje u %s\n",c, s);

   return 0;
}
```

Rešenje 3.5.31

Rešenje 3.5.7

```
/*
 Napisati funkciju int podniska(char s[], char t[]) koja proverava da
     li je niska t podniska niske s.
 Napisati i program koji ucitava dve niske maksimalne duzine 10
    karaktera i testira rad napisane funkcije.

 Napomena: u biblioteci string.h postoji funkcija strstr
 char* strstr(const char* s, const char* t)
 koja vraca adresu pocetka prvog pojavljivanja niske t u niski s ili
    NULL ako se niska t ne javlja
 u niski s.
 */

#include<stdio.h>
#define MAX 11

/*Funkcija koja proverava da li je t podniska od s*/
int podniska(char s[], char t[]){
  int i, j, k;

  /*Spoljna petlja ide redom po niski s*/
  for(i=0; s[i] != '\0'; i++){

  /*Unutrasnja petlja ide redom po niski t*/
  /*Promenljiva k pamti vrednost i, sluzi za poredjenje s[k] i t[j] i
   *zatim se vrsi pomeranje i po niski s i po niski t (k++, j++) */
  /*Cim naidjemo na situaciju da se karakteri ne poklapaju, izlazimo
    iz  unutrasnje petlje*/
  for(j=0, k = i; t[j] != '\0'; j++, k++){
    if(s[k] != t[j])
    break;
  }

  /*Ako smo prosli celu unutrasnju petlju (do kraja niske t), to
    znaci da su se svi karakteri iz t poklopili
   sa karakterima iz s i onda vracamo 1*/
  if(t[j] == '\0')
    return 1;
  }

  /*Na kraju vracamo 0*/
  return 0;
}


int main(){
  char s[MAX], t[MAX];

  /*Ucitavamo prvu nisku*/
  printf("Unesite nisku s: ");
```

```
46    scanf("%s", s);

48    /*Ucitavamo drugu nisku*/
      printf("Unesite nisku t: ");
50    scanf("%s", t);

52    /*Pozivamo funkciju i ispisujemo odgovarajucu poruku*/
      if(podniska(s,t))
54    printf("t je podniska niske s!\n");
      else
56    printf("t nije podniska niske s!\n");

58    return 0;
}
```

Rešenje 3.5.8

```
/*
2     Napisati funkciju skrati koja uklanja beline sa
      kraja datog stringa.
4
      Napisati glavni program koji testira napisanu
6     funkciju na stringu "rep belina                              ".

8  */

10 #include <stdio.h>
   #include <ctype.h>
12
   /*
14     Funkcija koja racuna duzinu niza
       ne racunajuci '\0'.
16
       U biblioteci string.h definisan je veliki
18     broj funkcija za rad sa stringovima,
       ukljucujuci i funkciju strlen koja racunana
20     duzinu stringa.

22     Funkcija strlen_klon predstavlja jednu
       implementaciju funkcije strlen.
24
       U zadacima cemo uvek koristiti ugradjenu
26     funkciju strlen osim ako u tekstu zadatka
       nije naglaseno da se ona ne sme koristiti.
28     Funkcija strlen_klon sluzi da pokaze na koji
       nacin radi ugradjena funkcija strlen.
30
       Ugradjena funkcija strlen poziva se na
32     isti nacin kao funkcija strlen_klon:
       strlen(s1)
34
```

```
*/
int strlen_klon(char s[])
{
  int i=0;
  while(s[i]) /* ASCII kod karaktera '\0' je jednak nuli. */
     i++;

  return i;
}

void skrati(char s[])
{
  /*
     Poslednji karakter stringa s(ne racunajuci '\0') ima
     indeks strlen_klon(s)-1; ideja je da pocnemo od poslednjeg
     karaktera stringa i da smanjujemo indeks dokle god
     je karakter na poziciji i blanko znak.

  */
  int i;
  for(i=strlen_klon(s)-1;i>=0;i--)
      if (!isspace (s[i])) /* Ako s[i] nije blanko znak, prekidamo
    petlju. */
          break;

  s[i+1]='\0';  /* DOdajemo terminirajucu nulu iza indeksa i (prvi
    neblanko karakter gledano sdesna nalevo).*/

  /*
     Ugradjena funkcija isspace nalazi se u biblioteci ctype.h i
    vraca 1 ako
     je dati karakter blanko znak a 0 u suprotnom.

     Unarni logicki operator ! oznacava negaciju.

  */
}

int main()
{

  /*
     Ukoliko string ne zelimo da ucitavamo po pokretanju programa
     vec da ga unapred zadamo, to mozemo uraditi na sledeci nacin:
  */
  char s[]="rep belina                              ";
  /* U ovom slucaju nije neophodno navoditi dimenziju stringa vec
     ce ona biti automatski postavljena na broj karaktera u stringu +
     1 za
     terminirajucu nulu. */


```

```
  printf("Pre skracivanja: *%s*\n", s);
84  skrati(s);
  printf("Posle skracivanja: *%s*\n", s);
86
  return 0;
88 }
```

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.11

```
1 /*
    Napisati program koji ucitava string src i formira string dst
3   trostrukim nadovezivanjem stringa src. Program treba da ispise
    string dst. Na primer, za uneti string "dan", string dst treba
5   da bude "dandandan". Pretpostaviti da string src nije duzi od
    30 karaktera.
7 */

9 #include <stdio.h>
  #include <string.h>
11
  #define MAX 30
13 /*
     Na stringove ne mozemo primeniti naredbu dodele.
15   Ukoliko zelimo da jedan string "dodelimo" drugom,
     mozemo koristiti ugradjenu funkciju strcpy(s,t)
17   koja kopira karaktere stringa t
     u string s zajedno za terminirajucom nulom.
19
     Funkcija strcpy se nalazi u biblioteci string.h.
21
     Funkcija strcpy_klon predstavlja jednu
23   implementaciju funkcije strcpy.

25   Karakteri stringa original se, jedan po jedan,
     kopiraju u string kopija. Nakon kopiranja,
27   na kraj stringa kopija dodaje se terminalna
     nula.
29
     U zadacima cemo uvek koristiti ugradjenu
31   funkciju strcpy osim ako u tekstu zadatka
     nije naglaseno da se ona ne sme koristiti.
33   Funkcija strcpy_klon sluzi da pokaze na koji
     nacin radi ugradjena funkcija strcpy.
35
     Ugradjena funkcija strcpy poziva se na
```

```
37      isti nacin kao funkcija strcpy_klon:
        strcpy(dst,src)
39      gde karaktere stringa src kopiramo
        u string dst.
41

43  */
    void strcpy_klon(char kopija[], char original[])
45  {
      int i;
47    for(i=0; original[i]; i++)
        kopija[i]=original[i];
49
      kopija[i] = '\0';
51  }

53  int main()
    {
55    char src[MAX+1];   /* src, skraceno od source (izvor, odnosno sta
        kopiramo) */
      char dst[3*MAX+1];   /* dst, skraceno od destination (odrediste,
        odnosno gde kopiramo) */
57
      /*
59       Vazno je izdvojiti dovoljno memorijskog prostora
         za string dst: on treba da bude tri puta veci od
61       maksimalne duzine stringa src + jedan karakter za
         terminirajucu nulu.
63    */

65    printf("Unesi jedan string:");
      scanf("%s", src);
67
      strcpy_klon(dst,src);
69
      /*
71      Funkcija strcat(s,t) nadovezuje karaktere stringa
        t na kraj stringa s i novi string terminira
73      karakterom '\0' .

75      Funkcija strcat se nalazi u biblioteci string.h.
      */
77    strcat(dst,src);
      strcat(dst,src);
79
      printf("Kada nadovezemo string %s triput: %s\n",src,dst);
81
      return 0;
83  }
```

Rešenje 3.5.31

Rešenje 3.5.13

```c
/*
Napisati funkciju void kopiraj_n(char t[], char s[], int n) koja
    kopira najvise n karaktera niske s u nisku t.
Napisati i program koji ucitava nisku maksimalne duzine 20 karaktera
    i jedan ceo broj i testira rad napisane funkcije.
*/

#include <stdio.h>
#define MAX_DUZINA 20

void kopiraj_n(char t[], char s[], int n){
  int i;

  /* Brojac i oznacava tekucu poziciju u niski */
  /* Uslov i<n je neophodan zbog kopiranja najvise n karaktera */
  /* Uslov s[i]!='\0' (ili skraceno samo s[i]) je neophodan kako bi
    bili sigurni da na poziciji i postoji karakter u niski s */
  for(i=0; i<n && s[i]!='\0'; i++){
    t[i]=s[i];
  }

  /* Upisujemo terminirajucu nulu u novodobijenu nisku */
  t[i]='\0';
}


int main(){
  int n;
  char s[MAX_DUZINA+1], t[MAX_DUZINA+1];

  /* Ucitavamo nisku */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Ucitavamo broj n i proveravamo korektnost unosa */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<0 || n>MAX){
    printf("Greska: pogresan unos!\n");
    return 0;
  }

  /* Pozivamo funkciju za kopiranje */
  kopiraj_n(t, s, n);

  /* Ispisujemo dobijenu nisku */
  printf("Rezultujuca niska: %s\n", t);

  return 0;
}
```

Rešenje 3.5.14

```c
/*
Napisati funkciju void dupliranje(char t[], char s[]) koja na osnovu
    niske s formira nisku t tako sto duplira svaki
karakter niske s. Napisati i program koji ucitava nisku maksimalne
    duzine 20 karaktera i testira rad napisane funkcije.
*/

#include <stdio.h>
#define MAX_DUZINA 20

void dupliranje(char t[], char s[]){
  int i, j;

  /* Brojac i oznacava tekucu poziciju u niski s */
  /* Brojac j oznacava tekucu poziciju u niski t */
  for(i=0, j=0; s[i]!='\0'; i++, j+=2){
    t[j]=s[i];
    t[j+1]=s[i];
  }

  /* Upisujemo terminirajucu nulu u novodobijenu nisku */
  t[j]='\0';
}


int main(){
  int n;
  char s[MAX_DUZINA+1], t[2*MAX_DUZINA+1];

  /* Ucitavamo nisku */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Pozivamo funkciju za dupliranje */
  dupliranje(t, s);

  /* Ispisujemo dobijenu nisku */
  printf("%s\n", t);

  return 0;
}
```

Rešenje 3.5.15

```c
/*
```

```
  Napisati program koji pretvara nisku u ceo broj.
  Npr. za ulaz "-1238" se generise rezultat -1238
  Pogledati funkcije atoi i atof koje postoje u biblioteci stdlib.h
*/

#include <stdio.h>
#include <ctype.h>
#define MAX 10
/*
  String b se sastoji od karaktera koji
  cine jedan ceo broj, onim redom kojim
  se karakteri pojavljuju u zapisu broja.

  Ako je prvi karakter stringa b '-',
  to znaci da je broj negativan i
  funkcija znak_broja vraca -1

  U suprotnom, broj je pozitivan i
  funkcija znak_broja vraca 1

*/

int znak_broja(char b[])
{
   if(b[0]=='-')
      return -1;
   return 1;
}


/*
  Funkcija formiraj_broj na osnovu
  karaktera koji cine broj iz stringa
  b vraca ceo broj koji odgovara
  zapisu datom u stringu b.

  Ako su cifre broja a,b,c i d, tada
  broj mozemo kreirati kao:
  a*10^3 + b*10^2 + c*10^1 + d*10^0

  Medjutim, efikasnije je koristiti
  Hornerovu semu:

  10*(10*(10*(10*0 + a)+b)+c)+d

*/

int formiraj_broj(char b[])
{
   int i;
   int n=0;
   int znak = znak_broja(b);
```

```
54
    /*
56      Ako je broj negativan, cifre u nizu b
        pocinju od indeksa 1
58  */

60  i=0;
    if(znak==-1)
62      i=1;

64  /*
      Funkcija isdigit proverava da li je broj
66    cifra. Nalazi se u biblioteci ctype.h

68    Proveravamo da li je karakter u zapisu
      broja cifra kako bismo se osigurali
70    od nekorektnog unosa, npr ako korisnik
      unese -123abc. Ovaj unos je moguc jer
72    se vrsi sa scanf("%s",broj), gde unosimo
      karaktere do prvog blanko znaka
74
      Ako naidjemo na karakter koji nije cifra,
76    prekidamo petlju

78  */
    for(; b[i]!='\0'; i++)
80      if(isdigit(b[i]))
            n = n*10 + b[i] - '0';
82      else
            break;
84
    /* Formirani broj mnozimo znakom: */
86
    n*=znak;
88  return n;

90  }

92  int main()
    {
94      char broj[MAX];
        int n;
96
        /* Ucitavamo broj: */
98      scanf("%s", broj);

100     /* Ispisujemo rezultat: */
        printf("Broj zapisan kao int: %d\n", formiraj_broj(broj));
102
        return 0;
104 }
```

Rešenje 3.5.16

```
/*
  Napisati program koji pretvara zadatu broj u nisku.
  Npr. za broj -453 treba generisati nisku "-453"
*/

#include <stdio.h>
#include <string.h>
#define MAX 10
/*

   Funkcija transformisi_negativan vraca
   1 ako je broj negativan i 0 u suprotnom, a
   uz to, ako broj jeste negativan, funkcija
   treba da ga konvertuje u  njegovu apsolutnu
   vrednost. S obzirom da funkcija treba da vrati dve
   vrednosti, to realizujemo na sledeci nacin:
   1. indikator da li je broj negativan
   ce vratiti kao povratnu vrednost
   2. apsolutnu vrednost broja ce vratiti
   preko liste argumenata, zbog cega broj
   prenosimo preko pokazivaca

*/
int transformisi_negativan(int* pn)
{
   if(*pn<0)
   {
      *pn = -(*pn);
      return 1;
   }
   return 0;
}

int formiraj_niz_cifara(int n, char b[], int neg)
{
   int i=0;
   char cifra;

   do
   {
      cifra = n%10;

      /* Promenljiva b predstavlja string.
         Da bismo na neku poziciju u stringu
         upisali karakter koji odgovara nekoj
         cifri, npr '2', neophodno je da
         odgovarajucoj poziciji dodelimo vrednost
         ASCII koda te cifre, konkretno za '2'
         ASCII kod je '0'+2.

```

```
            Greska bi bila navesti b[i]=2
            jer 2 nije ASCII kod koji odgovara karakteru
            '2'.
        */
        b[i]=cifra+'0';

        n/=10;
        i++;
    } while(n);

    /* Ako je broj negativan, dodajemo znak minus: */
    if(neg)
    {
        b[i]='-';
        i++;
    }

    /* Svaki string se zavrsava terminirajucom nulom: */
    b[i]='\0';
}

void obrni(char s[])
{

    char t;
    int i,j;
    /*
     Karaktere stringa obrcemo tako sto razmenimo karaktere na
     pozicijama 0 i n-1,
     zatom 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
      druge
    */

    for(i=0,j=strlen(s)-1;i<j;i++, j--)
    {
          t = s[i];
          s[i] = s[j];
          s[j] = t;
    }

}

void broj_u_niz_cifara(int n, char broj[])
{
    int negativan;

    /* Odredjujemo znak broja: */
    negativan=transformisi_negativan(&n);

    /* Izdvajamo cifre broja i smestamo ih u niz: */
    formiraj_niz_cifara(n, broj, negativan);

```

```
    /* S obzirom da cifre izdvajamo sa kraja broja, u nizu ce biti u
     obrnutom redosledu.
       Na primer, za broj 234 niz ce sadrzati cifre 4 3 2. */
    obrni(broj);
}

int main()
{
    int n;
    char broj[MAX];
    int negativan;

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

    /* Kreiramo broj na osnovu niza cifara: */
    broj_u_niz_cifara(n,broj);

    /* Ispisujemo rezultat: */
    printf("Broj zapisan kao string: %s\n", broj);

    return 0;
}
```

Rešenje 3.5.17

```
/*
 Napisati funkciju int heksa_broj(char s[]) koja proverava da li je
    niskom s zadat korektan heksadekadni broj.
 Heksadekadni broj je korektno zadat ako pocinje prefiksom 0x ili 0X
    i ako sadrzi samo cifre i mala ili velika slova A, B, C, D, E i F
    .
 Funkcija treba da vrati vrednost 1 ako je niska korektan
    heksadekadni broj, odnosno 0 ako nije.
 Napisati i program koji ucitava nisku maksimalne duzine 7 karaktera
    i ispisuje rezultat rada funkcije.
 */

#include<stdio.h>
#define MAX 8


/*
 Funkcija koja proverava da li je prosledjeni karakter ispravna
    heksadekadna cifra (broj ili slovo a,b,c,d,e,f)
 Ukoliko jeste, funkcija vraca 1, u suprotnom 0.
 */
int heksa_cifra(char c){
  /*Pretvaramo karakter c u veliko slovo*/
  c = toupper(c);

```

```
  /*Proveravamo da li je u pitanju cifra ili slovo A,B,C,D,E,F i
    ukoliko jeste, vracamo 1*/
  if(isdigit(c) || (c >= 'A' && c <= 'F'))
  return 1;

  /*Ukoliko nije, vracamo 0*/
  return 0;
}

/*Funkcija koja proverava da li prosledjena niska s predstavlja
    ispravan heksadekadni broj */
int heksa_broj(char s[]){
  int i;

  /*Kako heksadekadni brojevi pocinju sa 0x ili 0X, prvo proveravamo
    da li je taj uslov ispunjen,
  tj. da li je s[0] jednak 0 i da li je s[1] jednak X i ako taj uslov
     nije ispunjen, onda
  niska s ne predstavlja korektan heksadekadni broj */
   if(s[0] != '0' || toupper(s[1]) != 'X')
   return 0;

   /*Prolazimo kroz nisku, pocev od pozicije 2 (jer su prve dve
    pozicije 0x) i za svaki karakter do kraja
   niske proveravamo da li je ispravna heksadekadna cifra.
   Ako naidjemo na bilo koji koji ne ispunjava taj uslov, onda niska
    s nije korektan heksadekadni broj
   i vracamo 0. */
   for(i=2; s[i] != '\0'; i++)
   if(!heksa_cifra(s[i]))
     return 0;

   /*Ako smo stigli do kraja, znaci da su svi karakteri date niske
    ispravne heksadekadne cifre
     i zato vracamo 1 */
   return 1;
}

int main(){
  char s[MAX];

  /*Ucitavamo nisku*/
  printf("Unesite nisku:");
  scanf("%s", s);

  /*Pozivamo funckiju i stampamo odgovarajucu poruku*/
  if(heksa_broj(s))
  printf("Korektan heksadekadni broj!\n");
  else
  printf("Nekorektan heksadekadni broj!\n");

  return 0;
```

```
65 }
```

Rešenje 3.5.18

```
/* Napisati funkciju int heksa_broj(char s[]) koja izracunava dekadnu
     vrednost heksadekadnog broja zadatog niskom s.
 * Napisati i program koji ucitava nisku maksimalne duzine 7
    karaktera i ispisuje rezultat rada funkcije.
 * Pretpostaviti da je uneta niska korektan heksadekadni broj. */

#include<stdio.h>
#include<string.h>

#define MAX 8

/*Funkcija koja racuna dekadnu vrednost heksadekadnog broja*/
int heksa_broj(char s[]){
  int i,k;
  char c;

  /*Racunamo duzinu niske koja predstavlja heksadekadni broj*/
  int n = strlen(s);

  /*Inicijalizujemo vrednost v na 0*/
  int v = 0;

  /*Prolazimo petljom kroz nisku, krenuvsi sa desne strane
  npr: 1a8e = e*1 + 8*16 + a*256 + 1*4096
  Promenljiva k ce nam biti mnozilac i ona uzima vrednosti 1, 16,
    256, 4096, ...
  Promenljiva c ce nam cuvati trenutnu heksadekadnu cifru (u nasem
    primeru e, 8, a, 1)
  U svakom koraku treba na ispravan nacin da pomnozimo c i k
  */
  for(i=n-1, k=1; i>=2; i--, k*=16)
  {
  /*U c smestamo trenutnu heksadekadnu cifru.
   Pozivamo funkciju toupper da bi obezbedili da radimo samo sa
    velikim slovima.
   Ako je s[i] cifra, funkcija toupper je nece promeniti.
   */
  c = toupper(s[i]);

  if(isdigit(c)){
    /*Ako je c cifra, onda samu vrednost te cifre dobijamo sa c-'0'
     NAPOMENA: Nije ispravno napisati c*k jer je c karakter!
     */
    v += (c-'0')*k;
  }else{
    /*Ako je c slovo izmedju A i F, mi treba da dobijemo odgovarajucu
     vrednost izmedju 10 i 15.
```

```
     Ova vrednost se dobija sa 10 + c - 'A'. npr. za A ce biti 10 + '
    A' - 'A' = 10, za B: 10 + 'B' - 'A' = 11, ...*/
    v += (c - 'A' + 10)*k;
  }
  }

  /*Na kraju vracamo izracunatu vrednost */
   return v;
}

int main(){
  char s[MAX];

  /*Ucitavamo nisku*/
  printf("Unesite nisku:");
  scanf("%s", s);

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

  return 0;
}
```

Rešenje 3.5.19

```
/*
   Napisati funkciju int ucitaj_liniju(char s[], int n)
   koja ucitava liniju maksimalne duzine n u string s
   i vraca duzinu ucitane linije. Linija moze da sadrzi
   blanko znakove ali ne moze da sadrzi \n ili EOF.

   Napisati potom glavni program koji ucitava linije
   do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko
   ima vise linija maksimalne duzine, ispisati prvu. Mozemo
   pretpostviti da svaka linija sadrzi najvise 80 karaktera,
   zajedno sa \n.

*/

#include<stdio.h>
#include<string.h>
#define MAX 81

/*
   Ukoliko zelimo da ucitamo string koji sadrzi beline
   (npr liniju teksta), ne mozemo koristiti funkciju
   scanf jer ona ucitava string do prvog blanko znaka.

   Zbog toga je neophodno napisati funkciju koja ucitava
   string karakter po karakter.
```

```
27    Ova funkcija ne dopusta unosenje vise karaktera od
      unapred odredjene granice (argument n).
29
      U standardnoj biblioteci stdio.h postoji definisana
31    funkcija char *gets(char *s) koja ucitava karaktere
      dok se ne pojavi novi red ili EOF. Ova funkcija
33    dopusta unosenje vise karaktera nego sto string
      s sadrzi, sto moze dovesti do neocekivanog ponasanja
35    programa.

37    Pored funkcije gets, koja vrsi ucitavanje sa standardnog
      ulaza, u standardnoj biblioteci stdio.h postoji
39    i ugradjena funkcija fgets koja vrsi ucitavanje iz
      datoteke. Nju cemo koristiti za nekoliko casova
41    kada budemo radili datoteke. Prototim funkcije fgets je
      ovakav:
43
      char *fgets(char *s, int size, FILE *stream);
45
      Argumenti funkcije fgets su:
47    s - string u koji vrsimo ucitavanje
      size - maksimalna duzina unetog stringa
49    stream - datoteka iz koje vrsimo ucitavanje

51    Funkcija fgets, za razliku od funkcije gets, ne dopusta
      unos vise karaktera od date vrednosti size. Zbog toga
53    je ona sigurnija nego funkcija gets. Funkciju fgets
      mozemo koristiti i za unos sa standardnog ulaza
55    ukoliko kao treci argument navedemo stdin.

57 */
   int ucitaj_liniju(char s[], int n)
59 {
     int i=0;
61   int c;

63   while((c=getchar())!='\n' && i<n-2 && c!=EOF)
     {
65     s[i] = c;
       i++;
67   }

69   /* Ucitavamo najvise n-2 karaktera jer na kraju dodajemo jos
        dva: '\n' i '\0' */
71
     s[i]='\n';
73   s[i+1]='\0';

75   return i;

77 }
```

```
int main()
{
  char linija[MAX];
  char najduza_linija[MAX];
  int max_duzina=0;
  int duzina;

  /*
     Petlja se zavrsava ukoliko je promenljiva duzina
     jednaka nuli, sto cemo postici zadavanjem linije koja ne sadrzi
     nijedan karakter osim EOF.
  */

  while ((duzina=ucitaj_liniju(linija, MAX))>0)
  {
     /*
        Proveravamo da li je uneta linija duza od trenutnog
        maksimuma i azuriramo promenljive max_duzina i najduza_linija
     .
      */
     if (max_duzina<duzina)
     {
        max_duzina = duzina;
        strcpy(najduza_linija,linija);
     }
  }

  printf("Najduza linija: %s duzine: %d\n", najduza_linija,
    max_duzina);

  return 0;
}
```

Rešenje 3.5.31

Rešenje 3.5.21

Rešenje 3.5.22

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.31

Rešenje 3.5.31

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 greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti matricu celih brojeva:
 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:
 Uneti dimenzije matrice: 5 3
 Uneti matricu celih brojeva:
 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:
 Uneti dimenzije matrice: 500 3
   Neispravna dimenzija matrice.
```

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 greške ispisati odovarajuću poruku. Uputstvo: *Euklidska norma matrice je kvadratni koren sume kvadrata svih elemenata matrice.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenzije matrice: 3 4
 Uneti matricu celih brojeva:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Euklidska norma je: 25.495
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenzije matrice: 5 3
 Uneti matricu celih brojeva:
 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:
 Uneti dimenzije matrice: 500 3
   Neispravna dimenzija matrice.
```

Zadatak 3.7.3 Napisati funkcije za rad sa celobrojnim matricama:

(a) `void ucitavanje(int mat[][MAX], int* n, int* m)` kojom se prvo učitava dimezija matrice (brojevi m i n), a potom i vrednosti matrice celih brojeva mat,

(b) `void ispis(int mat[][MAX], int n, int m)` kojom se ispisuju vrednosti matrice mat 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 greške ispisati odovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenzije matrice: 3 4
 Uneti matricu celih brojeva:
 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:
 Uneti dimenzije matrice: 5 3
 Uneti matricu celih brojeva:
 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:
 Uneti dimenzije matrice: 500 3
   Neispravna dimenzija matrice.
```

Zadatak 3.7.4 Napisati funkciju `void transponovana(int a[][max], int m, int n, int b[][max])` koja određuje matricu b koja je dobijena transponovanjem matrice a. Napisati program koji za učitanu matricu celih brojeva[1] ispisuje odgvarajuću transponovanu matricu. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti matricu celih brojeva:
 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:
 Uneti dimenzije matrice: 5 3
 Uneti matricu celih brojeva:
 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:
 Uneti dimenzije matrice: 500 3
   Neispravna dimenzija matrice.
```

Zadatak 3.7.5 Napisati funkciju `void razmeni(int mat[][max], int m, int n, int k, int t)` u kojoj se razmenjuju elemeti k–te i t–te vrste matrice *mat* 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 greške ispisati odovarajuću poruku.

[1] Pod pojmom *učitati matricu* ili *za datu matricu* uvek se podrazumeva da se prvo unose dimenzije matrice, a potom i sama matrica.

```
- - - - - - 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.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenzije matrice: 3 4
 Uneti matricu celih brojeva:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Uneti indekse vrsta: 0 2
 9 10 11 12
 5 6 7 8
 1 2 3 4
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenzije matrice: 5 3
 Uneti matricu celih brojeva:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Uneti indekse vrsta: 1 3
 1 1 2
 1 2 4
 7 8 9
 5 0 2
 0 1 1
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenzije matrice: 5 3
 Uneti matricu celih brojeva:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Uneti indekse vrsta: -1 50
 Neispravni indeksi vrsta.
```

Zadatak 3.7.6 Napisati program koji za učitanu matricu celih brojeva ispisuje indekse onih elemenata matrice koji su jednaki zbiru svih svojih susednih elemenata. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju greške ispisati odovarajuću poruku. 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:
 Uneti dimenzije matrice: 4 5
 Uneti 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:
 Uneti dimenzije matrice: 3 4
 Uneti 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
```

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 greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenzije matrice: 4 5
 Uneti 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:
 Uneti dimenzije matrice: 3 4
 Uneti elemente matrice:
 7 10 12 20
 -1 -3 1 7
 0 -47 2 0
 Dobijeni niz je:
 12.25 1 -11.25
```

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 broj 1, a nije u relaciji ukoliko se tu nalazi broj 0. Napisati funkcije:

(a) `int refleksivna(int a[][MAX], int n)` kojom se za relaciju zadatom matricom `a` dimenzije $n \times n$ ispituje da li je refleksivna;

(b) `int simetricna(int a[][MAX], int n)` kojom se za relaciju zadatom matricom `a` dimenzije $n \times n$ ispituje da li je simetrična;

(c) `int tranzitivna(int a[][MAX], int n)` kojom se za relaciju zadatom matricom `a` ispituje dimenzije $n \times n$ da li je tranzitivna;

(d) `int ekvivalencija(int a[][MAX], int n)` kojom se za relaciju koja je zadata metricom `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 može imati za vrednosti elemenata samo brojeve 0 i 1. U slučaju greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 4
 Uneti 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:
 Uneti dimenziju matrice: 4
 Uneti 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.
```

Zadatak 3.7.9 Data je kvadratna matrica dimenzije $n \times n$.

(a) Napisati funkciju `float trag(float a[][MAX], int n)` koja računa trag matrice, odnosno zbir elemenata na glavnoj dijagonali matrice.

(b) Napisati funckciju `float suma_sporedna(float a[][MAX], int n)` koja računa zbir elemenata na sporednoj dijagonali matrice.

(c) Napisati funkciju `float suma_iznad(float a[][MAX], int n)` koja određuje sumu elememenata iznad glavne dijagonale.

(d) Napisati funkciju `float suma_ispod(float a[][MAX], 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 greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 4
 Uneti 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 53.200.
 Suma na sporednoj dijagonali je 0.900.
 Suma iznad glavne dijagonale je 31.700.
 Suma ispod sporedne dijagonale je -1.820.
```

Primer 2

```
Interakcija sa programom:
 Uneti dimenziju matrice: 5
 Uneti 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 21.000.
 Suma na sporednoj dijagonali je 17.000.
 Suma iznad glavne dijagonale je 33.000.
 Suma ispod sporedne dijagonale je 24.000.
```

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 greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 5
 Uneti 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:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 2 -2 1
 1 2 2
 2 1 -2
 Matrica nije donje trougaona.
```

Primer 3

```
Interakcija sa programom:
 Uneti dimenziju matrice: 200
 Neispravna dimenzija matrice.
```

Primer 4

```
Interakcija sa programom:
 Uneti dimenziju matrice: 4
 Uneti elemente matrice:
 2 0 0 0
 7 80 0 0
 -9 4 4 0
 14 23 -8 1
 Matrica jeste donje trougaona.
```

Zadatak 3.7.11 Napisati program koji za učitanu kvadratnu matricu ispisuje redni broj kolone koja ima najveći zbir elemenata. Pretpostaviti da je maksimalna dimenzija matrice 50×50. U slučaju greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 1 2 3
 7 3 4
 5 3 1
 Indeks kolone je: 0
```

Primer 2

```
Interakcija sa programom:
 Uneti dimenziju matrice: 4
 Uneti elemente matrice:
 7 8 9 10
 7 6 11 4
 3 1 2 -2
 8 3 9 9
 Indeks kolone je: 2
```

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 greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 2 3.2 4
 7 8.8 1
 2.3 1 1
 Razlika je: 2.20
```

Primer 2

```
Interakcija sa programom:
 Uneti dimenziju matrice: 5
 Uneti 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
```

Zadatak 3.7.13 Napisati program koji za učitanu 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 greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti matricu celih brojeva:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Uneti dva cela broja:  3 3
 Sume podmatrica su: 54 63
```

Primer 2

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti matricu celih brojeva:
 1 2 3 4
 5 6 7 8
 9 10 11 12
 Uneti dva cela broja:  2 3
 Sume podmatrica su: 24 30 48 54
```

Primer 3

```
Interakcija sa programom:
 Uneti dimenzije matrice: -3 200
 Neispravna dimenzija matrice.
```

Primer 4

```
Interakcija sa programom:
 Uneti dimenzije matrice: 5 3
 Uneti matricu celih brojeva:
 1 1 2
 5 0 2
 7 8 9
 1 2 4
 0 1 1
 Uneti dva cela broja:  2 2
 Sume podmatrica su: 7 5 20 19 18 23 4 8
```

Zadatak 3.7.14 Napisati program koji za učitanu 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 greške ispisati odovarajuću poruku.

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.
```

Zadatak 3.7.15 Napisati program koji za učitanu kvadratnu matricu ispituje da li su zbirovi elemenata njenih kolona uređeni u strogo rastućem poretku. Pretpostaviti da je maksimalna dimenzija matrice koja se unosi 10×10. U slučaju greške ispisati odovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 4
 Uneti 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:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 1 2 3
 4 5 6
 7 8 9
 Sume jesu uredjenje strogo rastuce.
```

Primer 3

```
Interakcija sa programom:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 2 -2 1
 1 2 2
 2 1 -2
 Sume nisu uredjenje strogo rastuce.
```

Primer 4

```
Interakcija sa programom:
 Uneti dimenziju matrice: 5
 Uneti 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.
```

Zadatak 3.7.16 Matrica je *ortonormirana* ako je skalarni proizvod svakog para različitih vrsta jednak 0, a skalarni proizvod vrste sa samom sobom 1. Napisati program koji za unetu kvadratnu matricu proverava da li je ortonormirana. U slučaju greške ispisati odgovarajuću poruku. 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:
 Uneti dimenziju matrice: 4
 Uneti 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:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 1 2 3
 4 5 6
 7 8 9
 Matrica nije ortonormirana.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 2 -2 1
 1 2 2
 2 1 -2
 Matrica nije ortonormirana.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenziju matrice: 5
 Uneti 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.
```

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 kvadratna matrica magični kvadrat i ispisuje odgovarajuću poruku na standardni izlaz. Pretpostaviti da je maksimalna dimenzija matrice je 50×50. U slučaju greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Uneti dimenziju matrice: 4
 Uneti 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:
 Uneti dimenziju matrice: 3
 Uneti elemente matrice:
 1 2 3
 4 5 6
 -1 3 3
 Matrica nije magicni kvadrat.
```

*** Zadatak 3.7.18** Ispisati elemente učitane kvadratne matrice celih brojeva u grupama koje su paralelne sa sporednom dijagonalom matrice počevši od gornjeg levog ugla matrice. Pretpostaviti da je maksimalna dimenzija matrice 100×100. U slučaju greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 3
 Uneti 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:
 Uneti dimenziju matrice: 5
 Uneti 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
```

*** Zadatak 3.7.19** Napisati funkciju `void mnozenje(int a[][max], int m, int n, int b[][max], int k, int t, int c[][max])` 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. Ukoliko množenje matrica nije moguće ili je došlo do greške prilikom unosa podataka ispisati odgovarajuću poruku. Maksimalna dimenzija obe matrice je 50×50.

Primer 1

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti elemente matrice:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Uneti dimenzije matrice: 4 2
 Uneti elemente matrice:
 11 5
 6 7
 8 9
 0 -3
 Rezultat mnozenja je:
 87 64
 2 24
 145 83
```

Primer 2

```
Interakcija sa programom:
 Uneti dimenzije matrice: 5 2
 Uneti elemente matrice:
 1 7
 9 0
 -10 2
 92 3
 14 -8
 Uneti dimenzije matrice: 2 4
 Uneti elemente matrice:
 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 2

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti elemente matrice:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Uneti dimenzije matrice: 5 2
 Uneti elemente matrice:
 11 5
 6 7
 8 9
 0 -3
 4 4
 Mnozenje matrica nije moguce.
```

*** 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 greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenzije matrice: 2 3
 1 2 3
 0 5 6
 Sedlo: 0 0 1
```

Primer 2

```
Interakcija sa programom:
 Uneti dimenziju matrice: 3 3
 10 3 20
 15 5 100
 30 -1 200
 Sedlo: 1 1 5
```

*** 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 greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti dimenziju matrice: 3 3
 Uneti 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:
 Uneti dimenziju matrice: 5 7
 Uneti 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
```

*** 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.

Primer 1

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti elemente matrice:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Uneti dimenzije matrice: 2 2
 Uneti elemente matrice:
 2 3
 4 10
 Druga matrica je sadrzana u prvoj matrici.
```

Primer 2

```
Interakcija sa programom:
 Uneti dimenzije matrice: 3 4
 Uneti elemente matrice:
 1 2 8 9
 -4 5 2 3
 7 6 4 10
 Uneti dimenzije matrice: 2 2
 Uneti elemente matrice:
 2 8
 6 4
 Druga matrica nije sadrzana u prvoj matrici.
```

Primer 3

```
Interakcija sa programom:
 Uneti dimenzije matrice: 5 5
 Uneti 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
 Uneti dimenzije matrice: 3 4
 Uneti elemente matrice:
 90 11 0 5
 12 -9 14 23
 80 6 88 17
 Druga matrica je sadrzana u prvoj matrici.
```

3.8 Rešenja

Rešenje 3.7.1

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

#define MAX 50

int main()
{
  int mat[MAX][MAX];
  int m, n;

  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", &m, &n);

  if (n <= 0 || n > MAX || m <= 0 || m > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<m; i++)
    for(j=0; j<n; j++)
      scanf("%d", &mat[i][j]);

  /* Ispis elemenata matrice. */
  for(i=0; i<m; i++)
  {
    for(j=0; j<n; j++)
      printf("%d ", mat[i][j]);
    printf("\n");
  }

  return 0;
}
```

Rešenje 3.7.2

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

#define MAX 50
```

```c

int main()
{
  int mat[MAX][MAX];
  int m, n;
  int suma = 0;

  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", &m, &n);

  if (n <= 0 || n > MAX || m <= 0 || m > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<m; i++)
    for(j=0; j<n; j++)
      scanf("%d", &mat[i][j]);


  for(i=0; i<m; i++)
    for(j=0; j<n; j++)
      suma += mat[i][j] * mat[i][j];

  printf("Euklidska norma je %.3lf.\n", sqrt(suma));

  return 0;

}
```

Rešenje 3.7.3

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

#define MAX 50

void ucitavanje(int mat[][MAX], int* m, int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", m, n);

  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
  {
```

```
15      printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
17    }

19    printf("Uneti matricu celih brojeva\n");

21    for(i=0; i<*m; i++)
        for(j=0; j<*n; j++)
23        scanf("%d", &mat[i][j]);
  }
25
  void ispis(int mat[][MAX], int m, int n)
27{
    int i, j;
29
    for(i=0; i<m; i++)
31  {
      for(j=0; j<n; j++)
33      printf("%d ", mat[i][j]);
      printf("\n");
35  }
  }
37
  int main()
39{
    int mat[MAX][MAX];
41  int m, n;

43  ucitavanje(mat, &m, &n);
    ispis(mat, m, n);
45
    return 0;
47}
```

Rešenje 3.7.4

```
1 #include <stdio.h>
  #include <stdlib.h>
3
  #define MAX 50
5
  void ucitavanje(int mat[][MAX], int* m, int* n)
7 {
    int i, j;
9
    printf("Uneti dimenzije matrice: ");
11  scanf("%d%d", m, n);

13  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
    {
15    printf("Neispravna dimenzija matrice\n");
```

```
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*m; i++)
    for(j=0; j<*n; j++)
      scanf("%d", &mat[i][j]);
}

void ispis(int mat[][MAX], int m, int n)
{
  int i, j;

  for(i=0; i<m; i++)
  {
    for(j=0; j<n; j++)
      printf("%d ", mat[i][j]);
    printf("\n");
  }
}

void transponovana(int a[][MAX], int m, int n, int b[][MAX])
{
  int i, j;

  for(i=0; i<m; i++)
    for(j=0; j<n; j++)
      b[j][i] = a[i][j];
}

int main()
{
   int mat[MAX][MAX], t[MAX][MAX];
  int m, n;

  ucitavanje(mat, &m, &n);
  transponovana(mat, m, n, t);
  ispis(t, n, m);

  return 0;
}
```

Rešenje 3.7.5

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

#define MAX 50

void ucitavanje(int mat[][MAX], int* m, int* n)
```

```
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", m, n);

  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*m; i++)
    for(j=0; j<*n; j++)
      scanf("%d", &mat[i][j]);
}

void ispis(int mat[][MAX], int m, int n)
{
  int i, j;

  for(i=0; i<m; i++)
  {
    for(j=0; j<n; j++)
      printf("%d ", mat[i][j]);
    printf("\n");
  }
}

void razmeni(int mat[][MAX], int m, int n, int k, int t)
{
  int j, pom;

  for(j = 0; j< n; j++)
  {
    pom = mat[k][j];
    mat[k][j] = mat[t][j];
    mat[t][j] = pom;
  }
}

int main()
{
  int mat[MAX][MAX];
  int m, n;
  int k, t;

  ucitavanje(mat, &m, &n);

  printf("Uneti indekse vrsta: ");
```

```
  scanf("%d%d", &k, &t);

  if (k < 0 || k >= m || t < 0 || t>= m)
  {
    printf("Neispravni indeksi vrsta.\n");
    return -1;
  }

  razmeni(mat, m, n, k, t);

  ispis(mat, m, n);

  return 0;
}
```

Rešenje 3.7.6

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

#define MAX 50

void ucitavanje(int mat[][MAX], int* m, int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", m, n);

  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*m; i++)
    for(j=0; j<*n; j++)
      scanf("%d", &mat[i][j]);
}


int main()
{
  int mat[MAX][MAX];
  int m, n, i, j, suma;
  int k, t;

  ucitavanje(mat, &m, &n);

```

```
  printf("Indeksi elemenata koji su jednaki zbiru suseda su:\n");
  for(i=0; i<m; i++)
    for(j=0; j<n; j++)
    {
      suma = 0;

      for(k=-1;k<=1; k++)
        for(t=-1; t<=1; t++)
          if (i+k >= 0 && i+k < n && j+t >= 0 && j+t < n)
            suma += mat[i+k][j+t];

      if (suma - mat[i][j] == mat[i][j])
        printf("%d %d\n", i, j);
    }

  return 0;
}
```

Rešenje 3.7.7

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

#define MAX 50

void ucitavanje(int mat[][MAX], int* m, int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", m, n);

  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*m; i++)
    for(j=0; j<*n; j++)
      scanf("%d", &mat[i][j]);
}

void kreiraj_niz(int mat[][MAX], int m, int n, double b[])
{
  int i, j, suma;

  for(i=0; i<m; i++)
  {
```

```
32      suma = 0;
        for(j=0; j<n; j++)
34        suma += mat[i][j];

36      b[i] = (double)suma/n;
    }
38  }

40  int main()
    {
42    int mat[MAX][MAX];
      double  b[MAX];
44    int m, n, i;

46    ucitavanje(mat, &m, &n);

48    kreiraj_niz(mat, m, n, b);

50    printf("Dobijeni niz je:\n");
      for(i=0; i<m; i++)
52      printf("%g ", b[i]);
      printf("\n");
54
      return 0;
56  }
```

Rešenje 3.7.8

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    #define MAX 50
5
    void ucitavanje(int mat[][MAX], int* n)
7   {
      int i, j;
9
      printf("Uneti dimenzije matrice: ");
11    scanf("%d", n);

13    if (*n <= 0 || *n > MAX)
      {
15      printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
17    }

19    printf("Uneti matricu celih brojeva\n");

21    for(i=0; i<*n; i++)
        for(j=0; j<*n; j++)
23        scanf("%d", &mat[i][j]);
```

```
}

int refleksivna(int a[][MAX], int n)
{
  int i;

  for(i=0; i<n; i++)
    if (a[i][i] != 1)
      return 0;

  return 1;
}

int simetricna(int a[][MAX], int n)
{
  int i, j;

  for(i=0; i<n; i++)
    for(j=0; j<n; j++)
      if (a[i][j] != a[j][i])
        return 0;

  return 1;
}

int tranzitivna(int a[][MAX], int n)
{
  int i, j, k;

  for(i=0; i<n; i++)
    for(j=0; j<n; j++)
      for(k=0; k<n; k++)
        if (a[i][j] == 1 && a[j][k] == 1 && a[i][k] == 0)
          return 0;

  return 1;
}

int ekvivalencija(int a[][MAX], int n)
{
  if (refleksivna(a, n) && simetricna(a, n) && tranzitivna(a, n))
    return 1;

  return 0;
}

int main()
{
  int a[MAX][MAX];
  int n;

  ucitavanje(a, &n);
```

```
77    if (refleksivna(a, n))
        printf("Relacija jeste refleksivna.\n");
79    else
        printf("Relacija nije refleksivna.\n");
81
    if (simetricna(a, n))
83      printf("Relacija jeste simetricna.\n");
    else
85      printf("Relacija nije simatricna.\n");

87  if (tranzitivna(a, n))
      printf("Relacija jeste tranzitivna.\n");
89  else
      printf("Relacija nije tranzitivna.\n");
91
    if (ekvivalencija(a, n))
93    printf("Relacija jeste ekvivalencija.\n");
    else
95    printf("Relacija nije ekvivalencija.\n");

97  return 0;
}
```

Rešenje 3.7.9

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

4 #define MAX 50

6 void ucitavanje(float mat[][MAX], int* n)
{
8   int i, j;

10  printf("Uneti dimenzije matrice: ");
    scanf("%d", n);
12
    if (*n <= 0 || *n > MAX)
14  {
      printf("Neispravna dimenzija matrice\n");
16    exit(EXIT_FAILURE);
    }
18
    printf("Uneti matricu celih brojeva\n");
20
    for(i=0; i<*n; i++)
22    for(j=0; j<*n; j++)
        scanf("%f", &mat[i][j]);
24 }
```

```
26 float trag(float a[][MAX], int n)
   {
28   float suma = 0;
     int  i;
30
     for(i=0; i<n; i++)
32     suma += a[i][i];

34   return suma;
   }
36
   float suma_sporedna(float a[][MAX], int n)
38 {
     float suma = 0;
40   int  i;

42   for(i=0; i<n; i++)
       suma += a[i][n-i-1];
44
     return suma;
46 }

48 float suma_iznad(float a[][MAX], int n)
   {
50   float suma = 0;
     int  i, j;
52
     for(i=0; i<n; i++)
54     for(j=i+1; j<n; j++)
         suma += a[i][j];
56
     return suma;
58 }

60 float suma_ispod(float a[][MAX], int n)
   {
62   float suma = 0;
     int  i, j;
64
     for(i=0; i<n; i++)
66     for(j=n-i-1; j>i; j--)
         suma += a[i][j];
68
     return suma;
70 }

72 int main()
   {
74   float a[MAX][MAX];
     int n;
76
     ucitavanje(a, &n);
```

```
78
   printf("Trag je %.3f.\n", trag(a, n));
80 printf("Suma na sporednoj dijagonali je %.3f.\n", suma_sporedna(a,
     n));
   printf("Suma iznad glavne dijagonale je %.3f.\n", suma_iznad(a, n))
     ;
82 printf("Suma ispod sporedne dijagonale je %.3f.\n", suma_ispod(a, n
     ));

84 return 0;
}
```

Rešenje 3.7.10

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

4 #define MAX 50

6 void ucitavanje(int mat[][MAX], int* n)
{
8   int i, j;

10  printf("Uneti dimenzije matrice: ");
    scanf("%d", n);
12
    if (*n <= 0 || *n > MAX)
14  {
      printf("Neispravna dimenzija matrice\n");
16    exit(EXIT_FAILURE);
    }
18
    printf("Uneti matricu celih brojeva\n");
20
    for(i=0; i<*n; i++)
22    for(j=0; j<*n; j++)
        scanf("%d", &mat[i][j]);
24 }

26 int donje_trougaona(int a[][MAX], int n)
{
28  int i, j;

30  for(i=0; i<n; i++)
      for(j=i+1; j<n; j++)
32      if (a[i][j] != 0)
          return 0;
34
    return 1;
36 }
```

```
38 int main()
   {
40   int a[MAX][MAX];
     int n;
42
     ucitavanje(a, &n);
44
     if (donje_trougaona(a, n))
46     printf("Matrica jeste donje trougaona.\n");
     else
48     printf("Matrica nije donje trougaona.\n");

50   return 0;
   }
```

Rešenje 3.7.11

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

4  #define MAX 50

6  void ucitavanje(int mat[][MAX], int* n)
   {
8    int i, j;

10   printf("Uneti dimenzije matrice: ");
     scanf("%d", n);
12
     if (*n <= 0 || *n > MAX)
14   {
       printf("Neispravna dimenzija matrice\n");
16     exit(EXIT_FAILURE);
     }
18
     printf("Uneti matricu celih brojeva\n");
20
     for(i=0; i<*n; i++)
22     for(j=0; j<*n; j++)
         scanf("%d", &mat[i][j]);
24 }

26 int main()
   {
28   int a[MAX][MAX];
     int n, i, j;
30   int max_zbir, trenutni_zbir = 0, indeks_kolone = 0;

32   ucitavanje(a, &n);

34   for(i=0; i<n; i++)
```

```
    trenutni_zbir += a[i][0];

  max_zbir = trenutni_zbir;

  for(j=1; j<n; j++)
  {

    trenutni_zbir = 0;
    for(i=0; i<n; i++)
      trenutni_zbir += a[i][j];

    if (trenutni_zbir > max_zbir)
    {
      max_zbir = trenutni_zbir;
      indeks_kolone = j;
    }
  }

  printf("Indeks kolone je: %d\n", indeks_kolone);

  return 0;
}
```

Rešenje 3.7.12

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

#define MAX 50

void ucitavanje(float mat[][MAX], int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d", n);

  if (*n <= 0 || *n > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*n; i++)
    for(j=0; j<*n; j++)
      scanf("%f", &mat[i][j]);
}

int main()
```

```
{
    float a[MAX][MAX];
    int n, i, j;
    float gornji_trougao = 0, donji_trougao = 0;

    ucitavanje(a, &n);

    for(i=0; i<n/2; i++)
      for(j=i+1; j<n-i-1; j++)
        gornji_trougao += a[i][j];

    for(i=n/2; i<n; i++)
      for(j=n-i; j<i; j++)
        donji_trougao += a[i][j];

    printf("%f %f\n", gornji_trougao, donji_trougao);

    printf("Razlika je: %.2f\n", gornji_trougao - donji_trougao);

    return 0;
}
```

Rešenje 3.7.13

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

#define MAX 50

void ucitavanje(int mat[][MAX], int* m, int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", m, n);

  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
  {
      printf("Neispravna dimenzija matrice\n");
      exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*m; i++)
      for(j=0; j<*n; j++)
    scanf("%d", &mat[i][j]);
}

int main()
{
```

```
28  int a[MAX][MAX];
    int n, i, j, m, x, y, p, k;
30  int suma;

32  ucitavanje(a, &m, &n);

34  printf("Uneti dva cela broja: ");
      scanf("%d%d", &p, &k);
36
      printf("Sume podmatrica su: ");
38    for(i=0; i<= m-p; i++)
        for(j=0; j<= n-k; j++)
40      {
          suma = 0;
42        for(x=0; x<p; x++)
            for(y=0; y<k; y++)
44            suma += a[i+x][j+y];

46        printf("%d ", suma);
        }
48
      printf("\n");
50
    return 0;
52  }
```

Rešenje 3.7.14

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    #define MAX 50
5
    void ucitavanje(int mat[][MAX], int* n)
7   {
      int i, j;
9
      printf("Uneti dimenzije matrice: ");
11    scanf("%d", n);

13    if (*n <= 0 || *n > MAX)
      {
15      printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
17    }

19    printf("Uneti matricu celih brojeva\n");

21    for(i=0; i<*n; i++)
        for(j=0; j<*n; j++)
23        scanf("%d", &mat[i][j]);
```

```
}

int sortirana_kolona(int mat[][MAX], int n, int j)
{
  int i;

  for(i=0; i<n-1; i++)
    if (mat[i][j] >= mat[i+1][j])
      return 0;

  return 1;
}

int sortirani_po_kolonama(int mat[][MAX], int n)
{
  int j;

  for(j=0; j<n; j++)
    if (!sortirana_kolona(mat, n, j))
      return 0;

  return 1;
}

int sortirana_vrsta(int mat[][MAX], int n, int i)
{
  int j;

  for(j=0; j<n-1; j++)
    if (mat[i][j] >= mat[i][j+1])
      return 0;

  return 1;
}

int sortirani_po_vrstama(int mat[][MAX], int n)
{
  int i;

  for(i=0; i<n; i++)
    if (!sortirana_vrsta(mat, n, i))
      return 0;

  return 1;
}

int sortirana_glavna(int mat[][MAX], int n)
{
  int i;

  for(i=0; i<n-1; i++)
    if (mat[i][i] >= mat[i+1][i+1])
```

```
      return 0;

  return 1;
}

int sortirana_sporedna(int mat[][MAX], int n)
{
  int i;

  for(i=0; i<n-1; i++)
    if (mat[i][n-i-1] >= mat[i+1][n-i-2])
      return 0;

  return 1;
}

int sortirani_po_dijagonalama(int mat[][MAX], int n)
{
  if (!sortirana_glavna(mat, n))
    return 0;

  if (!sortirana_sporedna(mat, n))
    return 0;

  return 1;
}

int main()
{
  int mat[MAX][MAX];
  int n;

  ucitavanje(mat, &n);

  if (sortirani_po_kolonama(mat, n))
    printf("Elementi su sortirani po kolonama.\n");
  else
    printf("Elementi nisu sortirani po kolonama.\n");

  if (sortirani_po_vrstama(mat, n))
    printf("Elementi su sortirani po vrstama.\n");
  else
    printf("Elementi nisu sortirani po vrstama.\n");

  if (sortirani_po_dijagonalama(mat, n))
    printf("Elementi su sortirani po dijagonalama.\n");
  else
    printf("Elementi nisu sortirani po dijagonalama.\n");

  return 0;
}
```

Rešenje 3.7.15

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

#define MAX 10

void ucitavanje(int mat[][MAX], int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d", n);

  if (*n <= 0 || *n > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*n; i++)
    for(j=0; j<*n; j++)
      scanf("%d", &mat[i][j]);
}

int suma_kolone(int mat[][MAX], int n, int j)
{
  int suma = 0, i;

  for(i=0; i<n; i++)
    suma += mat[i][j];

  return suma;
}

int uredjene_sume(int mat[][MAX], int n)
{
  int suma1, suma2;
  int j;

  suma1 = suma_kolone(mat, n, 0);

  for(j=1; j<n; j++)
  {
    suma2 = suma_kolone(mat, n, j);

    if (suma1 >= suma2)
      return 0;

    suma1 = suma2;
```

```
51    }
53    return 1;
}
55
int main()
57{
    int mat[MAX][MAX];
59  int n;
61  ucitavanje(mat, &n);
63  if (uredjene_sume(mat, n))
      printf("Sume jesu uredjenje strogo rastuce.\n");
65  else
      printf("Sume nisu uredjenje strogo rastuce.\n");
67
69  return 0;
}
```

Rešenje 3.7.16

```
1 #include <stdio.h>
  #include <stdlib.h>
3
  #define MAX 200
5
  void ucitavanje(int mat[][MAX], int* n)
7 {
    int i, j;
9
    printf("Uneti dimenzije matrice: ");
11  scanf("%d", n);
13  if (*n <= 0 || *n > MAX)
    {
15    printf("Neispravna dimenzija matrice\n");
      exit(EXIT_FAILURE);
17  }
19  printf("Uneti matricu celih brojeva\n");
21  for(i=0; i<*n; i++)
      for(j=0; j<*n; j++)
23      scanf("%d", &mat[i][j]);
}
25
  int skalarni_proizvod(int mat[][MAX], int n, int i, int j)
27{
    int suma = 0, k;
```

```
29
   for(k=0; k<n; k++)
31   suma += mat[i][k] * mat[j][k];

33 return suma;
}
35
int ortonormirana(int mat[][MAX], int n)
37 {
   int i, j;
39
   for(i=0; i<n; i++)
41   for(j=0; j<n; j++)
       if (i==j && skalarni_proizvod(mat, n, i, i) != 1)
43       return 0;
       else if (i != j && skalarni_proizvod(mat, n, i, j) != 0)
45       return 0;

47 return 1;
}
49
int main()
51 {
   int mat[MAX][MAX];
53 int n;

55 ucitavanje(mat, &n);

57 if (ortonormirana(mat, n))
     printf("Matrica jeste ortonormirana.\n");
59 else
     printf("Matrica nije ortonormirana.\n");
61
   return 0;
63 }
```

Rešenje 3.7.17

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAX 50
5
   void ucitavanje(int mat[][MAX], int* n)
7  {
     int i, j;
9
     printf("Uneti dimenzije matrice: ");
11   scanf("%d", n);

13   if (*n <= 0 || *n > MAX)
```

```
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*n; i++)
    for(j=0; j<*n; j++)
      scanf("%d", &mat[i][j]);
}

int suma_kolone(int mat[][MAX], int n, int j)
{
  int i, suma = 0;

  for(i=0; i<n; i++)
    suma += mat[i][j];

  return suma;
}

int suma_vrste(int mat[][MAX], int n, int i)
{
  int j, suma = 0;

  for(j=0; j<n; j++)
    suma += mat[i][j];

  return suma;
}

int magicni_kvadrat(int mat[][MAX], int n)
{
  int suma = suma_kolone(mat, n, 0);
  int i, j;

  for(j=1; j<n; j++)
    if (suma_kolone(mat, n, j) != suma)
      return 0;

  for(i=0; i<n; i++)
    if (suma_vrste(mat, n, i) != suma)
      return 0;

  return 1;
}

int main()
{
  int mat[MAX][MAX];
  int n;
```

```
67    ucitavanje(mat, &n);
69    if (magicni_kvadrat(mat, n))
        printf("Matrica jeste magicni kvadrat.\n");
71    else
        printf("Matrica nije magicni kvadrat.\n");
73
      return 0;
75  }
```

Rešenje 3.7.18

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

4   #define MAX 100

6   void ucitavanje(int mat[][MAX], int* n)
    {
8     int i, j;

10    printf("Uneti dimenzije matrice: ");
      scanf("%d", n);
12
      if (*n <= 0 || *n > MAX)
14    {
        printf("Neispravna dimenzija matrice\n");
16      exit(EXIT_FAILURE);
      }
18
      printf("Uneti matricu celih brojeva\n");
20
      for(i=0; i<*n; i++)
22      for(j=0; j<*n; j++)
          scanf("%d", &mat[i][j]);
24  }

26  int main()
    {
28    int mat[MAX][MAX];
      int n;
30    int i, j, k;

32    ucitavanje(mat, &n);

34    for(k=0; k<n; k++)
      {
36      j = k;
        i = 0;
38
```

```
    while(j >= 0)
    {
      printf("%d ", mat[i][j]);
      i++;
      j--;
    }

    printf("\n");
  }


  for(k=1; k<n; k++)
  {
    i = k;
    j = n-1;

    while(i < n)
    {
      printf("%d ", mat[i][j]);
      i++;
      j--;
    }

    printf("\n");
  }

  return 0;
}
```

Rešenje 3.7.19

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

#define MAX 50

void ucitavanje(int mat[][MAX], int* m, int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", m, n);

  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

```

```
  for(i=0; i<*m; i++)
    for(j=0; j<*n; j++)
      scanf("%d", &mat[i][j]);
}

void ispis(int mat[][MAX], int m, int n)
{
  int i, j;

  for(i=0; i<m; i++)
  {
    for(j=0; j<n; j++)
      printf("%d ", mat[i][j]);
    printf("\n");
  }
}

void mnozenje(int a[][MAX], int m, int n, int b[][MAX], int k, int t,
     int c[][MAX])
{
  int i, j, w;

  for(i=0; i<m; i++)
    for(j=0; j<t; j++)
    {
      c[i][j] = 0;
      for(w=0; w<n; w++)
        c[i][j] += a[i][w] * b[w][j];
    }
}

int main()
{
  int a[MAX][MAX], b[MAX][MAX], c[MAX][MAX];
  int m, n;
  int k, t;

  ucitavanje(a, &m, &n);
  ucitavanje(b, &k, &t);

  if (n != k)
  {
    printf("Mnozenje matrica nije moguce.\n");
    return -1;
  }

  mnozenje(a, m, n, b, k, t, c);

  printf("Rezultat mnozenja je:\n");
  ispis(c, m, t);

  return 0;
```

```
72
}
```

Rešenje 3.7.20

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

#define MAX 50

void ucitavanje(double mat[][MAX], int* m, int* n)
{
  int i, j;

  printf("Uneti dimenzije matrice: ");
  scanf("%d%d", m, n);

  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
  {
    printf("Neispravna dimenzija matrice\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti matricu celih brojeva\n");

  for(i=0; i<*m; i++)
    for(j=0; j<*n; j++)
      scanf("%lf", &mat[i][j]);
}

int main()
{
  double mat[MAX][MAX];
  int m, n, k, i, j;

  int indeks_kolone;
  double max_kolone, min_vrste;

  ucitavanje(mat, &m, &n);

  for(i=0; i<m; i++)
  {
    min_vrste = mat[i][0];
    indeks_kolone = 0;

    for(j=1; j<n; j++)
      if (mat[i][j] < min_vrste)
      {
        min_vrste = mat[i][j];
        indeks_kolone = j;
      }
```

```
48      max_kolone = mat[0][indeks_kolone];

50      for(k=1; k<m; k++)
          if (mat[k][indeks_kolone] > max_kolone)
52          max_kolone = mat[k][indeks_kolone];

54      if (min_vrste == max_kolone)
          printf("Sedlo: %d %d %g\n", i, indeks_kolone, min_vrste);
56
    }
58
    return 0;
60  }
```

Rešenje 3.7.21

```
1   #include <stdio.h>
    #include <stdlib.h>
3
    #define MAX 50
5
    void ucitavanje(int mat[][MAX], int* m, int* n)
7   {
      int i, j;
9
      printf("Uneti dimenzije matrice: ");
11    scanf("%d%d", m, n);

13    if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
      {
15        printf("Neispravna dimenzija matrice\n");
          exit(EXIT_FAILURE);
17    }

19    printf("Uneti matricu celih brojeva\n");

21    for(i=0; i<*m; i++)
        for(j=0; j<*n; j++)
23        scanf("%d", &mat[i][j]);
    }
25
    int main()
27  {
      int mat[MAX][MAX];
29    int m, n, brojac, i, j;

31    int pravac = 1;
      int gore_i, dole_i, levo_j, desno_j;
33
      ucitavanje(mat, &m, &n);
```

```
35
   gore_i = 1;
37 dole_i = m-1;

39 levo_j = 0;
   desno_j = n-1;
41
   i = 0;
43 j = 0;

45 for(brojac=0; brojac < m*n; brojac++)
   {
47   printf("%d ", mat[i][j]);

49   switch(pravac)
     {
51     case 1:
         if (j == desno_j)
53       {
           pravac = 2;
55         desno_j--;
           i++;
57       }
         else
59         j++;
         break;
61     case 2:
         if (i == dole_i)
63       {
           pravac = 3;
65         dole_i--;
           j--;
67       }
         else
69         i++;
         break;
71     case 3:
         if (j == levo_j)
73       {
           pravac = 4;
75         levo_j++;
           i--;
77       }
         else
79         j--;
         break;
81     case 4:
         if (i == gore_i)
83       {
           pravac = 1;
85         gore_i++;
           j++;
```

```
87          }
            else
89            i--;
      }
91    }

93    return 0;
}
```

Rešenje 3.7.22

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

4 #define MAX 50

6 void ucitavanje(int mat[][MAX], int* m, int* n)
{
8   int i, j;

10  printf("Uneti dimenzije matrice: ");
    scanf("%d%d", m, n);
12
    if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
14  {
      printf("Neispravna dimenzija matrice\n");
16    exit(EXIT_FAILURE);
    }
18
    printf("Uneti matricu celih brojeva\n");
20
    for(i=0; i<*m; i++)
22    for(j=0; j<*n; j++)
        scanf("%d", &mat[i][j]);
24 }

26 int podmatrica(int a[][MAX], int m, int n, int b[][MAX], int k, int t
       )
{
28  int i, j, x, y;
    int jeste_pod;
30
    for(i=0; i<= m-k; i++)
32    for(j=0; j<= n-t; j++)
      {
34      jeste_pod = 1;
        for(x=0; x<k && jeste_pod; x++)
36        for(y=0; y<t && jeste_pod; y++)
            if (a[i+x][j+y] != b[x][y])
38            jeste_pod = 0;
```

```
40        if (jeste_pod)
            return 1;
42      }

44    return 0;
}
46
int main()
48{
    int a[MAX][MAX], b[MAX][MAX];
50  int m, n;
    int k, t;
52
    ucitavanje(a, &m, &n);
54  ucitavanje(b, &k, &t);

56  if (podmatrica(a, m, n, b, k, t))
      printf("Druga matrica je sadrzana u prvoj matrici.\n");
58  else
      printf("Druga matrica nije sadrzana u prvoj matrici.\n");
60
}
```

3.9 Strukture

Zadatak 3.9.1 Definisati strukturu kojom se predstavlja 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.

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
```

Zadatak 3.9.2 Definisati strukturu kojom se predstavlja razlomak. Napisati funkcije koje izračunavaju zbir i proizvod dva razlomka. Napisati program koji za uneti broj n i unetih n razlomaka ispisuje njihov zbir i proizvod.

Primer 1

```
Interakcija sa programom:
 Unesi broj razlomaka:  5
 Uneti 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:
 Unesi broj razlomaka:  10
 Uneti 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.
```

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 program koji učitava podatke o voćkama sve do unosa reči `KRAJ` i ispisuje ime voćke sa najviše vitamina C. Pretpostaviti da broj voćki neće biti veći od 50.

Primer 1

```
Interakcija sa programom:
 Unesite ime voćke i njenu količinu vitamina C:  jabuka 4.6
 Unesite ime voćke i njenu količinu vitamina C:  limun 51
 Unesite ime voćke i njenu količinu vitamina C:  kivi 92.7
 Unesite ime voćke i njenu količinu vitamina C:  banana 8.7
 Unesite ime voćke i njenu količinu vitamina C:  pomorandza 53.2
 Unesite ime voćke i njenu količinu vitamina C:  KRAJ
 Voce sa najvise C vitamina je: kivi
```

Zadatak 3.9.4 Definisati strukturu `Grad` u kojoj se nalazi ime grada (niska dužine 20 karaktera) i prosečna temperatura u toku decembra (realan broj). Napisati program koji učitava imena n ($0 < n < 50$) gradova i njihove prosečne temperature, a zatim ispisuje one gradove koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Unesite grad i temperaturu:  Beograd 7
Unesite grad i temperaturu:  Uzice 1.5
Unesite grad i temperaturu:  Subotica 4
Unesite grad i temperaturu:  Zrenjanin 9
Gradovi sa idealnom temperaturom za klizanje u decembru:
Beograd
Subotica
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 2
Unesite grad i temperaturu:  Varsava 11
Unesite grad i temperaturu:  Prag 2
Gradovi sa idealnom temperaturom za klizanje u decembru:
```

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, ukupan broj parova reči je maksimalno 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
```

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 greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti broj prodavnica:  5
 idea 58.9
 maxi 58.2
 roda 55.1
 tempo 54.5
 interex 57.99
 Uneti zeljenu cenu:  57.0
 Povoljne prodavnice su:
 roda
 tempo
```

Primer 2

```
Interakcija sa programom:
 Uneti broj prodavnica:  4
 dm 43.2
 lily 45.99
 benu_apoteke 43.99
 sephora 50.99
 Uneti zeljenu cenu:  47.00
 Povoljne prodavnice su:
 dm
 lily
 benu_apoteke
```

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.

Primer 1

```
Interakcija sa programom:
 Uneti broj dece:  5
 Uneti podatke za svako dete, pol,
 broj godina i ocenu:
 m 3 5
 z 3 4
 m 4 2
 m 5 4
 m 3 4
 Uneti pol i broj godina:  m 3
 Prosecna ocena je: 4.500.
```

Primer 2

```
Interakcija sa programom:
 Uneti broj dece:  10
 Uneti 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
 Uneti pol i broj godina:  z 4
 Prosecna ocena je: 4.200.
```

Primer 3

```
Interakcija sa programom:
 Uneti broj dece:  15
 Uneti podatke za svako dete, pol,
 broj godina i ocenu:
 m 3 2
 z 7 5
 Neispravan broj godina.
```

Primer 4

```
Interakcija sa programom:
 Uneti broj dece:  2
 Uneti podatke za svako dete, pol,
 broj godina i ocenu:
 m 3 2
 z 3 5
 Uneti pol i broj godina:  h 5
 Neispravan pol.
```

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 sve njih. Maksimalan broj studenata je 2000. U slučaju greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti broj studenata:  5
 Uneti 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
 Uneti 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:
 Uneti broj studenata:  4
 Uneti podatke o studentima:
 0. student:  Djordje Lazarevic N 9.05
 1. student:  Minja Peric W 7.70
 Nekorektan smer.
```

Zadatak 3.9.9 Program učitava podatke o učenicima do kraja unosa. Učenika može biti najviše 30. Za svakog učenika dato je njegovo ime (maksimalne dužine 20 karaktera) i 9 ocena (ocene su celi brojvi od 1 do 5). Ispisati:

(a) Reč `NEDOVOLJNI:`, a potom imena nedovoljnih učenika. Učenik je nedovoljan ako ima barem jednu jedinicu.

(b) Potom ispisati reč `ODLICNI:`, a potom imena odličnih učenika. Učenik je odličan ako ima prosek ocena veći ili jednak 4.5.

U slučaju greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti podatke o djaku:  Maja 4 5 2 3 4 4 3 3 4
 Uneti podatke o djaku:  Nikola 5 4 5 5 5 4 4 5 5
 Uneti podatke o djaku:  Jasmina 2 2 1 1 2 3 3 1 3
 Uneti podatke o djaku:  Pera 5 4 5 3 5 5 1 5 5
 Uneti podatke o djaku:  Pavle 4 3 2 4 3 2 4 3 2
 Uneti podatke o djaku:

 NEDOVOLJNI: Jasmina Pera
 ODLICNI: Nikola
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 Uneti podatke o djaku:   Uros 3 4 2 3 4 2 3 4 4
 Uneti podatke o djaku:   Nebojsa 4 5 5 5 4 5 5 5 5
 Uneti podatke o djaku:   Sreten 2 3 2 4 5 4 4 4 2
 Uneti podatke o djaku:

 NEDOVOLJNI:
 ODLICNI: Nebojsa
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 Uneti podatke o djaku:   Mirko 2 3 4 4 4 3 3 3 4
 Uneti podatke o djaku:   Mihailo 2 3 10 5 5 2 3 4 2
 Neispravna ocena.
```

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 greške ispisati odgovarajuću poruku. Može se smatrati da je svaka email adresa dobro zadata i sadrži samo jedno pojavljivanje znaka `@`.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Uneti broj osoba:   3
 Uneti 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:
 Uneti broj osoba:   3
 Uneti podatke o osobama:
 ime, prezime i email.
 Homer Simpson homer@yahoo.com
 Mardz Simpson mardz@matf.bg.ac.rs
 Vlasnici gmail naloga su:
```

*** 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. Možemo pretpostaviti da nijedan potrošač neće kupiti više od 20 artikala, kao i da naziv svakog artikla sadrži

maksimalno 30 karaktera. U slučaju greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Uneti broj potrosackih korpi:  3
 Uneti podatke o korpi:
 Broj artikala:  4
 Unesi artikal, naziv, kolicinu i cenu:  jabuke 10 22.4
 Unesi artikal, naziv, kolicinu i cenu:  dezodorans 1 120.99
 Unesi artikal, naziv, kolicinu i cenu:  C_supa 3 36.56
 Unesi artikal, naziv, kolicinu i cenu:  sunka 1 230.99
 Uneti podatke o korpi:
 Broj artikala:  2
 Unesi artikal, naziv, kolicinu i cenu:  Jafa_keks 55.78
 Unesi artikal, naziv, kolicinu i cenu:  Najlepse_zelje 62.99
 Uneti podatke o korpi:
 Broj artikala:  3
 Unesi artikal, naziv, kolicinu i cenu:  prasak_za_ves 1 1199.99
 Unesi artikal, naziv, kolicinu i cenu:  omeksivac 1 279.99
 Unesi 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
```

Zadatak 3.9.12 Uvesti tip podatka `Sifra` kojim se opisuje način šifrovanja alfanumeričkih karaktera. Svaka šifra se opisuje pozitivnom celobrojnom vrednošću b koja određuje broj pozicija pomeranja, kao i karakterom `'L'` ili `'D'` koji određuje smer pomeranja (levo ili desno).

(a) Napisati funkciju `char sifruj(char c, Sifra s)` koja transformiše zadati karakter `c` po šifri `s`. Karakter se šifruje tako što se svako slovo zamenjuje slovom za b mesta levo ili desno od njega u abecedi, i to ciklično, a isto tako i za cifre. Na primer: za $b = 2$, i smer=`'D'` : `a` se menja sa `c`, `b` sa `d`, ..., `x` sa `z`, `y` sa `a`, `z` sa `b`, 1 sa 3, ..., 8 sa 0, 9 sa 1. Funkcija vraća novodobijeni

karakter.

(b) Načini šifrovanja se zadaju do kraja unosa i to u obliku `2 D 5 L`. Potom se zadaju karajteri do kraja unosa. Izmeniti alfanumeričke karaktere prema svim zadatim šiframa i ispisati dobijeni rezultat. Maksimalan broj karaktera može biti 5000. Maksimalan broj šifri može biti 100. U slučaju greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti sifre u obliku: broj, smer:
 23 D
 Uneti tekst za sifrovanje:
 Temperatura danas je 23 stepena Celzijusova.
 Rckncpxrspx bxlxq hc 56 qrcnclx Zcjxghsqmtx.
```

Primer 2

```
Interakcija sa programom:
 Uneti sifre u obliku: broj, smer:
 3 l 7 a
 Neispravan smer.
```

Primer 3

```
Interakcija sa programom:
 Uneti sifre u obliku: broj, smer:
 23 D 3 L 14 D 20 L 1 L 2 L 5 D
 Uneti tekst za sifrovanje:
 Temperatura danas je 23 stepena Celzijusova.
 Kudguiqkliq tqeqj zu 89 jkugueq Sucqyzljfmq.
```

Zadatak 3.9.13 Definisati strukturu `Lopta` sa poljima `poluprecnik` (ceo broj u centimetrima) i `boja` (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Zatim učitati informacije o n lopti ($0 < n < 50$) i ispisati ukupnu zapreminu, kao i broj crvenih lopti. U slučaju greške ispisati odgovarajuću poruku.

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
 Nekorektan unos.
```

Zadatak 3.9.14 Napisati program za predstavljanje poligona i izračunavanje njegovog obima i dužine stranica.

(a) Definisati tip podataka `TACKA` pogodan za predstavljanje tačke Dekartovske ravni (čije su x i y koordinate podaci tipa `double`).

(b) Definisati funkciju `double rastojanje(TACKA a, TACKA b)` koja izračunava rastojanje između dve tačke.

(c) Definisati funkciju `unsigned ucitaj_poligon(TACKA* tacke, unsigned 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, unsigned 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, unsigned n)` koja izračunava dužinu najduže stranice poligona sa n tačaka u zadatom nizu.

(f) Napisati funkciju `double povrsina_trougla(TACKA A, TACKA B, TACKA C)` za računanje površine trougla.

(g) Napisati funkciju `double povrsina(TACKA* poligon, unsigned 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 uneseni poligon konveksan. Poligon mora imati barem 3 temena. U slučaju greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Uneti 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:
 Uneti 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:
 Uneti maksimalan broj tacaka poligona:  4
 0 0
 Neispravan broj tacaka poligona.
```

*** Zadatak 3.9.15** Definisati strukturu `IZRAZ` kojom se opisuje numerički izraz nad celim brojevima koji se sastoji od dva celobrojna operanda, 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 1 ako jeste a 0 u suprotnom. Podrazumevamo 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 odredjuje vrednost izraza.

(c) Napisati funkciju koja učitava dati izraz. Funkcija treba da učita sa standardnog ulaza izlaz koji je zadat prefiksno — prvo operacija, a potom dva operanda. Funkcija vraća 1 ako je učitavanje bilo uspešno, tj. ako je izraz bio korektno zadat ili 0 u suprotnom.

(d) Napisati funkciju koja štampa dati izraz infiksno, u obliku "$operand_1$ $operacija$ $operand_2 = vrednost$".

Napisati glavni 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.

Primer 1

```
Interakcija sa programom:
 Uneti broj izraza:  4
 Uneti 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:
 Uneti broj izraza:  10
 Uneti 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:
 Uneti broj izraza:  3
 Uneti izraze u prefiksnoj notaciji:
 * 1 2
 / 3 0
 Deljenje nulom!
 Nekorektan unos
```

*** Zadatak 3.9.16** Definisati strukturu kojom se zadaje polinom. Polinom je dat svojim stepenom (može biti najviše 10) i realnim koeficijentima.

(a) Napisati funkciju koja učitava jedan polinom dat stepenom i koeficijentima.

(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 greške ispisati odgovarajuću poruku.

Primer 1

```
Interakcija sa programom:
 Uneti stepen:  3
 Uneti koeficijente polinoma:
 1 0 3 1
 Uneti stepen:  4
 Uneti koeficijente polinoma:
 7 9 4 0 4
 Uneti 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:
 Uneti stepen:  3
 Uneti koeficijente polinoma:
 1 0 -4 1
 Uneti stepen:  2
 Uneti koeficijente polinoma:
 1 2 -3
 Uneti stepen:  1
 Uneti koeficijente polinoma:
 0 -1
 Uneti 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
```

3.10 Rešenja

Rešenje 3.9.1

```c
#include <stdio.h>

/* Struktura koja opisuje kompleksni broj obuhvata polje za realni
 * i polje za imaginarni deo broja.
 */
typedef struct Complex {

  float re;
  float im;
} Complex;

/* Funkcija kojom se izracunava zbir kompleksnih brojeva. */
Complex saberi(Complex *a, Complex *b) {

  Complex c;
  c.re = a->re + b->re;
  c.im = a->im + b->im;
  return c;
}

/* Funkcija kojom se izracunava razlika kompleksnih brojeva. */
Complex oduzmi(Complex *a, Complex *b) {

  Complex c;
  c.re = a->re - b->re;
```

```
  c.im = a->im - b->im;
  return c;
}

/* Funkcija kojom se izracunava proizvod kompleksnih brojeva. */
Complex pomnozi(Complex *a, Complex *b) {

  Complex c;
  c.re = a->re * b->re - a->im * b->im;
  c.im = b->re * a->im + a->re * b->im;
  return c;
}

/* Funkcija kojom se izracunava kolicnik kompleksnih brojeva. */
Complex podeli(Complex *a, Complex *b) {

  Complex c;
  c.re = (a->re * b->re + a->im * b->im) / (b->re*b->re + b->im*b->im
    );
  c.im = (b->re * a->im - a->re * b->im) / (b->re*b->re + b->im*b->im
    );
  return c;
}

int main() {

  Complex a, b;
  Complex c;

  /* Ucitavamo kompleksne brojeve. */
  printf("Unesite realni i imaginarni deo prvog broja: ");
  scanf("%f%f", &a.re, &a.im);

  printf("Unesite realni i imaginarni deo drugog broja: ");
  scanf("%f%f", &b.re, &b.im);

  c = saberi(&a, &b);
   /* Ukoliko je imaginarni deo negativan,
    * njegov zapis vec ukljucuje znak,
    * te to treba proveriti.
    * Inace, broj je oblika a+b*i.
    */
  printf("Zbir: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im);

  c = oduzmi(&a, &b);
  printf("Razlika: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im)
    ;

  c = pomnozi(&a, &b);
  printf("Proizvod: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im
    );

```

```
  if(b.re != 0 || b.im != 0) {
75  c = podeli(&a, &b);
    printf("Kolicnik: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.
    im);
77  }
  /* U polju kompleksnih brojeva
79  * nije dozvoljeno deljenje nulom.
    */
81  else
    printf("Kolicnik ne postoji.\n");
83

85  return 0;
}
```

Rešenje 3.9.2

```
1 #include <stdio.h>

3 typedef struct
  {
5   int brojilac;
    int imenilac;
7 }razlomak;

9 int nzd(int a, int b)
  {
11  int pom;

13  if (a < b)
    {
15    pom = a;
      a = b;
17    b = pom;
    }
19
    while(b != 0)
21  {
      pom = a % b;
23    a = b;
      b = pom;
25  }

27  return a;
  }
29
  razlomak zbir(razlomak a, razlomak b)
31 {
    razlomak c;
33  int nzd_razlomka;
```

```
35    c.brojilac = a.brojilac * b.imenilac + b.brojilac*a.imenilac;
      c.imenilac = a.imenilac*b.imenilac;
37
      /* Brojilac i imenilac dobijenog zbira se dele najvecim zajednickim
39     * deliocom.
       */
41    nzd_razlomka = nzd(c.brojilac, c.imenilac);

43    c.brojilac = c.brojilac/nzd_razlomka;
      c.imenilac = c.imenilac/nzd_razlomka;
45
      return c;
47  }

49  razlomak proizvod(razlomak a, razlomak b)
    {
51    razlomak c;
      int nzd_razlomka;
53
      c.brojilac = a.brojilac*b.brojilac;
55    c.imenilac = a.imenilac*b.imenilac;

57    /* Brojilac i imenilac dobijenog zbira se dele najvecim zajednickim
       * deliocom.
59     */
      nzd_razlomka = nzd(c.brojilac, c.imenilac);
61
      c.brojilac = c.brojilac/nzd_razlomka;
63    c.imenilac = c.imenilac/nzd_razlomka;

65    return c;
    }
67
    int main()
69  {
      int n, i;
71
      razlomak suma, proizvod_svih, r;
73
      printf("Unesi broj razlomaka: ");
75    scanf("%d", &n);


77
      suma.brojilac = 0;
79    suma.imenilac = 1;

81    proizvod_svih.brojilac = 1;
      proizvod_svih.imenilac = 1;
83
      printf("Uneti razlomke:\n");
85    for(i=0; i<n; i++)
      {
```

```
87      scanf("%d%d", &r.brojilac, &r.imenilac);

89      suma = zbir(suma, r);
        proizvod_svih = proizvod(proizvod_svih, r);
91    }

93    printf("Suma svih razlomaka je %d/%d.\n", suma.brojilac, suma.
        imenilac);
      printf("Proizvod svih razlomaka je %d/%d.\n", proizvod_svih.
        brojilac, proizvod_svih.imenilac);
95
      return 0;
97  }
```

Rešenje 3.9.3

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

4   #define MAX_DUZINA 21
    #define MAX_BR_VOCKI 50
6
    typedef struct vocka
8   {
      char ime[MAX_DUZINA];
10    float vitamin;
    } VOCKA;
12


14  int main()
    {
16    VOCKA vocke[MAX_BR_VOCKI];
      int i = 0, n, max_vocka;
18    char ime[MAX_DUZINA];

20    /*
        Program ucitava podatke o vockama i smesta ih u niz
22      sve dok se ne unese rec KRAJ ili ucita MAX_BR_VOCKI vocki.
      */
24    do
      {
26      printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
        scanf("%s",ime);
28      /*
          Kada se unese rec KRAJ prekida se petlja.
30      */
        if(strcmp(ime, "KRAJ") == 0)
32        break;

34      /*
          Inace ucitava se kolicina vitamina
```

```
36        i ta vrednost se smesta u vocku na poziciji "i".
      */
38      strcpy(vocke[i].ime,ime);
      scanf("%f",&vocke[i].vitamin);
40      i++;
    }
42    while(i<MAX_BR_VOCKI);

44    n = i;

46    /*
      Pretpostavka je da prva vocka ima najvise vitamina.
48      Petljom se prolazi niz vocki i ukoliko se naidje na vocku koja
      ima vise vitamina
      od one koja trenutno ima najvise, azurira se vrednosti maksimalne
       vocke.
50
      Sve vreme se cuva indeks vocke sa najvise vitamina C.
52    */

54    max_vocka = 0;
    for(i=1;i<n;i++)
56      if(vocke[i].vitamin > vocke[max_vocka].vitamin)
      {
58        max_vocka = i;
      }
60
    printf("Voce sa najvise C vitamina je: %s\n", vocke[max_vocka].ime)
      ;
62
    return 0;
64  }
```

Rešenje 3.9.4

```
1   #include <stdio.h>
    #define MAX_DUZINA 20
3   #define MAX_BR_GRADOVA 50

5   typedef struct Grad{
      char ime_grada[MAX_DUZINA+1];
7     float temperatura;
    }Grad;
9

11  int main(){
      int n, i;
13    Grad grad[MAX_BR_GRADOVA];

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

```
17    if(n<0 || n>MAX_BR_GRADOVA){
        printf("Greska: pogresan unos!\n");
19      return 0;
      }
21
      for(i=0; i<n; i++){
23      printf("Unesite grad i temperaturu: ");
        scanf("%s %f", grad[i].ime_grada, &grad[i].temperatura);
25    }

27    printf("Gradovi sa idealnom temperaturom za klizanje u decembru:\n"
        );
      for(i=0; i<n; i++){
29      if(grad[i].temperatura>=3 && grad[i].temperatura<=8){
          printf("%s\n", grad[i].ime_grada);
31      }
      }
33
      return 0;
35  }
```

Rešenje 3.9.5

```
  #include <stdio.h>
2 #include <string.h>
  #define MAX_DUZINA 21
4 #define MAX_BR_RECI 100

6 typedef struct ParReci{
    char sr[MAX_DUZINA+1];
8   char en[MAX_DUZINA+1];
  }ParReci;
10


12 /*
    Funkcija koja u recniku koji sadrzi n reci trazi prevod reci rec i
      upisuje ga u prevod.
14  Ukoliko se rec ne nalazi u recniku, prevod se sastoji od zvezdica
      pri cemu broj zvezdica odgovara
    duzini nepoznate reci.
16 */

18 void pronadji_prevod(ParReci recnik[], int n, char rec[], char prevod
       []){
    int i;
20
    /* Pretrazuje se recnik i trazi se zadata rec. */
22  for(i=0; i<n; i++)
    {
24    if(strcmp(recnik[i].sr, rec)==0)
      {
```

```
26        strcpy(prevod, recnik[i].en);
          return;
28      }
    }
30
    /* Ukoliko rec nije pronadjena, formira se prevod reci koji se
       sastoji od zvezdica. */
32  for(i=0; rec[i]; i++){
      prevod[i]='*';
34  }
    prevod[i]='\0';
36 }


38
int main(){
40  ParReci recnik[MAX_BR_RECI];
    int n;
42  char sr[MAX_DUZINA+1];
    char en[MAX_DUZINA+1];
44  int i, j, k;
    char rec[MAX_DUZINA+1];
46  char prevod[MAX_DUZINA+1];
    char c;
48
    /* Ucitavaju se parovi reci sa standardnog ulaza sve do kraja ulaza
       . */
50  i=0;
    while(scanf("%s %s", sr, en)!=EOF){
52    if(i==MAX_BR_RECI)
        break;
54
      strcpy(recnik[i].sr, sr);
56    strcpy(recnik[i].en, en);

58    i++;
    }
60  /* Broj parova reci se cuva u promenljivoj n. */
    n=i;
62
    printf("Unesite recenicu za prevod: \n");
64  do
    {
66    /* Ucitava se rec po rec date recenice i pronalazi se njen prevod
      . */
      scanf("%s", rec);
68
      pronadji_prevod(recnik, n, rec, prevod);
70    printf("%s ", prevod);

72    /* Ukoliko je karakter iza reci znak za novi red, onda se prekida
       sa unosom, a ako nije
       * ucitava se sledeca recenica.
```

```
74      */
        c = getchar();
76
    }while(c != '\n');
78
    putchar('\n');
80
    return 0;
82  }
```

Rešenje 3.9.6

```
1   #include <stdio.h>
3   #define MAX_PRODAVNICA 50
    #define DUZINA_RECI 21
5
    typedef struct
7   {
      char prodavnica[DUZINA_RECI];
9     double cena;
    }podatak;
11
    int main()
13  {
      podatak niz[MAX_PRODAVNICA];
15    double zeljena;
      int n, i;
17
      printf("Uneti broj prodavnica: ");
19    scanf("%d", &n);

21    if (n <=0 || n > MAX_PRODAVNICA)
      {
23      printf("Neispravan broj prodavnica.\n");
        return -1;
25    }

27    for(i=0; i<n; i++)
      {
29      scanf("%s%lf", niz[i].prodavnica, &niz[i].cena);

31      if (niz[i].cena <= 0)
        {
33        printf("Neispravna cena.\n");
          return -1;
35      }
      }
37
      printf("Uneti zeljenu cenu: ");
39    scanf("%lf", &zeljena);
```

```
41    printf("Povoljne prodavnice su:\n");
      for(i=0; i<n; i++)
43      if (niz[i].cena <= zeljena)
          printf("%s\n", niz[i].prodavnica);
45
      return 0;
47  }
```

Rešenje 3.9.7

```
#include <stdio.h>
2
#define MAX_DECE 200
4
typedef struct
6  {
     char pol;
8    int broj_godina;
     int ocena;
10 }dete;

12 int main()
   {
14   int n, i, broj_godina;
     dete niz[MAX_DECE];
16   char blanko, pol;
     int suma, broj_dece;
18
     printf("Uneti broj dece: ");
20   scanf("%d", &n);

22   if (n <= 0 || n > MAX_DECE)
     {
24     printf("Neispravan broj dece.\n");
       return -1;
26   }

28   printf("Uneti podatke za svako dete, pol, broj godina i ocenu:\n");
     for(i=0; i<n; i++)
30   {
       scanf("%c%c%d%d", &blanko, &niz[i].pol, &niz[i].broj_godina, &niz
       [i].ocena);
32
       /* Ispitivanje pogresnog unosa. */
34     if (niz[i].pol != 'm' && niz[i].pol != 'z')
       {
36       printf("Neispravan pol.\n");
         return -1;
38     }
       if (niz[i].broj_godina > 6 || niz[i].broj_godina < 3)
```

```
40      {
          printf("Neispravan broj godina.\n");
42        return -1;
        }
44      if (niz[i].ocena < 1 || niz[i].ocena > 5)
        {
46        printf("Neispravna ocena.\n");
          return -1;
48      }
    }
50
    printf("Uneti pol i broj godina: ");
52  scanf("%c%c%d", &blanko, &pol, &broj_godina);

54  /* Ispitivanje ispravnosti unetih podataka. */
    if (pol != 'm' && pol != 'z')
56  {
      printf("Neispravan pol.\n");
58    return -1;
    }
60  if (broj_godina > 6 || broj_godina < 3)
    {
62    printf("Neispravan broj godina.\n");
      return -1;
64  }

66  suma = 0;
    broj_dece = 0;
68
    for(i=0; i<n; i++)
70    if (niz[i].pol == pol && niz[i].broj_godina == broj_godina)
      {
72      suma += niz[i].ocena;
        broj_dece++;
74    }

76  if (broj_dece == 0)
      printf("Ne postoje deca sa takvim karakteristikama.\n");
78  else
      printf("Prosecna ocena je: %.3lf.\n", (double)suma/broj_dece);
80
    return 0;
82  }
```

Rešenje 3.9.8

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

4 #define MAXST 2000
  #define MAX 31
```

```
6
typedef struct Student
8 {
    char ime[MAX];
10  char prezime[MAX];
    char smer;
12  float prosek;
} STUDENT;
14
void provera(char smer)
16 {
    if (smer != 'R' && smer != 'I' && smer != 'V' && smer != 'N' &&
       smer != 'T' && smer != 'O')
18  {
      printf("Nekorektan smer.\n");
20    exit(EXIT_FAILURE);
    }
22 }

24 void ucitaj(STUDENT* s)
{
26  scanf("%s",s->ime);

28  scanf("%s",s->prezime);

30  getchar();
    scanf("%c",&s->smer);
32
    scanf("%f", &s->prosek);
34 }

36 /* II */
/*
38  Kada neku promenljivu prenosimo u funkciju kao argument, obicno
    je prenosimo po vrednosti (bez pokazivaca), ako se ona nece menjati
       u funkciji
40  ili po adresi (preko pokazivaca), ako ce se njena vrednost
      promeniti u funkciji.

42  Prilikom poziva funkcije, za svaki argument funkcije kreira se
      promenljiva
    koja predstavlja lokalnu kopiju argumenta i koja prestaje da
      postoji po zavrsetku
44  funkcije. S obzirom da se strukuture sastoje od vise polja,
      zauzimaju
    vise memorije nego nestrukturne promenljive. Zbog toga je za
      njihovo kopiranje
46  potrebno vise vremena i vise memorijskih resursa nego za kopiranje
      nestrukturnih
    promenljivih.
48
    Da bismo ucinili program efikasnijim, korisno je da strukturu uvek
```

```
    kao
  argument funkcije prenosimo po adresi (preko pokazivaca), bez
    obzira
  da li ce se struktura u toj funkciji menjati ili ne. Pokazivac na
    strukturu
  zauzima manje memorije nego sama struktura pa je izrada njegove
    kopije
  brza a kopija pokazivaca uzima manji memorijski prostor nego kopija
  strukture.

  Kada prenosimo strukturnu promenljivu u funkciju po adresi (preko
    pokazivaca), tada
  imamo mogucnost da je u funkciji menjamo. Ukoliko zelimo da
    onemogucimo promenu,
  uz argument dodajemo kljucnu rec const. Ako pokusamo da promenimo
    argument
  funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
     s->smer='X';),
  kompajler ce prijaviti gresku. Na ovaj nacin obezbedjujemo da
    promenljiva
  koju smo preneli po adresi ne da bismo je promenili vec radi
    povecanja
  efikasnosti programa, ne bude, cak ni slucajno, izmenjena u
    funkciji.

*/

void ispisi(const STUDENT* s)
{
  printf("%s %s, %c, %.2f\n",s->ime, s->prezime, s->smer, s->prosek);
}


float najveci_prosek(STUDENT studenti[], int n)
{
  float m;
  int i;

  m = studenti[0].prosek;
  for(i=1;i<n;i++)
    if(m<studenti[i].prosek)
       m=studenti[i].prosek;
  return m;
}

/*
   Struktura moze da bude povratna vrednost funkcije.
*/
STUDENT prvi_student_sa_najvecim_prosekom(STUDENT studenti[], int n,
    float m)
{
  STUDENT s;
```

```
90    int i;
      for(i=0;i<n;i++)
92      if(m == studenti[i].prosek)
        {
94        /*
            Na strukture se moze primenjivati
96          naredba dodele.
          */
98        s = studenti[i];
          break;
100     }
      return s;
102 }


104
    int main()
106 {
      STUDENT studenti[MAXST];
108   int n;
      int i;
110   float max_prosek;
      STUDENT student_sa_max_prosekom;
112   int indeks;
      char smer;
114
      printf("Uneti broj studenata: ");
116   scanf("%d", &n);

118   if (n<0 || n>MAXST)
      {
120     printf("Nekorektan unos\n");
        return -1;
122   }

124   printf("Uneti podatke o studentima:\n");
      for(i=0;i<n;i++)
126   {
        printf("%d. student: ", i);
128     ucitaj(&studenti[i]);
        provera(studenti[i].smer);
130   }

132   printf("Uneti smer: ");
      getchar();
134   scanf("%c", &smer);

136   provera(smer);

138   printf("Studenti sa R smera:\n");
      for(i=0;i<n;i++)
140     if(studenti[i].smer == smer)
          printf("%s %s\n",studenti[i].ime, studenti[i].prezime);
```

```
142   printf("----------------------\n");

144

      /* Stampamo podatke o svim studentima sa
146      maksimalnim prosekom.
      */
148

      max_prosek = najveci_prosek(studenti, n);
150   printf("Svi studenti koji imaju maksimalni prosek:\n");
      for(i=0;i<n;i++)
152      if(studenti[i].prosek == max_prosek)
            ispisi(&studenti[i]);
154

156   return 0;
}
```

Rešenje 3.9.9

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

4 #define IME 21
  #define OCENE 9
6 #define MAX_DJAKA 30

8 typedef struct
  {
10   char ime[IME];
     int ocena[OCENE];
12 }_djak;

14 void provera(int ocena)
  {
16   if (ocena < 1 || ocena > 5)
     {
18     printf("Neispravna ocena.\n");
       exit(EXIT_FAILURE);
20   }
  }
22

24 int main()
  {
26   _djak niz[MAX_DJAKA];
     int i = 0, n, j;
28   int suma;
     float prosek;

30   printf("Uneti podatke o djaku: ");
     while(scanf("%s", niz[i].ime) != EOF && i < MAX_DJAKA)
32   {
```

```
    for(j=0; j<9; j++)
    {
      scanf("%d", &niz[i].ocena[j]);
      provera(niz[i].ocena[j]);
    }

    i++;
    printf("Uneti podatke o djaku: ");
  }

  n = i;

  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");


  printf("ODLICNI: ");
  for(i=0; i<n; i++)
  {
    suma = 0;
    for(j=0; j<9; j++)
      suma += niz[i].ocena[j];

    prosek = (float)suma/9;

    if (prosek >= 4.5)
      printf("%s ", niz[i].ime);
  }
  printf("\n");

  return 0;
}
```

Rešenje 3.9.10

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

#define IME 21
#define PREZIME 31
#define EMAIL 51

#define MAX_OSOBA 50

```

```c
typedef struct
{
  char ime[IME];
  char prezime[PREZIME];
  char email[EMAIL];
}Osoba;

int gmail(char* s)
{
  char* deo = strtok(s, "@");
  deo = strtok(NULL, "");

  if (strcmp(deo, "gmail.com") == 0)
    return 1;
  else
    return 0;
}

int main()
{
  int n, i;
  Osoba osobe[MAX_OSOBA];

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

  if (n < 0 || n >= MAX_OSOBA)
  {
    printf("Greska u broju osoba.\n");
    return -1;
  }

  printf("Uneti podatke o osobama, ime, prezime i email.\n");
  for(i=0; i<n; i++)
    scanf("%s%s%s", osobe[i].ime, osobe[i].prezime, osobe[i].email);

  printf("Vlasnici gmail naloga su:\n");
  for(i=0; i<n; i++)
    if (gmail(osobe[i].email))
      printf("%s %s\n", osobe[i].ime, osobe[i].prezime);

  return 0;
}
```

Rešenje 3.9.11

```c
#include <stdio.h>

#define MAXART 20
#define MAXPOT 100
#define MAXNAZIV 31
```

```c

typedef struct artikal
{
   char naziv[MAXNAZIV];
   int kolicina;
   float cena;
} ARTIKAL;

typedef struct korpa
{
   int br_art;
   ARTIKAL artikli[MAXART];
} KORPA;

/*
   Funkcija ucitaj_artikal ucitava podatke za jedan
   artikal i vraca 1 ako je ucitavanje bilo uspesno
   a 0 u suprotnom. Ucitavanje je neuspesno ukoliko
   kolicina nekog artikla ili njegova cena nisu pozitivni
   brojevi.

   S obzirom da funkcija ucitaj artikal treba da vrati
   dve vrednosti (ucitanu strukturu i indikator uspesnosti),
   strukturu ARTIKAL prenosimo preko pokazivaca a
   indikator uspesnosti vracamo kao povratnu vrednost.

*/

int ucitaj_artikal(ARTIKAL* a)
{
   printf("Unesi artikal, naziv, kolicinu i cenu: ");
   scanf("%s", a->naziv);
   scanf("%d", &a->kolicina);

   if (a->kolicina<=0)
   {
      printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina
    );
      return 0;
   }

   scanf("%f",&a->cena);
   if (a->cena<0)
   {
      printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
      return 0;
   }

   return 1;
}

/*
```

```
57    Funkcija izracunaj_racun izracunava racun date
      potrosacke korpe u kojoj su inicijalizovani
59    podaci o broju artikala i o svakom pojedinacnom
      artiklu.
61 */
   float izracunaj_racun(const KORPA* k)
63 {
      int i;
65    float racun=0;
      for(i=0;i<k->br_art;i++)
67       racun+=k->artikli[i].kolicina * k->artikli[i].cena;
      return racun;
69 }

71 /*
      Pri ucitavanju korpe, zadaje se broj artikala a zatim
73    podaci za svaki artikal.

75    Funkcija ucitaj_korpu vraca 1 ako je ucitavanje uspesno
      i 0 u suprotnom. Do neuspesnog ucitavanja moze doci
77    ako broj artikala u korpi nije pozitivan ili ako dodje
      do neuspesnog ucitavanja nekog artikla.
79 */

81 int ucitaj_korpu(KORPA* k)
   {
83    int i;
      printf("Uneti podatke o korpi: \n");
85    printf("Broj artikala: ");
      scanf("%d", &k->br_art);
87    if (k->br_art<=0)
      {
89       printf("Nekorektan unos za broj artikala: %d\n", k->br_art);
         return 0;
91    }
      for(i=0; i<k->br_art;i++)
93       if (ucitaj_artikal(&k->artikli[i])==0)
            return 0;
95
      return 1;
97 }

99 /*
      Funkcija ucitaj_niz_korpi ucitava podatke
101   za niz od n potrosackih korpi. Funkcija
      vraca 1 ako je ucitavanje uspesno i 0 ako
103   nije. Ucitavanje je neuspesno ukoliko ne uspe
      ucitavanje jedne od korpi.
105 */

107 int ucitaj_niz_korpi(KORPA korpe[], int n)
    {
```

```
109     int i,j;
        for(i=0; i<n; i++)
111        if(ucitaj_korpu(&korpe[i])==0)
              return 0;
113
        return 1;
115
}
117
/*
119     Funkcija stampaj_racun ispisuje na
        standardni izlaz racun za datu korpu
121     tako sto za svaki artikal ispise
        naziv, cenu i kolicinu i na kraju
123     ukupnu cenu za kupljene artikle.
*/
125
void stampaj_racun(const KORPA* k)
127 {
        int i,j;
129     for(i=0;i<k->br_art;i++)
           printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
         kolicina, k->artikli[i].cena);
131     printf("------------------------\n");
        printf("\tukupno: %.2f\n", izracunaj_racun(k));
133
}
135
/*
137     Funkcija stampaj_racune_za_korpe
        ispisuje na standardni izlaz racune
139     za svaku korpu u nizu potrosackih
        korpi
141 */

143 void stampaj_racune_za_korpe(KORPA korpe[], int n)
    {
145     int i;
        for (i=0;i<n;i++)
147     {
           printf("\nKorpa %d:\n",i);
149        stampaj_racun(&korpe[i]);
        }
151 }

153 /*
        Funkcija prosek racuna prosecnu cenu
155     potrosacke korpe za dati niz potrosackih
        korpi
157 */
    float prosek(KORPA korpe[], int n)
159 {
```

```
    int i;
161 float p;

163 for(i=0;i<n;i++)
        p+=izracunaj_racun(&korpe[i]);
165
    return p/n;
167 }

169 int main()
    {
171 int n;
    KORPA korpe[MAXPOT];
173
    printf("Uneti broj potrosackih korpi:");
175 scanf("%d", &n);

177 if(n<0 || n>MAXPOT)
    {
179     printf("Nekorektan unos broja potrosackih korpi: %d\n",n);
        return -1;
181 }

183 if (ucitaj_niz_korpi(korpe, n)==0)
        return -1;
185
    stampaj_racune_za_korpe(korpe,n);
187 printf("Prosecna cena potrosacke korpe: %.2f\n", prosek(korpe, n))
      ;

189 return 0;
    }
```

Rešenje 3.9.12

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

4   #define BROJ_SIFRI 100
    #define BROJ_KARAKTERA 5000
6
    typedef struct
8   {
      int b;
10    char smer;
    }Sifra;
12
    char sifruj(char c, Sifra s)
14  {
      int pomeraj;
16
```

```
  if (!isalnum(c))
18  return c;

20  if (s.smer == 'L')
    {
22    if (isdigit(c))
      {
24      pomeraj = s.b % 10;

26      if (pomeraj > c - '0')
        {
28        pomeraj = pomeraj - (c - '0') - 1;
          c = '9' - pomeraj;
30      }
        else
32        c = c - pomeraj;

34      return c;
      }
36
      pomeraj = s.b % 26;
38
      if (islower(c))
40    {
        if (pomeraj > c - 'a')
42      {
          pomeraj = pomeraj - (c - 'a') - 1;
44        c = 'z' - pomeraj;
        }
46      else
          c = c - pomeraj;
48
        return c;
50    }

52    if (pomeraj > c - 'A')
        {
54        pomeraj = pomeraj - (c - 'A') - 1;
          c = 'Z' - pomeraj;
56      }
        else
58        c = c - pomeraj;

60      return c;
    }
62  else
    {
64    if (isdigit(c))
      {
66      pomeraj = s.b % 10;

68      if (pomeraj > '9' - c)
```

```
      {
        pomeraj = pomeraj - ('9' - c) - 1;
        c = '0' + pomeraj;
      }
      else
        c = c + pomeraj;

      return c;
    }

    pomeraj = s.b % 26;

    if (islower(c))
    {
      if (pomeraj > 'z' - c)
      {
        pomeraj = pomeraj - ('z' - c) - 1;
        c = 'a' + pomeraj;
      }
      else
        c = c + pomeraj;

      return c;
    }

    if (pomeraj > 'Z' - c)
      {
        pomeraj = pomeraj - ('Z' - c) - 1;
        c = 'A' + pomeraj;
      }
      else
        c = c + pomeraj;

      return c;
  }
}

int main()
{
  char linija[BROJ_KARAKTERA];
  Sifra sifre[BROJ_SIFRI];
  char c;
  int n = 0, j = 0, i;

  printf("Uneti sifre u obliku: broj, smer:\n");
  while (scanf("%d %c", &sifre[n].b, &sifre[n].smer) != EOF)
  {
    if (sifre[n].smer != 'L' && sifre[n].smer != 'D')
    {
      printf("Neispravan smer.\n");
      return -1;
    }
```

```
122     if (sifre[n].b < 0)
        {
124       printf("Neispravan broj za sifrovanje.\n");
          return -1;
126     }

128     n++;
        if (n == BROJ_SIFRI)
130       break;
    }
132
    printf("Uneti tekst za sifrovanje:\n");
134 while((c = getchar()) != EOF)
    {
136   for(i=0; i<n; i++)
       c = sifruj(c, sifre[i]);
138
      linija[j] = c;
140   j++;

142   if (j == BROJ_KARAKTERA)
        break;
144 }

146 linija[j] = 0;

148 printf("%s\n", linija);

150 return 0;
}
```

Rešenje 3.9.13

```
1  #include <stdio.h>
   #include <math.h>
3
   #define MAX 50
5
   typedef struct lopta {
7    int poluprecnik;
     enum {plava, zuta, crvena, zelena} boja;
9  } LOPTA;

11 float zapremina(LOPTA l) {
    return pow(l.poluprecnik, 3)*4/3*M_PI;
13 }


15
   float ukupna_zapremina(LOPTA lopte[], int n) {
17
```

```
  int i;
  float z = 0;

  for(i = 0; i < n; i++)
    z += zapremina(lopte[i]);

  return z;
}

/*
  Funkcija je opstija od trazene i broji sve lopte odredjene boje u
    nizu lopti.
  U zavisnosti od prosledjene boje funkciji, funkcija vraca
    odgovarajuci broj.
*/
int broj_lopti_u_boji(LOPTA lopte[], int n, int boja) {

  int br = 0;
  int i;
  for(i = 0; i < n; i++)
    if(lopte[i].boja == boja)
      br++;
  return br;
}

int main() {

  LOPTA lopte[MAX];
  int n;
  int i;
  int boja;

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

  if(n < 1 || n > MAX) {

    printf("Nekorektan unos.\n");
    return 0;
  }

  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%d", &lopte[i].poluprecnik, &boja);

  /* U zavisnosti od unetog celog broja,
     bira se boja lopte.
   */
    switch(boja) {
```

```
67
        case 1: lopte[i].boja = plava; break;
69      case 2: lopte[i].boja = zuta; break;
        case 3: lopte[i].boja = crvena; break;
71      case 4: lopte[i].boja = zelena; break;
        default:
73        printf("Nekorektan unos.\n");
        return 0;
75    }
  }
77
  printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));
79
  printf("Ukupno crvenih lopti: %d\n", broj_lopti_u_boji(lopte, n,
    crvena));
81
  return 0;
83 }
```

Rešenje 3.9.14

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

4 #define MAX_TACAKA 1000

6 typedef struct
{
8   int x, y;
}TACKA;
10
double rastojanje(TACKA a, TACKA b)
12 {
  return sqrt(pow(a.x - b.x, 2) + pow(a.y - b.y, 2));
14 }

16 unsigned ucitaj_poligon(TACKA* tacke, unsigned n)
{
18   int i = 0;

20   while(i < n && scanf("%d%d", &tacke[i].x, &tacke[i].y) != EOF)
       i++;
22
  return i;
24 }

26 double obim(TACKA* poligon, unsigned n)
{
28   double o = rastojanje(poligon[0], poligon[n-1]);
  int i;
30
```

```
  for(i=0; i<n-1; i++)
    o += rastojanje(poligon[i], poligon[i+1]);

  return o;
}

double maksimalna_stranica(TACKA* poligon, unsigned n)
{
  double max = 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 > max)
      max = stranica;
  }

  return max;
}

double povrsina_trougla(TACKA A, TACKA B, TACKA C)
{
  double a = rastojanje(B, C);
  double b = rastojanje(A, C);
  double c = rastojanje(A, B);

  double s = (a + b + c)/2;

  return sqrt(s*(s -a)*(s - b)*(s - c));
}

double povrsina(TACKA* poligon, unsigned n)
{
  double P = 0;
  int i;

  for(i=1; i<n-1; i++)
    P += povrsina_trougla(poligon[0], poligon[i], poligon[i+1]);

  return P;
}

int main()
{
  int N;
  unsigned m;
  TACKA poligon[MAX_TACAKA];

  printf("Uneti maksimalan broj tacaka poligona: ");
  scanf("%d", &N);
```

```
  if (N < 3 || N > MAX_TACAKA)
  {
    printf("Neispravan broj tacaka poligona.\n");
    return -1;
  }

  m = ucitaj_poligon(poligon, N);

  if (m < 3)
  {
    printf("Neispravan broj tacaka poligona.\n");
    return -1;
  }

  printf("Obim poligona je %.3lf.\n", obim(poligon, m));
  printf("Duzina maksimalne stranice je %.3lf.\n",
    maksimalna_stranica(poligon, m));
  printf("Povrsina poligona je %.3lf.\n", povrsina(poligon, m));

  return 0;
}
```

Rešenje 3.9.15

```
#include <stdio.h>
#define MAX 1000

typedef struct
{
  char o;
  int x;
  int y;
} IZRAZ;


int korektan_izraz(const IZRAZ izraz)
{
   if(izraz.o != '+' && izraz.o != '-' && izraz.o != '*' && izraz.o
    != '/')
   {
      printf("Nedozvoljena operacija!\n");
      return 0;
   }
   if(izraz.o == '/' && izraz.y == 0)
   {
      printf("Deljenje nulom!\n");
      return 0;
   }
   return 1;
}
```

```
/* Racunanje vrednosti izraza. */
int vrednost(const IZRAZ izraz)
{
   int v;

   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;
   }
}


/*
   Promenljiva izraz ce se promeniti u funkciji
   ucitaj_izraz tako sto ce njenim neinicijalizovanim
   poljima o,x,y biti dodeljene vrednosti ucitane
   sa ulaza. Zbog toga ovu promenljivu
   funkciji prosledjujemo po adresi, preko pokazivaca.

   S obzirom da ucitavanje karaktera nije prvo
   ucitavanje koje se obavlja u programu, funkcijom
   getchar() se ucita karakter kojim se razdvaja
   unos karaktera od prethodnog unosa (najcesce blanko
   znak ili znak za novi red).

*/

int ucitaj_izraz(IZRAZ* izraz)
{
   getchar();
   scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
   if (!korektan_izraz(*izraz))
      return 0;
   return 1;
}


void stampaj_izraz(const IZRAZ izraz)
{
   printf("%d %c %d = %d\n", izraz.x, izraz.o, izraz.y, vrednost(
    izraz));
}

int max_vr(IZRAZ izrazi[], int n)
```

```
{
   int i;
   int max;

   max=vrednost(izrazi[0]);

   for(i=1; i<n; i++)
      if(vrednost(izrazi[i])>max)
         max=vrednost(izrazi[i]);

   return max;
}

int main()
{
   int n;
   IZRAZ izrazi[MAX];
   int max;
   int i;

   printf("Uneti broj izraza: ");
   scanf("%d", &n);
   if(n<0 || n>MAX)
   {
      printf("Nekorektna vrednost broja n!\n");
      return -1;
   }


   printf("Uneti izraze u prefiksnoj notaciji:\n");
   for(i=0; i<n; i++)
      if(ucitaj_izraz(&izrazi[i])==0)
      {
         printf("Nekorektan unos\n");
         return -1;
      }


   max = max_vr(izrazi, n);
   printf("Maksimalna vrednost izraza:%d\n", max);

   printf("Izrazi cija je vrednost manja od polovine maksimalne
    vrednosti:\n");

   for(i=0; i<n; i++)
      if(vrednost(izrazi[i])<max/2)
         stampaj_izraz(izrazi[i]);

   return 0;
}
```

Rešenje 3.9.16

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

#define STEPEN 10
#define MAX_POLINOMA 100

typedef struct
{
  int  stepen;
  float koef[STEPEN+1];
}_polinom;

int ucitaj(_polinom* p)
{
  int i;

  printf("Uneti stepen: ");
  if (scanf("%d", &(p->stepen)) == EOF)
    return 0;

  if (p->stepen > STEPEN || p->stepen < 0)
  {
    printf("Greska u unosu stepena.\n");
    exit(EXIT_FAILURE);
  }

  printf("Uneti koeficijente polinoma:\n");
  for(i=0; i<=p->stepen; i++)
    scanf("%f", &(p->koef[i]));

  return 1;
}

int ispis_prvog_monoma(float koef, int stepen)
{
  if (koef != 0)
  {
    printf("%.2f", koef);

    if (stepen == 1)
      printf("*x ");
    else if (stepen > 1)
      printf("*x^%d ", stepen);

    return 1;
  }
  else
    return 0;
}

```

```
void ispis_monoma(float koef, int stepen)
{
  if (koef != 0)
  {
    if (koef > 0)
      printf("+ ");

    printf("%.2f", koef);

    if (stepen == 1)
      printf("*x ");
    else if (stepen > 1)
      printf("*x^%d ", stepen);
  }
}

void ispis(const _polinom *p)
{
  int prvi = 1;
  int i;

  for(i=0; i <= p->stepen; i++)
    if (prvi)
    {
      prvi = !ispis_prvog_monoma(p->koef[i], i);
    }
    else
      ispis_monoma(p->koef[i], i);

  printf("\n");
}

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()
{
  _polinom p[MAX_POLINOMA], integ;
  int i = 0, j;

  while(ucitaj(&p[i]))
    i++;
```

```
104   printf("\n\nIntegrali su:\n");
      for(j=0; j<i; j++)
106   {
        integral(&p[j], &integ);
108     ispis(&integ);
      }
110
      return 0;
112 }
```

4

Ulaz i izlaz programa

4.1 Datoteke

Zadatak 4.1.1 Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u datoteku izlaz.txt karakter po karakter.

Zadatak 4.1.2 Napisati program koji u datoteci cije se ime navodi kao prvi argument komandne linije odredjuje liniju maksimalne duzine i ispisuje je na standarni izlaz. Ukoliko ima vise takvih linija , ispisati onu koja je leksikografski prva. Mozemo pretpostaviti da datoteka ne sadrzi linije duze od 80 karaktera.

Zadatak 4.1.3 U datoteci cije se ime zadaje kao prvi argument komandne linije nalazi se prirodan broj n a zatim i n celih brojeva. Napisati program koji prebrojava koliko k-tocifrenih brojeva postoji u datoteci , pri cemu se prirodan broj k zadaje kao drugi argument komandne linije.

Zadatak 4.1.4 U datoteci cije se ime navodi kao prvi argument komandne linije navedena je rec r i niz linija. Napisati program koji u datoteku cije se ime navodi kao drugi argument komandne linije upisuje sve linije u kojima se rec r pojavljuje bar n puta , gde je n prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu broj_pojavljivanja : linija.

Zadatak 4.1.5 Program se pokrece tako sto se navedu nazivi dve datoteke(ulazna i izlazna) i opcije. U datoteci cije se ime navodi kao prvi argument komandne linije nalaze se podaci o razlomcima: u prvom redu se nalazi broj razlomaka , a u svakom sledecem redu brojilac i imenilac jednog razlomka. Potrebno je kreirati strukturu koja opisuje razlomak i ucitati niz razlomaka iz datoteke , a potom:

a) ukoliko je navedena opcija x, upisati u datoteku cije je ime drugi argument komandne linije reciprocni razlomak za svaki razlomak iz niza (npr. za 2/3 treba upisati 3/2)
b) ukoliko je navedena opcija y, upisati u datoteku cije je ime drugi argument komandne linije realnu vrednost reciprocnog razlomka svakog razlomka iz niza (npr. za 2/3 treba upisati 1.5)

Mozemo pretpostaviti da se u datoteci sa podacima o razlomcima nalazi najvise 100 razlomaka.

Zadatak 4.1.6 Za svaki automobil poznati su marka , model i cena. Iz datoteke cije se ime zadaje sa standardnog ulaza ucitava se broj automobila a potom i podaci za svaki automobil. Program treba da:
a) izracuna prosecnu cenu po marki kola
b) za maksimalnu cenu koju je kupac spreman da plati , a koja se zadaje kao argument komandne linije, da ispise automobile u tom cenovnom rangu zajednu sa prosecnom cenom odgovarajuce marke

Mozemo pretpostaviti da se model i marka sastoje od jedne reci i da svaka od njih sadrzi najvise 30 karaktera kao i da se u datoteci nalaze podaci za najvise 100 automobila.

Zadatak 4.1.7 Napisati program koji prebrojava mala slova u datoteci *test.txt*.

Primer 1

```
TEST.TXT
 Abcd EFGH+ijKLMN

IZLAZ:
 Broj malih slova je: 5
```

Primer 2

```
TEST.TXT
 PrograMiranje

IZLAZ:
 Broj malih slova je: 11
```

Zadatak 4.1.8 Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*.

Primer 1

```
ULAZ.TXT
 Volim programiranje.
IZLAZ.TXT
 Vipgmae
```

Zadatak 4.1.9 Kao argumenti komandne linije se zadaju ime datoteke i ceo broj k. Napisati program koji na standardni izlaz ispisuje sve linije zadate datoteke čija je dužina veća od k. Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera.

Primer 1

```
POKRETANJE: ./a.out test.txt 7
TEST.TXT
 Teme koje su obradjivane:
 Petlje
 Funkcije
 Nizovi
 Strukture

IZLAZ:
 Teme koje su obradjivane:
 Funkcije
 Strukture
```

Primer 2

```
POKRETANJE: ./a.out test.txt

IZLAZ:
 Greska: Pogresan broj argumenata!
```

Zadatak 4.1.10 Napisati program koji prebrojava koliko se linija datoteke *ulaz.txt* završava niskom s koja se učitava sa standardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera, kao i da dužina niske s neće biti veća od 20 karaktera.

Primer 1

```
ULAZ.TXT
 abcde abcde
 abcde aab
 abcde abcde abcde
 abcde abcde Aab
 abcde abcde ab
 abcde abcde abcde abcde

INTERAKCIJA SA PROGRAMOM:
 Unesite nisku s: ab
 Broj linija: 3
```

Primer 2

```
ULAZ.TXT
 abcde abcde
 abcde
 abcde abcde AB

INTERAKCIJA SA PROGRAMOM:
 Unesite nisku s: ab
 Broj linija: 0
```

Zadatak 4.1.11 Napisati program koji pronalazi maksimum brojeva zapisanih u datoteci *brojevi.txt*.

Primer 1

```
BROJEVI.TXT
 2.36 -16.11 5.96 8.88
 -265.31 54.96 38.4

IZLAZ:
 Najveci broj je: 54.96
```

Zadatak 4.1.12 U datoteci *studenti.txt* se nalaze informacije o studentima: prvo broj studenata, a zatim u pojedinačnim linijama korisničko ime i pet poslednjih ocena koje je student dobio. Napisati program koji pronalazi studenta koji je ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da broj

studenata neće biti veći od 100.

Primer 1

```
STUDENTI.TXT
 mr15239 10 9 9 8 10
 mi14005 8 8 9 8 10
 ml15112 9 8 8 7 10
 mr15007 10 10 10 10 10
 mn13208 7 7 9 6 10

IZLAZ:
 korisnicko ime: mr15007, prosek ocena: 10.00
```

Zadatak 4.1.13 U datoteci *tacke.txt* se nalazi prvo broj tačaka, a zatim u pojedinačnim linijama x i y koordinate tačke. Napisati program koji u datoteku *rastojanja.txt* upisuje rastojanje svake od pročitanih tačaka od koordinatnog početka, a na standardni izlaz koordinate tačke koja je najudaljenija. Koristiti strukturu *Tacka* sa poljima x i y, kao i funkciju kojom se računa rastojanje. Pretpostaviti da broj tačaka u datoteci neće biti veći od 50.

Primer 1

```
TACKE.TXT
 4
 11 -2
 3 5
 8 -8
 0 4

RASTOJANJA.TXT
 11.18
 5.29
 11.31
 4.00

IZLAZ:
 Najudaljenija je tačka: 8 -8
```

Primer 1

```
TACKE.TXT
 -2
 0 0
 9 -8

IZLAZ:
 Greska: Nedozvoljen broj tacaka!
```

Zadatak 4.1.14 Napisati program koji za reč s maksimalne dužine 20 karaktera koja se zadaje sa standardnog ulaza u datoteku *rotacije.txt* upisuje sve rotacije reči s.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite rec:  abcde

ROTACIJE.TXT
 abcde
 bcdea
 cdeab
 deabc
 eabcd
```

Zadatak 4.1.15 Napisati program koji linije koji se učitavaju sa standardnog ulaza sve do kraja ulaza prepisuje u datoteku *izlaz.txt* i to, ako je prilikom pokretanja zadata opcija *-v* ili *-V* samo one linije koje počinju velikim slovom, ako je zadata opcija *-m* ili *-M* samo one linije koje počinju malim slovom, a ako je opcija izostavljena sve linije. Pretpostaviti da linije neće biti duže od 80 karaktera.

Primer 1

```
POKRETANJE: ./a.out -m
INTERAKCIJA SA PROGRAMOM:
 Unesite recenice:
 programiranje u C-u je zanimljivo
 Volim programiranje!
 Kada porastem bicu programer!
 u slobodno vreme programiram

IZLAZ.TXT
 programiranje u C-u je zanimljivo
 u slobodno vreme programiram
```

Primer 2

```
POKRETANJE: ./a.out -V
INTERAKCIJA SA PROGRAMOM:
 Unesite recenice:
 programiranje u C-u je zanimljivo
 Volim programiranje!
 Kada porastem bicu programer!
 u slobodno vreme programiram

IZLAZ.TXT
 Volim programiranje!
 Kada porastem bicu programer!
```

Primer 3

```
POKRETANJE: ./a.out -k
INTERAKCIJA SA PROGRAMOM:
 Greska: Pogresno pokretanje programa!
```

Zadatak 4.1.16 Sa standarnog ulaza učitavaju se imena dve tekstualne datoteke i jedan karakter. Napisati program koji prepisuje datoteku čije se ime navodi kao prvo u datoteku čije ime se navodi kao drugo. Ukoliko je ucitan karakter `u` program prilikom prepisivanja treba da zamenjuje sva mala slova velikim, a ukoliko je učitan karakter `l` sva velika slova se zamenjuju malim. U slučaju greske ispisati -1. Greška može biti neuspešno otvaranje datoteke ili pogrešno zadat karakter. Maksimalna dužina naziva datoteka je 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 ulaz.txt izlaz.txt u
ULAZ.TXT
 danas je lep dan
 i Ja zelim
 da postanem programer
IZLAZ.TXT
 DANAS JE LEP DAN
 I JA ZELIM
 DA POSTANEM PROGRAMER
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 prva.dat druga.dat l
PRVA.DAT
 Cena soka je 30
 Cena vina je 150
 Cena limunade je 200
 Cena sendvica je 120
DRUGA.DAT
 cena soka je 30
 cena vina je 150
 cena limunade je 200
 cena sendvica je 120
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 primer.c prazna.txt V
PRIMER.C
 #include <stdio.h>
 int main()
 {
 }
PRAZNA.TXT

IZLAZ:
 -1
```

Zadatak 4.1.17 Sastaviti program koji sa standardnog ulaza prima ime datoteke koju treba otvoriti. Ispisati (na standardnom izlazu) koja cifra (među svim ciframa koje se pojavljuju u datoteci) ima najveći broj pojavljivanja. U slučaju greške pri otvaranju datoteke ispisati `-1`. Ukoliko nema cifara u datoteci ispisati `-1`. Maksimalna dužina naziva datoteka je 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 ulaz.txt
ULAZ.TXT
 danas je lep dan
 i Ja zelim
 da postanem programer
IZLAZ:
 -1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 prva.dat druga.dat l
PRVA.DAT
 Cena soka je 30
 Cena vina je 150
 Cena limunade je 200
 Cena sendvica je 120
IZLAZ:
 0
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
 primer.c
PRIMER.C
 #include <stdio.h>
 int main()
 {
 }
PRAZNA.TXT

IZLAZ:
 -1
```

Zadatak 4.1.18 Prvi red datoteke `matrice.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice A. Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji pronalazi sve elemente matrice A koji su jednaki zbiru svih svojih susednih elemenata i štampa ih u obliku

`(broj vrste, broj kolone, vrednost elementa).`

U slučaju greške prilikom otvaranja datoteke ispisati `-1`. Pretpostaviti da je sadržaj datoteke ispravan.

Primer 1

```
MATRICE.TXT
 1 2 3 4
 7 2 15 -3
 -1 3 1 3
IZLAZ:
 (1, 0, 7)
 (1, 2, 15)
```

Zadatak 4.1.19 Napisati program koji za dve datoteke čija su imena data kao prvi i drugo na standarnom ulazu, radi sledeće: za cifru u prvoj datoteci, u drugu datoteku se upisuje 0, za slovo se upisuje 1, a za sve ostale karaktere se upisuje 2. Maksimalna dužina naziva datoteka je 20 karaktera.

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  prva.dat druga.dat
PRVA.DAT
  Cena soka je 30
  Cena vina je 150
  Cena limunade je 200
  Cena sendvica je 120
DRUGA.DAT

      1111211112112002111121111211200021111211111111211200021111211111111211200
```

Zadatak 4.1.20 Ako je data tekstualna datoteka `plain.txt` napraviti tekstualnu datoteku `sifra.txt` tako što se svako slovo zamenjuje svojim prethodnikom (ciklično) suprotne velicine `'b'` sa `'A'`, `'B'` sa `'a'`, `'a'` sa `'Z'`, `'A'` sa `'z'`, itd. Podrazumevati da se na sistemu koristi tabela karaktera ASCII.

Zadatak 4.1.21 Sa standarnog ulaza se učitava ime tekstualne datoteke i prirodan broj k. Podrazumeva se da zadata datoteka sadrži samo slova i beline i da je svaka reč iz datoteke dužine najviše 100. Program treba da učitava reči iz datoteke, da svaku reč rotira za k mesta i da tako dobijenu reč upiše u datoteku čije je ime `rotirano.txt`. Maksimalna dužina naziva datoteka je 20 karaktera.

Zadatak 4.1.22 Napisati program koji u datoteku `izlaz.txt` prepisuje sve reči iz datoteke `ulaz.txt` čiji je zbir ascii kodova slova strogo veći od 1000. Reči su odvojene prazninama i nisu duže od 200 karaktera.

Primer 1

```
ULAZ.TXT
  Sa standardnog ulaza unosi se neoznacen
  ceo broj. Formirati novi broj koji se dobija
  izbacivanjem svake druge cifre iz polaznog
      broja.
IZLAZ.TXT
  standardnog izbacivanjem
```

Primer 2

```
ULAZ.TXT
  i sada jedan kratak primer
  p1: 1234567890
  p2: ABCDEFGHIJ
  p3: abcdefghij
IZLAZ.TXT
  abcdefghij
```

Primer 3

```
ULAZ.TXT
  konstruisanje test-primera sa
  i dugackim recima kao prestolonaslednik
  brojevima1234567890
IZLAZ.TXT
  konstruisanje test-primera
  prestolonaslednik
  brojevima1234567890
```

Primer 4

```
ULAZ.TXT
  ima jos dugackih reci: predskazanje,
  potom
  nelogicnosti, zanemarivati, odugovlaciti, a ima
  i i malih reci koje su kratke
  predosecaj
IZLAZ.TXT
  predskazanje, nelogicnosti,
  zanemarivati, odugovlaciti,
  predosecaj
```

Zadatak 4.1.23 U datoteci `razno.txt` nalazi se tekst. U datoteku `palindromi.txt` prepisati sve reči iz datoteke `razno.txt` koje su palindromi. Reč je palindrom ako se čita isto sa leve i desne strane. Za reč smatramo niz karaktera koji se nalazi izmeų belina i koji nije duži od 200 karaktera. Dozvoljeno je korišćenje specifikatora za čitanje reči. Maksimalan broj reči nije poznat. U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

Primer 1

```
RAZNO.TXT
 Ana i melem su primeri palindroma.
PALINDROMI.TXT:
 Ana i melem
```

Primer 2

```
RAZNO.TXT
 jabuka neven pomorandza kuk
 Oko kapAk pero radar caj
PALINDROMI.TXT:
 neven kuk\datoteka{Oko kapAk radar}
```

Zadatak 4.1.24 U datoteci čije se ime navodi na standarnom ulazu programa nalazi se broj `n`, a zatim i `n` reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

(a) ispisuje ga [3],

(b) iz njega uklanja sve duplikate i u datoteku `rez.txt` ispisuje transformisani niz [4]

U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 dat1.txt
DAT1.TXT
 12 jha14 hahaha deda mraz deda
 mraz deda deda jase konj konj konj
IZLAZ:
 jha14 hahaha deda mraz deda mraz deda
 deda jase konj konj konj
REZ.TXT:
 jha14 hahaha deda mraz jase konj
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
 dat2.txt
DAT2.TXT
 14
 so secer supa so ljuto secer kiselo slatko
 ljuto
 paprika, ljuta paprika, ljuto dete
IZLAZ:
 so secer supa so ljuto secer kiselo slatko
 ljuto paprika, ljuta paprika, ljuto dete
REZ.TXT:
 so secer supa ljuto kiselo slatko
 paprika, ljuta dete
```

Zadatak 4.1.25 U datoteci čije se ime navodi na standarnom ulazu programa nalazi se broj n, a zatim i n reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

(a) ispisuje ga, [3]

(b) u datoteku `rez.txt` upisuje sve reči koje sadrže prvu reč i podvlaku. [4]

U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT
  7 rec Opet _rec Reci rec_enica
  DVa recica_
IZLAZ:
  rec Opet _rec Reci rec_enica
  DVa recica_
REZ.TXT:
  _rec rec_enica recica_
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  dat2.txt
DAT2.TXT
  11 Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
      suncanica.
IZLAZ:
  Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
      suncanica.
REZ.TXT:
  Sunce_Moje Sunce123_123
```

Zadatak 4.1.26 Imena dve datoteke se zadaje na standarnom ulazu. U prvoj datoteci navedena je rec `r` i niz linija. Napisati program koji u drugu datoteku upisuje sve linije u kojima se reč `r` pojavljuje bar `n` puta, gde je n prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu `broj_pojavljivanja: linija`. Linije brojati počevši od `1`. Maksimalna dužina naziva datoteka je 20 karaktera.

Zadatak 4.1.27 Napisati program koji poredi dva fajla i ispisuje redni broj linija u kojima se fajlovi razlikuju. Imena fajlova se zadaju kao argumenti komandne linije. U slučaju neuspešnog otvaranja datoteka ispisati poruku o grešci. Pretpostaviti da je maksimalna dužina reda u datoteci 200 karaktera. Ukoliko nisu zadati potrebni argumenti komadne linije ispisati poruku o grešci. Linije brojati pov cevši od `1`.

Primer 1

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt
ULAZ.TXT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ.TXT:
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ:
```

Primer 2

```
POKRETANJE: ./a.out primer1.dat primer2.dat
PRIMER1.DAT
  danas vezbamo
  analizu
  ovo je primer kad
  su datoteke razlicite
PRIEMR2.DAT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke razlicite
IZLAZ:
  2 3 4
```

Primer 3

```
POKRETANJE: ./a.out prva.dat
IZLAZ:
 greska
```

Primer 2

```
POKRETANJE: ./a.out prva.dat druga.dat
PRVA.DAT
 ovo je primer
 kada su
 datoteke
 razlicite duzine
DRUGA.DAT
 kada su
 programiranje
 datoteke
 razlicite
 duzine
 i kada treba ispisati broj
 tih redova
IZLAZ:
 1 4 5 6 7
```

Zadatak 4.1.28 Definisati strukturu

```
typedef struct{
   unsigned int a, b;
   char ime[5];
}_pravougaonik;
```

kojom se opisuje pravougaonik dužinama svojih stranica i imenom. Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava pravougaonike (nepoznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine medju pravougaonicima koji nisu kvadrati. U slučaju unosa nekorektnih dužina stranica pravougaonika ili nekorektne vrednosti broja `n`, ispisati -1 i odmah prekinuti izvršavanje programa. Maksimalan broj pravougaonika je 200.

Primer 1

```
POKRETANJE: ./a.out pravougaonici.dat
PRAVOUGAONICI.DAT
 2 4 p1
 3 3 p2
 1 6 p3
IZLAZ:
 p2 8
```

Primer 2

```
POKRETANJE: ./a.out dva.dat
DVA.DAT
 5 2 pm
 4 7 pv
IZLAZ:
 28
```

Primer 3

```
POKRETANJE: ./a.out tri.dat
TRI.DAT
 5 5 m
 3 3 s
 8 8 xl
IZLAZ:
 m s xl
```

Primer 4

```
POKRETANJE: ./a.out primerx.dat
PRIMERX.DAT
 9 7 p
IZLAZ:
 63
```

Primer 5

```
POKRETANJE: ./a.out prazna.dat
PRAZNA.DAT
IZLAZ:
```

Zadatak 4.1.29 Ime datoteke dato je kao argument komandne linije. U datoteci se nalaze otvorene i zatvorene zagrade i još nekakav tekst. Proveriti da li su zagrade pravilno uparene. Npr. `ab( cd) ..` odgovor je `jesu`, a `..)ba()` odgovor je `nisu`. Ukoliko nisu zadati svi argumenti komadne linije ispisati poruku o grešci.

Primer 1

```
POKRETANJE: ./a.out
      zagrade.txt
ZAGRADE.TXT
 ab( cd) ..
 ((3+4)*5+1)*9
IZLAZ:
 jesu
```

Primer 2

```
POKRETANJE: ./a.out
      primer2.dat
PRIMER2.DAT
 (7+8
 nisu(
 uparene
IZLAZ:
 nisu
```

Primer 3

```
POKRETANJE: ./a.out
      primer3.dat
PRIMER3.DAT
 )) 7 + 6 ((
IZLAZ:
 nisu
```

Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
 greska
```

Zadatak 4.1.30 Napraviti strukturu `STUDENT` koja sadrži:

- `ime` (u polju se čuva ime i prezime studenta, npr. ”Marko Markovic”, maksimalna dužina polja je 100 karaktera),
- `oc` (sadrži najviše 10 ocena studenta)
- `br_ocena` (ukupan broj ocena za studenata)
- `pr_oc` (prosečna ocena)

U datoteci se nalaze podaci o studentima. Za svakog studenta unosi se ime i prezime razdvojeno razmakom (uputstvo: može se koristiti `strcat` da spoji ime i prezime koji se mogu pročitati sa specifikatorom `%s`), a potom ocene koje se završavaju sa 0. Pronaći studenta koji ima najveći prosek i ispisati sve njegove podatke (prosek ispisati na 2 decimale). Maksimalan broj studenta je 100. Ime datoteke se zadaje kao argument komandne linije.

Primer 1

```
POKRETANJE: ./a.out
      studenti.txt
STUDENTI.TXT
  Marko Markovic 5 6 7 8 9 0
  Jelena Jankovic 10 10 10 0
  Filip Viskovic 10 9 8 7 6 0
  Jana Peric 10 10 9 9 8 8 7 7
      0
IZLAZ:
  Jelena Jankovic 10 10 10 0
      10.00
```

Primer 2

```
POKRETANJE: ./a.out
IZLAZ:
  greska
```

Zadatak 4.1.31

a) Napisati C funkciju `int unesiSkup(char *s, FILE* f)` kojom se unosi skup elemenata iz datoteke F. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiđe na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspesno učitani.

b) Napisati funkciju `void prebroj(char *s, int *br_slova,int *br_cifara)` kojom se određuje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.

c) Napisati glavni program gde se unose podaci o skupu elemenata. Ime datoteke se zadaje kao argument komandne linije. Na stadardni izlaz ispisati informacije o broju slova i cifara (koristiti funkcije pod a) i b)).

Primer 1

```
POKRETANJE: ./a.out skup.txt
SKUP.TXT
  abc56ighj9012hjFGHH
IZLAZ:
  broj slova: 13
  broj cifara: 6
```

Primer 2

```
POKRETANJE: ./a.out skup2.txt
SKUP2.TXT
  ovdeimamo$dolar
IZLAZ:
  broj slova: 9
  broj cifara: 0
```

Primer 3

```
POKRETANJE: ./a.out skup3.txt
SKUP3.TXT
  broJ3
    broj5
IZLAZ:
  broj slova: 4
  broj cifara: 1
```

Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
 greska
```

Zadatak 4.1.32 Definisati strukturu

```
typedef struct{
   int x;
   int y;
   int z;
} vektor;
```

kojom se opisuje trodimenzioni vektor. U datoteci `vektori.txt` nalazi se nepoznati broj vektora (maksimalno ih može biti 200). Učitati ih u niz i ispisuje na standardnom izlazu koordinate vektora sa najvećom dužinom. Dužina vektora se izračunava po formuli:

$$|v| = \sqrt{x^2 + y^2 + z^2}$$

U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

Primer 1

```
VEKTORI.TXT
 2
 4 -1 7
 3 1 2
IZLAZ:
 4 -1 7
```

Primer 2

```
VEKTORI.TXT
 67
IZLAZ:
 -1
```

Primer 3

```
VEKTORI.TXT
 3
 0 0 0
 0 1 0
 1 0 0
IZLAZ:
 0 1 0
```

Primer 4

```
VEKTORI.TXT
 4
 3 0 1
 4 5 2
 1 0 0
 2 -1 2
IZLAZ:
 4 5 2
```

Zadatak 4.1.33 Prvi red datoteke `ulaz.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice A. Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji nalazi i štampa sve četvorke oblika `(A(i,j), A(i+1,j), A(i,j+1),A(i+1,j+1))` u kojima su svi elementi međusobno različiti.

4.2 Rešenja

Rešenje 4.1.1

```
/*
    Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
    datoteku izlaz.txt karakter po karakter.
*/

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

int main()
{
   int c;
   FILE *ulaz, *izlaz;

   /*
      Promenljive ulaz i izlaz predstavljaju
      pokazivace na ugradjenu strukturu FILE.
      Unutar ove strukture nalaze se polja neophodna
      za rad sa datotekama.

      Kada zelimo da radimo sa nekom datotekom,
      moramo je prvo otvoriti. Ugradjena funkcija
      fopen(dat, mode) otvara datoteku sa nazivom
      dat. Datoteka moze biti otvorena za citanje,
      pisanje ili nadovezivanje, sto odredjuje
      argument mode koji moze imati vrednost "r" (read),
      "w"(write) ili "a"(append).
   */

   ulaz=fopen("ulaz.txt","r");

   /*
      Do neuspesnog otvaranja datoteke moze doci
      ukoliko ne postoji datoteka sa datim nazivom
      ili je putanja do datoteke pogresna. U tom
      slucaju, funkcija fopen vraca pokazivac na NULL
      i tada treba prijaviti gresku. Datoteka stderr
      predstavlja standardnu datoteku u koju se upisuju
      greske. Stderr je podrazumevano postavljen
      na standardni izlaz.

      Ugradjena funkcija exit prouzrokuje zavrsetak programa.
      Argument ove funkcije je jedna od konstanti definisanih
      u biblioteci stdlib.h koje pokazuju da li se program
      zavrsio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).

   */
```

```
    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.
62     Povratna vrednost ove funkcije je ascii kod unetog
       karaktera.
64
       Funkcija fputc ispisuje karakter c u datoteku izlaz.
66  */

68  while((c=fgetc(ulaz))!=EOF)
       fputc(c,izlaz);
70
    /*
72     Nakon zavrsetka rada sa datotekama, neophodno ih je
       zatvoriti pomocu ugradjene funkcije fclose.
74  */
    fclose(ulaz);
76  fclose(izlaz);
    return 0;
78 }
```

Rešenje 4.1.2

```
/*
2   Napisati program koji u datoteci cije se ime navodi kao prvi
    argument komandne linije odredjuje liniju maksimalne duzine i
4   ispisuje je na standarni izlaz. Ukoliko ima vise takvih linija,
    ispisati onu koja je leksikografski prva. Mozemo pretpostaviti
6   da datoteka ne sadrzi linije duze od 80 karaktera.
*/
8 #include <stdio.h>
  #include <stdlib.h>
10 #include <string.h>
  #define MAX_LEN 81
12
  int main(int argc, char* argv[])
14 {
```

```
    char linija[MAX_LEN];
16  char max_linija[MAX_LEN];
    int duzina;
18  int max_duzina;

20  FILE *ulaz, *izlaz;

22  /*
      Proveravamo da li poziv programa ima dovoljan broj argumenata.
24  */
    if(argc!=2)
26  {
       fprintf(stderr,"Greska: program se pokrece sa: %s
     ime_ulazne_datoteke\n", argv[0]);
28     exit(EXIT_FAILURE);
    }
30

    ulaz=fopen(argv[1],"r");
32  if(ulaz==NULL)
    {
34     fprintf(stderr,"error fopen(): Neuspelo otvoranje datoteke %s
     za citanje.\n", argv[1]);
       exit(EXIT_FAILURE);
36  }

38  /*
       Funkcija fgets ucitava jednu liniju teksta maksimalne duzine
     MAX_LEN
40     iz datoteke ulaz u string linija. Ukoliko ucitavanje ne uspe (
     na primer,
       zato sto smo dosli do kraja datoteke), povratna vrednost ove
     funkcije
42     bice prazan pokazivac (NULL).
    */
44

    max_duzina=0;
46  while(fgets(linija, MAX_LEN, ulaz)!=NULL)
    {
48     duzina = strlen(linija);
       /*
50        Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u
      petlju.
          Ovu promenljivu menjamo kada je duzina ucitana linije
52        veca od max_duzina ili kada su jednake, ali je ucitana
     linija
          leksikografski ispred trenutne linije sa maksimalnom duzinom
     .
54

          Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii
     kodova prva dva
56        razlicita karaktera stringova s1 i s2 na istim indeksima,
     ukoliko oni
```

```
            postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga
     osetljiva
58          na mala i velika slova (npr 'D' je leksikografski ispred 'p
     ').

60        */

62        if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija,
     max_linija)<0))
          {
64           strcpy(max_linija, linija);
             max_duzina=duzina;
66        }
    }
68
    /*
70     Funkcija fputs ispisuje string koji je njen prvi argument u
     datoteku
       koja je njen drugi argument. Sve funkcije za ucitavanje iz
     datoteka i
72     upis u datoteke (fgetc, fputc, fgets, fputs, ...) mozemo
     koristiti
       i kada radimo sa standardnim ulazom i standardnim izlazom. Kao
     nazive
74     datoteka tada navodimo stdin i stdout.
    */
76  fputs(max_linija, stdout);

78  fclose(ulaz);
    return 0;
80 }
```

Rešenje 4.1.3

```
/*
2   U datoteci cije se ime zadaje kao prvi argument komandne linije
     nalazi se
    prirodan broj n a zatim i n celih brojeva. Napisati program koji
     prebrojava
4   koliko k-tocifrenih brojeva postoji u datoteci, pri cemu se
     prirodan broj k
    zadaje kao drugi argument komandne linije.
6 */

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

12 /*
    Funkcija ucitaj_i_prebroj ucitava brojeve
14  iz datoteke na koju pokazuje f i prebrojava
```

```
   koliko je medju njima k-tocifrenih brojeva
*/
int ucitaj_i_prebroj(FILE* f, int k)
{
   int n;
   int x;
   int i;
   int br;

   /* U datoteci je prvo naveden ukupan broj brojeva. */
   fscanf(f,"%d",&n);

   /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
    */
   if(n<=0)
   {
      fprintf(stderr, "Greska: broj n mora biti prirodan\n");
      exit(EXIT_FAILURE);
   }

   br=0;
   for(i=0;i<n;i++)
   {
      fscanf(f,"%d",&x);
      if(broj_cifara(x)==k)
         br++;
   }

   return br;
}

int broj_cifara(int x)
{
   int br_c;

   br_c=0;

   /*
      Do while petlja je pogodnija od petlji sa preduslovom
      jer tacno racuna broj cifara i za broj 0.
   */
   do
   {
     br_c++;
     x/=10;
   } while(x);

   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 4.1.4

```
/*
   U datoteci cije se ime navodi kao prvi argument komandne
   linije navedena je rec r i niz linija. Napisati
   program koji u datoteku cije se ime navodi kao
   drugi argument komandne linije upisuje sve linije
   u kojima se rec r pojavljuje bar n puta, gde je
   n prirodan broj koji se unosi sa standardnog ulaza. Ispis
   treba da bude u formatu broj_pojavljivanja: linija.
*/
```

```c
#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++;
   }

   return br;
}
int main(int argc, char* argv[])
{
   char rec[MAXR];
   char linija[MAXL];
   FILE* in, *out;
   int n;
   int br;

   if(argc!=3)
   {
      fprintf(stderr,"Greska: program se pokrece sa: %s
    ime_ulazne_datoteke ime_izlazne_datoteke\n", argv[0]);
      exit(EXIT_FAILURE);
```

```
61      }
63      in= fopen(argv[1],"r");
        if(in==NULL)
65      {
           fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
         .\n", argv[1] );
67         exit(EXIT_FAILURE);
        }
69
        out= fopen(argv[2],"w");
71      if(out==NULL)
        {
73         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
         .\n", argv[2] );
           exit(EXIT_FAILURE);
75      }

77      printf("Unesi n:");
        scanf("%d", &n);
79
        if(n<=0)
81      {
           fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
83         exit(EXIT_FAILURE);
        }
85
        fscanf(in,"%s",rec);
87
        while(fgets(linija,MAXL,in)!=NULL)
89      {
           br = broj_pojavljivanja(linija,rec);
91         if (br>=n)
              fprintf(out,"%d: %s\n", br, linija);
93      }
        fclose(in);
95      fclose(out);
        return 0;
97  }
```

Rešenje 4.1.5

```
1   /* Program se pokrece tako sto se navedu nazivi dve datoteke(ulazna i
         izlazna) i opcije.
       U datoteci cije se ime navodi kao prvi argument komandne linije
        nalaze se podaci o razlomcima:
3      u prvom redu se nalazi broj razlomaka, a u svakom sledecem redu
        brojilac i imenilac jednog razlomka.
       Potrebno je kreirati strukturu koja opisuje razlomak i ucitati niz
         razlomaka
5      iz datoteke, a potom:
```

```
      a) ukoliko je navedena opcija x, upisati u datoteku cije je ime
     drugi argument komandne linije
         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;
```

```
53
    fscanf(f,"%d", dim);
55  for (i=0; i<*dim; i++)
    {
57      fscanf(f,"%d %d", &niz[i].br, &niz[i].im);
        if (niz[i].im==0)
59          return 0;
    }
61  return 1;
}
63
RAZLOMAK reciprocni(RAZLOMAK* r)
65{
    RAZLOMAK rec;
67  rec.im = r->br;
    rec.br = r->im;
69  return rec;
}
71
float vrednost(RAZLOMAK* r)
73{
    return 1.0*r->br/r->im;
75}

77int main(int argc, char* argv[])
{
79  FILE *in, *out;
    char c;
81  int i;
    int j;
83  int xoption=0;
    int yoption=0;
85  int dim;
    RAZLOMAK razlomci[MAX];
87  RAZLOMAK r;

89  /*
        Prilikom pokretanja programa se, pored naziva ulazne i izlazne
91      datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
        obe opcije, sto znaci da je minimalni broj argumenata 3.
93
        Moguci nacini pokretanja:
95      ./a.out ulaz.txt izlaz.txt -x
        ./a.out ulaz.txt izlaz.txt -y
97      ./a.out ulaz.txt izlaz.txt -yx
        ./a.out ulaz.txt izlaz.txt -xy
99
    */
101
    if(argc!=4)
103 {
        fprintf(stderr,"Greska: program se pokrece sa: %s
```

```
    ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n", argv[0]);
105       exit(EXIT_FAILURE);
      }
107
109   in= fopen(argv[1],"r");
      if(in==NULL)
111   {
         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
      .\n", argv[1] );
113      exit(EXIT_FAILURE);
      }
115
      out= fopen(argv[2],"w");
117   if(out==NULL)
      {
119      fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
      .\n", argv[2] );
         exit(EXIT_FAILURE);
121   }

123   /* Ispitujemo da li je treca opcija u trazenom formatu. Prvi
      karakter mora biti '-'.*/

125   if (argv[3][0] != '-')
      {
127      fprintf(stderr, "Greska u zadavanju opcija: program se pokrece
      sa: %s ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n",
      argv[0]);
         exit(EXIT_FAILURE);
129   }

131   /* Ostali karakteri mogu biti 'x' ili 'y'. U zavisnosti od date
      opcije, postavljamo
        vrednosti indikatorskih promenljivih xoption i yoption. */
133
      for(j=1;argv[3][j]!='\0';j++)
135       switch(argv[3][j])
          {
137          case 'x': xoption=1;
                       break;
139          case 'y': yoption=1;
                       break;
141          default:
                       fprintf(stderr, "Greska: nedozvoljeni karakter\n"
        );
143                    exit(EXIT_FAILURE);
           }
145

147    if(ucitaj_razlomke(razlomci, &dim, in)==0)
       {
```

```
149         fprintf(stderr, "Greska pri zadavanju razlomaka\n");
            exit(EXIT_FAILURE);
151     }

153     /*
           U zavisnosti od datih opcija, vrsimo upis reciprocnih
155        razlomaka u trazenom formatu.

157        Funkcija fprintf se koristi na isti nacin kao
           funkcija printf uz dodatni prvi argument koji
159        oznacava naziv datoteke u koju se vrsi upis.
        */
161     for (i=0; i<dim;i++)
        {
163        /*
              Ukoliko je brojilac razlomka jednak nuli,
165           nema smisla traziti njegovu reciprocnu vrednost
           */
167        if (razlomci[i].br==0)
              continue;
169
           r = reciprocni(&razlomci[i]);
171
           if (xoption)
173           fprintf(out,"%d/%d ", r.br, r.im);

175        if (yoption)
              fprintf(out, "%f ", vrednost(&r));
177
           fprintf(out,"\n");
179     }

181     fclose(in);
        fclose(out);
183
        return 0;
185 }
```

Rešenje 4.1.6

```
1  /*
     Za svaki automobil poznati su marka, model i cena. Iz datoteke cije
3    se ime zadaje sa standardnog ulaza ucitava se broj automobila a
       potom
     i podaci za svaki automobil. Program treba da:
5    a) izracuna prosecnu cenu po marki kola
     b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
       zadaje
7    kao argument komandne linije, da ispise automobile u tom cenovnom
     rangu zajednu sa prosecnom cenom odgovarajuce marke
9
```

```
   Mozemo pretpostaviti da se model i marka sastoje od jedne reci i
   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.
```

```
63     Povratna vrednost ove funkcije je indeks pojavljivanja, ukoliko
       se marka m pojavljuje u nizu proseka, ili -1 u suprotnom.
65 */

67 int sadrzi(INFO p[], int n, char m[])
   {
69     int i;
       for(i=0;i<n;i++)
71         if(strcmp(p[i].marka,m)==0)
               return i;
73
       return -1;
75 }

77 /*
       Funkcija informacije_o_markama za niz automobila a dimenzije n
79     racuna proseke cena automobila po markama i smesta ih u niz
       p. Na dimenziju niza p pokazuje pokazivac pn.
81
       Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
83     sumu cena automobila te marke (koju cemo smestiti u polje vrednost
         strukture
       INFO), i broj automobila te marke (koju cemo smestiti u polje
85     n strukture INFO) i da na kraju podelimo ove dve promenljive
       i tako dobijemo prosecnu vrednost cene.
87
       Za svaki automobil a[i] proveravamo da li se njegova marka vec
89     nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
       vredost cene automobila a[i] i uvecavamo broj automobila sa
91     tom markom. U suprotnom, dodajemo novi element u niz p. Posto
       ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
93     na koju pokazuje pokazivac *pn.
   */
95 void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
   {
97     int i,j;
       int ind;
99     for(i=0;i<n;i++)
       {
101        /* Proveravamo da li se marka automobila a[i] vec nalazi u
              nizu p (niz proseka po markama) */
103        ind = sadrzi(p,*pn1,a[i].marka);
           if(ind==-1) /* Ako se ne nalazi, uvodimo novi element niza na
        kraj, na poziciju *pn. */
105        {
               strcpy(p[*pn1].marka, a[i].marka);
107            p[*pn1].vrednost = a[i].cena;
               p[*pn1].n = 1;
109            (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
         */
           }
```

```
111         else /* Ako se nalazi, azuriramo polja strukture. */
            {
113             p[ind].vrednost+=a[i].cena;
                p[ind].n++;
115         }
        }
117
        /* Na osnovu sume cena i broja automobila racunamo prosecnu
         vrednost. */
119     for(i=0;i<*pn1;i++)
            p[i].vrednost = p[i].vrednost/p[i].n;
121
}
123
void stampaj_informacije(INFO p[], int n)
125 {
        printf("Informacije o broju automobila i prosecnoj ceni po markama
         :\n");
127     int i;
        for(i=0;i<n;i++)
129         printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
}
131
/*
133     Funkcija stampa automobile cija je cena manja od maksimalne
        cene koju je korisnik naveo u komandnoj liniji da je spreman
135     da plati, zajedno sa prosecnom cenom za tu marku automobila
*/
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
         n1)
{
139     /*
            S obzirom da je niz p formiran na osnovu niza a, marka svakog
141         automobila iz niza a se sigurno nalazi u nizu p. Zbog toga
            nije neophodno proveravati da li je povratna vrednost funkcije
143         sadrzi razlicita od -1.
        */
145     int i;
        printf("Kola u vasem cenovnom rangu:\n");
147     for(i=0;i<n;i++)
            if(a[i].cena<g)
149             printf("%s %s %.2f\n", a[i].marka, a[i].model, p[sadrzi(p,n1
         ,a[i].marka)].vrednost);
}
151
int main(int argc, char* argv[])
153 {
        AUTOMOBIL kola[MAXA];
155     FILE* f;
        char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
         ulaza. */
157     float granica; /* Maksimalna cena koju je korisnik spreman da
```

```
     plati.
                        Zadaje se kao argument komandne linije.
159                  */
    INFO infos[MAXA];
161 int dim_kola,dim_infos;
    int i;
163
    if(argc!=2)
165 {
       fprintf(stderr,"Greska: program se pokrece sa: %s
     gornja_granica_cene \n", argv[0]);
167    exit(EXIT_FAILURE);
    }
169
    /* Argumenti komandne linije su stringovi. Da bismo od stringa
     dobili
171    realan broj, koristimo ugradjenu funkciju atof. */
    granica = atof(argv[1]);
173
    printf("Unesi naziv datoteke:");
175 scanf("%s", dat);

177 f=fopen(dat, "r");

179 if(f==NULL)
    {
181    fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
     .\n", dat);
       exit(EXIT_FAILURE);
183 }

185 if (ucitaj_podatke(f,kola,&dim_kola,MAXA)==0)
    {
187    fprintf(stderr, "Greska pri ucitavanju podataka\n");
       exit(EXIT_FAILURE);
189 }


191
    informacije_o_markama(kola, dim_kola, infos, &dim_infos);
193
    stampaj_informacije(infos,dim_infos);
195
    stampaj_kandidate(kola, dim_kola, granica, infos, dim_infos);
197
    fclose(f);
199 return 0;
}
```

Rešenje 4.1.7

```c
/* 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 4.1.8

```c
/* 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;

  /* Citamo karakter po karakter iz datoteke sve dok ne stignemo do
    kraja datoteke */
  while((c=fgetc(in)) != EOF){

    /* Ukoliko je procitani karakter na poziciji koja je deljiva sa 3
     prepisujemo ga */
    if(rbr_karaktera%3==0)
    fputc(c, out);

  /* Uvecavamo redni broj karaktera */
  rbr_karaktera++;
  }

  /*Zatvaramo obe datoteke koje smo otvorili*/
  fclose(out);
  fclose(in);
  return 0;
}
```

Rešenje 4.1.9

```
/* Kao argumenti komandne linije se zadaju ime datoteke i ceo broj k.
     Napisati program koji na standardni izlaz
ispisuje sve linije zadate datoteke cija je duzina veca od k. Moze se
     pretpostaviti da duzina linije nece biti veca
od 80 karaktera */

#include<stdio.h>
#include<string.h>

#define MAXL 81

int main(int argc, char* argv[]){

  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*/
40    fclose(in);
      return 0;
42  }
```

Rešenje 4.1.10

```
/* 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];
  int broj_linija = 0;
  int duzina_s = strlen(s);
  int duzina_linije;

  while(fgets(linija, MAXL, in) != NULL){
  duzina_linije = strlen(linija);

  /* Ukoliko je znak za novi red bio indikacija kraja linije,
    uklanjamo ga kako bi mogli da izvrsimo
   *ispravno poredjenje (jer niska s nema novi red na kraju) */
  if(linija[duzina_linije-1]=='\n'){
    linija[duzina_linije-1] = '\0';
    duzina_linije--;
  }

  /*linija + duzina_linije ce nas odvesti na kraj tog stringa, a kada
     oduzmemo duzinu stringa s,
    a kada od toga oduzmemo duzinu niske s, dobicemo bas onoliko
    poslednjih karaktera, koliko
    nam i treba. U primeru uspravna crta (|) oznacava pokazivac
       s              ab
       duzina_s           2
       Linija:            aaabbbdfsssab
                     |
       Linija + duzina linije   aaabbbdfsssab
                           |
       Linija + duzina linije - 2 aaabbbdfsssab
                           |
      kada kazemo strcmp(linija + duzina_linije - duzina_s, s), mi
    cemo u nasem primeru zaista porediti "ab" i "ab".
   */
  if(strcmp(linija + duzina_linije - duzina_s, s) == 0)
    broj_linija++;
  }
  return broj_linija;
}

int main(){

  FILE* in;
  char s[MAXS];

  /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
    uspesno otvorili*/
  in = fopen("ulaz.txt", "r");
  if(in == NULL){
  printf("Greska: neuspesno otvaranje datoteke!\n");
  return 0;
  }

  /*Ucitavamo nisku*/
```

```
62    printf("Unesite nisku s: ");
      scanf("%s", s);
64
      /*Ispisujemo koliko linija iz datoteke se zavrsava sa niskom s*/
66    printf("Broj linija: %d\n", brojLinija(in, s));

68    /*Zatvaramo datoteku*/
      fclose(in);
70
      return 0;
72  }
```

Rešenje 4.1.11

```
/* Napisati program koji pronalazi maksimum brojeva zapisanih u
     datoteci brojevi.txt */
2
#include<stdio.h>
4
int main(){
6
  FILE* in;
8 float broj, max_broj;

10 /*Otvaramo datoteku brojevi.txt za citanje i proveravamo da li smo
     je uspesno otvorili*/
   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;
}
```

Rešenje 4.1.12

```
/* U datoteci studenti. txt se nalaze informacije o studentima: prvo
    broj studenata, a zatim u pojedinacnim linijama
korisnicko ime i pet poslednjih ocena koje je student dobio. Napisati
     program koji pronalazi studenta koji je
ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da
    broj studenata nece biti veci od 100. */

#include<stdio.h>

#define MAXS 100

/*Definisemo strukturu za cuvanje studenata*/
typedef struct st{
  char korisnicko_ime[8];
  float prosek;
}STUDENT;

int main(){

  FILE *ulaz;
  STUDENT studenti[MAXS];

  int ocena1,ocena2,ocena3,ocena4,ocena5, i=0, i_max_prosek;
  float max_prosek = 0;

  /*Otvaramo datoteku studenti.txt za citanje*/
  ulaz = fopen("studenti.txt", "r");
  if(ulaz == NULL){
  printf("Greska pri otvaranju datoteke!\n");
  return 0;
  }

  /*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.
   Zatim, na osnovu ocena racunamo prosek.

   Ovde paralelno sa ucitavanjem pronalazimo i studenta sa najvecim
    prosekom i
   pamtimo njegov prosek i njegovu poziciju u nizu studenata,
   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.

   */
```

```
  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;

  if(studenti[i].prosek > max_prosek){
     max_prosek= studenti[i].prosek;
     i_max_prosek = i;
  }
  i++;
  }

  /*Ispisujemo rezultat*/
  printf("korisnicko ime: %s, prosek ocena: %.2f\n", studenti[
    i_max_prosek].korisnicko_ime, studenti[i_max_prosek].prosek);

  /*Zatvaramo datoteku*/
  fclose(ulaz);

  return 0;
}
```

Rešenje 4.1.13

Rešenje 4.1.14

```
/* Napisati program koji za rec s maksimalne duzine 20 karaktera koja
     se zadaje sa standardnog ulaza u datoteku
rotacije.txt upisuje sve rotacije reci s */

#include<stdio.h>
#include<string.h>

#define MAXS 21

/*Funkcija rotira nisku za jedno mesto u desno.
 Duzina niske n nije obavezan argument. Mogli smo
 i da je racunamo u okviru funkcije, ali kako ce sve niske
 sa kojima radimo biti iste duzine, efikasnije je da jednom
 izracunamo tu duzinu u glavnom programu,
 pa da je prosledjujemo kao argument.*/
void rotiraj_za_1(char* s, int n){
  int i;
  char c = s[0];
  for(i=0; i<n-1; i++){
  s[i] = s[i+1];
  }
  s[n-1] = c;
}

```

```
int main(){

  char s[MAXS];
  int n, i;
  FILE * izlaz;

  /*Otvaramo datoteku rotacije.txt za pisanje i proveravamo da li smo
     je uspesno otvorili*/
  izlaz = fopen("rotacije.txt", "w");
  if(izlaz == NULL){
  printf("Greska pri otvaranju fajla!");
  return 0;
  }

  /*Sa standardnog ulaza ucitavamo rec koju treba da rotiramo*/
  scanf("%s", s);

  /*Racunamo njenu duzinu*/
  n = strlen(s);

  /*U petlji, ispisujemo tu rec u datoteku, pa je rotiramo za jedno
    mesto u desno.*/
  for(i=0; i<n; i++){
  fprintf(izlaz, "%s\n", s);
  rotiraj_za_1(s,n);
  }

  /*Zatvaramo datoteku rotacije.txt*/
  fclose(izlaz);

  return 0;

}
```

Rešenje 4.1.15

```
/* Napisati program koji linije koje se ucitavaju sa standardnog
    ulaza sve do kraja ulaza prepisuje u datoteku
izlaz:txt i to, ako je prilikom pokretanja zadata opcija -v ili -V
    samo one linije koje pocinju velikim slovom,
ako je zadata opcija -m ili -M samo one linije koje pocinju malim
    slovom, a ako je opcija izostavljena sve linije.
Pretpostaviti da linije nece biti duze od 80 karaktera.
*/

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

#define MAXL 81
```

```
13 int main(int argc, char* argv[]){

15   char linija[MAXL];
     FILE* izlaz;
17
     /*Indikatori koji oznacavaju koja opcija je navedena kao argument
       komandne linije
19   vind - ispisuju se recenice koje pocinju velikim slovom
     mind - ispisuju se recenice koje pocinju malim slovom
21    */
     int vind=0, mind = 0;
23
     /*Proveravamo da li je program ispravno pozvan*/
25   if(argc > 2){
     printf("Greska pri pozivanju programa!\n");
27   return 0;
     }
29
     /*Ako opcije nisu zadate, onda treba da se ispisuju sve recenice,
       pa postavljamo oba indikatora na 1*/
31   if(argc == 1){
     vind = mind = 1;
33   }else{

35   /*Proveravamo da li je postavljena neka od opcija -v,-V,-m, -M
      Ako jeste, postavljamo odgovarajuci indikator
37    Ako nije, onda ispisujemo poruku o gresci*/
     if(strcmp(argv[1], "-v") == 0 || strcmp(argv[1], "-V") == 0)
39     vind = 1;
     else if(strcmp(argv[1], "-m") == 0 || strcmp(argv[1], "-M") == 0)
41     mind = 1;
     else{
43     printf("Greska pri zadavanju opcije!\n");
       return 0;
45   }
     }
47


49   /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo da li smo je
        uspesno otvorili*/
     izlaz = fopen("izlaz.txt", "w");
51   if(izlaz == NULL){
     printf("Greska pri otvaranju datoteke!\n");
53   return 0;
     }
55
     /*Citamo liniju po liniju sa standardnog ulaza i ispisujemo je u
       datoteku.
57    Liniju ispisujemo ukoliko je ispunjen neki od dva uslova:
       1. Izabrana je opcija za ispis malih slova i linija pocinje malim
        slovom
59     2. Izabrana je opcija za velika slova i linija pocinje velikim
```

```
    slovom
    NAPOMENA: Kada dodje do kraja ulaza, funkcija fgets vraca NULL
61  */
  while(fgets(linija, MAXL, stdin) != NULL){
63  if( mind && islower(linija[0]) || vind && isupper(linija[0]) ||
      mind && vind)
      fputs(linija, izlaz);
65  }

67  /*Zatvaramo datoteku izlaz.txt*/
  fclose(izlaz);
69
  return 0;
71 }
```

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33