

## PROGRAMIRANJE 1



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

# **PROGRAMIRANJE 1**

## **Zbirka zadataka**

**Beograd  
2017.**

Autori:

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

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

*Danijela Simić*, asistent na Matematičkom fakultetu u Beogradu

*Anđelka Zečević*, asistent na Matematičkom fakultetu u Beogradu

*Aleksandra Kocić*, asistent na Matematičkom fakultetu u Beogradu

PROGRAMIRANJE 1

Zbirka zadataka

# Sadržaj

|          |                                     |            |
|----------|-------------------------------------|------------|
| <b>1</b> | <b>Uvodni zadaci</b>                | <b>1</b>   |
| 1.1      | Naredba izraza . . . . .            | 1          |
| 1.2      | Rešenja . . . . .                   | 12         |
| <b>2</b> | <b>Kontrola toka</b>                | <b>29</b>  |
| 2.1      | Naredbe grananja . . . . .          | 29         |
| 2.2      | Rešenja . . . . .                   | 41         |
| 2.3      | Petlje . . . . .                    | 71         |
| 2.4      | Rešenja . . . . .                   | 101        |
| 2.5      | Funkcije . . . . .                  | 161        |
| 2.6      | Rešenja . . . . .                   | 174        |
| <b>3</b> | <b>Predstavljanje podataka</b>      | <b>211</b> |
| 3.1      | Nizovi . . . . .                    | 211        |
| 3.2      | Rešenja . . . . .                   | 231        |
| 3.3      | Pokazivači . . . . .                | 288        |
| 3.4      | Rešenja . . . . .                   | 291        |
| 3.5      | Niske . . . . .                     | 300        |
| 3.6      | Rešenja . . . . .                   | 313        |
| 3.7      | Višedimenzioni nizovi . . . . .     | 350        |
| 3.8      | Rešenja . . . . .                   | 365        |
| 3.9      | Strukture . . . . .                 | 400        |
| 3.10     | Rešenja . . . . .                   | 412        |
| <b>4</b> | <b>Ulaz i izlaz programa</b>        | <b>443</b> |
| 4.1      | Argumenti komandne linije . . . . . | 443        |
| 4.2      | Rešenja . . . . .                   | 446        |
| 4.3      | Datoteke . . . . .                  | 451        |
| 4.4      | Rešenja . . . . .                   | 470        |



# 1

## Uvodni zadaci

### 1.1 Naredba izraza

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Zdravo svima!
```

[Rešenje 1.1.1]

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

*Primer 1*

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

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite ceo broj: -14  
|| Kvadrat: 196  
|| Kub: -2744
```

[Rešenje 1.1.2]

**Zadatak 1.1.3** Napisati program koji za uneta dva cela broja  $x$  i  $y$  ispisuje njihov zbir, razliku, proizvod, ceo deo pri deljenju prvog broja drugim brojem i

## 1 Uvodni zadaci

---

ostatak pri deljenju prvog broja drugim brojem. NAPOMENA: *Pretpostaviti da je unos ispravan.*

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.3]

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.4]

**Zadatak 1.1.5** Napisati program koji za unetu količinu jabuka u kilogramima i unetu cenu po kilogramu ispisuje ukupnu vrednost date količine jabuka. Obe ulazne vrednosti su pozitivni celi brojevi. NAPOMENA: *Pretpostaviti da je unos ispravan.*

### Primer 1

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

### Primer 1

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

[Rešenje 1.1.5]

**Zadatak 1.1.6** Napisati program koji pomaže kasirki da obračuna kusur koji treba da vrati kupcu. Za unetu cenu artikla, količinu artikla i iznos koji je



kupac dao, program treba da ispiše vrednost kusura. Sve ulazne vrednosti su pozitivni celi brojevi. NAPOMENA: *Pretpostaviti da je unos ispravan.*

### Primer 1

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

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite cenu, kolicinu i iznos:  
59 6 2000  
Kusur je 1646 dinara.
```

[Rešenje 1.1.6]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vreme poletanja: 8 5  
Unesite vreme sletanja: 12 41  
Duzina trajanja leta je 4 h i 36 min
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vreme poletanja: 13 20  
Unesite vreme sletanja: 18 45  
Duzina trajanja leta je 5 h i 25 min
```

[Rešenje 1.1.7]

**Zadatak 1.1.8** Date su dve celobrojne promenljive  $x$  i  $y$ . Napisati program koji razmenjuje njihove vrednosti.

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.8]

**Zadatak 1.1.9** Date su dve celobrojne promenljive  $a$  i  $b$ . Napisati program koji promenljivoj  $a$  dodeljuje njihovu sumu, a promenljivoj  $b$  njihovu razliku. NAPOMENA: *Ne koristiti pomoćne promenljive.*

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.9]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.10]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.11]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.12]

**Zadatak 1.1.13** Napisati program koji za uneti pozitivan četvorocifreni broj:

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

NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.13]

**Zadatak 1.1.14** Napisati program koji ispisuje broj koji se dobija izbacivanjem cifre desetica u unetom pozitivnom celom broju. NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.14]

## 1 Uvodni zadaci

---

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite pozitivan ceo broj: 12345  
|| Unesite pozitivan dvocifreni broj: 67  
|| Novi broj je 1267345
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite pozitivan ceo broj: 50000000  
|| Unesite pozitivan dvocifreni broj: 12  
|| Novi broj je 705044704
```

[Rešenje 1.1.15]

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj inča: 4.69  
|| 4.69 in = 11.91 cm
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj inča: 71.426  
|| 71.43 in = 181.42 cm
```

[Rešenje 1.1.16]

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

*Primer 1*

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

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj milja: 327.128  
|| 327.128 mi = 526.46 km
```

[Rešenje 1.1.17]

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

## Primer 1

```

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

```

## Primer 2

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj funti: 89.437
|| 89.437 lb = 40.57 kg

```

[Rešenje 1.1.18]

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

## Primer 1

```

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

```

## Primer 2

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite temperaturu u F: 25.562
|| 25.562 F = -3.58 C

```

[Rešenje 1.1.19]

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

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

Pri ispisu vrednost zaokružiti na 4 decimale.

## Primer 1

```

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

```

## Primer 2

```

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

```

## Primer 3

```

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

```

## Primer 4

```

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

```

[Rešenje 1.1.20]

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

## 1 Uvodni zadaci

---

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.21]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite poluprecnik: 4.2  
|| Obim: 26.39  
|| Povrsina: 55.42
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite poluprecnik: 14.932  
|| Obim: 93.82  
|| Povrsina: 700.46
```

[Rešenje 1.1.22]

**Zadatak 1.1.23** Napisati program koji za unetu realnu vrednost dužine stranice jednakostraničnog trougla ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.* UPUTSTVO: *Za računanje korena broja koristiti funkciju `sqrt` čija se deklaracija nalazi u zaglavlju `math.h`.*

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.23]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 1.1.24]

**Zadatak 1.1.25** Pravougaonik čije su stranice paralelne koordinatnim osama zadan je svojim realnim koordinatama suprotnih temena (gornje levo i donje desno teme). Napisati program koji ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 1.1.25]

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

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite tri cela broja: 11 5 4
Aritmeticka sredina: 6.67

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite tri cela broja: 3 -8 13
Aritmeticka sredina: 2.67

```

[Rešenje 1.1.26]

**Zadatak 1.1.27** Napisati program koji pomaže moleru da izračuna površinu zidova prostorije koju treba da okreči. Za unete celobrojne vrednosti dimenzije

## 1 Uvodni zadaci

---

sobe u metrima (dužinu, širinu i visinu), program treba da ispiše površinu zidova za krečenje pod pretpostavkom da na vrata i prozore otpada oko 20%. Omogućiti i da na osnovu unete celobrojene cene usluge po kvadratnom metru program izračuna ukupnu cenu krečenja. Sve realne vrednosti ispisati zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije sobe: 4 4 3
Unesite cenu po m2: 500
Moler treba da okreći 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 okreći 320.80 m2
Cena krecenja je 152380.00
```

[Rešenje 1.1.27]

**Zadatak 1.1.28** Napisati program koji za unete pozitivne cele brojeve  $x$ ,  $p$  i  $c$  ispisuje broj koji se dobija ubacivanjem cifre  $c$  u broj  $x$  na poziciju  $p$ . Pretpostaviti da numeracija cifara počinje od nule, odnosno da se cifra najmanje težine nalazi se na nultoj poziciji. NAPOMENA: *Pretpostaviti da je unos ispravan.* UPUTSTVO: *Koristiti funkciju  $\text{pow}$  čija se deklaracija nalazi u zaglavlju  $\text{math.h}$ .*

### Primer 1

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

### Primer 1

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

[Rešenje 1.1.28]

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

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

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

### Primer 1

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

### Primer 2

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



[Rešenje 1.1.29]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 1.1.30]

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

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva cela broja: 4 8
|| Minimum je 4

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva cela broja: -3 -110
|| Minimum je -110

```

[Rešenje 1.1.31]

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

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

zaokruženu na dve decimale.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva realna broja: 5.7 11.2
|| Rezultat je: 0.05

```

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva realna broja: -9.34 8.99
|| Rezultat je: -0.11

```

[Rešenje 1.1.32]

### 1.2 Rešenja

#### Rešenje 1.1.1

```
#include<stdio.h>
2
int main()
4 {
    /* Ispisuje se trazena poruka. Na kraju poruke se ispisuje i
6     novi red. */
    printf("Zdravo svima!\n");
8
    /* Povratna vrednost 0 se obicno koristi da oznaci da je prilikom
10     izvršavanja programa sve proslo u redu. */
    return 0;
12 }
```

#### Rešenje 1.1.2

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

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

12     /* Ispis kvadratne vrednosti unetog broja. */
    printf("Kvadrat: %d\n", n * n);
14

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

18     return 0;
}
```

#### Rešenje 1.1.3

```
#include<stdio.h>
2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     int x, y, rezultat;
```

```

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

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

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

20  /* II nacin ispisa: direktan ispis vrednosti izraza, bez njegovog
   dodeljivanja posebnoj promenljivoj. */
22  printf("%d - %d = %d\n", x, y, x - y);
24  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,
   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 celobrojnem deljenju dve
   celobrojne promenljive.
   Da bi se odstampao karakter %, u naredbi printf se pise %%. */
34  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

```

2  #include <stdio.h>

```

## 1 Uvodni zadaci

---

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

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

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

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

    return 0;
}
```

### Rešenje 1.1.7

```
#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 celobrojn timer deljenjem broja sekundi
       sa 3600.
       Preostali broj minuta se dobija deljenjem preostalog broja
    */
}
```

```

32     sekundi sa 60. */
    duzina_sat = duzina / 3600;
    duzina_minut = (duzina - duzina_sat * 3600) / 60;
34
    /* II nacin: duzina_minut = (duzina % 3600) / 60; */
36
    /* Ispis rezultata. */
38    printf("Duzina trajanja leta je %u h i %u min\n", duzina_sat,
        duzina_minut);
40
    return 0;
42 }

```

### Rešenje 1.1.8

```

#include<stdio.h>
2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     int x, y;
    int p;
8
    /* Ucitavaju se vrednosti x i y. */
10    printf("Unesite vrednosti x i y: ");
    scanf("%d%d", &x, &y);
12
    /* Ispis vrednosti promenljivih pre zamene. */
14    printf("Pre zamene: x=%d, y=%d\n", x, y);
16
    /* Pomocna promenljiva p je potrebna da sacuva vrednost
       promenljive x pre nego sto se ona izmeni i dobije vrednost
       promenljive y. */
18    p = x;
20    x = y;
    y = p;
22
    /* Ispis vrednosti promenljivih nakon zamene. */
24    printf("Posle zamene: x=%d, y=%d\n", x, y);
26
    return 0;
}

```

### Rešenje 1.1.9

```

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

```

## 1 Uvodni zadaci

---

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

7

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

11

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

15  /* 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;

17

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

21  return 0;
}
```

### Rešenje 1.1.10

```
1  #include <stdio.h>

3  int main()
   {
5     /* Deklaracija neoznacnog broja. */
     unsigned int x;

7

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

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

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

21

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

29 }
```

```

31  /* II nacín: Ispis rezultata bez uvođenja dodatnih promenljivih
    cifra_jedinice, cifra_desetice i cifra_stotine:

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

35      return 0;
}

```

### Rešenje 1.1.11

```

1  #include <stdio.h>

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

9

11     /* Vrednost x/5000 predstavlja maksimalan broj novčanica od 5000
        dinara koje je moguće iskoristiti za plaćanje računa.
        Na primer, neka je uneta cena 8367 dinara, vrednost izraza
13     8367/5000 je jednaka 1. */
    printf("%u = %u*5000 + ", x, x / 5000);

15

17     /* Da bi se isti postupak primenio i na ostale novčanice, potrebno
        je izračunati preostali iznos. Jedan način da se to uradi je
        računanje ostatka pri deljenju unete vrednosti x
19     (u primeru 8367) sa 5000. On iznosi 3367. Ovu vrednost
        dodeljujemo promenljivoj x. */
21     x = x % 5000;

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

```

## 1 Uvodni zadaci

---

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

### Rešenje 1.1.12

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      unsigned int x;
7      unsigned int obrnuto_x;
8      char cifra_jedinice;
9      char cifra_desetice;
10     char cifra_stotine;
11
12     /* Ucitava se neoznaceni trocifreni broj. */
13     printf("Unesite trocifreni broj: ");
14     scanf("%u", &x);
15
16     /* Izdvajaju se pojedinačne cifre broja. */
17     cifra_jedinice = x % 10;
18     cifra_desetice = (x / 10) % 10;
19     cifra_stotine = x / 100;
20
21     /* Formira se rezultujući broj. */
22     obrnuto_x = cifra_jedinice * 100 + cifra_desetice * 10 +
23         cifra_stotine;
24
25     /* Ispis rezultata. */
26     printf("Obrnuto: %u\n", obrnuto_x);
27
28     return 0;
29 }
```

### Rešenje 1.1.13

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      unsigned int n, broj_obrnuto, broj_zamena;
7      char jedinice, desetice, stotine, hiljade;
8      int proizvod_cifara, razlika_cifara, suma_kvadrata;
9
10     /* Ucitava se jedan neoznaceni broj. */
11     printf("Unesite četvorocifreni broj: ");
```



```

scanf("%u", &n);

13
/* Izdvajaju se cifre ucitanog broja. */
15
jedinice = n % 10;
desetice = (n / 10) % 10;
17
stotine = (n / 100) % 10;
hiljade = n / 1000;
19

/* Izracunava se proizvod cifara. */
21
proizvod_cifara = jedinice * desetice * stotine * hiljade;
printf("Proizvod cifara: %d\n", proizvod_cifara);
23

/* Izracunava se razlika sume krajnjih i srednjih cifara. */
25
razlika_cifara = (hiljade + jedinice) - (stotine + desetice);
printf("Razlika sume krajnjih i srednjih: %d\n", razlika_cifara);
27

/* Izracunava se suma kvadrata cifara. */
29
suma_kvadrata = jedinice * jedinice + desetice * desetice +
                stotine * stotine + hiljade * hiljade;
31
printf("Suma kvadrata cifara: %d\n", suma_kvadrata);
33

/* Izracunava se broj zapisan istim ciframa ali u obrnutom
   redosledu. */
35
broj_obrnuto = jedinice * 1000 + desetice * 100 + stotine * 10 +
               hiljade;
printf("Broj u obrnutom poretku: %u\n", broj_obrnuto);
37

/* Izracunava se broj u kojem su cifra jedinica i cifra stotina
   zamenile mesta. */
39
broj_zamena = hiljade * 1000 + jedinice * 100 + desetice * 10 +
              stotine;
41
printf("Broj sa zamenjenom cifrom jedinica i stotina: %u\n",
       broj_zamena);
43
return 0;
}

```

### Rešenje 1.1.14

```

1 #include <stdio.h>

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

9     /* Ucitava se neoznaceni broj. */
    printf("Unesite broj: ");
11    scanf("%u", &broj);

```

## 1 Uvodni zadaci

---

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

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

21

23  /* Rezultat se dobija spajanjem levog i desnog dela.
    U datom primeru: 12*10 + 4 = 124. */
    novibroj = levo*10 + desno;

25

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

29  return 0;
}
```

### Rešenje 1.1.15

```
#include <stdio.h>

2  int main()
3  {
4      /* Deklaracija potrebnih promenljivih. */
5      unsigned int n, novibroj;
6      unsigned int levi, desni, m;
7
8
9      /* Ucitavaju se brojevi n i m. */
10     printf("Unesite pozitivan ceo broj: ");
11     scanf("%u", &n);
12     printf("Unesite pozitivan dvocifreni broj: ");
13     scanf("%u", &m);
14
15     /* Levi deo rezultata su sve cifre levo od cifre stotina.
16     Na primer, ako je n=12345, levi deo rezultata je 12.
17     On se dobija deljenjem unetog broja sa 1000. */
18     levi = n / 1000;
19
20     /* Desni deo rezultata su sve cifre desno od cifre hiljada.
21     Za n=12345, desni deo rezultata je 345. */
22     desni = n % 1000;
23
24     /* Srednji deo rezultata je broj m.
25     U navedenom primeru, rezultat se dobija nadovezivanjem
26     brojeva 12, 67 i 345. Ovo se radi mnozenjem delova sa
27     odgovarajucim stepenom broja 10 i njihovim sabiranjem. */
28     novibroj = levi * 100000 + m * 1000 + desni;
29
30     /* Ispis rezultata. */
```

```
printf("Novi broj je %u\n", novibroj);
return 0;
}
```

### Rešenje 1.1.16

```
1 #include <stdio.h>
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     float in, cm;
7
8     /* Ucitava se realna vrednost koja predstavlja broj inca. */
9     printf("Unesite broj inca: ");
10    scanf("%f", &in);
11
12    /* Izracunava se rezultat (1 in = 2.54 cm) */
13    cm = in * 2.54;
14
15    /* Ispis rezultata (na dve decimale). */
16    printf("%.2f in = %.2f cm\n", in, cm);
17
18    return 0;
19 }
```

### 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

```
1 #include <stdio.h>
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     float a11, a12, a21, a22;
7     float determinanta;
```

## 1 Uvodni zadaci

---

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

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

15

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

19  return 0;
   }
```

### Rešenje 1.1.21

```
#include <stdio.h>

2
int main()
{
4  /* Deklaracija potrebnih promenljivih. */
   float a, b;
   float obim, površina;

8

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

12

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

16  /* Izracunava se površina pravougaonika. */
   površina = a * b;

18

20  /* Ispis rezultata na dve decimala. */
   printf("Obim: %.2f\n", obim);
   printf("Površina: %.2f\n", površina);

22  return 0;
24 }
```

### Rešenje 1.1.22

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

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

```

8
/* Ucitava se poluprecnik kruga. */
10 printf("Unesite poluprecnik: ");
scanf("%f", &r);

12
/* Racunaju se obim i povrsina.
14     M_PI je konstanta koja se nalazi u zaglavlju math.h
    i njena vrednost odgovara pribilznoj vrednosti broja pi. */
16 obim = 2 * r * M_PI;
povrsina = r * r * M_PI;

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

22 return 0;
}

```

### Rešenje 1.1.23

```

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

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

8
    /* Ucitava se duzina stranice. */
10 printf("Unesite duzinu stranice trougla: ");
scanf("%f", &a);

12
    /* Racunaju se obim i povrsina. */
14 obim = 3 * a;
povrsina = (a * a * sqrt(3)) / 4;

16
    /* Ispis rezultata na dve decimale. */
18 printf("Obim: %.2f\n", obim);
printf("Povrsina: %.2f\n", povrsina);

20 return 0;

22 }

```

### Rešenje 1.1.24

```

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

3
int main()
5 {

```

## 1 Uvodni zadaci

---

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

9
/* Ucitavaju se duzine stranica. */
11 printf("Unesite duzine stranica trougla:\n");
   scanf("%f%f%f", &a, &b, &c);

13
/* Racuna se obim. */
15 obim = a + b + c;

17
/* Racuna se površina koriscenjem Heronovog obrasca. */
   s = obim / 2;
19 površina = sqrt(s * (s - a) * (s - b) * (s - c));

21
/* Ispis rezultata. */
   printf("Obim: %.2f\n", obim);
23 printf("Površina: %.2f\n", površina);

25 return 0;
}
```

### 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;
       float as;

7
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
          pretvori u realan broj. */
19      as = (a + b + c) / 3.0;

21
```

```

23  /* Drugi ispravni nacini:
    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: %.2f\n", as);
29
    return 0;
31 }

```

## Rešenje 1.1.27

```

1  #include <stdio.h>

3  int main()
{
5  /* Deklaracija potrebnih promenljivih. */
    unsigned int duzina, sirina, visina;
7  unsigned int cena;
    float površina_za_krecenje;
9  float ukupna_cena;

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

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

19 /* Povrsina za krecenje odgovara površini kvadra
    umanjena za površinu poda jer se on ne kreci. */
21 površina_za_krecenje = 0.8 * (duzina * sirina +
                                2 * duzina * visina +
23                                2 * sirina * visina);

25 /* Racuna se ukupna cena. */
    ukupna_cena = površina_za_krecenje * cena;
27

29 /* Ispis rezultata. */
    printf("Moler treba da okreći %.2f m2\n", površina_za_krecenje);
    printf("Cena krecenja je %.2f\n", ukupna_cena);
31
    return 0;
33 }

```

## Rešenje 1.1.28

## 1 Uvodni zadaci

---

```
1 #include <stdio.h>
  #include <math.h>
3
4 int main()
5 {
6     /* Deklaracija potrebnih promenljivih. */
7     unsigned int x, p, c;
8     unsigned int levo, desno;
9     unsigned int novo_x;
10
11     /* Ucitavaju se broj, pozicija i cifra. */
12     printf("Unesite redom x, p i c: ");
13     scanf("%u%u%u", &x, &p, &c);
14
15     /* Racuna se deo broja koji se nalazi desno od pozicije p.
16        Funkcija pow kao povratnu vrednost vraca realan broj dvostruke
17        tacnosti, a operacija % ocekuje celobrojne operande. Iz tog
18        razloga je neophodno izvršiti pretvaranje povratne vrednosti
19        u tip unsigned int. */
20     desno = x % (unsigned int) pow(10, p);
21
22     /* Racuna se deo broja koji se nalazi levo od pozicije p. */
23     levo = x / (unsigned int) pow(10, p);
24
25     /* Rezultat se racuna nadovezivanjem levog dela, cifre c
26        i desnog dela. */
27     novo_x = levo * (unsigned int) pow(10, p + 1) +
28             c * (unsigned int) pow(10, p) + desno;
29
30     /* Ispis rezultata. */
31     printf("Rezultat je: %u\n", novo_x);
32
33     return 0;
34 }
35
```

### Rešenje 1.1.29

```
1 #include <stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     int a, b;
7     int rezultata, rezultatb, rezultatc;
8
9     /* Ucitavaju se dva cela broja. */
10    printf("Unesite dva cela broja: ");
11    scanf("%d%d", &a, &b);

```



```

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

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

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

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

33  return 0;
}

```

### Rešenje 1.1.30

```

1  #include <stdio.h>

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

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

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

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

17  return 0;

19 }

```

### Rešenje 1.1.31

Zadatak se rešava analogno zadatku 1.1.30

### Rešenje 1.1.32

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      float a, b, rez;
7      float min, max;
8
9      /* Ucitavaju se dva realna broja. */
10     printf("Unesite dva realna broja: ");
11     scanf("%f%f", &a, &b);
12
13     /* Racunaju se minimalna i maksimalna vrednost unetih brojeva. */
14     min = (a < b) ? a : b;
15     max = (a > b) ? a : b;
16
17     /* Racuna se vrednost rezultata. */
18     rez = (min + 0.5) / (1 + max * max);
19
20     /* Ispis rezultata. */
21     printf("Rezultat je %.2f\n", rez);
22
23     return 0;
24 }
```

## 2

# Kontrola toka

## 2.1 Naredbe grananja

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite tri cela broja: 5 18 -1  
| Najmanji: -1
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite tri cela broja: 0 43 16  
| Najmanji: 0
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite tri cela broja: 3 3 3  
| Najmanji: 3
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite tri cela broja: -5 -5 -5  
| Najmanji: -5
```

[Rešenje 2.1.1]

**Zadatak 2.1.2** Napisati program koji za uneti realan broj ispisuje njegovu apsolutnu vrednost zaokruženu na dve decimale.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite jedan realan broj: 7.42  
| Apsolutna vrednost: 7.42
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite jedan realan broj: -562.428  
| Apsolutna vrednost: 562.43
```

## 2 Kontrola toka

---

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan broj: 0  
|| Apsolutna vrednost: 0.00
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan broj: 52  
|| Apsolutna vrednost: 52.00
```

[Rešenje 2.1.2]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan ceo broj: 22  
|| Recipročna vrednost: 0.0455
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan ceo broj: -9  
|| Recipročna vrednost: -0.1111
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan ceo broj: 0  
|| Greška: nedozvoljeno je deljenje nulom.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan ceo broj: 57298  
|| Recipročna vrednost: 0.0000
```

[Rešenje 2.1.3]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cela broja: 1 3 -6  
|| Zbir pozitivnih: 4
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cela broja: -15 81 0  
|| Zbir pozitivnih: 81
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cela broja: -719 -48 -123  
|| Zbir pozitivnih: 0
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cela broja: 16 2 576  
|| Zbir pozitivnih: 594
```

[Rešenje 2.1.4]

**Zadatak 2.1.5** U prodavnici je organizovana akcija da svaki kupac dobije najjeftiniji od tri artikla za jedan dinar. Napisati program koji za unete cene

tri artikla izračunava ukupnu cenu, kao i koliko dinara se uštedi zahvaljujući popustu. Cene artikala su pozitivni celi brojevi. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tri cene: 247 -133 126  
Greska: neispravan unos cene.
```

[Rešenje 2.1.5]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vreme: 18:19  
Do ponoci: 5 sati i 41 minuta
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vreme: 23:7  
Do ponoci: 0 sati i 53 minuta
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vreme: 24:20  
Greska: neispravan unos vremena.
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vreme: 14:0  
Do ponoci: 10 sati i 0 minuta
```

[Rešenje 2.1.6]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite godinu: 2016  
Godina je prestupna.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite godinu: 1997  
Godina nije prestupna.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite godinu: 2000  
|| Godina je prestupna.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite godinu: 1900  
|| Godina nije prestupna.
```

[Rešenje 2.1.7]

**Zadatak 2.1.8** Napisati program koji za učitani karakter ispisuje uneti karakter i njegov ASCII kod. Ukoliko je uneti karakter malo (veliko) slovo, ispisati i odgovarajuće veliko (malo) slovo i njegov ASCII kod.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: 0  
|| Uneti karakter: 0  
|| ASCII kod: 48
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: ?  
|| Uneti karakter: ?  
|| ASCII kod: 63
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: A  
|| Uneti karakter: v  
|| ASCII kod: 65  
|| Odgovarajuće malo slovo: a  
|| ASCII kod: 97
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: v  
|| Uneti karakter: v  
|| ASCII kod: 118  
|| Odgovarajuće veliko slovo: V  
|| ASCII kod: 86
```

[Rešenje 2.1.8]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: A 5 3  
|| Proizvod cifara: 15
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: k ! m  
|| Medju unetim karakterima nema cifara.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: 9 9 9  
|| Proizvod cifara: 729
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: a 8 0  
|| Proizvod cifara: 0
```

[Rešenje 2.1.9]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite sifru: aBc  
| abc
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite sifru: a?!  
| Greška: ? je neispravan karakter.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite karaktere: 5A5  
| 5a5
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite karaktere: 123  
| 123
```

[Rešenje 2.1.10]

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

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite četvorocifreni broj: 7777  
| Najveća cifra je: 7
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite četvorocifreni broj: 238  
| Greska: niste uneli četvorocifreni broj.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite četvorocifreni broj: -2002  
| Najveća cifra je: 2
```

[Rešenje 2.1.11]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite pozitivan trocifreni broj:  
|| 84  
|| Greska: niste uneli pozitivan trocifreni broj.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite pozitivan trocifreni broj:  
|| 371  
|| Broj je Armstrongov.
```

[Rešenje 2.1.12]

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

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite četvorocifreni broj: 3579  
|| Nema parnih cifara.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite četvorocifreni broj: 288  
|| Greska: niste uneli četvorocifreni broj.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite četvorocifreni broj: -1234  
|| Proizvod parnih cifara: 8
```

[Rešenje 2.1.13]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite četvorocifreni broj: 2863  
|| Rezultat: 8263
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite četvorocifreni broj: 1192  
|| Rezultat: 1912
```



### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: 247
|| Greska: niste uneli cetvorocifreni broj.
```

### Primer 4

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

[Rešenje 2.1.14]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: 1389
|| Cifre su uredjene neopadajuće.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: -9622
|| Cifre su uredjene nerastuće.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: 88
|| Greska: niste uneli cetvorocifreni broj.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: 6792
|| Cifre nisu uredjene.
```

[Rešenje 2.1.15]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koordinate tacke A: 1.5 6
|| Unesite koordinate tacke B: 2.33 9.8
|| Tacke se nalaze u istom kvadrantu.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koordinate tacke A: -3 6
|| Unesite koordinate tacke B: 0.33 -5
|| Tacke se ne nalaze u istom kvadrantu.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koordinate tacke A: 0 -6
|| Unesite koordinate tacke B: -1 -99.66
|| Tacke se nalaze u istom kvadrantu.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koordinate tacke A: 3 -6
|| Unesite koordinate tacke B: -0.33 0
|| Tacke se ne nalaze u istom kvadrantu.
```

[Rešenje 2.1.16]

**Zadatak 2.1.17** Napisati program koji ispituje da li se tačke  $A(x_1, y_1)$ ,  $B(x_2, y_2)$  i  $C(x_3, y_3)$  nalaze na istoj pravoj.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 1.5 6
Unesite koordinate tacke B: -2.5 -10
Unesite koordinate tacke C: 3 12
Tacke se nalaze na istoj pravoj.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: -1.5 3
Unesite koordinate tacke B: -0.4 9.8
Unesite koordinate tacke C: 2 3
Tacke se ne nalaze na istoj pravoj.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 1.55 6
Unesite koordinate tacke B: -8.4 9.8
Unesite koordinate tacke C: 5 4.682412
Tacke se nalaze na istoj pravoj.
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 5.5 3.5
Unesite koordinate tacke B: 5.5 3.5
Unesite koordinate tacke C: 5.5 3.5
Tacke se nalaze na istoj pravoj.
```

*Primer 5*

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 1 2
Unesite koordinate tacke B: 1 2
Unesite koordinate tacke C: -56 1.3
Tacke se nalaze na istoj pravoj.
```

*Primer 6*

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 3.4 3.5
Unesite koordinate tacke B: -10 -1
Unesite koordinate tacke C: -10 -1
Tacke se nalaze na istoj pravoj.
```

[Rešenje 2.1.17]

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

- dužinu preseka datih intervala
- presečni interval datih intervala
- dužinu prave koju pokrivaju dati intervali
- najmanji interval koji sadrži date intervale.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite a1, b1, a2 i b2: 2 9 4 11
Duzina preseka: 5
Presecni interval: [4,9]
Duzina koju pokrivaju: 9
Najmanji interval: [2, 11]
```

*Primer 2*

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

[Rešenje 2.1.18]

**Zadatak 2.1.19** Napisati program koji za unete koeficijente kvadratne jednačine ispisuje koliko realnih rešenja jednačina ima i ako ih ima, ispisuje ih zaokružene na dve decimale.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koeficijente A, B i C: 1 3 2
|| Jednacina ima dva razlicita realna resenja:
|| -1.00 i -2.00
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koeficijente A, B i C: 1 1 1
|| Jednacina nema resenja.
```

[Rešenje 2.1.19]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite k: 200
|| Greska: neispravan unos pozicije.
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite k: 10
|| Na 10-toj poziciji je broj 1.
```

[Rešenje 2.1.20]

**Zadatak 2.1.21** Data je funkcija  $f(x) = 2 \cdot \cos(x) - x^3$ . Napisati program koji za učitane vrednost realne promenljive  $x$  i vrednost celobrojne promenljive  $k$  koje može biti 1, 2 ili 3 izračunava vrednost funkcije  $F(x, k)$  koja se dobija tako što se funkcija  $f$  primeni  $k$ -puta ( $F(x, 1) = f(x)$ ,  $F(x, 2) = f(f(x))$ ,  $F(x, 3) = f(f(f(x)))$ ) i ispisuje je zaokruženu na dve decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom x i k: 2.31 2
|| F(2.31, 2)=2557.52
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom x i k: 12 1
|| F(12, 1)=-1726.31
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom x i k: 2.31 0  
|| Greska: nedozvoljena vrednost za k.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom x i k: 1 3  
|| F(1, 3)=-8.74
```

[Rešenje [2.1.21](#)]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 8  
|| Greska: neispravan unos dana.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2  
|| U pitanju je: utorak
```

[Rešenje [2.1.22](#)]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan karakter: A  
|| Uneti karakter je samoglasnik.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan karakter: i  
|| Uneti karakter je samoglasnik.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan karakter: f  
|| Uneti karakter nije samoglasnik.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan karakter: 4  
|| Uneti karakter nije samoglasnik.
```

[Rešenje [2.1.23](#)]

**Zadatak 2.1.24** Napisati program koji učitava dva cela broja i jedan od karaktera +, -, \*, / ili % i ispisuje vrednost izraza dobijenog primenom date

operacije na date argumente. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite izraz: 8 - 11  
| Rezultat je: -3
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite izraz: 14 / 0  
| Greska: deljenje nulom.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite izraz: 5 ? 7  
| Greska: nepoznat operator.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite izraz: 19 / 5  
| Rezultat je: 3
```

[Rešenje 2.1.24]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dan i mesec: 14.10.  
| jesen
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dan i mesec: 2.8.  
| leto
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dan i mesec: 27.2.  
| zima
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dan i mesec: 19.5.  
| prolece
```

[Rešenje 2.1.25]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite godinu: 2018  
| Unesite mesec: 1  
| Januar, 31 dan
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite godinu: 2000  
| Unesite mesec: 2  
| Februar, 29 dana
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite godinu: 2018  
|| Unesite mesec: 13  
|| Greska: neispravan unos meseca.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite godinu: 1998  
|| Unesite mesec: 2  
|| Februar, 28 dana
```

[Rešenje 2.1.26]

**Zadatak 2.1.27** Napisati program koji za uneti datum u formatu *dan.me-sec.godina.* proverava da li je korektan.

### Primer 1

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

### Primer 2

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

[Rešenje 2.1.27]

**Zadatak 2.1.28** Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum prethodnog dana.

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 1.1.2019.  
|| Prethodni datum: 31.12.2018.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 10.12.2015.  
|| Prethodni datum: 9.11.2015.
```

[Rešenje 2.1.28]

**Zadatak 2.1.29** Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum narednog dana.

### Primer 1

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

### Primer 2

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

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite datum: 31.12.2008.
Naredni datum: 1.1.2009.

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite datum: 5.5.2005.
Naredni datum: 6.5.2005.

```

[Rešenje 2.1.29]

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

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

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

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite (k,l): 1 1
Unesite (m,n): 2 2
Polja su iste boje.
Kraljica sa (1,1) ugrozava (2,2).
Konj sa (1,1) ne ugrozava (2,2).

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite (k,l): 1 1
Unesite (m,n): 3 2
Polja su razlicite boje.
Kraljica sa (1,1) ne ugrozava (3,2).
Konj sa (1,1) ugrozava (3,2).

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite (k,l): 5 4
Unesite (m,n): 3 3
Polja su razlicite boje.
Kraljica sa (5,4) ne ugrozava (3,3).
Konj sa (5,4) ugrozava (3,3).

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite (k,l): 0 1
Unesite (m,n): 3 9
Greska: neispravna pozicija.

```

[Rešenje 2.1.30]

## 2.2 Rešenja

### Rešenje 2.1.1

```
1 #include <stdio.h>
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     int a, b, c, najmanji;
7
8     /* Ucitavaju se ulazne vrednosti. */
9     printf("Unesite tri cela broja: ");
10    scanf("%d%d%d", &a, &b, &c);
11
12    /* Najmanji broj se inicijalizuje na vrednost prvog broja. */
13    najmanji = a;
14
15    /* Ako je vrednost drugog broja manji od vrednosti tekuceg
16       minimuma, vrednost minimuma se azurira. */
17    if (b < najmanji)
18        najmanji = b;
19
20    /* Postupak se ponavlja za treci broj. */
21    if (c < najmanji)
22        najmanji = c;
23
24    /* Ispis rezultata. */
25    printf("Najmanji: %d\n", najmanji);
26
27    return 0;
28 }
```

### Rešenje 2.1.2

```
1 #include<stdio.h>
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     float x, apsolutno_x;
7
8     /* Ucitava se vrednost broja. */
9     printf("Unesite jedan realan broj:");
10    scanf("%f", &x);
11
12    /* Racuna se apsolutna vrednost unetog broja. */
13    apsolutno_x = x;
14    if (x < 0)
15        apsolutno_x = -x;
16
17    /* Ispis rezultata. */
18    printf("Apsolutna vrednost: %.2f\n", apsolutno_x);
19 }
```



```

21  /* II nacin: koriscenjem funkcije fabs cija se deklaracija nalazi
    u zaglavlju math.h: apsolutno_x=fabs(x); */
23  return 0;
}

```

### Rešenje 2.1.3

```

1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int x;
7      float recipročno_x;
8
9      /* Ucitavanje vrednosti broja x. */
10     printf("Unesite jedan ceo broj:");
11     scanf("%d", &x);
12
13     /* Vrsi se provera ispravnosti ulaznih podataka. Napomena: za
14     razliku od izlaza iz programa sa kodom 0 (return 0;) koji
15     služi kao indikator da se program završio uspešno, izlaz iz
16     programa sa izlaznim kodom koji se razlikuje od nule služi
17     kao indikator da je pri izvršavanju programa došlo do neke
18     greske. */
19     if (x == 0) {
20         printf("Greska: nedozvoljeno je deljenje nulom.\n");
21         return -1;
22     }
23
24     /* Racuna se recipročna vrednost. */
25     recipročno_x = 1.0 / x;
26
27     /* Ispis rezultata. */
28     printf("Recipročna vrednost: %.4f\n", recipročno_x);
29
30     return 0;
31 }

```

### Rešenje 2.1.4

```

1  #include<stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int a, b, c, suma;
7

```

## 2 Kontrola toka

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

11

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

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

21
23 if (b > 0)
    suma += b;

25
27 if (c > 0)
    suma += c;

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

31 return 0;
}
```

### Rešenje 2.1.5

```
1 #include <stdio.h>

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

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

13

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

19

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

23    if (b < najjeftiniji)
```

```

    najjeftiniji = b;
25
    if (c < najjeftiniji)
27        najjeftiniji = c;

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

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

37    return 0;
}

```

### Rešenje 2.1.6

```

1  #include<stdio.h>

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

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

15    /* Vrsi se provera ispravnosti ulaznih podataka. */
    if (sati > 24 || sati < 0 || minuti > 59 || minuti < 0) {
17        printf("Greska: neispravan unos vremena.\n");
        return -1;
19    }

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

25    if (preostali_minuti == 60) {
        /* Uvecavanje vrednosti broja za 1 se moze uraditi na vise
27        nacina. Neki od njih su:
            broj = broj + 1;
29            broj += 1;
            broj++; */
31        preostali_sati++;
        preostali_minuti = 0;
33    }
}

```

## 2 Kontrola toka

---

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

39  return 0;
}
```

### Rešenje 2.1.7

```
1  #include <stdio.h>

3  int main()
{
5     /* Deklaracija potrebne promenljive. */
    unsigned int x;

7     /* Ucitava se vrednost godine. */
9     printf("Unesite godinu:");
    scanf("%u", &x);

11

13     /* Proverava se da li je godina prestupna ili ne i ispisuje se
        odgovarajuca poruka. Godina je prestupna ukoliko vazi jedan od
        narednih uslova:
        1. da je deljiva sa 4, a nije sa 100
        2. da je deljiva sa 400. */
15     if ((x % 4 == 0 && x % 100 != 0) || x % 400 == 0)
17         printf("Godina je prestupna.\n");
    else
19         printf("Godina nije prestupna.\n");

21
23     return 0;
}
```

### Rešenje 2.1.8

```
1  #include <stdio.h>

2
3  int main()
4  {
5     /* Deklaracija karakterske promenljive. */
6     char c;

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

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

```

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

16 /* Karakteri koji odgovaraju velikim slovima su u ASCII tablici
   smesteni sekvencijalno. Na primer, ASCII kod karaktera 'A' je
18 65, 'B' je 66, ..., 'Z' je 90. Isto vazi i za mala slova: 'a'
   je 97, 'b' je 98, ..., 'z' je 122.

20
   Oдавде, ako se vrsi provera da li je neki karakter veliko
22 slovo, dovoljno je proveriti da li se njegov ASCII kod nalazi
   izmedju ASCII kodova slova 'A' i slova 'Z'.

24
   Dodatno, moze se primetiti da je razlika izmedju ASCII koda
26 svakog malog i odgovarajuceg velikog slova konstanta koja ima
   vrednost 'a'-'A', sto je isto sto i 'b'-'B', itd. Zbog toga,
28 ako je potrebno od velikog slova dobiti malo, onda je
   dovoljno ASCII kodu velikog slova dodati pomenutu konstantu.
30 Za mala slova, vazi obrnuto - da bi se dobilo veliko slovo,
   ova konstanta se oduzima. */

32
   if (c >= 'A' && c <= 'Z') {
34     printf("Odgovarajuce malo slovo: %c\n", c + ('a' - 'A'));
     printf("ASCII kod: %d\n", c + ('a' - 'A'));
36   }

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

42   return 0;
44 }

```

### Rešenje 2.1.9

```

1  #include <stdio.h>

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

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

```

```
19  c1 = getchar();
    getchar();
    c2 = getchar();
21  getchar();
    c3 = getchar();
23
    /* II nacin ucitavanja ulaza: koriscenjem funkcije scanf()
25     Blanko znaci se navode kao deo ocekivanog formata ulaza.
        char c1, c2, c3;
27     scanf("%c %c %c", &c1, &c2, &c3); */

29  /* Pogresan nacin ucitavanja ulaza:
        scanf("%c%c%c", &c1, &c2, &c3);
31     U ovom slucaju ce u c1 biti upisan prvi karakter, u c2
        blanko i u c3 drugi karakter. */
33
35  /* Karakteri koji predstavljaju cifre su u ASCII tablici takodje
        smesteni sekvencijalno. Na primer, '0' ima ASCII kod 48, '1'
        49, ..., '9' ima ASCII kod 57.

37
39     Oдавде, ako se proverava da li je karakter cifra, dovoljno je
        proveriti da li se njegov ASCII kod nalazi izmedju '0' i '9'.

41
43     Dodatno, ako je potrebno izracunati dekadnu vrednost karaktera
        koji je cifra, dovoljno je od ASCII koda tog karaktera,
        oduzeti ASCII kod karaktera '0'. Na primer, '4'-'0' = 52 - 48
        = 4. */
45
47  /* Racuna se proizvod onih karaktera koji su cifre. */
    if (c1 >= '0' && c1 <= '9') {
        proizvod_cifara *= (c1 - '0');
49     broj_cifara++;
    }

51
53     if (c2 >= '0' && c2 <= '9') {
        proizvod_cifara *= (c2 - '0');
        broj_cifara++;
55     }

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

61
63  /* Ispis rezultata. */
    if (broj_cifara == 0)
        printf("Medju unetim karakterima nema cifara.\n");
65     else
        printf("Proizvod cifara: %u\n", proizvod_cifara);
67
69     return 0;
}
```

## Rešenje 2.1.10

```
1  #include <stdio.h>
2  #include <ctype.h>
3
4  int main()
5  {
6      /* Deklaracija potrebnih promenljivih. */
7      int c1, c2, c3;
8
9      /* Ucitava se sifra artikla. */
10     printf("Unesite sifru: ");
11     c1 = getchar();
12     c2 = getchar();
13     c3 = getchar();
14
15     /* Funkcije islower, isupper i isdigit proveravaju da li je
16        prosledjeni karakter malo slovo, veliko slovo ili cifra.
17        Deklaracije ovih funkcija se nalaze u zaglavlju ctype.h.
18
19        Ukoliko prvi karakter nije ni malo slovo ni veliko slovo, ni
20        cifra, ispisuje se odgovarajuca poruka o gresci i izlazi se
21        iz programa. */
22     if (!islower(c1) && !isupper(c1) && !isdigit(c1)) {
23         printf("Greska: %c je neispravan karakter.\n", c1);
24         return -1;
25     }
26
27     /* Postupak se ponavlja za druga dva karaktera. */
28     if (!islower(c2) && !isupper(c2) && !isdigit(c2)) {
29         printf("Greska: %c je neispravan karakter.\n", c2);
30         return -1;
31     }
32
33     if (!islower(c3) && !isupper(c3) && !isdigit(c3)) {
34         printf("Greska: %c je neispravan karakter.\n", c3);
35         return -1;
36     }
37
38     /* Funkcija tolower(c) radi sledece: ako je c veliko slovo, kao
39        povratnu vrednost vraca odgovarajuce malo slovo, u suprotnom
40        vraca c. Dakle, tolower('A') je 'a', a tolower('6') = '6',...
41
42        Slicno, samo obrnuto, radi i funkcija toupper(c). Deklaracije
43        ovih funkcija se takodje nalaze u zaglavlju ctype.h. */
44     c1 = tolower(c1);
45     c2 = tolower(c2);
46     c3 = tolower(c3);
47
48     printf("%c%c%c\n", c1, c2, c3);
49
50     return 0;
```

51 | }

### Rešenje 2.1.11

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     /* Deklaracija potrebnih promenljivih. */
7     int n;
8     char jedinica, desetica, stotina, hiljada, najveca_cifra;
9
10    /* Ucitava se vrednost broja n. */
11    printf("Unesite cetvorocifreni broj: ");
12    scanf("%d", &n);
13
14    /* Da bi program radio ispravno i za negativne brojeve, uzima se
15     apsolutna vrednost broja n. */
16    n = abs(n);
17
18    /* Vrsi se provera ispravnosti ulaznih podataka. */
19    if (n < 1000 || n > 9999) {
20        printf("Greska: niste uneli cetvorocifreni broj.\n");
21        return -1;
22    }
23
24    /* Izdvajaju se cifre broja n. */
25    jedinica = n % 10;
26    desetica = (n / 10) % 10;
27    stotina = (n / 100) % 10;
28    hiljada = n / 1000;
29
30    /* Racuna se najveca cifra broja n. */
31    najveca_cifra = jedinica;
32
33    if (desetica > najveca_cifra)
34        najveca_cifra = desetica;
35
36    if (stotina > najveca_cifra)
37        najveca_cifra = stotina;
38
39    if (hiljada > najveca_cifra)
40        najveca_cifra = hiljada;
41
42    /* Ispis rezultata */
43    printf("Najveca cifra je: %d\n", najveca_cifra);
44
45    return 0;
46 }
```



## Rešenje 2.1.12

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     /* Deklaracija potrebnih promenljivih. */
7     int n;
8     char jedinica, desetica, stotina;
9
10    /* Ucitava se vrednost broja n. */
11    printf("Unesite pozitivan trocifreni broj: ");
12    scanf("%d", &n);
13
14    /* Vrsi se provera ispravnosti ulaznih podataka. */
15    if (n < 100 || n > 999) {
16        printf("Greska: niste uneli pozitivan trocifreni broj.\n");
17        return -1;
18    }
19
20    /* Izdvajaju se cifre broja n. */
21    jedinica = n % 10;
22    desetica = (n / 10) % 10;
23    stotina = n / 100;
24
25    /* Ispis rezultata. */
26    if (n == jedinica * jedinica * jedinica +
27        desetica * desetica * desetica + stotina * stotina * stotina)
28        printf("Broj je Armstrongov.\n");
29    else
30        printf("Broj nije Armstrongov.\n");
31
32    return 0;
33 }
```

## Rešenje 2.1.13

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     /* Deklaracija potrebnih promenljivih. */
7     int n;
8     char jedinica, desetica, stotina, hiljada;
9     int broj_parnih, proizvod_parnih;
10
11    /* Ucitava se vrednost broja n. */
12    printf("Unesite cetvorocifreni broj: ");
```

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

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

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

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

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

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

41    if (desetica % 2 == 0) {
43        proizvod_parnih = proizvod_parnih * desetica;
        broj_parnih++;
45    }

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

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

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

63    return 0;
```

65 }

## Rešenje 2.1.14

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   int main()
5  {
   /* Deklaracija potrebnih promenljivih. */
7  int n, n_abs;
   char jedinica, desetica, stotina, hiljada;
9  int najveca, najmanja, stepen_najvece, stepen_najmanje;
   int rezultat;
11
   /* Ucitava se broj vrednost broja n. */
13  printf("Unesite cetvorocifreni broj: ");
   scanf("%d", &n);
15
   /* Da bi program radio ispravno i za negativne vrednosti, uzima
      se apsolutna vrednost broja n. */
17  n_abs = abs(n);
19
   /* Vrsi se provera ispravnosti ulaznih podataka. */
21  if (n_abs < 1000 || n_abs > 9999) {
       printf("Greska: niste uneli cetvorocifreni broj.\n");
23     return -1;
   }
25
   /* Izdvajaju se cifre broja n. */
27  jedinica = n_abs % 10;
   desetica = (n_abs / 10) % 10;
29  stotina = (n_abs / 100) % 10;
   hiljada = n_abs / 1000;
31
   /* Po algoritmu za trazenje najvece/najmanje cifre (koji je
      prikazan u zadatku 2.1.11) racunaju se najveca i najmanja
      cifra broja n, kao i pozicija na kojoj se one nalaze.
      Radi lakseg izracunavanja, pozicija se pamti kao stepen broja
      10. Na primer, pozicija cifre jedinica je 1, cifre desetica
      10, itd... */
33
   najveca = jedinica;
35  stepen_najvece = 1;
37
   if (desetica > najveca) {
       najveca = desetica;
       stepen_najvece = 10;
41   }
43
   if (stotina > najveca) {
       najveca = stotina;
45   }
47

```

```

    stepen_najvece = 100;
49 }

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

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

59 if (desetica < najmanja) {
61     najmanja = desetica;
    stepen_najmanje = 10;
63 }

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

69 if (hiljada < najmanja) {
71     najmanja = hiljada;
    stepen_najmanje = 1000;
73 }

75 /* Ideja: U broju 4179, najmanja cifra je 1 i njen stepen je 100,
    a najveca cifra je 9 i njen stepen je 1. Zamenja mesta se vrši
77     tako što se oduzme 9 i doda 1, a zatim oduzme 100 i doda 900. */
    rezultat = n_abs - najveca * stepen_najvece
79                 + najmanja * stepen_najvece
                - najmanja * stepen_najmanje
81                 + najveca * stepen_najmanje;

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

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

91 return 0;
}
```

### Rešenje 2.1.15

```

1 #include <stdio.h>
  #include <stdlib.h>
3
```

```

5 int main()
6 {
7     /* Deklaracija potrebnih promenljivih. */
8     int n;
9     char jedinica, desetica, stotina, hiljada;
10
11     /* Ucitava se vrednost broja n. */
12     printf("Unesite cetvorocifreni broj: ");
13     scanf("%d", &n);
14
15     /* Da bi program radio ispravno i za negativne vrednosti, uzima
16        se apsolutna vrednost broja n. */
17     n = abs(n);
18
19     /* Vrsi se provera ispravnosti ulaznih podataka. */
20     if (n < 1000 || n > 9999) {
21         printf("Greska: niste uneli cetvorocifreni broj.\n");
22         return -1;
23     }
24
25     /* Izdvajaju se cifre broja n. */
26     jedinica = n % 10;
27     desetica = (n / 10) % 10;
28     stotina = (n / 100) % 10;
29     hiljada = n / 1000;
30
31     /* Ispis rezultata. */
32     if (hiljada <= stotina && stotina <= desetica
33         && desetica <= jedinica)
34         printf("Cifre su uredjene neopadajuce. \n");
35     else if (hiljada >= stotina && stotina >= desetica
36         && desetica >= jedinica)
37         printf("Cifre su uredjene nerastuce. \n");
38     else
39         printf("Cifre nisu uredjene.\n");
40
41     return 0;
42 }

```

### Rešenje 2.1.16

```

1 #include<stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     float xa, ya, xb, yb;
7
8     /* Ucitavaju se koordinate tacaka A i B. */
9     printf("Unesite koordinate tacke A: ");
10    scanf("%f%f", &xa, &ya);

```

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

14

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

24   return 0;
26 }
```

### Rešenje 2.1.17

```
1  #include<stdio.h>

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

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

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

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

19    /* Ako su bilo koje dve tacke jednake, onda se sigurno sve tri
       nalaze na jednoj pravoj. */
21    if ((xa == xb && ya == yb) ||
        (xa == xc && ya == yc) || (xb == xc && yb == yc)) {
23        printf("Tacke se nalaze na istoj pravoj.\n");
        return 0;
25    }

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

```

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

51     /* II nacin: Tacke su kolinearne ako je:
52        |xa ya 1 |
53        |xb yb 1 | = 0
54        |xc yc 1 |
55        odnosno, ako je:
56        xa*yb + ya*xc + xb*yc - ya*xb - xa*yc - yb*xc = 0

57        if(xa*yb + ya*xc + xb*yc - ya*xb - xa*yc - yb*xc == 0)
58            printf("Tacke se nalaze na istoj pravoj. \n");
59        else
60            printf("Tacke se ne nalaze na istoj pravoj. \n"); */

61     return 0;
62 }

```

### Rešenje 2.1.18

```

#include<stdio.h>

2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     int a1, a2, b1, b2;

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

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

```

```

18     printf("Presecni interval: [%d, %d]\n", a2, b1);
    printf("Duzina koju pokrivaju: %d\n", b2 - a1);
    printf("Najmanji interval: [%d, %d]\n", a1, b2);
20 } else if (a2 <= a1 && b2 >= a1) {
    /* II slucaj: intervali se seku i [a2,b2] je pre [a1,b1]. */
22     printf("Duzina preseka:: %d\n", b2 - a1);
    printf("Presecni interval: [%d, %d]\n", a1, b2);
24     printf("Duzina koju pokrivaju: %d\n", b1 - a2);
    printf("Najmanji interval: [%d, %d]\n", a2, b1);
26 } else if (a1 >= a2 && b1 <= b2) {
    /* III slucaj: interval [a1,b1] se nalazi unutar [a2,b2]. */
28     printf("Duzina preseka:: %d\n", b1 - a1);
    printf("Presecni interval: [%d, %d]\n", a1, b1);
30     printf("Duzina koju pokrivaju: %d\n", b2 - a2);
    printf("Najmanji interval: [%d, %d]\n", a2, b2);
32 } else if (a2 >= a1 && b2 <= b1) {
    /* IV slucaj: interval [a2,b2] se nalazi unutar [a1,b1]. */
34     printf("Duzina preseka:: %d\n", b2 - a2);
    printf("Presecni interval: [%d, %d]\n", a2, b2);
36     printf("Duzina koju pokrivaju: %d\n", b1 - a1);
    printf("Najmanji interval: [%d, %d]\n", a1, b1);
38 } else {
    /* V slucaj: intervali su disjunktni. */
40     printf("Duzina preseka:: 0\n");
    printf("Presecni interval: prazan\n");
42     printf("Duzina koju pokrivaju: %d\n", b1 - a1 + b2 - a2);
    if (a1 < a2)
44         printf("Najmanji interval: [%d, %d]\n", a1, b2);
    else
46         printf("Najmanji interval: [%d, %d]\n", a2, b1);
    }
48
50     return 0;
}

```

## Rešenje 2.1.19

```

1 #include <stdio.h>
#include <math.h>
3
4 int main()
5 {
6     /* Deklaracija potrebnih promenljivih. */
7     float a, b, c;
8     float D;
9
10    /* Ucitavaju se koeficijenti kvadratne jednacine. */
11    printf("Unesite koeficijente A, B i C:");
12    scanf("%f%f%f", &a, &b, &c);
13
14    /* Racunaju se resenja jednacine u zavisnosti od vrednosti

```



```

15     koeficijenta a, b i c i ispisuje se odgovarajući rezultat. */
16     if (a == 0) {
17         if (b == 0) {
18             if (c == 0) {
19                 /* Slučaj a==0 && b==0 && c==0: beskonacno mnogo resenja. */
20                 printf("Jednacina ima beskonacno mnogo resenja\n");
21             } else {
22                 /* Slučaj a==0 && b==0 && c!=0: nema resenja. */
23                 printf("Jednacina nema resenja\n");
24             }
25         } else {
26             /* Slučaj a=0 && b!=0: jedinstveno resenje. */
27             printf("Jednacina ima jedinstveno realno resenje %.2f\n",
28                 -c / b);
29         }
30     } else {
31         /* Slučaj a != 0: racuna se diskriminanta. */
32         D = b * b - 4 * a * c;
33
34         /* U zavisnosti od vrednosti diskriminante, ispisuje se
35            rezultat. */
36         if (D < 0) {
37             printf("Jednacina nema realnih resenja\n");
38         } else if (D > 0) {
39             printf("Jednacina ima dva realna resenja %.2f i %.2f\n",
40                 (-b + sqrt(D)) / (2 * a), (-b - sqrt(D)) / (2 * a));
41         } else {
42             printf("Jednacina ima jedinstveno realno resenje %.2f\n",
43                 -b / (2 * a));
44         }
45     }
46
47     return 0;
48 }

```

### Rešenje 2.1.20

```

1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int k, broj;
7
8      /* Ucitava se trazena pozicija. */
9      printf("Unesite k: ");
10     scanf("%d", &k);
11
12     /* Vrsi se provera ispravnosti ulaznih podataka. */
13     if (k < 1 || k > 189) {
14         printf("Greska: neispravan unos pozicije.\n");
15     }
16 }

```

## 2 Kontrola toka

```
15     return -1;
16 }
17
18 /* Racuna se rezultat. */
19 if (k < 10) {
20     /* I slucaj: trazi se jednocifreni broj. */
21     printf("Na %d-toj poziciji je broj %d.\n", k, k);
22 } else {
23     /* II slucaj: trazi se dvocifreni broj. */
24
25     /* Ideja: izracunati broj na koji pokazuje pozicija k. Zatim,
26        ako je k parno, uzeti cifru desetica tog broja, a ako je k
27        neparno, uzeti cifru jedinica tog broja.
28
29        Na primer, za k=14 i k=15, broj koji se nalazi na ovim
30        pozicijama je 12, pa u slucaju da je k=14, treba ispisati 1,
31        a u slucaju da je k=15, treba ispisati 2. */
32
33     /* Odredjivanje odgovarajuceg broja: Kada bi niz izgledao
34        10111213...9899, za dato k, broj bi se dobio kao  $9 + k/2 + 1$ 
35        za neparne vrednosti k, odnosno  $9 + k/2$  za parne (dodaje se
36        vrednost detet jer je prvi broj u nizu desетка.) Na primer:
37        k=1, broj =  $9 + 1/2 + 1 = 9 + 0 + 1 = 10$  k=2, broj =  $9 + 2/2$ 
38        = 10 k=3, broj =  $9 + 3/2 + 1 = 9 + 1 + 1 = 11$  k=4, broj =  $9$ 
39        +  $4/2 = 11$  ... Posto ovde postoji i 9 pozicija ispred,
40        potrebno je i njih uzeti u obzir - odatle: broj =  $9 +$ 
41         $(k-9)/2 + 1$  za neparne vrednosti k, odnosno broj =  $9 +$ 
42         $(k-9)/2$  za parne vrednosti k. */
43     if (k % 2 != 0) {
44         broj = 9 + (k - 9) / 2;
45         printf("Na %d-toj poziciji je broj %d.\n", k, broj % 10);
46     } else {
47         broj = 9 + (k - 9) / 2 + 1;
48         printf("Na %d-toj poziciji je broj %d.\n", k, broj / 10);
49     }
50 }
51
52 return 0;
53 }
```

### Rešenje 2.1.21

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     /* Deklaracija potrebnih promenljivih. */
7     float x, Fx;
8     int k;
9 }
```

```

11  /* Ucitavaju se vrednosti x i k. */
    printf("Unesite redom x i k: ");
    scanf("%f %d", &x, &k);
13
15  /* Vrsi se provera ispravnosti ulaznih podataka. */
    if (k < 1 || k > 3) {
        printf("Greska: nedozvoljena vrednost za k.\n");
17         return 0;
    }
19
21  /* U zavisnosti od vrednosti k, data funkcija ce se izracunati
    jednom, dva puta ili tri puta. */
    Fx = 2 * cos(x) - x * x * x;
23    if (k > 1)
        Fx = 2 * cos(Fx) - Fx * Fx * Fx;
25    if (k > 2)
        Fx = 2 * cos(Fx) - Fx * Fx * Fx;
27
29    /* Ispis rezultata. Napomena: ispis realnih brojeva sa %g
        rezultuje ispisom na onaj broj decimala koliko sam broj ima.
        Dakle, broj 1 ce se ispisati kao 1, broj 2.33 kao 2.33, broj
31        0.9999 kao 0.9999. */
    printf("F(%g,%d)=%.2f\n", x, k, Fx);
33
    return 0;
35 }

```

### Rešenje 2.1.22

```

1  #include <stdio.h>
3
4  int main()
5  {
6      /* Deklaracija potrebnih promenljivih. */
7      int dan;
8
9      /* Ucitava se redni broj dana u nedelji. */
10     printf("Unesite broj: ");
11     scanf("%d", &dan);
12
13     /*I nacin: koriscenjem if-else naredbe.
        if(dan == 1)
            printf("ponedeljak\n");
15     else if(dan == 2)
            printf("utorak\n");
17     else if(dan == 3)
            printf("sreda\n");
19     else if(dan == 4)
            printf("cetvrtak\n");
21     else if(dan == 5)
            printf("petak\n");

```

## 2 Kontrola toka

---

```
23     else if(dan == 6)
24         printf("subota\n");
25     else if(dan == 7)
26         printf("nedelja\n");
27     else
28         printf("Greska: neispravan unos dana.\n"); */
29
30     /* II nacin: koriscenjem switch naredbe.*/
31     switch (dan) {
32     case 1:
33         /* Ako dan ima vrednost 1, ispisuje se ponedeljak. */
34         printf("ponedeljak\n");
35
36         /* Ako se naredba break ne navede, izvorsice se i sledeca
37            naredba, tj. ispis ce biti "ponedeljak utorak". */
38         break;
39     case 2:
40         /* Postupak se ponavlja i za ostale dane. */
41         printf("utorak\n");
42         break;
43     case 3:
44         printf("sreda\n");
45         break;
46     case 4:
47         printf("cetvrtak\n");
48         break;
49     case 5:
50         printf("petak\n");
51         break;
52     case 6:
53         printf("subota\n");
54         break;
55     case 7:
56         printf("nedelja\n");
57         break;
58     default:
59         /* Ako vrednost promenljive dan nije ni jedna od vrednosti
60            izmedju 1 i 7, onda je uneta vrednost neispravna. */
61         printf("Greska: neispravan unos dana.\n");
62     }
63
64     return 0;
65 }
```

### Rešenje 2.1.23

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
```

```
7   char c;

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

11  /* Proverava se da li je karakter c samoglasnik, tj. da li
13     odgovara nekom od sledecih karaktera: A,E,I,O,U,a,e,i,o,u. */
   switch (c) {
15     case 'A':
16     case 'E':
17     case 'I':
18     case 'O':
19     case 'U':
20     case 'a':
21     case 'e':
22     case 'i':
23     case 'o':
24     case 'u':
25         printf("Uneti karakter je samoglasnik.\n");
26         break;
27     default:
28         printf("Uneti karakter nije samoglasnik.\n");
29         break;
30   }

31   return 0;
32 }
33 }
```

### Rešenje 2.1.24

```
1  #include <stdio.h>

3  int main()
4  {
5     /* Deklaracija potrebnih promenljivih. */
6     char op;
7     int x, y;

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

13    /* U zavisnosti od unete operacije, racuna se vrednost izraza. */
14    switch (op) {
15        case '+':
16            printf("Rezultat je: %d\n", x + y);
17            break;
18        case '-':
19            printf("Rezultat je: %d\n", x - y);
20            break;
21    }
```

```
21     case '*':
22         printf("Rezultat je: %d\n", x * y);
23         break;
24     case '/':
25         if (y == 0)
26             printf("Greska: deljenje nulom.\n");
27         else
28             printf("Rezultat je: %d\n", x / y);
29         break;
30     case '%':
31         printf("Rezultat je: %d\n", x % y);
32         break;
33     default:
34         printf("Greska: nepoznat operator.\n");
35 }
36
37 return 0;
}
```

### Rešenje 2.1.25

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int dan, mesec;
7
8      /* Ucitava se vrednost datuma koji je zadat u formatu:
9       dan.mesec. */
10     printf("Unesite dan i mesec");
11     scanf("%d.%d.", &dan, &mesec);
12
13     /* Odredjuje se godisnje doba. */
14     switch (mesec) {
15     case 1:
16     case 2:
17         /* Ako je mesec januar ili februar, onda je sigurno u pitanju
18          zima. */
19         printf("zima\n");
20         break;
21     case 3:
22         /* Ako je mesec mart, onda se godisnje doba odredjuje u
23          zavisnosti od dana u mesecu. */
24         if (dan < 21)
25             printf("zima\n");
26         else
27             printf("prolece\n");
28         break;
29     case 4:
30     case 5:
```

```

31     /* Ako je mesec april ili maj, onda je sigurno u pitanju
        prolece. */
33     printf("prolece\n");
        break;
35 case 6:
    /* Ako je mesec jun, onda se godisnje doba odredjuje u
37     zavisnosti od dana u mesecu. */
        if (dan < 21)
39         printf("prolece\n");
        else
41         printf("leto\n");
        break;
43 case 7:
44 case 8:
    /* Ako je mesec jul ili avgust, onda je sigurno u pitanju
45     leto. */
        printf("leto\n");
        break;
47 case 9:
    /* Ako je mesec septembar, onda se godisnje doba odredjuje u
51     zavisnosti od dana u mesecu. */
        if (dan < 23)
53         printf("leto\n");
        else
55         printf("jesen\n");
        break;
57 case 10:
58 case 11:
    /* Ako je mesec oktobar ili novembar, onda je sigurno u pitanju
59     jesen. */
        printf("jesen\n");
        break;
61 case 12:
    /* Ako je mesec decembar, onda se godisnje doba odredjuje u
63     zavisnosti od dana u mesecu. */
        if (dan < 22)
65         printf("jesen\n");
        else
67         printf("zima\n");
69     }
71     return 0;
73 }

```

### Rešenje 2.1.26

```

#include <stdio.h>
2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */

```

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

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

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

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

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

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



```

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

76 return 0;
78 }

```

### Rešenje 2.1.27

```

1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
        int dan, mesec, godina, dozvoljeni_broj_dana;
7
8      /* Ucitava se datum. */
9      printf("Unesite datum: ");
        scanf("%d.%d.%d", &dan, &mesec, &godina);
11
12     /* Vrsi se provera korektnosti vrednosti unete godine. */
13     if (godina < 0) {
        printf("Datum nije korektan.\n");
15         return 0;
    }
17
18     /* Vrsi se provera korektnosti vrednosti unetog meseca. */
19     if (mesec < 1 || mesec > 12) {
        printf("Datum nije korektan.\n");
21         return 0;
    }
23
24     /* Vrsi se provera korektnosti vrednosti unetog dana. */
25     switch (mesec) {
        case 1:
27         case 3:

```

## 2 Kontrola toka

```

29  case 5:
30  case 7:
31  case 8:
32  case 10:
33  case 12:
34      /* Dozvoljeni broj dana za januar, mart, maj, jul, avgust,
35         oktobar i decembar je 31 */
36      dozvoljeni_broj_dana = 31;
37      break;
38  case 2:
39      /* Dozvoljeni broj dana za februar je 28 ili 29 u zavisnosti od
40         toga da li je godina prestupna ili ne. */
41      if ((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
42          dozvoljeni_broj_dana = 29;
43      else
44          dozvoljeni_broj_dana = 28;
45      break;
46  case 4:
47  case 6:
48  case 9:
49  case 11:
50      /* Dozvoljeni broj dana za april, jun, septembar i novembar je
51         30. */
52      dozvoljeni_broj_dana = 30;
53      break;
54  }
55
56  if (dan < 0 || dan > dozvoljeni_broj_dana) {
57      printf("Datum nije korektan.\n");
58      return 0;
59  }
60
61  /* Kako su sve provere korektnosti prosle, datum se smatra
62     korektnim. */
63  printf("Datum je korektan.\n");
64
65  return 0;
66 }
```

### Rešenje 2.1.28

```

1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int dan, mesec, godina;
7      int prethodni_dan, prethodni_mesec, prethodni_godina;
8
9      /* Ucitava se datum. */
10     printf("Unesite datum: ");
```

```
11  scanf("%d.%d.%d.", &dan, &mesec, &godina);
13
14  /* Racunaju se dan, mesec i godina prethodnog dana. */
15  prethodni_dan = dan - 1;
16  prethodni_mesec = mesec;
17  prethodni_godina = godina;
18
19  /* Ako je potrebno, vrse se korekcije. */
20  if (prethodni_dan == 0) {
21      prethodni_mesec = mesec - 1;
22      if (prethodni_mesec == 0) {
23          prethodni_mesec = 12;
24          prethodni_godina = godina - 1;
25      }
26
27      switch (prethodni_mesec) {
28          case 1:
29          case 3:
30          case 5:
31          case 7:
32          case 8:
33          case 10:
34          case 12:
35              prethodni_dan = 31;
36              break;
37          case 2:
38              if ((prethodni_godina % 4 == 0 && prethodni_godina % 100 != 0)
39                  || prethodni_godina % 400 == 0)
40                  prethodni_dan = 29;
41              else
42                  prethodni_dan = 28;
43              break;
44          case 4:
45          case 6:
46          case 9:
47          case 11:
48              prethodni_dan = 30;
49      }
50
51  /* Ispis rezultata. */
52  printf("Prethodni datum: %d.%d.%d.\n",
53         prethodni_dan, prethodni_mesec, prethodni_godina);
54
55  return 0;
56 }
```

### Rešenje 2.1.29

Rešenje je analogno rešenju zadatka 2.1.28.

### Rešenje 2.1.30

```
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  int main()
5  {
6      /* Deklaracija potrebnih promenljivih. */
7      int k, l, m, n;
8
9      /* Ucitavaju se vrednosti pozicija na tabli. */
10     printf("Unesite (k,l): ");
11     scanf("%d%d", &k, &l);
12
13     printf("Unesite (m,n): ");
14     scanf("%d%d", &m, &n);
15
16     /* Vrsi se provera ispravnosti ulaznih podataka. */
17     if (k < 1 || k > 8 || l < 1 || l > 8 ||
18         m < 1 || m > 8 || n < 1 || n > 8) {
19         printf("Greska: neispravna pozicija.\n");
20         return -1;
21     }
22
23     if(k == m && l == n){
24         printf("Greska: pozicije moraju biti razlicite.\n");
25         return -1;
26     }
27
28     /* Proverava se da li su (k,l) i (m,n) iste boje. Polja su iste
29        boje ako su: 1) oba reda parna i obe kolone parne ILI 2) oba
30        reda neparna i obe kolone neparne. */
31     if (((k % 2 == m % 2) && (l % 2 == n % 2))
32         || ((k % 2 != m % 2) && (l % 2 != n % 2)))
33         printf("Polja su iste boje.\n");
34     else
35         printf("Polja su razlicite boje.\n");
36
37     /* Proverava se da li kraljica sa (k,l) napada polje (m,n).
38        Kraljica napada polje u sledecim situacijama:
39        1) Ako se nalaze u istom redu (k==m)
40        2) Ako se nalaze u istoj koloni (l==n)
41        3) Ako se nalaze na istoj dijagonali. Dijagonala moze biti:
42           a) paralelna glavnoj dijagonali (abs(k-l) == abs(m-n))
43           b) paralelna sporednoj dijagonali (k+l == m+n) */
44     if ((k == m) || (l == n) || (abs(k - l) == abs(m - n))
45         || (k + l == m + n)){
46         printf("Kraljica sa (%d, %d) ugrozava (%d, %d).\n",
47             k, l, m, n);
48     }
49     else {
50         printf("Kraljica sa (%d, %d) ne ugrozava (%d, %d).\n",
```

```

52         k, l, m, n);
53     }
54     /* Proverava se da li konj sa (k, l) napada polje (m, n). Postoji
55        8 mogucih vrednosti za polja koja konj napada. Vrsi se
56        provera da li je (m,n) jednako nekom od tih polja. */
57     if ((abs(k-m) == 2 && abs(n-l) == 1) || (abs(n-l) == 2 && abs(m-k)
58         == 1))
59         printf("Konj sa (%d, %d) ugrozava (%d, %d).\n",
60             k, l, m, n);
61     else
62         printf("Konj sa (%d, %d) ne ugrozava (%d, %d).\n",
63             k, l, m, n);
64     return 0;
65 }

```

## 2.3 Petlje

**Zadatak 2.3.1** Napisati program koji pet puta ispisuje tekst *Mi volimo da programiramo.*

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Mi volimo da programiramo.
Mi volimo da programiramo.
Mi volimo da programiramo.
Mi volimo da programiramo.
Mi volimo da programiramo.

```

[Rešenje 2.3.1]

**Zadatak 2.3.2** Napisati program koji učitava pozitivan ceo broj  $n$  i  $n$  puta ispisuje tekst *Mi volimo da programiramo.* U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 6
Mi volimo da programiramo.
Mi volimo da programiramo.
Mi volimo da programiramo.
Mi volimo da programiramo.
Mi volimo da programiramo.
Mi volimo da programiramo.

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Greska: pogresan unos broja n.

```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -5  
|| Greska: pogresan unos broja n.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 1  
|| Mi volimo da programiramo.
```

[Rešenje 2.3.2]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 4  
|| 0 1 2 3 4
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -10  
|| Greska: pogresan unos broja n.
```

[Rešenje 2.3.3]

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

- (a) Koristiti `while` petlju.
- (b) Koristiti `for` petlju.
- (c) Koristiti `do-while` petlju.

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite granice intervala: -2 4  
|| -2 -1 0 1 2 3 4
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite granice intervala: 10 6  
|| Greska: pogresan unos granica.
```

[Rešenje 2.3.4]

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

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 18
|| 18! = 6402373705728000

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 8
|| 8! = 40320

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 40
|| Pri racunanju 40! ce doći do prekoracenja.

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -5
|| Greska: neispravan unos.

```

[Rešenje 2.3.5]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 11.43 -6
|| Greska: neispravan unos broja n.

```

*Primer 4*

```

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

```

[Rešenje 2.3.6]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 2.3.7]

**Zadatak 2.3.8** Pravi delioci celog broja su svi delioci sem jedinice i samog tog broja. Napisati program za uneti pozitivan ceo broj  $n$  ispisuje sve njegove prave delioce. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

## 2 Kontrola toka

---

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 100  
|| 2 4 5 10 20 25 50
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -6  
|| Greska: neispravan unos.
```

[Rešenje 2.3.8]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -1400  
|| -14
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 147  
|| 147
```

[Rešenje 2.3.9]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite ceo broj: 6789  
|| 9 8 7 6
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite ceo broj: -892345  
|| 5 4 3 2 9 8
```

[Rešenje 2.3.10]

**Zadatak 2.3.11** Napisati program koji za uneti pozitivan ceo broj ispisuje da li je on deljiv sumom svojih cifara. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 12  
|| Broj 12 je deljiv sa 3.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2564  
|| Broj 2564 nije deljiv sa 17.
```



*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -4
|| Greska: neispravan ulaz.

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 0
|| Greska: neispravan ulaz.

```

[Rešenje 2.3.11]

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

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj t: 7
|| Unesite transakcije:
|| 8 -50 45 2007 -67 -123 14
|| Prihod: 2074
|| Rashod: -240
|| Zarada: 1834

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj t: 5
|| Unesite transakcije:
|| -5 -20 -4 -200 -8
|| Prihod: 0
|| Rashod: -237
|| Gubitak: 237

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj t: -6
|| Greska: neispravan unos.

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 0
|| Nema evidentiranih transakcija.

```

*Primer 5*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj t: 2
|| Unesite transakcije:
|| 120 -120
|| Prihod: 120
|| Rashod: -120
|| Zarada: 0

```

[Rešenje 2.3.12]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva:
1 -5 -6 3 -11
Zbir neparnih i negativnih: -16
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -4
Greska: neispravan unos.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Unesite n brojeva:
5 8 13 17
Zbir neparnih i negativnih: 0
```

### Primer 4

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

[Rešenje 2.3.13]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: :2 35 5 -175 -20
Suma je -15.
```

### Primer 2

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

### Primer 3

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

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 6
Unesite brojeve:
2205 -1904 2 7 -540 5
Suma je -535.
```

[Rešenje 2.3.14]

**Zadatak 2.3.15** Napisati program koji učitava cele brojeve sve dok se ne unese nula i ispisuje proizvod onih unetih brojeva koji su pozitivni.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve:
-87 12 -108 -13 56 0
Proizvod pozitivnih brojeva je 672.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 0
Nije unet nijedan broj.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve:
|| -5 -200 -43 0
|| Medju unetim brojevima nema pozitivnih.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 1 0
|| Proizvod pozitivnih brojeva je 1.
```

[Rešenje 2.3.15]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1857
|| Broj 1857 sadrzi cifru 5.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 84
|| Broj 84 ne sadrzi cifru 5.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -2515
|| Broj -2515 sadrzi cifru 5.
```

[Rešenje 2.3.16]

**Zadatak 2.3.17** Napisati program koji učitava cele brojeve sve do unosa broja nula, a zatim izračunava i ispisuje aritmetičku sredinu unetih brojeva na četiri decimale.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 8 5 6 3 0
|| Aritmeticka sredina: 5.5000
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 0
|| Nisu uneti brojevi.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 762 -12 800 2010 -356 899 -101 0
|| Aritmeticka sredina: 571.7143
```

[Rešenje 2.3.17]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite cene: 8 5.2 6.11 3 0
Prosečna cena: 5.5775
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite cene: 6.32 -9
Greska: neispravan unos cene.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite cene: 0
Nisu unete cene.
```

[Rešenje 2.3.18]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 9
Unesite brojeve:
7.82 4.3 -1.2 56.8 -3.4 -72.1 8.9 11.2 -11.2
Broj promena je 5.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite brojeve:
-23.8 -11.2 0 5.6 7.2
Broj promena je 1.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -6
Greska: neispravan unos.
```

### Primer 4

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

[Rešenje 2.3.19]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj artikla: 6
Unesite cene artikala:
12 3.4 90 100.53 53.2 12.8
Najmanja cena: 3.400000
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj artikla: 3
Unesite cene artikala:
4 -8 92
Greska: neispravan unos cene.
```

*Primer 3*

```

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

```

[Rešenje 2.3.20]

**Zadatak 2.3.21** Nikola želi da obradi bazu i da joj kupi jedan poklon u radnji. On na raspolaganju ima  $m$  dinara. U radnji se nalazi  $n$  artikala i zanima ga koliko ima artikala u radnji čija cena je manja ili jednaka  $m$ . Napisati program koji pomaže Nikoli da brzo odredi broj artikala. Program učitava realan nenegativan broj  $m$ , ceo nenegativan broj  $n$  i  $n$  pozitivnih realnih brojeva. Ispisati koliko artikala ima cenu čija je vrednost manja ili jednaka  $m$ . **NAPOMENA:** *Pretpostaviti da je unos ispravan.*

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 2.3.21]

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

- (a) Broj sa najvećom cifrom desetica. Ukoliko ima više takvih, ispisati prvi.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| Unesite brojeve:
|| 18 365 25 1 78
|| Broj sa najvećom cifrom desetica: 78.

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 8
|| Unesite brojeve:
|| 14 1576 -1267 -89 109 122 306 918
|| Broj sa najvećom cifrom desetica: -89.

```

*Primer 3*

```

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

```

*Primer 4*

```

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

```

## 2 Kontrola toka

---

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: 18 -365 251 1 78
Najviše cifara ima broj -365.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 7
Unesite n brojeva:
3 892 18 21 639 742 85
Najviše cifara ima broj 892.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Nisu uneti brojevi.
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -7
Greska: neispravan unos.
```

### Primer 5

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: 0 1 2 -3 4
Najviše cifara ima broj 0.
```

### Primer 6

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: -5 4 -3 2 1
Najviše cifara ima broj -5.
```

- (c) Broj sa najvećom vodećom cifrom. Vodeća cifra je cifra najveće težine u zapisu broja. Ukoliko ima više takvih, ispisati prvi.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: 8 964 -32 511 27
Broj sa najvećom vodećom cifrom je 964.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
Unesite n brojeva: 0 0 0
Broj sa najvećom vodećom cifrom je 0.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
Unesite n brojeva: 41 669 -8
Broj sa najvećom vodećom cifrom je -8.
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Nisu uneti brojevi.
```

[Rešenje 2.3.22]

**Zadatak 2.3.23** Vršena su merenja nadmorskih visina na određenom delu teritorije i naučnike zanima razlika između najveće i najmanje nadmorske visine. Napisati program koji učitava cele brojeve sve do unosa 0 koji označavaju nadmorske visine i ispisuje razliku najveće i najmanje nadmorske visine.

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 0
|| Nisu unete nadmorske visine.

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: -500 0
|| Razlika: 0

```

*Primer 5*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: -500 -300 -5000 0
|| Razlika: 4700

```

[Rešenje 2.3.23]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve n i d: 5 0
|| Unesite n brojeva: 1 1 1 1 1
|| Broj parova: 4

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve n i d: 1 3
|| Greska: neispravan unos.

```

[Rešenje 2.3.24]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.25]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -67123  
|| Rezultat: -613
```

[Rešenje 2.3.26]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -5  
|| Greska: neispravan unos.
```

[Rešenje 2.3.27]

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



*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 25452
|| Broj je palindrom.

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 895
|| Broj nije palindrom.

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 5
|| Broj je palindrom.

```

[Rešenje 2.3.28]

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

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 10
|| F[10] = 55

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -100
|| Greska: neispravan unos.

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 40
|| F[40] = 102334155

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 20
|| F[20] = 6765

```

[Rešenje 2.3.29]

**Zadatak 2.3.30** Niz prirodnih brojeva formira se prema sledećem pravilu:

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

Napisati program koji za uneti početni član niza  $a_0$  (pozitivan ceo broj) štampa niz brojeva sve do onog člana niza koji je jednak 1. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite prvi clan: 56
|| 56 28 14 7 11 17 26 13 20 10
|| 5 8 4 2 1

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite prvi clan: -48
|| Greska: neispravan unos.

```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite prvi clan: 67
|| 67 101 152 76 38 19 29 44 22 11
|| 17 26 13 20 10 5 8 4 2 1
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite prvi clan: 33
|| 33 50 25 38 19 29 44 22
|| 11 17 26 13 20 10 5 8 4 2 1
```

[Rešenje 2.3.30]

**\* Zadatak 2.3.31** Papir  $A_0$  ima površinu  $1m^2$  i odnos stranica  $1 : \sqrt{2}$ . Papir  $A_1$  dobija se podelom papira  $A_0$  po dužoj ivici. Papir  $A_2$  dobija se podelom  $A_1$  papira po dužoj ivici itd. Napisati program koji za uneti nenegativan broj  $k$  ispisuje dimenzije papira  $A_k$  u milimetrima. Rezultat ispisati kao celobrojne vrednosti. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite format papira: 4
|| 210 297
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite format papira: 0
|| 840 1189
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite format papira: -7
|| Greska: neispravan unos.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite format papira: 9
|| 37 52
```

[Rešenje 2.3.31]

**Zadatak 2.3.32** Napisati program koji učitava karaktere dok se ne unese karakter tačka, i ako je karakter malo slovo ispisuje odgovarajuće veliko, ako je karakter veliko slovo ispisuje odgovarajuće malo, a u suprotnom ispisuje isti karakter kao i uneti.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Danas je Veoma Lep DAN.
|| dANAS JE vEOMA lEP dan
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| PROGRAMIRANJE 1 je zanimljivo!.
|| programiranje 1 JE ZANIMLJIVO!
```

[Rešenje 2.3.32]

**Zadatak 2.3.33** Napisati program koji učitava karaktere sve do kraja ulaza, a potom ispisuje broj velikih slova, broj malih slova, broj cifara, broj belina i zbir unetih cifara.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Tekst sa brojevima: 124, -8900, 23...
velika: 1, mala: 15
cifre: 9, beline: 5
suma cifara: 29

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
NEMA cifara!
velika: 4, mala: 6
cifre: 0, beline: 1
suma cifara: 0

```

[Rešenje 2.3.33]

**Zadatak 2.3.34** Program učitava pozitivan ceo broj  $n$ , a potom i  $n$  karaktera. Za svaki od samoglasnika ispisati koliko puta se pojavio među unetim karakterima. Ne praviti razliku između malih i velikih slova. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

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

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 7
Unesite n karaktera: jk+EEae
Samoglasnik a: 1
Samoglasnik e: 3
Samoglasnik i: 0
Samoglasnik o: 0
Samoglasnik u: 0

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n karaktera: UuUuU
Samoglasnik a: 0
Samoglasnik e: 0
Samoglasnik i: 0
Samoglasnik o: 0
Samoglasnik u: 5

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -7
Greska: neispravan unos.

```

[Rešenje 2.3.34]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.35]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 14
Suma kubova: 11025
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 25
Suma kubova: 105625
```

### Primer 3

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

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -5
Greska: neispravan unos.
```

[Rešenje 2.3.36]

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

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| k=1, suma=1
|| k=2, suma=9
|| k=3, suma=36
|| k=4, suma=100
|| k=5, suma=225

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 8
|| k=1, suma=1
|| k=2, suma=9
|| k=3, suma=36
|| k=4, suma=100
|| k=5, suma=225
|| k=6, suma=441
|| k=7, suma=784
|| k=8, suma=1296

```

*Primer 3*

```

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

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -5
|| Greska: neispravan unos.

```

[Rešenje 2.3.37]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 5.5 0
|| Greska: neispravan unos.

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: -0.5 -5
|| Greska: neispravan unos.

```

[Rešenje 2.3.38]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 5.5 0
|| Greska: neispravan unos.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: -0.5 -5
|| Greska: neispravan unos.
```

[Rešenje 2.3.39]

**\* Zadatak 2.3.40** Napisati program koji učitava realne brojeve  $x$  i  $eps$  i sa zadatom tačnošću  $eps$  izračunava i ispisuje sumu  $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$ . Izračunati sumu u odnosu na tačnost  $eps$  znači uporediti poslednji član sume sa  $eps$  i ukoliko je taj poslednji član manji od  $eps$  prekinuti dalja izračunavanja. UPUTSTVO: Prilikom računanja sume koristiti prethodni izračunati član sume u računanju sledećeg člana sume. Naime, ako je izračunat član sume  $\frac{x^n}{n!}$  na osnovu njega se lako može dobiti član  $\frac{x^{n+1}}{(n+1)!}$ . Nikako ne računati stepen i faktorijel odvojeno zbog neefikasnosti takvog rešenja i zbog mogućnosti prekoračenja.

### Primer 1

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

### Primer 2

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

[Rešenje 2.3.40]

**\* Zadatak 2.3.41** Napisati program koji učitava realne brojeve  $x$  i  $eps$  i sa zadatom tačnošću  $eps$  izračunava i ispisuje sumu  $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} + \dots$ . NAPOMENA: Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.

### Primer 1

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

### Primer 2

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

[Rešenje 2.3.41]

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

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 2.3.42]

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

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.43]

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

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

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x i n: 5.6 8
|| S=0.779792
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x i n: 14.32 11
|| S=-6714.066406
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x i n: 2 -6
|| Greska: neispravan unos.
```

[Rešenje 2.3.44]

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

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

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 2.3.45]

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

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

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 20
|| Greska: neispravan unos.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -3
|| Greska: neispravan unos.
```



[Rešenje 2.3.46]

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

$$\underbrace{((\dots((x+a)^2+a)^2+a)^2+\dots a)^2}_n.$$

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve x i a: 3.2 0.2
|| Unesite broj n: 5
|| Izraz = 135380494030332048.000000
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve x i a: 2 1
|| Unesite broj n: 3
|| Izraz = 10201.000000
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve x i a: 2.6 0.3
|| Unesite broj n: 3
|| Izraz = 5800.970129
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve x i a: 5.4 7
|| Unesite broj n: -2
|| Greska: neispravan unos.
```

[Rešenje 2.3.47]

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

- (a) Napisati program koji za unetu vrednost  $n$  ispisuje tablicu množenja.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| 1 2 3
|| 2 4 6
|| 3 6 9
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 4
|| 1 2 3 4
|| 2 4 6 8
|| 3 6 9 12
|| 4 8 12 16
```

- (b) Napisati program koji za unetu  $n$  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  $i, j = 0, \dots, n$ . Prav ugao se nalazi u gornjem levom uglu slike i njegova koordinata je  $(0, 0)$ . Koordinata  $i$  se uvećava po vrsti, a koordinata  $j$  po koloni, pa je zato koordinata tačke koja je ispod tačke  $(0, 0)$  jednaka  $(1, 0)$ , a koordinata tačke koja je desno od tačke  $(0, 0)$  jednaka  $(0, 1)$ .

### Primer 1

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 1
|| (0,0)

```

### Primer 2

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 2
|| (0,0) (0,1)
|| (1,0)

```

### Primer 3

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| (0,0) (0,1) (0,2)
|| (1,0) (1,1)
|| (2,0)

```

### Primer 4

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 4
|| (0,0) (0,1) (0,2) (0,3)
|| (1,0) (1,1) (1,2)
|| (2,0) (2,1)
|| (3,0)

```

[Rešenje 2.3.48]

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

- (a) Slika predstavlja kvadrat stranice  $n$  sastavljen od zvezdica.

Primer 1

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

Primer 2

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

- (b) Slika predstavlja rub kvadrata dimenzije  $n$ .

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 1

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

[Rešenje 2.3.49]

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

### Primer 1

```

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

```

### Primer 2

```

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

```

[Rešenje 2.3.50]

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

### Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
+
+
+++++
+
+

```

### Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
+
+++
+

```

[Rešenje 2.3.51]

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

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

### Primer 1

```

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

```

### Primer 1

```

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

```

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

### Primer 1

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

### Primer 2

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

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

### Primer 1

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

### Primer 1

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

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

### Primer 1

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

### Primer 2

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

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

### Primer 1

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

### Primer 2

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

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.52]

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

- (a) Slika predstavlja jednakostranični trougao stranice  $n$  koji je sastavljen od zvezdica.

*Primer 1*

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

*Primer 2*

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

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

*Primer 1*

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

*Primer 2*

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

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

Primer 1

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

Primer 2

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

- (d) Slika predstavlja rub jednakostraničnog trougla čija stranica je dužine  $n$ .

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

\* **Zadatak 2.3.54** Napisati program koji za uneti pozitivan ceo broj  $n$  iscrtava strelice dimenzije  $n$ . NAPOMENA: *Pretpostaviti da je unos ispravan.*

### Primer 1

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

### Primer 2

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

[Rešenje 2.3.54]

\* **Zadatak 2.3.55** Napisati program koji učitava pozitivan ceo broj  $n$ , i iscrta sliku koja se dobija na sledeći način: u prvom redu je jedna zvezdica, u drugom redu su dve zvezdice razdvojene razmakom, treći red je sastavljen od zvezdica i iste je dužine kao i drugi red, četvrti red se sastoji od tri zvezdice razdvojene razmakom, a peti red je sastavljen od zvezdica i iste je dužine kao i četvrti red itd. Ukupna visina slike je  $n$ . NAPOMENA: *Pretpostaviti da je unos ispravan.*

### Primer 1

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

[Rešenje 2.3.55]

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



## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve n i m: 5 3
*****
*       *       *       *
*       *       *       *
*       *       *       *
*       *       *       *
*****

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve n i m: 4 4
*****
*       *       *       *
*       *       *       *
*       *       *       *
*       *       *       *
*****

```

[Rešenje 2.3.56]

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

## Primer 1

```

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

```

## Primer 2

```

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

```

[Rešenje 2.3.57]

**Zadatak 2.3.58** Napisati program koji učitava ceo broj  $n$  ( $n \geq 2$ ) i koji iscrtava sliku kuće sa krovom: kuća je kvadrat stranice  $n$ , a krov jednakostranični trougao stranice  $n$ . Pretpostaviti da je unos korektan.

## Primer 1

```

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

```

## Primer 1

```

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

```

[Rešenje 2.3.58]

\* **Zadatak 2.3.59** Napisati program koji učitava pozitivan ceo broj  $n$  i ispisuje brojeve od 1 do  $n$ , zatim od 2 do  $n - 1$ , 3 do  $n - 2$ , itd. Ispis se završava kada nije moguće ispisati ni jedan broj. NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.3.59]

\* **Zadatak 2.3.60** Napisati program koji učitava pozitivan ceo broj  $n$  i ispisuje sve brojeve od 1 do  $n$ , zatim svaki drugi broj od 1 do  $n$ , zatim svaki treći broj od 1 do  $n$  itd., završavajući sa svakim  $n$ -tim (tj. samo sa 1). NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

[Rešenje 2.3.60]

## 2.4 Rešenja

### Rešenje 2.3.1

```
1 #include <stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     int i;
7
8     /* Promenljiva i kontrolise koliko puta ce se petlja izvesti i
9        naziva se brojac petlje. Njena pocetna vrednost se postavlja na
10        0 jer se u pocetku petlja nije ni jednom izvela. */
11    i = 0;
12
13    /* Petlja ce se izvesti za i=0,1,2,3,4. Kada i dostigne vrednost
14       5 uslov i < 5 nece biti ispunjen i prelazi se na prvu sledecu
15       naredbu nakon tela petlje. */
16    while (i < 5) {
17
18        /* Ispis poruke. */
19        printf("Mi volimo da programiramo.\n");
20
21        /* Uvecavanje brojaca za 1. */
22        i++;
23    }
24
25    return 0;
26 }
```

### Rešenje 2.3.2

```
1 #include<stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     int i, n;
7
8     /* Ucitava se vrednost broja n. */
9     printf("Unesite broj n: ");
10    scanf("%d", &n);
11
12    /* Vrsi se provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: pogresan unos broja n.\n");
15        return -1;
16    }
17 }
```

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

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

27  return 0;
}
```

### Rešenje 2.3.3

```
1  #include <stdio.h>

3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int i, n;

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

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

17     /* Inicijalizacija brojaca. */
18     i = 0;

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

25         /* Ispisuje se trenutna vrednost brojaca. */
26         printf("%d\n", i);

27         /* Prelazi se na sledeci broj. */
28         i++;
29     }

31     return 0;
33 }
```

## Rešenje 2.3.4

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int n, m, i;
7
8      /* Ucitavaju se vrednosti granica intervala. */
9      printf("Unesite granice intervala: ");
10     scanf("%d%d", &n, &m);
11
12     /* Vrsi se provera ispravnosti ulaznih podataka. */
13     if (m < n) {
14         printf("Greska: pogresan unos granica.\n");
15         return -1;
16     }
17
18     /* a) I nacin: koriscenjem while petlje. */
19     /* Inicijalizacija brojaca na levu granicu intervala. */
20     i = n;
21
22     /* Ispisuju se sve vrednosti brojaca izmedju leve i desne
23     granice intervala, ukljucujuci i same granice. */
24     while (i <= m) {
25         printf("%d ", i);
26         i++;
27     }
28
29     /* b) II nacin: koriscenjem for petlje.
30
31     Naredba i=n se izvorsava jednom, pre prve iteracije.
32     Uslov petlje i<=m se proverava pre svake iteracije.
33     Naredba i++ se izvorsava nakon svake iteracije.
34
35     for (i = n; i <= m; i++){
36         printf("%d ", i);
37     } */
38
39     /* c) III nacin: koriscenjem do while petlje.
40
41     Uslov petlje se proverava na kraju svake iteracije.
42     Zbog toga se do while petlja izvorsava bar jednom, cak i u
43     slucaju da uslov petlje nikada nije ispunjen. U ovom slucaju
44     je to ispravno jer je poznato da ce interval imati bar
45     jedan element. U opstem slucaju to ne mora da vaziti.
46
47     i = n;
48     do {
49         printf("%d ", i);
50         i++;
```

```
51 }
    while (i <= m); */
53
    printf("\n");
55
    return 0;
57 }
```

### Rešenje 2.3.5

```
1  #include<stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int n, tekuca_vrednost;
7
8      /* Za cuvanje vrednosti faktoriijela se koristi tip unsigned long
9         jer izracunata vrednost moze da bude jako veliki broj. */
10     unsigned long faktorijel;
11
12     /* Ucitava se vrednost broja n. */
13     printf("Unesite broj n: ");
14     scanf("%d", &n);
15
16     /* Vrsi se provera ispravnosti ulaza. */
17     if (n < 0) {
18         printf("Greska: neispravan unos..\n");
19         return -1;
20     }
21
22     if (n >= 22) {
23         printf("Pri racunanju %d! ce doci do prekoracenja.\n", n);
24         return -1;
25     }
26
27     /* Tekuca vrednost uzima vrednosti n, n-1, n-2, ..., 2.
28        Na pocetku se inicijalizuje na n, a zatim se u svakoj
29        iteraciji umanjuje za 1. */
30     tekuca_vrednost = n;
31
32     /* Inicijalizacija vrednosti faktoriijela. */
33     faktorijel = 1;
34
35     /* Racuna se vrednost faktoriijela tako sto se trenutni rezultat
36        u svakoj iteraciji mnozi sa promenljivom cija vrednost krece
37        od n, a zatim se u svakoj iteraciji umanjuje za 1. */
38     while (tekuca_vrednost > 1) {
39         faktorijel = faktorijel * tekuca_vrednost;
40         tekuca_vrednost--;
41     }
```

```

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

```

### Rešenje 2.3.6

```

1  #include <stdio.h>

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

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

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

19  /* Inicijalizacija rezultata. */
    rezultat = 1;

21  /* Vrednost n-tog stepena broja x se dobija tako sto se tekuca
23   vrednost rezultata n puta pomnozi sa brojem x.
        (rezultat = x * x * ... * x) = x^n */
25   for (i=0; i<n; i++)
        rezultat = rezultat * x;

27  /* Ispis rezultata. */
29   printf("Rezultat: : %.5f\n", rezultat);

31   return 0;
}

```

### Rešenje 2.3.7

```

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

3
   int main()
5   {

```

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

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

13 /* Pamti se znak stepena i uzima se apsolutna vrednost stepena. */
15 znak = 1;
if(n < 0){
17     znak = -1;
    n = abs(n);
19 }
/* Inicijalizacija rezultata. */
21 rezultat = 1;

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

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

33 return 0;
}
```

### Rešenje 2.3.8

```
1 #include<stdio.h>

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

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

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

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



```

21     nuli). Ako je uslov ispunjen, taj broj se ispisuje.
    for (i = 2; i < n; i++) {
23         if (n % i == 0)
            printf("%d ", i);
25     }
    printf("\n");
    */

27     /* II nacin (brzi): Provera se ne vrši za sve brojeve iz
29     intervala [2, n-1], vec za brojeve iz intervala
        [2, sqrt(n)], tj. za sve brojeve k za koje vazi da je
31     k*k <= n. */
    for (i = 2; i*i <= n; i++) {
33         /* Ako i deli n, treba razlikovati dva slucaja. */
        if (n % i == 0){
35             if (i == n / i) {
                /* I slucaj: kada je i koren broja, npr. 4 za 16,
37                 ispisuje se samo broj i. */
                printf("%d ", i);
39             }
            else {
41                 /* II slucaj: u suprotnom, ispisuje se taj broj i
                    broj n / i, npr. 2 za 16, ispisuju se i 2 i 8. */
43                 printf("%d %d ", i, n / i);
            }
45         }
    }
47     printf("\n");

49     return 0;
}

```

### Rešenje 2.3.9

```

1  #include <stdio.h>

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

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

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

```

```
19 }
20
21 /* Dok god je poslednja cifra broja n nula, broj n se deli sa
22    10 i na taj nacin se iz broja uklanja poslednja cifra. */
23 while (n % 10 == 0)
24     n = n / 10;
25
26 /* Ispis rezultata. */
27 printf("%d\n", n);
28
29 return 0;
30 }
```

### Rešenje 2.3.10

```
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  int main()
5  {
6      /* Deklaracija potrebnih promenljivih. */
7      int x;
8
9      /* Ucitava se vrednost broja x. */
10     printf("Unesi ceo broj:");
11     scanf("%d", &x);
12
13     /* Uzima se apsolutna vrednost broja da bi izdvojene cifre bile
14        pozitivni brojevi. Na primer, 123%10 je 3, a -123%10 je -3. */
15     x = abs(x);
16
17     /* Slucuj kada je uneti broj 0 se posebno obradjuje. */
18     if(x == 0)
19     {
20         printf("0\n");
21         return 0;
22     }
23
24     /* U petlji se obradjuje cifra po cifra broja, dok god ima
25        neobradjenih cifara u broju. */
26     while (x != 0) {
27         /* Ispisuje se poslednja cifra broja x. */
28         printf("%d ", x % 10);
29
30         /* Uklanja se poslednja cifra broja x. */
31         x /= 10;
32     }
33     printf("\n");
34
35     return 0;
36 }
```

## Rešenje 2.3.11

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int n, suma, pom_n;
7
8      /* Ucitava se vrednost broja n. */
9      printf("Unesite broj: ");
10     scanf("%d", &n);
11
12     /* Vrsi se provera ispravnosti ulaza. */
13     if (n <= 0) {
14         printf("Greska: neispravan unos.\n");
15         return -1;
16     }
17
18     /* Pravi se kopija originalnog broja, da bi originalna vrednost
19        n ostala nepromenjena. */
20     pom_n = n;
21
22     /*Inicijalizacija sume cifara. */
23     suma = 0;
24
25     /* Racuna se suma cifara. */
26     while (pom_n != 0) {
27         /* Na sumu se dodaje poslednja cifra broja. */
28         suma += pom_n % 10;
29         /* Sa broja se skida poslednja cifra. */
30         pom_n /= 10;
31     }
32
33     /* Ispis rezultata. */
34     if (n % suma == 0)
35         printf("Broj %d je deljiv sa %d.\n", n, suma);
36     else
37         printf("Broj %d nije deljiv sa %d.\n", n, suma);
38
39     return 0;
40 }
```

## Rešenje 2.3.12

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

```
3  int main()
5  {
    /* Deklaracija potrebnih promenljivih. */
7   int t, x, i;
    int ukupan_prihod, ukupan_rashod, ukupan_rashod_abs;

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

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

23  /* Inicijalizacija suma. */
25  ukupan_prihod = 0;
    ukupan_rashod = 0;

27  /* Ucitavanje transakcija i izracunavanje suma. */
29  printf("Unesite transakcije: ");
    i = 0;
31  while (i < t) {
        /* Ucitava se jedna transakcija. */
33     scanf("%d", &x);

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

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

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

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

```
55 return 0;
}
```

### Rešenje 2.3.13

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int n, x, i;
7      int zbir = 0;
8
9      /* Ucitava se vrednost broja n. */
10     printf("Unesite broj n: ");
11     scanf("%d", &n);
12
13     /* Vrsi se provera ispravnosti ulaza. */
14     if (n <= 0) {
15         printf("Greska: neispravan unos.\n");
16         return -1;
17     }
18
19     /* Ucitava se n brojeva i izracunava se trazeni zbir. */
20     printf("Unesite n brojeva: ");
21     i = 0;
22     while (i < n) {
23         /* Ucitava se jedan broj. */
24         scanf("%d", &x);
25
26         /* Ako je ucitani broj negativan i neparan,
27            dodaje se na zbir. */
28         if (x < 0 && x % 2 != 0)
29             zbir = zbir + x;
30
31         /* Uvecava se brojac. */
32         i++;
33     }
34
35     /* Ispis rezultata. */
36     printf("Zbir neparnih i negativnih: %d\n", zbir);
37
38     return 0;
39 }
```

### Rešenje 2.3.14

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

```
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     int n, broj;
7     int suma = 0;
8     int i;
9
10    /* Ucitava se vrednost broja n. */
11    printf("Unesite broj n: ");
12    scanf("%d", &n);
13
14    /* Vrsi se provera ispravnosti ulaza. */
15    if (n <= 0) {
16        printf("Greska: neispravan unos.\n");
17        return -1;
18    }
19
20    /* Ucitava se n brojeva i izracunava se trazena suma. */
21    printf("Unesite brojeve: ");
22    for (i = 0; i < n; i++) {
23        scanf("%d", &broj);
24
25        if (broj % 5 == 0 && broj % 7 != 0)
26            suma += broj;
27    }
28
29    /* Ispis rezultata. */
30    printf("Suma je %d.\n", suma);
31
32    return 0;
33 }
```

### Rešenje 2.3.15

```
1 #include <stdio.h>
2 int main()
3 {
4     /* Deklaracija potrebnih promenljivih. */
5     int x, proizvod;
6
7     /* Indikator koji oznacava da li je korisnik uneo bar jedan
8        broj. */
9     int unet_bar_jedan = 0;
10
11    /* Indikator koji oznacava da li je korisnik uneo bar jedan
12       pozitivan broj. */
13    int unet_pozitivan = 0;
14
15    /* Inicijalizacija proizvoda. */
16    proizvod = 1;
17 }
```

```

19 printf("Unesite brojeve:");
21 while (1) {
23     /* Ucitava se jedan broj. */
24     scanf("%d", &x);
25
26     /* Ako je uneta nula, petlja se prekida naredbom break. */
27     if (x == 0)
28         break;
29
30     /* Ako petlja nije prekinuta, znaci da je unet bar jedan broj.
31        Iz tog razloga se vrednost indikatora za unete brojeve
32        postavlja na 1. */
33     unet_bar_jedan = 1;
34
35     /* Proverava se da li je broj x pozitivan. */
36     if(x > 0){
37         /* Ako jeste, znaci da je unet bar jedan pozitivan broj i iz
38            tog razloga se vrednost odgovarajuceg indikatora postavlja
39            na 1. */
40         unet_pozitivan = 1;
41
42         /* Azurira se vrednost proizvoda pozitivnih brojeva. */
43         proizvod = proizvod * x;
44     }
45 }
46
47 /* Ispis rezultata. */
48 if (unet_bar_jedan == 0)
49     printf("Nije unet nijedan broj.\n");
50 else if (unet_pozitivan == 0)
51     printf("Medju unetim brojevima nema pozitivnih.\n");
52 else
53     printf("Proizvod pozitivnih brojeva je %d.\n", proizvod);
54
55 return 0;
56 }

```

### Rešenje 2.3.16

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      /* Deklaracije potrebnih promenljivih. */
7      int n, cifra, n_original;
8      int pronadjena_petica = 0;
9

```

## 2 Kontrola toka

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

13
    /* Pamti se originalna vrednost unetog broja. */
15    n_original = n;

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

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

25
        /* Proverava se da li je ona jednaka broju 5 */
27        if (cifra == 5) {
            /* Ako jeste, vrednost odgovarajuceg indikatora se postavlja
29                na 1 i petlja se prekida. */
            pronadjena_petica = 1;
            break;
31        }

33
        /* Ako petlja nije prekinuta, iz broja se uklanja poslednja
35            cifra i postupak se ponavlja dok god ima neobradjenih
            cifara. */
        n = n / 10;
37    }

39
    /* Ispis rezultata.
41        Napomena: Koristi se unapred zapamcena promenljiva n_original
            jer je promenljiva n izmenjena u petlji. */
43    if (pronadjena_petica == 0)
        printf("Broj %d sadrzi cifru 5.\n", n_original);
45    else
        printf("Broj %d ne sadrzi cifru 5.\n", n_original);

47    return 0;
49 }
```

### Rešenje 2.3.17

```
1  #include <stdio.h>

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



```

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

14
15         if (x == 0)
16             break;

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

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

22
23     /* Ispis rezultata.
24      Napomena: primetiti da su i suma i broj_brojeva celi brojevi
25      i da je neophodno bar jednu od te dve vrednosti pretvoriti
26      u realan broj kako deljenje ne bi bilo celobrojno. */
27     if (broj_brojeva == 0)
28         printf("Nisu uneti brojevi.\n");
29     else
30         printf("Aritmeticka sredina: %.4f\n",
31                (double) suma / broj_brojeva);

32
33     return 0;
34 }

```

### Rešenje 2.3.18

```

1  #include <stdio.h>

3  int main()
4  {
5      /* Deklaracije potrebnih promenljivih. */
6      float cena, suma = 0;
7      int broj_artikla = 0;

8
9      /* Cene se ucitavaju sve do unosa broja 0. */
10     printf("Unesite cenu: ");
11     while (1) {
12         scanf("%f", &cena);

13
14         if (cena == 0)
15             break;

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

```

## 2 Kontrola toka

---

```
21     }

23     /* Suma se uvecava za vrednost unete cene. */
    suma += cena;

25     /* Broj unetih artikala se uvecava za 1. */
27     broj_artikla++;
    }

29     /* Ispis rezultata. */
31     if (broj_artikla == 0)
        printf("Nisu unete cene.\n");
33     else
        printf("Prosecna cena: %.4f\n", suma / broj_artikla);

35     return 0;
37 }
```

### Rešenje 2.3.19

```
1  #include <stdio.h>

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

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

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

19    printf("Unesite brojeve: ");
    /* Provera promene znaka se vrsi za svaka dva susedna uneta
21    broja. Prvi broj se ucitava pre petlje i smesta se u
    promenljivu prethodni. Zatim se u petlji ucitava drugi i
23    njihov znak se poredi. Postupak se ponavlja za sve parove,
    tako sto se uvek na kraju petlje poslednji ucitani broj
25    postavi da bude prethodni za sledecu iteraciju. */
    scanf("%lf", &prethodni);

27

29    /* Kako je vec jedan broj unet, brojac se postavlja na 1, a ne
    na 0. */
    for (i = 1; i < n; i++) {
31
```

```

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

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

45      /* Trenutni broj postaje prethodni za sledecu iteraciju. */
      prethodni = trenutni;
47  }

49      /* Ispis rezultata. */
      printf("Broj promena je %d.\n", broj_promena);
51
53      return 0;
}

```

### Rešenje 2.3.20

```

1  #include <stdio.h>

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

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

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

19     printf("Unesite cene artikala:");

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

```

## 2 Kontrola toka

---

```
27     }
    min_cena = cena;
29
    /* Ucitava se i preostalih n-1 cena i racuna se najmanja. */
31    for(i=1; i<n; i++){
        scanf("%f", &cena);
33
        if (cena <= 0) {
35            printf("Greska: neispravan unos cene.\n");
            return -1;
37        }
39        if (cena < min_cena)
            min_cena = cena;
41        i++;
    }
43
    /* Ispis rezultata. */
45    printf("Najmanja cena: %f\n", min_cena);
47
    return 0;
}
```

### Rešenje 2.3.21

```
1  #include <stdio.h>
3  int main()
{
5     /* Deklaracije potrebnih promenljivih. */
    float cena, m;
7     unsigned int n, i, broj_artikala = 0;
9
    /* Ucitava se vrednost broja m. */
    printf("Nikolin budzet: ");
11    scanf("%f", &m);
13
    /* Ucitava se broj artikala. */
    printf("Unesite broj artikala: ");
15    scanf("%u", &n);
17
    /* Unose se cene artikala i racuna se rezultat. */
    printf("Unesite cene artikala: ");
19
    for(i=0; i<n; i++){
21        /* Ucitava se cena artikla. */
        scanf("%f", &cena);
23
        /* Provera se da li Nikola moze da kupi trenutni artikal. */
25        if (cena <= m)
            broj_artikala++;
    }
}
```

```

27     }

29     /* Ispis rezultata. */
    printf("Ukupno artikala: %d\n", broj_artikala);

31     return 0;

33 }

```

### Rešenje 2.3.22

Rešenje (a)

```

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

3

5  int main()
6  {
   /* Deklaracije potrebnih promenljivih. */
7  int n, i, x, rezultat;
   int x_desetica, najveca_desetica;

9

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

13

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

19

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

25

   printf("Unesite brojeve: ");

27

   /* Prvi broj se ucitava pre petlje, zbog ispravne
29     inicijalizacije. */
   scanf("%d", &x);
31 /* Promenljiva najveca_desetica se postavlja na cifru desetica
   ucitanog broja. Napomena: pri racunanju se uzima apsolutna
33     vrednost broja jer je npr. (-123/10)= -12 i -12 % 10 = -2,
   a cifra desetica treba da bude 2. */
35 najveca_desetica = (abs(x) / 10) % 10;
   /* Kako je na kraju potrebno ispisati broj cija je cifra desetica
37     najveca, trenutna vrednost rezultata se postavlja na vrednost
   ucitanog broja. */

```

## 2 Kontrola toka

```
39  rezultat = x;

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

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

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

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

61  najveca_desetica = -1;
    for(i=0; i<n; i++)
63  {
        scanf("%d", &x);

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

67      if (x_desetica > najveca_desetica) {
        najveca_desetica = x_desetica;
69      rezultat = x;
    }
71 }
73 */

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

### Rešenje (b)

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

```
9  int x, x_kopija, broj_cifara;
10 int najveći_broj_cifara, rezultat;
11
12 /* Učitava se vrednost broja n. */
13 printf("Unesite broj n: ");
14 scanf("%d", &n);
15
16 /* Vrsi se provera ispravnosti ulaza. */
17 if (n < 0) {
18     printf("Greska: neispravan unos.\n");
19     return -1;
20 }
21
22 /* Ako nema unetih brojeva, ispisuje se odgovarajuća poruka. */
23 if (n == 0) {
24     printf("Nisu uneti brojevi.\n");
25     return 0;
26 }
27
28 /* Maksimalan broj cifara se postavlja na 0 jer svaki broj ima
29    više od 0 cifara. */
30 najveći_broj_cifara = 0;
31
32 printf("Unesite n brojeva: ");
33 for (i = 0; i < n; i++) {
34     scanf("%d", &x);
35
36     /* Racuna se broj cifara unetog broja x. */
37     x_kopija = abs(x);
38     broj_cifara = 0;
39     do {
40         broj_cifara++;
41         x_kopija = x_kopija / 10;
42     } while (x_kopija != 0);
43
44     /* Ako je broj cifara unetog broja veći od najvećeg broja
45        cifara, azuriraju se vrednosti najvećeg broja cifara i
46        tekućeg rezultata. */
47     if (broj_cifara > najveći_broj_cifara) {
48         najveći_broj_cifara = broj_cifara;
49         rezultat = x;
50     }
51 }
52
53 /* Ispis rezultata. */
54 printf("Najviše cifara ima broj %d.\n", rezultat);
55
56 return 0;
57 }
```

Rešenje (c)

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      /* Deklaracije potrebnih promenljivih. */
7      int n, i;
8      int x, x_kopija, vodeca_cifra;
9      int najveca_vodeca_cifra, rezultat;
10
11      /* Ucitava se vrednost broja n. */
12      printf("Unesite broj n: ");
13      scanf("%d", &n);
14
15      /* Vrsi se provera ispravnosti ulaza. */
16      if (n < 0) {
17          printf("Greska: neispravan unos.\n");
18          return -1;
19      }
20
21      /* Ako nema unetih brojeva, ispisuje se odgovarajuca poruka. */
22      if (n == 0) {
23          printf("Nisu uneti brojevi.\n");
24          return 0;
25      }
26
27      /* Inicijalizacija najvece vodece cifre na -1. */
28      najveca_vodeca_cifra = -1;
29
30      printf("Unesite n brojeva: ");
31      for (i = 0; i < n; i++) {
32          scanf("%d", &x);
33
34          /* Racuna se vodeca cifra ucitanog broja x. */
35          x_kopija = abs(x);
36          while (x_kopija > 10) {
37              x_kopija = x_kopija / 10;
38          }
39          vodeca_cifra = x_kopija;
40
41          /* Ako je izdvojena cifra veca od najvece vodece cifre,
42             azuriraju se vrednosti najvece vodece cifre i rezultata. */
43          if (vodeca_cifra > najveca_vodeca_cifra) {
44              najveca_vodeca_cifra = vodeca_cifra;
45              rezultat = x;
46          }
47      }
48
49      /* Ispis rezultata. */
50      printf("%d\n", rezultat);
```



```
52     return 0;
    }
```

### Rešenje 2.3.23

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int x;
7      int najmanja, najveca;
9      printf("Unesite brojeve: ");
10     /* Prvi broj se ucitava izvan petlje zbog inicijalizacije
11        najvece i najmanje vrednosti nadmorske visine.
12        Napomena: Ovde bi inicijalizacija najveca=-1 bila pogresna
13        jer moze da se desi da su svi uneti brojevi negativni i manji
14        od -1 i onda bi najveca i nakon izvorsavanja tela petlje ostala
15        -1. */
16     scanf("%d", &x);
17     najveca = x;
18     najmanja = x;
19
20     /* Ako nema unetih nadmorskih visina, ispisuje se odgovarajuca
21        poruka. */
22     if(x == 0)
23     {
24         printf("Nisu unete nadmorske visine.");
25         return 0;
26     }
27
28     /* Za svaki ucitani broj se proverava da li je manji od najmanje
29        ili veci od najvece i vrsi se azuriranje odgovarajucih
30        vrednosti. Petlja se prekida kada se unese broj 0.*/
31     while (1) {
32         scanf("%d", &x);
33
34         if(x == 0)
35             break;
36
37         if (x > najveca)
38             najveca = x;
39
40         if (x < najmanja)
41             najmanja = x;
42     }
43
44     /* Ispis rezultata. */
45     printf("Razlika: %d\n", najveca - najmanja);
```

```
47     return 0;
    }
```

### Rešenje 2.3.24

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      /* Deklaracija potrebnih promenljivih. */
7      int n, d, i;
8      int x, y;
9      int broj_parova = 0;
10
11     /* Ucitavaju se vrednosti n i d. */
12     printf("Unesite brojeve n i d: ");
13     scanf("%d %d", &n, &d);
14
15     /* Vrsi se provera ispravnosti ulaza. */
16     if (n <= 1 || d < 0) {
17         printf("Greska: neispravan unos.\n");
18         return -1;
19     }
20
21     printf("Unesite n brojeva: ");
22
23     /* Prvi broj se ucitava pre petlje. */
24     scanf("%d", &x);
25
26     for (i = 1; i < n; i++) {
27         scanf("%d", &y);
28
29         /* Provera se da li su brojevi na rastojanju d. */
30         if (abs(y - x) == d)
31             broj_parova++;
32
33         /* Broj iz tekuce iteracije se cuva kako bi mogao da se
34            upotrebljava u narednoj iteraciji. */
35         x = y;
36     }
37
38     /* Ispis rezultata. */
39     printf("Broj parova: %d\n", broj_parova);
40
41     return 0;
42 }
```

### Rešenje 2.3.25

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int n, cifra;
7      unsigned int rezultat;
8      int pozicija;
9
10     /* Ucitava se vrednost broja n. */
11     printf("Unesite broj: ");
12     scanf("%d", &n);
13
14     /* Vrsi se provera ispravnosti ulaza. */
15     if (n <= 0) {
16         printf("Greska: neispravan unos.\n");
17         return -1;
18     }
19
20     /* Inicijalizacija pozicije i rezultata.
21        Pozicija oznacava tezinu trenutne cifre i moze imati vrednosti
22        1, 10, 100, 1000, ... */
23     pozicija = 1;
24     rezultat = 0;
25
26     /* U petlji se izdvaja cifra po cifra, dok god ima neobradjenih
27        cifara. */
28     while (n > 0) {
29         /* Izdvaja se poslednja cifra iz zapisa i ako je njena vrednost
30            paran broj, uvecava se za 1. */
31         cifra = n % 10;
32         if (cifra % 2 == 0)
33             cifra++;
34
35         /* Novi broj se formira tako sto se izdvojena cifra pomnozi
36            odgovarajucom tezinom (stepenom) pozicije i doda na tekuci
37            rezultat. */
38         rezultat += cifra * pozicija;
39
40         /* Uklanja se poslednja cifra broja. */
41         n /= 10;
42
43         /* Pozicija se mnozi sa 10. */
44         pozicija *= 10;
45     }
46
47     /* Ispisuje se izracunatu vrednost. */
48     printf("Rezultat: %d\n", rezultat);
49
50     return 0;
51 }
```

### Rešenje 2.3.26

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     /* Deklaracije potrebnih promenljivih. */
7     int x, pozicija, rezultat, cifra;
8     int znak = 1;
9
10    /* Ucitava se vrednost polaznog broja. */
11    printf("Unesite broj: ");
12    scanf("%d", &x);
13
14    /* Ako je broj negativan, uzima se njegova apsolutna vrednost
15     i azurira se vrednost znaka broja. */
16    if (x < 0) {
17        x = abs(x);
18        znak = -1;
19    }
20
21    /* Pozicija oznacava tezinu trenutne cifre rezultata i moze imati
22     vrednosti 1, 10, 100, ... */
23    pozicija = 1;
24    rezultat = 0;
25
26    /* Ideja: u rezultatu se zadrzavaju cifre jedinice, stotine,...
27     Na primer, x=12345
28     Pre petlje: pozicija = 1, rezultat = 0
29     1. iteracija:
30     cifra = 5, rezultat = 0+5*1=5, x = 123, pozicija = 10
31     2. iteracija:
32     cifra = 3, rezultat = 5+3*10 = 35, x = 12, pozicija = 100
33     3. iteracija:
34     cifra = 4, rezultat = 35+4*100, x = 1, pozicija = 1000
35     Petlja se zavrшава jer x ima vrednost 0. */
36    while (x > 0) {
37        /* Izdvajanje poslednje cifre. */
38        cifra = x % 10;
39
40        /* Rezultat se uvecava za vrednost cifre pomnozene sa vrednoscu
41         tezine njene pozicije u broju. */
42        rezultat += cifra * pozicija;
43
44        /* Iz polaznog broja se uklanjaju poslednje dve cifre jer u
45         rezultatu treba da ostane svaka druga cifra polaznog
46         broja.*/
47        x /= 10;
48
49        /* Pozicija se mnozi sa 10, kako bi imala ispravnu vrednost u
50         sledecoj iteraciji. */
51    }
```

```

51     pozicija *= 10;
52 }
53
54 /* Ispis rezultata */
55 printf("Rezultat: %d\n", znak * rezultat);
56
57 return 0;
58 }

```

### Rešenje 2.3.27

```

1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int n;
7      int c1, c2, c3;
8      int pozicija, rezultat;
9
10     /* Ucitava se vrednost broja n. */
11     printf("Unesite broj: ");
12     scanf("%d", &n);
13
14     /* Vrsi se provera ispravnosti ulaza. */
15     if (n <= 0) {
16         printf("Greska: neispravan unos.\n");
17         return -1;
18     }
19
20     /* Ako broj nema bar tri cifre, rezultat ima vrednost unetog
21        broja. */
22     if(n <= 99)
23     {
24         printf("Rezultat: %d\n", n);
25         return 0;
26     }
27
28     /* Izdvajaju se poslednje tri cifre polaznog broja. */
29     c1 = n%10;
30     c2 = (n/10)%10;
31     c3 = (n/100)%10;
32
33     /* Poslednja cifra se uvek nalazi u rezultatu jer ona nema
34        oba suseda. Zato rezultat inicijalizujemo na poslednju cifru,
35        a poziciju na 10. */
36     rezultat = c1;
37     pozicija = 10;
38
39     /* Petlja se izvrsava dok god broj ima bar tri cifre. */
40     while(n>99)

```

## 2 Kontrola toka

```
41 {
42     /* Proverava se da li c2 treba da se nadje u rezultatu. Ako
43        treba, rezultat se uvecava za vrednost cifre pomnozenu sa
44        vrednoscu tezine njene pozicije u rezultatu i tezina
45        pozicije se mnozi sa 10. */
46     if(c2 != c1 + c3)
47     {
48         rezultat += c2*pozicija;
49         pozicija *= 10;
50     }
51
52     /* Vrsi se pomeranje na sledece tri cifre polaznog broja.
53        Iz polaznog broja brisemo poslednju cifru. Prva i druga
54        cifra su vec izracunate, samo se vrsi njihovo premestanje
55        iz c2 i c3 u c1 i c2. Cifra c3 se racuna. */
56     n = n/10;
57     c1 = c2;
58     c2 = c3;
59     c3 = (n/100)%10;
60 }
61
62 /* Po završetku petlje, broj n je dvocifren i njegova cifra
63    desetica odgovara vodecoj cifri polaznog broja. Vodeca cifra
64    polaznog broja uvek treba da se nadje u rezultatu jer nema
65    oba suseda i iz tog razloga je dodajemo na tekuci rezultat. */
66 rezultat += (n/10)*pozicija;
67
68 /* Ispis rezultata. */
69 printf("%d\n", rezultat);
70
71 return 0;
72 }
```

### Rešenje 2.3.28

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      int x, x_kopija, x_obrnuto;
7
8      /* Ucitava se vrednost pocetnog broja. */
9      printf("Unesite broj: ");
10     scanf("%d", &x);
11
12     /* Uzima se apsolutna vrednost unetog broja. */
13     if (x < 0)
14         x = -x;
15
16     /* Racuna se broj koji se dobije kada se broju x obrnu cifre.
17
18     */
19 }
```

```

17     Na primer, od 12345 treba da se dobije 54321.
    Broj se obrće tako što se u svakoj iteraciji njegova vrednost
19 pomnoži sa 10 i doda mu se sledeća cifra polaznog broja.
    Za x_kopija=12345, x_obrnuto = 0
21     1. iteracija: x_obrnuto = 0*10 + 5 = 5, x_kopija = 1234
    2. iteracija: x_obrnuto = 5*10 + 4 = 54, x_kopija = 123,
23     3. iteracija: x_obrnuto = 54*10 + 3 = 543, itd.*/
x_kopija = x;
25 x_obrnuto = 0;
while (x_kopija != 0) {
27     x_obrnuto = x_obrnuto * 10 + x_kopija % 10;
    x_kopija /= 10;
29 }

31 /* Broj je palindrom ako je jednak broju koji se dobije
    obrtanjem njegovih cifara.
33     Npr. x = 12321, x_obrnuto je takodje 12321.*/
if (x == x_obrnuto)
35     printf("Broj je palindrom.\n");
else
37     printf("Broj nije palindrom.\n");
39 return 0;
}

```

### Rešenje 2.3.29

```

1  #include <stdio.h>

3  int main()
{
5     /* Deklaracija potrebnih promenljivih. */
    int n, i;
7     int fib1 = 0, fib2 = 1, fib3;

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

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

19    /* Ako je n=0, F[0] = 0, slicno ako je n=1 F[1] = 1. */
    if(n < 2){
21        printf("F[%d] = %d\n",n, n);
        return 0;
23    }
}

```

## 2 Kontrola toka

---

```
25 fib3 = fib1 + fib2;
26 for(i=2; i<n; i++) {
27     /* Vrsi se pomeranje na sledecu trojku. */
28     fib1 = fib2;
29     fib2 = fib3;
30     fib3 = fib1 + fib2;
31 }
32
33 /* Ispis rezultata. */
34 printf("F[%d] = %d\n", n, fib3);
35
36 return 0;
37 }
```

### Rešenje 2.3.30

```
#include<stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     int an;
7
8     /* Ucitava se vrednost prvog clana. */
9     printf("Unesite prvi clan:");
10    scanf("%d", &an);
11
12    /* Vrsi se provera ispravnosti ulaza. */
13    if (an <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return -1;
16    }
17
18    /* Dok se ne dodje do clana koji je 1, stampa se vrednost
19       trenutnog clana i vrsi se izracunavanje narednog, po
20       zadatoj formuli. */
21    while (an != 1) {
22        printf("%d ", an);
23
24        if (an % 2 != 0)
25            an = (3 * an + 1) / 2;
26        else
27            an = an / 2;
28    }
29
30    /* Na kraju se stampa i jedinica. */
31    printf("1\n");
32
33    return 0;
34 }
```



## Rešenje 2.3.31

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      /* Deklaracija potrebnih promenljivih. */
7      int format, i;
8      double sirina, duzina, nova_duzina;
9
10     /* Ucitava se format papira. */
11     printf("Unesite format papira: ");
12     scanf("%d", &format);
13
14     /* Vrsi se provera ispravnosti ulaza. */
15     if (format < 0) {
16         printf("Greska: neispravan unos.\n");
17         return -1;
18     }
19
20     /* duzina/sirina = 1/sqrt(2)
21        duzina*sirina = 1000mm x 1000mm
22        =>
23        duzina = sirina/sqrt(2)
24        duzina*sirina = 1000mm x 1000mm
25        =>
26        sirina*sirina/sqrt(2) = 1000*1000
27        sirina*sirina = sqrt(2) * 1000 * 1000
28        sirina = sqrt(sqrt(2) * 1000 * 1000)
29        =>
30        duzina = sirina/sqrt(2) */
31     sirina = sqrt(1000 * 1000 * sqrt(2));
32     duzina = sirina / sqrt(2);
33
34     /* U petlji se racunaju duzina i sirina za uneti format. */
35     for (i = 1; i <= format; i++) {
36         nova_duzina = sirina / 2;
37         sirina = duzina;
38         duzina = nova_duzina;
39     }
40
41     /* Ispis rezultata. Napomena: duzina i sirina celi brojevi. */
42     printf("%d %d\n", (int) duzina, (int) sirina);
43
44     return 0;
45 }
```

## Rešenje 2.3.32

```
1 #include <stdio.h>
3 int main()
4 {
5     char c;
7     /* I nacin ucitavanja:
8        U samom uslovu petlje se vrši ucitavanje jednog karaktera,
9        njegovo smestanje u promenljivu c i provera da li je ucitani
10       karakter tacka. Zgrade oko (c=getchar()) su obavezne jer
11       relacioni operator != ima veci prioritet od dodele i kada ne
12       bi postojale zgrade, redosled operacija bi bio:
13       (c = (getchar() != '.')), sto znaci da bi se u c smestio
14       rezultat poredjenja, odnosno 0 ili 1. */
15     while ((c = getchar()) != '.') {
16         /* Proveravaju se uslovi i vrši se ispis odgovarajuceg
17            karaktera.*/
18         if (c >= 'A' && c <= 'Z')
19             putchar(c + 'a' - 'A');
20         else if (c >= 'a' && c <= 'z')
21             putchar(c - 'a' + 'A');
22         else
23             putchar(c);
24     }
25
26     /*II nacin:
27     while(1) {
28         c = getchar();
29         if(c == '.')
30             break;
31
32         if (c >= 'A' && c <= 'Z')
33             putchar(c + 'a' - 'A');
34         else if (c >= 'a' && c <= 'z')
35             putchar(c - 'a' + 'A');
36         else
37             putchar(c);
38     } */
39     return 0;
41 }
```

### Rešenje 2.3.33

```
1 #include <stdio.h>
3 int main()
4 {
5     /* Deklaracije i inicijalizacije. */
6     char c;
7     int broj_velikih = 0, broj_malih = 0;
```

```

9      int broj_cifara = 0, suma_cifara = 0, broj_belina = 0;

11     /* Petlja se završava kada korisnik zada konstantu oznaku za kraj
        ulaza (konstanta EOF čija je vrednost -1). Ova konstanta se
        zadaje kombinacijom tastera CTRL+D. */
13     while ((c = getchar()) != EOF) {
        if (c >= 'A' && c <= 'Z')
15         broj_velikih++;
        else if (c >= 'a' && c <= 'z')
17         broj_malih++;
        else if (c >= '0' && c <= '9') {
19             broj_cifara++;
            suma_cifara = suma_cifara + c - '0';
21         }
        else if (c == '\t' || c == '\n' || c == ' ')
23             broj_belina++;
    }

25     /* Ispis rezultata. */
27     printf("velika: %d, mala: %d\n", broj_velikih, broj_malih);
    printf("cifre: %d, beline: %d\n", broj_cifara, broj_belina);
29     printf("suma cifara: %d\n", suma_cifara);

31     return 0;
}

```

### Rešenje 2.3.34

```

1  #include <stdio.h>

3  int main()
{
5     /* Deklaracija i inicijalizacija potrebnih promenljivih. */
    int n, i;
7     int broj_a = 0, broj_e = 0, broj_i = 0, broj_o = 0, broj_u = 0;
    char c;

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

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

19    /* Kako je korisnik nakon unosa broja n uneo oznaku za novi red,
        potrebno je preskociti taj novi red jer bi u suprotnom on bio
        ucitan kao prvi od n karaktera (oznaka za novi red je
21        regularan karakter kao sto je to 'a' ili ' ').*/
23

```

```
25     getchar();

26     /* Ucitavaju se karakteri i broje se samoglasnici. */
27     for (i = 0; i < n; i++) {
28         scanf("%c", &c);

29         switch (c) {
30             case 'a':
31             case 'A':
32                 broj_a++;
33                 break;
34             case 'e':
35             case 'E':
36                 broj_e++;
37                 break;
38             case 'i':
39             case 'I':
40                 broj_i++;
41                 break;
42             case 'o':
43             case 'O':
44                 broj_o++;
45                 break;
46             case 'u':
47             case 'U':
48                 broj_u++;
49                 break;
50         }
51     }
52 }

53
54 /* Ispis rezultata. */
55 printf("Samoglasnik a: %d\n", broj_a);
56 printf("Samoglasnik e: %d\n", broj_e);
57 printf("Samoglasnik i: %d\n", broj_i);
58 printf("Samoglasnik o: %d\n", broj_o);
59 printf("Samoglasnik u: %d\n", broj_u);

60
61 return 0;
62 }
```

### Rešenje 2.3.35

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     /* Deklaracija i inicijalizacija potrebnih promenljivih. */
7     int n, i;
8     int broj_Z = 0, broj_i = 0, broj_m = 0, broj_a = 0;
9     char novi_red, c;
```

```

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

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

21  /* Ucitavaju se karakteri. */
    for (i = 1; i <= n; i++) {
23      printf("Unestite %d. karakter: ", i);

25      /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
        pa tek posle procitane beline se cita uneti karakter. */
27      scanf("%c%c", &novi_red, &c);

29      /* Obradjuje se ucitani karakter. */
        switch (c) {
31          case 'Z':
            broj_Z++;
33            break;
            case 'i':
            broj_i++;
35            break;
            case 'm':
            broj_m++;
37            break;
            case 'a':
            broj_a++;
41            break;
43        }
    }

45  /* Ako su svi brojacii razliciti od nule, rec "Zima" se moze
47  napisati pomocu unetih karaktera. */
    if (broj_Z && broj_i && broj_m && broj_a)
49      printf("Moze se napisati rec Zima.\n");
    else
51      printf("Ne moze se napisati rec Zima.\n");

53  return 0;
}

```

### Rešenje 2.3.36

```

1  #include <stdio.h>

3  int main()

```

```
{
5  /* Deklaracije potrebnih promenljivih. */
   int n, i;
7  int suma_kubova;

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

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

19  /* Racuna se suma kubova svih brojeva iz intervala [1,n]. */
   suma_kubova = 0;
21  for (i = 1; i <= n; i++)
       suma_kubova += i * i * i;

23  /* Ispis rezultata. */
25  printf("Suma kubova: %d\n", suma_kubova);

27  return 0;
}
```

Rešenje 2.3.37      Rešenje je analogno rešenju zadatka 2.3.36.

### Rešenje 2.3.38

```
1  #include <stdio.h>

3  int main()
   {
5     /* Deklaracija potrebnih promenljivih. */
       int n, i;
       float x, suma, x_i;

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

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

19     /* Vrednost sume se inicijalizuje na nulu, a vrednost x^i
       na x. */
   }
```

```
21 suma = 0;
22 x_i = x;
23
24 /* Promenljiva x^i ima vrednosti [x, x^2, ..., x^n].
25    Vrednost sume se u svakoj iteraciji uvecava za i*x^i. */
26 for (i = 1; i <= n; i++) {
27     suma += i * x_i;
28     x_i *= x;
29 }
30
31 /* Ispis rezultata. */
32 printf("S=%f\n", suma);
33
34 return 0;
35 }
```

## Rešenje 2.3.39

```
1 #include <stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     int n, i;
7     float x, suma, x_i;
8
9     /* Ucitavaju se vrednosti x i n. */
10    printf("Unesite redom brojeve x i n: ");
11    scanf("%f %d", &x, &n);
12
13    /* Vrsi se provera ispravnosti ulaza. */
14    if (n <= 0) {
15        printf("Greska: neispravan unos.\n");
16        return -1;
17    }
18
19    /* Racuna se trazena suma. */
20    suma = 1;
21    x_i = x;
22    for (i = 1; i <= n; i++) {
23        suma += 1 / x_i;
24        x_i *= x;
25    }
26
27    /* Ispis rezultata. */
28    printf("S=%f\n", suma);
29
30    return 0;
31 }
```

### Rešenje 2.3.40

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     /* Deklaracije potrebnih promenljivih. */
7     int i;
8     float suma, clan;
9     float x, eps;
10
11     /* Ucitavaju se vrednosti x i eps. */
12     printf("Unesite x: ");
13     scanf("%f", &x);
14
15     printf("Unesite tacnost eps: ");
16     scanf("%f", &eps);
17
18     /* Inicijalizacija sume, prvog clana i brojaca. */
19     suma = 0;
20     clan = 1;
21     i = 1;
22
23     /* U svakoj iteraciji na sumu se dodaje prethodno izracunati
24     clan sume i zatim se racuna sledeci clan. Petlja se prekida
25     kada vrednost sledeceg clana postane manja ili jednaka eps. */
26     while (clan > eps) {
27         suma += clan;
28         clan = clan * x / i;
29         i++;
30     }
31
32     /* Ispis rezultata. */
33     printf("S=%f\n", suma);
34
35     return 0;
36 }
```

### Rešenje 2.3.41

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     /* Deklaracije potrebnih promenljivih. */
7     int i;
8     float suma;
9     float x, eps, clan;
```



```

10  /* Ucitavaju se vrednosti x i eps. */
12  printf("Unesite x: ");
13  scanf("%f", &x);
14
15  printf("Unesite tacnost eps: ");
16  scanf("%f", &eps);
17
18  /* Inicijalizacije. */
19  suma = 0;
20  clan = 1;
21  i = 1;
22
23  /* Kako clanovi sume mogu biti negativni, potrebno je posmatrati
24     apsolutnu vrednost clana. */
25  while (fabs(clan) > eps) {
26      suma += clan;
27
28      /* U svakoj iteraciji se racuna novi clan i mnozi se sa -1.
29         Na taj nacin se postize da je vrednost clana naizmenicno
30         pozitivna i negativna. */
31      clan = clan * x / i;
32      clan *= -1;
33
34      i++;
35  }
36
37  /* Ispis rezultata. */
38  printf("S=%f\n", suma);
39
40  return 0;
41 }

```

### Rešenje 2.3.42

```

1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      /* Deklaracije potrebnih promenljivih. */
7      int n, i;
8      double x, x_i, proizvod;
9
10     /* Ucitavaju se vrednosti x i n. */
11     printf("Unesite redom brojeve x i n: ");
12     scanf("%lf %d", &x, &n);
13
14     /* Vrsi se provera ispravnosti ulaza. */
15     if (n <= 0) {
16         printf("Greska: neispravan unos.\n");
17     }
18 }

```

## 2 Kontrola toka

---

```
17     return -1;
18 }
19
20 /* Racuna se trazeni proizvod. */
21 x_i = 1;
22 proizvod = 1;
23 for (i = 0; i < n; i++) {
24     x_i *= x;
25     proizvod *= 1 + cos(x_i);
26 }
27
28 /* Ispis rezultata. */
29 printf("P = %lf\n", proizvod);
30
31 return 0;
32 }
```

### Rešenje 2.3.43

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracije potrebnih promenljivih. */
6      int n, i;
7      double razlomak;
8
9      /* Ucitava se vrednost broja n. */
10     printf("Unesite broj n: ");
11     scanf("%d", &n);
12
13     /* Vrsi se provera ispravnosti ulaza. */
14     if (n <= 0) {
15         printf("Greska: neispravan unos.\n");
16         return -1;
17     }
18
19     /* Razlomak se izracunava "od nazad", odnosno, krece se od
20        najnizeg razlomka 1/n i od njega se nadalje formira sledeci,
21        "visi" razlomak itd. Zavrшава se kada se stigne do koraka 0 +
22        1/R. */
23     razlomak = n;
24     for (i = n - 1; i >= 0; i--)
25         razlomak = i + 1 / razlomak;
26
27     /* Ispis rezultata. */
28     printf("R = %lf\n", razlomak);
29
30     return 0;
31 }
```

## Rešenje 2.3.44

```
1 #include <stdio.h>
3 int main () {
4     /* Deklaracije potrebnih promenljivih. */
5     int i , n;
6     float suma, x, clan;
7
8     /* Ucitavaju se vrednosti x i n. */
9     printf("Unesite redom brojeve x i n: ");
10    scanf("%f%d", &x, &n);
11
12    /* Vrsi se provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return -1;
16    }
17
18    /* Inicijalizacije. */
19    suma = 1;
20    clan = 1;
21    i = 2;
22
23    /* Racuna se trazena suma. */
24    while (i <= 2 * n) {
25        /* Svaki clan suma se od prethodnog clana razlikuje za
26            $x^2/(i*(i-1))$ . */
27        clan = clan * x * x / (i * (i - 1));
28        clan *= -1;
29        suma += clan;
30        i += 2;
31    }
32
33    /* Ispis rezultata. */
34    printf("S=%f\n", suma);
35
36    return 0;
37 }
```

## Rešenje 2.3.45

```
1 #include <stdio.h>
3 int main()
4 {
5     /* Deklaracije potrebnih promenljivih. */
6     int n, i;
7     double clan, proizvod = 1;
```

## 2 Kontrola toka

---

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

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

19  /* Racuna se trazeni proizvod. */
   clan = 1;
21  for (i = 2; i <= n; i++) {
      clan = clan / i;
23      proizvod *= 1 + clan;
   }

25  /* Ispis rezultata. */
27  printf("P = %lf\n", proizvod);

29  return 0;
}
```

### Rešenje 2.3.46

```
1  #include <stdio.h>

3  int main()
   {
5      /* Deklaracija potrebnih promenljivih. */
      int n, i;
      long int clan, suma = 0;

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

13     /* Vrsi se provera ispravnosti ulaza. */
      if (n < 5 || n % 2 == 0) {
15         printf("Greska: neispravan unos.\n");
         return -1;
17     }

19     /* Izracunava se trazena suma. */
      clan = -1 * 3;
21     for (i = 5; i <= n; i += 2) {
         clan = -1 * clan * i;
23         suma += clan;
     }

25     /* Ispis rezultata. */
```

```

27     printf("S = %ld\n", suma);
29     return 0;
}

```

### Rešenje 2.3.47

```

#include <stdio.h>

2  int main()
4  {
    /* Deklaracije potrebnih promenljivih. */
    int n, i;
    double rezultat;
    double x, a;

10     /* Ucitavaju se vrednosti ulaznih promenljivih. */
    printf("Unesite brojeve x i a: ");
12     scanf("%lf%lf", &x, &a);
    printf("Unesite broj n: ");
14     scanf("%d", &n);

16     /* Vrsi se provera ispravnosti ulaza. */
    if (n <= 0) {
18         printf("Greska: neispravan unos.\n");
        return -1;
20     }

22     /* Racuna se vrednost zadatog izraza. Krece se od
        rezultat = (x+a)^2 i ide se ka spolja.
24     Svaki put vrednost rezultata treba zameniti sa
        (rezultat + a)^2. */
    rezultat = x;
    for (i = 0; i < n; i++)
28         rezultat = (rezultat + a) * (rezultat + a);

30     /* Ispis rezultata. */
    printf("Izraz = %lf\n", rezultat);
32     return 0;
34 }

```

### Rešenje 2.3.48

Rešenje (a)

```

#include <stdio.h>

2

```

## 2 Kontrola toka

---

```
1 int main()
2 {
3     /* Deklaracije potrebnih promenljivih. */
4     unsigned int n, i, j;
5
6     /* Ucitava se vrednost broja n. */
7     printf("Unesite broj n: ");
8     scanf("%u", &n);
9
10    /* Ispis tablice mnozenja dimenzije n*n. */
11    for (i = 1; i <= n; i++) {
12        for (j = 1; j <= n; j++){
13            /* Vrednost svakog polja je proizvod vrste i kolone. */
14            printf("%3d ", i * j);
15        }
16        /* Na kraju svake vrste se ispisuje novi red. */
17        printf("\n");
18    }
19
20    return 0;
21 }
```

### Rešenje (b)

```
1 #include <stdio.h>
2
3 int main()
4 {
5     /* Deklaracije potrebnih promenljivih. */
6     unsigned int n, i, j;
7
8     /* Ucitava se vrednost broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Brojac koji broji koliko brojeva je ispisano u jednom redu. */
13    j = 0;
14    for (i = 1; i <= n * n; i++) {
15        printf("%3d ", i);
16
17        j++;
18        /* Kada je ispisano n brojeva u jednom redu, ispisuje se znak
19         za novi red, da bi ispis krenuo u novom redu i vrednost
20         brojaca j se postavlja na 0 jer u novom redu jos ni jedan
21         broj nije ispisan. */
22        if (j == n) {
23            j = 0;
24            printf("\n");
25        }
26    }
27 }
```

```
    return 0;
29 }
```

*Rešenje (c)*

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracije potrebnih promenljivih. */
6      unsigned int n, i, j;
7
8      /* Ucitava se vrednost broja n. */
9      printf("Unesite broj n: ");
10     scanf("%u", &n);
11
12     /* Ispis trazene tablice. */
13     for (i = 1; i <= n; i++) {
14         for (j = 0; j < n; j++)
15             if ((j + i) % n == 0)
16                 printf("%3d", n);
17             else
18                 printf("%3d", (j + i) % n);
19
20         printf("\n");
21     }
22
23     return 0;
24 }
```

*Rešenje (d)*

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracije potrebnih promenljivih. */
6      unsigned int n, i, j;
7
8      /* Ucitava se vrednost broja n. */
9      printf("Unesite broj n: ");
10     scanf("%u", &n);
11
12     /* Ispis trazenog trougla. */
13     for (i = 0; i < n; i++) {
14         for (j = 0; j < n - i; j++)
15             printf("(%d, %d)", i, j);
16
17         printf("\n");
18     }
19 }
```

```
18     }
20     return 0;
    }
```

### Rešenje 2.3.49

#### Rešenje (a)

```
#include <stdio.h>

2 int main()
3 {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitava se vrednost broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11     /* Kvadrat predstavlja tabelu sa n vrsta gde svaka vrsta sadrzi
12        n polja, a svako polje je isto i predstavlja karakter *. */
13     for (i = 0; i < n; i++) {
14         for (j = 0; j < n; j++)
15             printf("*");
16         printf("\n");
17     }
18
19     return 0;
20 }
```

#### Rešenje (b)

```
1 #include <stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */
6     unsigned int n, i, j;
7
8     /* Ucitava se vrednost broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter *,
13       a untrasnjost kvadrata je karakter blanko. */
14    for (i = 0; i < n; i++) {
15        for (j = 0; j < n; j++){
```



```

17     /* Provera se da li je u pitanju ivica. */
18     if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
19         printf("*");
20     else
21         printf(" ");
22 }
23 printf("\n");
24 }
25 return 0;
26 }

```

Rešenje (c)

```

#include <stdio.h>

2 int main()
3 {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;

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

9     /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter *,
10    a untrasnjost kvadrata je karakter blanko osim na mestima na
11    kojima je glavna dijagonala. */
12    for (i = 0; i < n; i++) {
13        for (j = 0; j < n; j++){
14            /* Provera da li je ivica ili glavna dijagonala. */
15            if (j == 0 || j == n - 1 || i == 0 || i == n - 1 || i == j)
16                printf("*");
17            else
18                printf(" ");
19        }
20        printf("\n");
21    }
22    return 0;
23 }

```

### Rešenje 2.3.50

```

1 #include <stdio.h>
2
3 int main()
4 {
5     /* Deklaracija potrebnih promenljivih. */

```

## 2 Kontrola toka

```
    unsigned int n, i, j;

7
    /* Ucitava se vrednost broja n. */
9    printf("Unesite broj n: ");
    scanf("%u", &n);

11
    /* Veliko slovo X se dobija tako sto se na dijagonalama kvadrata
13     ispisuju karakteri *, a na ostalim mestima blanko. */
    for (i = 0; i < n; i++) {
15        for (j = 0; j < n; j++){
17            /* Provera da li je mesto glavne ili sporedne dijagonale. */
            if (i == j || i + j == n - 1)
                printf("*");
19            else
                printf(" ");
21        }
        printf("\n");
23    }

25    return 0;
}
```

### Rešenje 2.3.51

```
1 #include <stdio.h>

3 int main()
{
5     /* Deklaracija potrebnih promenljivih. */
    unsigned int n, i, j;

7
    /* Ucitava se vrednost broja n. */
9    printf("Unesite broj n: ");
    scanf("%u", &n);

11
    /* Iscrtava se znak plus tako sto se na pozicijama koje
13     odgovaraju sredisnjoj vrsti i sredisnjoj kolini ispisuje
    +, a na ostalim pozicijama se ispisuje blanko. */
15    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
25    return 0;
}
```

## Rešenje 2.3.52

Rešenje (a)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      unsigned int n, i, j;
7
8      /* Ucitava se vrednost broja n. */
9      printf("Unesite broj n: ");
10     scanf("%u", &n);
11
12     /* Iscrtava se trazeni trougao. */
13     for (i = 0; i < n; i++) {
14         for (j = 0; j < n - i; j++)
15             printf("*");
16         printf("\n");
17     }
18
19     return 0;
20 }
```

Rešenje (b)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      unsigned int n, i, j;
7
8      /* Ucitava se vrednost broja n. */
9      printf("Unesite broj n: ");
10     scanf("%u", &n);
11
12     /* Iscrtava se trazeni trougao. */
13     for (i = 0; i < n; i++) {
14         for (j = 0; j <= i; j++)
15             printf("*");
16         printf("\n");
17     }
18
19     return 0;
20 }
```

Rešenje (c)

## 2 Kontrola toka

---

```
#include <stdio.h>

2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     unsigned int n, i, j;

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

    /* Iscrtava se trazeni trougao. */
12    for (i = 0; i < n; i++) {
14        /* Prvo se ispisuju beline koje prethode karakterima *. */
        for (j = 0; j < i; j++)
16            printf(" ");
        /* Posle belina se ispisuje potreban broj karaktera *. */
18        for (j = 0; j < n - i; j++)
            printf("*");
20        printf("\n");
    }
22
24    return 0;
}
```

Rešenje (d)

```
#include <stdio.h>

2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     unsigned int n, i, j;

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

    /* Iscrtava se trazeni trougao. */
12    for (i = 0; i < n; i++) {
14        /* Prvo se ispisuju beline koje prethode karakterima *. */
        for (j = 0; j < n - i - 1; j++)
16            printf(" ");
        /* Posle belina se ispisuje potreban broj karaktera *. */
18        for (j = 0; j <= i; j++)
            printf("*");
20        printf("\n");
    }
22
    return 0;
}
```

24 | }

*Rešenje (e)*

```

#include <stdio.h>

2 int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     unsigned int n, i, j;

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

    /* Iscrtava se gornji deo trazenog trougla. */
12     for (i = 0; i < n; i++) {
        /* Prvo se ispisuju beline koje prethode karakterima *. */
14         for (j = 0; j < n - i - 1; j++)
            printf(" ");
        /* Posle belina se ispisuje potreban broj karaktera *. */
18         for (j = 0; j <= i; j++)
            printf("*");
20         printf("\n");
    }

22     /* Iscrtava se donji deo trazenog trougla. */
24     for (i = 1; i < n; i++) {
        /* Prvo se ispisuju beline koje prethode karakterima *. */
26         for (j = 0; j < i; j++)
            printf(" ");
        /* Posle belina se ispisuje potreban broj karaktera *. */
28         for (j = 0; j < n - i; j++)
            printf("*");
30         printf("\n");
    }

32 }

34 return 0;
}

```

*Rešenje (f)*

```

1 #include <stdio.h>

3 int main()
{
5     /* Deklaracije potrebnih promenljivih. */
    unsigned int n, i, j;
7     char c, novi_red;

```

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

13  /* Ucitava se karakter koji ce se koristiti za iscrtavanje.
   Napomena: voditi racuna da treba preskociti novi red koji
15  korisnik zadaje nakon unosa broja n. */
   printf("Unesite karakter c: ");
17  scanf("%c%c", &novi_red, &c);

19  /* Iscrtavanje trazenog trougla. Iscrtavaju se samo ivice
   trougla, ostalo se popunjava belinama. */
21  for (i = 0; i < n; i++) {
       for (j = 0; j <= i; j++)
23         if (i == n - 1 || j == 0 || j == i)
             printf("%c", c);
25         else
             printf(" ");
27         printf("\n");
   }
29
31  return 0;
}
```

### Rešenje 2.3.53

#### Rešenje (a)

```
1  #include <stdio.h>

3  int main()
   {
5     /* Deklaracija potrebnih promenljivih. */
       unsigned int n, i, j;

7

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

11

13    /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
       for (i = 0; i < n; i++) {
           /* Prvo se ispisuju beline koje prethode karakterima *. */
15         for (j = 0; j < n - i - 1; j++)
             printf(" ");
17         /* Posle belina se ispisuje potreban broj karaktera *. */
           for (j = 0; j < 2 * i + 1; j++)
19             printf("*");
           printf("\n");
21     }
}
```

```
23     return 0;
    }
```

*Rešenje (b)*

```
#include <stdio.h>

2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     unsigned int n;
    int i, j;

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

12    /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
14    izracunavanja koliko zvezdica i praznina je potrebno ispisati
    u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
16    iteraciji petlje. */
    for (i = n - 1; i >= 0; i--) {
18        /* Prvo se ispisuju beline koje prethode karakterima *. */
        for (j = 0; j < n - i - 1; j++)
20            printf(" ");
        /* Posle belina se ispisuje potreban broj karaktera *. */
22        for (j = 0; j < 2 * i + 1; j++)
            printf("*");
24        printf("\n");
    }

26    return 0;
28 }
```

*Rešenje (c)*

```
#include <stdio.h>

2
int main()
4 {
    /* Deklaracija potrebnih promenljivih. */
6     unsigned int n;
    int i, j;

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

12
```

```
14  /* Slika se crta iz dva dela. */
16  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
17  for (i = 0; i < n; i++) {
18      /* Prvo se ispisuju beline koje prethode karakterima *. */
19      for (j = 0; j < n - i - 1; j++)
20          printf(" ");
21      /* Posle belina se ispisuje potreban broj karaktera *. */
22      for (j = 0; j < 2 * i + 1; j++)
23          printf("*");
24      printf("\n");
25  }
26
27  /* II deo: crtanje donjeg trougla. Kako je prvi red donjeg
28     trougla vec ispisan (poslednji red gornjeg trougla), potrebno
29     je naciniti jednu iteraciju manje. */
30  for (i = n - 2; i >= 0; i--) {
31      /* Prvo se ispisuju beline koje prethode karakterima *. */
32      for (j = 0; j < n - i - 1; j++)
33          printf(" ");
34      /* Posle belina se ispisuje potreban broj karaktera *. */
35      for (j = 0; j < 2 * i + 1; j++)
36          printf("*");
37      printf("\n");
38  }
39
40  return 0;
}
```

### Rešenje (d)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      unsigned int n;
7      int i, j;
8
9      /* Ucitava se vrednost broja n. */
10     printf("Unesite broj n: ");
11     scanf("%u", &n);
12
13     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
14     for (i = 0; i < n; i++) {
15         /* Prvo se ispisuju beline koje prethode karakterima *. */
16         for (j = 0; j < n - i - 1; j++)
17             printf(" ");
18         /* Posle belina se ispisuje sam trougao. Ako je brojac na
19            ivici onda se ispisuje karakter *, a inace praznina.
20            Takodje, proverava se da li se ispisuje poslednji red (i==n)
21            */
22     }
```



```

    i u njemu se ispisuje svaki drugi put *, a inace praznina. */
22   for (j = 0; j < 2 * i + 1; j++)
    if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
24       printf("*");
    else
26       printf(" ");
    printf("\n");
28 }

30 return 0;
}

```

## Rešenje (c)

```

1  #include <stdio.h>

3  int main()
{
5   /* Deklaracija potrebnih promenljivih. */
   unsigned int n;
7   int i, j;

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

13  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
   for (i = 0; i < n; i++) {
15     /* Prvo se ispisuju beline koje prethode karakterima *. */
     for (j = 0; j < n - i - 1; j++)
17         printf(" ");
     /* Posle belina se ispisuje sam trougao. */
19     for (j = 0; j < 2 * i + 1; j++)
        if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
21         printf("*");
        else
23         printf(" ");
     printf("\n");
25 }

27 /* II deo: crtanje donjeg trougla. Kako je prvi red donjeg
   trougla vec ispisan (poslednji red gornjeg trougla), potrebno
29   je naciniti jednu iteraciju manje. */
   for (i = n - 2; i >= 0; i--) {
31     /* Prvo se ispisuju beline koje prethode karakterima *. */
     for (j = 0; j < n - i - 1; j++)
33         printf(" ");
     /* Posle belina se ispisuje potreban broj karaktera *. */
35     for (j = 0; j < 2 * i + 1; j++)
        if (j == 0 || j == 2 * i)
37         printf("*");

```

```
        else
39         printf(" ");
        printf("\n");
41     }

43     return 0;
}
```

### Rešenje 2.3.54

```
1  #include <stdio.h>

3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      unsigned int n;
7      int i, j;

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

13     /* Strelica se moze posmatrati kao spojena dva pravougla trougla
14        kojima se ispisuje hipotenuza i jedna kateta. */

15     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
16     for (i = 0; i < n; i++) {
17         for (j = 0; j <= i; j++)
18             /* Proverava se da li se ispisuje karakter na hipotenuzi
19                (j == i) ili da se ispisuje poslednji red (i == n-1). */
20             if (j == i || i == n - 1)
21                 printf("*");
22             else
23                 printf(" ");
24             printf("\n");
25         }

27     /* II deo: crtanje donjeg dela slike, odnosno donji trougao.
28        Brojac i odredjuje koji red donjeg trougla se trenutno iscrtava.
29        Kako je prvi red donjeg trougla vec iscrtan (to je poslednji
30        red gornjeg trougla), brojac se postavlja na 1. */
31     for (i = 1; i < n; i++) {
32         for (j = 0; j < n - i; j++)
33             /* Provera da li se ispisuje hipotenuza. */
34             if (j == n - i - 1)
35                 printf("*");
36             else
37                 printf(" ");
38             printf("\n");
39         }
40     }
41 }
```

```
43     return 0;
    }
```

### Rešenje 2.3.55

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Deklaracija potrebnih promenljivih. */
6      unsigned int n;
7      int i, j, k;
8
9      /* Ucitava se vrednost broja n. */
10     printf("Unesite broj n: ");
11     scanf("%u", &n);
12
13     /* Brojac j odredjuje koliko ukupno karaktera (praznina i
14     karakteri *) u svakom redu se ispisuje. U svakom drugom redu
15     ovaj broj se povecava za 2. Na pocetku je 1 (jer se ispisuje
16     samo jedna zvezda). */
17     j = 1;
18
19     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
20     for (i = 1; i <= n; i++) {
21         /* U svakom drugom redu broj karaktera koji treba da se
22         ispisu se uvecava za 2. */
23         if (i % 2 == 0)
24             j += 2;
25
26         /* Ispisuje se j karaktera. */
27         for (k = 0; k < j; k++)
28             /* U svakom parnom redu se naizmenicno
29             ispisuju * i praznina. */
30             if (i % 2 == 0) {
31                 if (k % 2 == 0)
32                     printf("*");
33                 else
34                     printf(" ");
35             }
36             else
37             {
38                 /*U svakom neparnom redu se ispisuju samo *. */
39                 printf("*");
40             }
41         printf("\n");
42     }
43
44     return 0;
45 }
```

### Rešenje 2.3.56

```
2  #include <stdio.h>
4  int main()
6  {
8      /* Deklaracija potrebnih promenljivih. */
10     unsigned int n, m;
12     int i, j;

14     /* Ucitavaju se dimenzije slike. */
16     printf("Unesite brojeve n i m: ");
18     scanf("%u%u", &n, &m);

20     /* Brojac i odredjuje koji red slike se trenutno ispisuje.
22     Ukupno ima m redova. */
24     for (i = 1; i <= m; i++) {
26         /* Brojac j oznacava koja kolona se trenutno ispisuje.
28         Za svaki kvadrat se racuna duzina bez poslednje ivice.
30         Kvadrat je sastavljen od (m-1) zvezdice i (m-1) praznine
32         (praznine se nalaze izmedju zvezdica). Znac i ukupna duzina
34         je 2*(m-1) karakter, a kako ima n kvadrata plus jedna kolona
36         za najdesniju ivicu, duzina je n*2*(m-1) + 1. */
38         for (j = 0; j <= n * 2 * (m - 1); j++)
40             /* Provera da li se ispisuje prvi ili poslednji red. */
42             if (i == 1 || i == m)
44                 /* Naizmenicno se ispisuje * i praznina. */
46                 if (j % 2 == 0)
48                     printf("*");
50                 else
52                     printf(" ");
54             else
56                 /* Na ivicama kvadrata se iscrtavaju * a na ostalim mestima
58                 beline. */
60                 if (j % (2 * (m - 1)) == 0)
62                     printf("*");
64                 else
66                     printf(" ");

68         printf("\n");
70     }

72     return 0;
74 }
```

### Rešenje 2.3.57

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

```

5  {
6  /* Deklaracija potrebnih promenljivih. */
7  unsigned int n;
8  int i, j;
9
10 /* Ucitava se vrednost broja n. */
11 printf("Unesite broj n: ");
12 scanf("%u", &n);
13
14 /* Romb se crta crtanjem dva spojena trougla koji se nezavisno
15    iscrtavaju. */
16
17 /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
18 for (i = 0; i < n; i++) {
19     /* Prvo se ispisuju * koje prethode karakterima -. */
20     for (j = 0; j < n - i; j++)
21         printf("*");
22     /* Potom se ispisuju karakteri -. */
23     for (j = 0; j < 2 * i; j++)
24         printf("-");
25     /* Potom se ispisuju * koje su nakon karaktera -. */
26     for (j = 0; j < n - i; j++)
27         printf("*");
28     printf("\n");
29 }
30
31 /* II deo: crtanje donjeg trougla. Kako je prvi red donjeg
32    trougla vec ispisan (poslednji red gornjeg trougla), potrebno
33    je naciniti jednu iteraciju manje. */
34 for (i = n - 2; i >= 0; i--) {
35     /* Prvo se ispisuju * koje prethode karakterima -. */
36     for (j = 0; j < n - i; j++)
37         printf("*");
38     /* Potom se ispisuju karakteri -. */
39     for (j = 0; j < 2 * i; j++)
40         printf("-");
41     /* Potom se ispisuju * koje su nakon karaktera -. */
42     for (j = 0; j < n - i; j++)
43         printf("*");
44     printf("\n");
45 }
46
47 return 0;
48 }

```

### Rešenje 2.3.58

```

1  #include <stdio.h>
2
3  int main()
4  {

```

## 2 Kontrola toka

---

```
5  /* Deklaracija potrebnih promenljivih. */
   unsigned int n;
7  int i, j;

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

13 /* Slika se sastoji iz dva dela, trougla i kvadrata i svaki deo
   se nezavisno iscrtava. */

15
17 /* I deo: crtanje trougla (krova). */
   for (i = 0; i < n - 1; i++) {
       /* Prvo se ispisuju beline koje prethode karakterima *. */
19       for (j = 0; j < n - i - 1; j++)
           printf(" ");

21       /* Posle belina se ispisuje sam trougao.*/
23       for (j = 0; j < 2 * i + 1; j++)
           if (j == 0 || j == 2 * i)
25               printf("*");
           else
27               printf(" ");
       printf("\n");
29   }

31 /* II deo: crtanje kvadrata. Da bi iscrtavanje bilo lakse
   istovremeno se ispisuju dva karaktera. */
33   for (i = 0; i < n; i++) {
       for (j = 0; j < n; j++)
35           /* Provera da li je ivica. */
           if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
37               printf("* ");
           else
39               printf(" ");
       printf("\n");
41   }

43   return 0;
}
```

### Rešenje 2.3.59

```
1  #include <stdio.h>

3  int main()
   {
5      /* Deklaracija potrebnih promenljivih. */
       unsigned int n;
7      int i, j;
```

```

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

13  /* Prva petlja oznacava broj 'serija' koje ce se ispisati.
   Na primer, za n=5, prva serija je 1 2 3 4 5, druga serija je
15  2 3 4 i treca serija je 3.
   Kako se u svakoj sledecoj seriji broj brojeva smanjuje za 2,
17  do 0 karaktera u seriji se dolazi posle n/2 koraka, ali
   zaokruženo navise (5/2 = 2.5 --> 3), a to je isto sto i
19  celobrojno (n+1)/2. */
   for (i = 1; i <= (n + 1) / 2; i++) {
21     /* U svakoj seriji se ispisuju brojevi izmedju i i n-i+1. */
     for (j = i; j <= n + 1 - i; j++)
23         printf("%d ", j);
   }
25
   return 0;
27 }

```

### Rešenje 2.3.60

```

1  #include <stdio.h>

3  int main()
   {
5     /* Deklaracija potrebnih promenljivih. */
     unsigned int n;
     int i, j;

7

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

13    /* Brojac i je redni broj vrste koja se ispisuje. */
     for (i = 1; i <= n; i++) {
15         /* U svakoj vrsti se ispisuju brojevi izmedju 1 i n,
           sa korakom i. */
17         for (j = 1; j <= n; j+=i)
             printf("%d ", j);

19         printf("\n");
21     }

23     return 0;
   }

```

## 2.5 Funkcije

**Zadatak 2.5.1** Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji učitava tri cela broja i ispisuje njihov minimum.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 19 8 14  
|| Minimum: 8
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: -6 11 -12  
|| Minimum: -12
```

[Rešenje 2.5.1]

**Zadatak 2.5.2** Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja  $x$ . Napisati program koji učitava jedan realan broj i ispisuje njegov razlomljeni deo na šest decimala.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 8.235  
|| Razlomljeni deo: 0.235000
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -5.11  
|| Razlomljeni deo: 0.110000
```

[Rešenje 2.5.2]

**Zadatak 2.5.3** Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja  $n$ . Napisati program koji učitava ceo pozitivan broj  $k$  i ispisuje zbir delilaca svakog broja od 1 do  $k$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj k: 6  
|| 1 3 4 7 6 12
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj k: -2  
|| Greska: neispravan unos.
```

[Rešenje 2.5.3]

**Zadatak 2.5.4** Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva broja  $x$  i  $n$  utvrđuje da li je  $x$  neki stepen broja  $n$ . Ukoliko jeste, funkcija vraća izložilac stepena, a u suprotnom vraća  $-1$ . Napisati program koji učitava dva neoznačena broja i ispisuje da li vrednost prvog broja odgovara vrednosti nekog stepena drugog broja.



*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva broja: 81 3  
| Jeste: 81 = 3^4
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva broja: 162 11  
| Broj 162 nije stepen broja 11.
```

[Rešenje 2.5.4]

**Zadatak 2.5.5** Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najvećeg zajedničkog delioca primenom Euklidovog algoritma. Napisati program koji učitava dva cela broja i ispisuje vrednost njihovog najvećeg zajedničkog delioca.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva cela broja: 1024 832  
| Najveci zajednicki delilac: 64
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva cela broja: -900 112  
| Najveci zajednicki delilac: 4
```

[Rešenje 2.5.5]

**Zadatak 2.5.6** Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato  $n$  vraća zbir recipročnih vrednosti brojeva od 1 do  $n$ . Napisati program koji učitava ceo pozitivan broj  $n$  i ispisuje odgovarajući zbir zaokružen na dve decimalne. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj n: 10  
| Zbir reciprocnih: 2.93
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj n: 100  
| Zbir reciprocnih: 5.19
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj n: -100  
| Greska: neispravan unos.
```

[Rešenje 2.5.6]

**Zadatak 2.5.7** Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj  $x$  pojavljuje u nizu brojeva koji se unose sve do unosa broja nula. Napisati program koji učitava vrednost broja  $x$  i ispisuje koliko puta se njegova vrednost pojavila u unetom nizu.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj x: 2.84  
|| Unesite brojeve:  
|| 8.13 2.84 5 21.6 2.84 11.5 0  
|| Broj pojavljivanja broja 2.84: 2
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj x: -1.17  
|| Unesite brojeve:  
|| -128.35 8.965 8.968 89.36 0  
|| Broj pojavljivanja broja -1.17: 0
```

[Rešenje 2.5.7]

**Zadatak 2.5.8** Broj je prost ako je deljiv samo sa 1 i sa samim sobom.

- Napisati funkciju `int prost(int x)` koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati jedinicu ako je broj prost i nulu u suprotnom.
- Napisati funkciju `void prvih_n_prostih(int n)` koja ispisuje prvih  $n$  prostih brojeva.
- Napisati funkciju `void prosti_brojevi_manji_od_n(int n)` koja ispisuje sve proste brojeve manje od broja  $n$ .

Napisati program koji učitava pozitivan ceo broj  $n$  i ispisuje prvih  $n$  prostih brojeva, kao i sve proste brojeve manje od  $n$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 5  
|| Prvih n prostih: 2 3 5 7 11  
|| Prosti manji od n: 2 3
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 2  
|| Prvih n prostih: 2 3  
|| Prosti manji od n: ne postoje
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -11  
|| Greska: neispravan unos.
```

[Rešenje 2.5.8]

**Zadatak 2.5.9** Rešiti sledeće zadatke korišćenjem funkcija.

- Zadatak 1.1.2 rešiti korišćenjem funkcija `int kvadrat(int x)` koja računa kvadrat datog broja i `int kub(int x)` koja računa kub datog broja.
- Zadatak 2.1.2 rešiti korišćenjem funkcije `float apsolutna_vrednost(float x)` koja izračunava apsolutnu vrednost datog broja.

- c) Zadatak 2.3.7 rešiti korišćenjem funkcije `float stepen(float x, int n)` koja računa vrednost  $n$ -tog stepena realnog broja  $x$ .
- d) Zadatak 2.3.29 rešiti korišćenjem funkcije `int fibonaci(int n)` koja računa  $n$ -ti element Fibonačijevog niza.

**Zadatak 2.5.10** Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji učitava ceo broj i ispisuje aritmetičku sredinju njegovih cifara zaokruženu na tri decimale.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 461
|| 3.667
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1001
|| 0.500
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -84723
|| 4.800
```

[Rešenje 2.5.10]

**Zadatak 2.5.11** Napisati funkciju `int sadrzi(int x, int c)` koja ispituje da li se cifra  $c$  nalazi u zapisu celog broja  $x$ . Funkcija treba da vrati jedinicu ako se cifra nalazi u broju, a nulu inače. Napisati program koji učitava jedan ceo broj i jednu cifru i u zavisnosti od toga da li se uneta cifra nalazi u zapisu unetog broja, ispisuje odgovarajuću poruku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: 17890 7
|| Cifra 7 se nalazi u zapisu broja 17890.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: 19 6
|| Cifra 6 se ne nalazi u zapisu broja 19.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: 17890 26
|| Greska: neispravan unos.
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: -1982 9
|| Cifra 9 se nalazi u zapisu broja -1982.
```

[Rešenje 2.5.11]

**Zadatak 2.5.12** Napisati funkciju `int broj_neparnih_cifara(int x)` koja određuje broj neparnih cifara u zapisu datog celog broja. Napisati program koji učitava cele brojeve sve do unosa broja nula i ispisuje broj neparnih cifara svakog unetog broja.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cele brojeve:
|| 2341
|| Broj neparnih cifara: 2
|| 78
|| Broj neparnih cifara: 1
|| 800
|| Broj neparnih cifara: 0
|| -99761
|| Broj neparnih cifara: 4
|| 0
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cele brojeve:
|| 987611
|| Broj neparnih cifara: 4
|| 135
|| Broj neparnih cifara: 3
|| -701
|| Broj neparnih cifara: 2
|| 602
|| Broj neparnih cifara: 0
|| -884
|| Broj neparnih cifara: 0
|| 79901
|| Broj neparnih cifara: 4
|| 0
```

[Rešenje 2.5.12]

**Zadatak 2.5.13** Napisati program za ispitivanje svojstava cifara datog celog broja.

- (a) Napisati funkciju `int sve_parne_cifre(int x)` koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati jedinicu ako su sve cifre broja parne, a nulu inače.
- (b) Napisati funkciju `int sve_cifre_jednake(int x)` koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati jedinicu ako su sve cifre broja jednake, a nulu inače.

Program učitava ceo broj `i` u zavisnosti od toga da li su navedena svojstva ispunjena ili ne, ispisuje odgovarajuću poruku.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 86422
|| Sve cifre broja su parne.
|| Cifre broja nisu jednake.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 55555
|| Broj sadrzi bar jednu neparnu cifru.
|| Cifre broja su jednake.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -88
|| Sve cifre broja su parne.
|| Cifre broja su jednake.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: -342
|| Broj sadrzi bar jednu neparnu cifru.
|| Cifre broja nisu jednake.
```

[Rešenje 2.5.13]

**Zadatak 2.5.14** Napisati funkciju `int ukloni(int n, int p)` koja menja broj  $n$  tako što iz njegovog zapisa uklanja cifru na poziciji  $p$ . Pozicije se broje sa desna na levo. Cifra jedinica ima poziciju 1. Napisati program koji učitava redni broj pozicije  $i$  zatim za cele brojeve koji se unose sve do unosa broja nula, ispisuje brojeve kojima je uklonjena cifra na poziciji  $p$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite poziciju: 3
Unesite broj: 1210
110
Unesite broj: 18
18
Unesite broj: 3856
356
Unesite broj: 0
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite poziciju: 1
Unesite broj: -9632
-963
Unesite broj: -2
0
Unesite broj: 246
24
Unesite broj: -52
-5
Unesite broj: 0
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite poziciju: 0
Greska: neispravan unos.
```

[Rešenje 2.5.14]

**Zadatak 2.5.15** Napisati funkciju `int zapis(int x, int y)` koja proverava da li se brojevi  $x$  i  $y$  zapisuju pomoću istih cifara. Funkcija treba da vrati jedinicu ako je uslov ispunjen, a nulu inače. Napisati program koji učitava dva cela broja i ispisuje da li je za njih pomenuti uslov ispunjen ili ne.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 251 125
Uslov je ispunjen.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 8898 9988
Uslov nije ispunjen.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: -7391 1397
Uslov je ispunjen.
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: -777 77
Uslov nije ispunjen.
```

[Rešenje 2.5.15]

**Zadatak 2.5.16** Napisati funkciju `int neopadajuce(int n)` koja ispituje da li su cifre datog celog broja u neopadajućem poretku. Funkcija treba da vrati jedinicu ako cifre ispunjavaju uslov, a nulu inače. Napisati program koji učitava ceo broj  $i$  i ispisuje poruku da li su cifre unetog broja u neopadajućem poretku.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2289  
|| Cifre su u neopadajućem poretku.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 5  
|| Cifre su u neopadajućem poretku.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 6628  
|| Cifre nisu u neopadajućem poretku.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -23  
|| Cifre su u neopadajućem poretku.
```

[Rešenje 2.5.16]

**Zadatak 2.5.17** Napisati funkciju `int par_nepar(int n)` koja ispituje da li su cifre datog celog broja naizmenično parne i neparne. Funkcija treba da vrati jedinicu ako cifre ispunjavaju uslov, a nulu inače. Napisati program koji učitava ceo broj i ispisuje da li on ispunjava pomenuti uslov ili ne.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 2749  
|| Broj ispunjava uslov.
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -963  
|| Broj ispunjava uslov.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 27449  
|| Broj ne ispunjava uslov.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 5  
|| Broj ispunjava uslov.
```

[Rešenje 2.5.17]

**Zadatak 2.5.18** Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog celog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sve do unosa broja nula ispisuje odgovarajuće rotirane brojeve.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 146  
|| 461  
|| Unesite broj: 18  
|| 81  
|| Unesite broj: 3856  
|| 8563  
|| Unesite broj: 7  
|| 7  
|| Unesite broj: 0
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 89  
|| 98  
|| Unesite broj: -369  
|| -693  
|| Unesite broj: -55281  
|| -52815  
|| Unesite broj: 0
```

[Rešenje 2.5.18]

**Zadatak 2.5.19** Za dati broj može se formirati niz tako da je svaki sledeći član niza dobijen kao suma cifara prethodnog člana niza. Broj je *srećan* ako se dati niz završava jedinicom. Napisati funkciju `int srećan(int x)` koja vraća jedinicu ako je broj srećan, a nulu inače. Napisati program koji za uneti pozitivan ceo broj  $n$  ispisuje sve srećne brojeve od 1 do  $n$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 100
Srećni brojevi:
1 10 19 28 37 46 55 64 73 82 91 100
```

*Primer 2*

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

[Rešenje 2.5.19]

**Zadatak 2.5.20** Prirodan broj  $a$  je Armstrongov ako je jednak sumi  $n$ -tih stepena svojih cifara, pri čemu je  $n$  broj cifara broja  $a$ . Napisati funkciju `int armstrong(int x)` koja vraća jedinicu ako je broj Armstrongov, a nulu inače. Napisati program koji za učitani pozitivan ceo broj proverava da li je Armstrongov. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

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

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 1634
Broj je Armstrongov.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 118
Broj nije Armstrongov.
```

*Primer 4*

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

[Rešenje 2.5.20]

**Zadatak 2.5.21** Napisati funkciju `double e_na_x(double x, double eps)` koja računa vrednost  $e^x$  kao parcijalnu sumu reda  $\sum_{n=0}^{\infty} \frac{x^n}{n!}$ , pri čemu se sumiranje vrši dok je razlika sabiraka u redu po apsolutnoj vrednosti manja od  $\varepsilon$ . Napisati program koji učitava dva realna broja  $x$  i  $eps$  i ispisuje izračunatu vrednost  $e^x$ .

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj x: 5  
|| Unesite eps: 0.001  
|| Rezultat: 148.412951
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj x: -3  
|| Unesite eps: 0.0001  
|| Rezultat: 0.049796
```

[Rešenje 2.5.21]

**Zadatak 2.5.22** Napisati funkciju `void ispis(float x, float y, int n)` koja za dva realna broja  $x$  i  $y$  i jedan pozitivan ceo broj  $n$  ispisuje vrednosti sinusne funkcije u  $n$  ravnomerno raspoređenih tačaka intervala  $[x, y]$ . Napisati program koji učitava granice intervala i broj tačaka i ispisuje odgovarajuće vrednosti sinusne funkcije, zaokružene na četiri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva realna broja: 7 32  
|| Unesite broj n: 10  
|| 0.6570 -0.3457 -0.0108 0.3659 -0.6731  
|| 0.8922 -0.9945 0.9666 -0.8122
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva realna broja: 20.5 -8.32  
|| Unesite broj n: 5  
|| -0.8934 -0.8979 -0.1920 0.6658 0.9968
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva realna broja: 8 8  
|| Greska: neispravan unos.
```

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva realna broja: 7 32  
|| Unesite broj n: -10  
|| Greska: neispravan unos.
```

[Rešenje 2.5.22]

**Zadatak 2.5.23** Napisati funkciju `char sifra(char c, int k)` koja za dati karakter  $c$  određuje šifru na sledeći način: ukoliko je  $c$  slovo, šifra je karakter koji se nalazi  $k$  pozicija pre njega u abecedi. Karakteri koji nisu slova se ne šifruju. Šifrovanje treba da bude kružno, što znači da je, na primer, šifra za karakter  $b$  i pomeraj 2 karakter  $z$ . Napisati program koji učitava nenegativan ceo broj  $k$ , a zatim i karaktere sve do kraja ulaza i nakon svakog učitanoj karaktera ispisuje njegovu šifru. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.



*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj k: 2
Unesite tekst (CTRL+D za prekid):
c
a
8
8
+
+
Z
X

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj k: -2
Greska: neispravan unos.

```

[Rešenje 2.5.23]

**Zadatak 2.5.24** Rešiti sledeće zadatke korišćenjem funkcija.

- Zadatak 2.3.32 rešiti korišćenjem funkcije `char konverzija(char c)` koja malo slovo pretvara u odgovarajuće veliko i obrnuto.
- Zadatak 2.3.33 rešiti korišćenjem funkcije `void prebrojavanje()` koja učitava karaktere sve do kraja ulaza i ispisuje broj malih slova, velikih slova, cifara, belina, kao i sumu svih unetih cifara.

**Zadatak 2.5.25** Napisati program koji učitava tri cela broja i ispisuje datum sledećeg dana. Zadatak rešiti korišćenjem narednih funkcija.

- `int prestupna(int godina)` koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati jedinicu ako je godina prestupna ili nulu ako nije.
- `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu.
- `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan.
- `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum ispisuje datum sledećeg dana.

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite datum: 24.8.1998.
Datum sledeceg dana je: 25.8.1998.

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite datum: 31.12.1789.
Datum sledeceg dana je: 1.1.1790.

```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2003.  
|| Datum sledeceg dana je: 1.3.2004.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Greska: neispravan unos.
```

[Rešenje 2.5.25]

**Zadatak 2.5.26** Napisati funkciju `int od_nove_godine(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do datog datuma. Napisati program koji učitava tri cela broja i ispisuje koliko dana je proteklo od Nove godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 24.8.1998.  
|| Broj dana od Nove godine je: 235
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.12.1680.  
|| Broj dana od Nove godine je: 366
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2003.  
|| Broj dana od Nove godine je: 58
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Greska: neispravan unos.
```

[Rešenje 2.5.26]

**Zadatak 2.5.27** Napisati funkciju `int do_kraja_godine(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati program koji učitava tri cela broja i ispisuje broj dana do kraja godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 24.8.1998.  
|| Broj dana do Nove godine je: 129
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.12.1680.  
|| Broj dana do Nove godine je: 0
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2004.  
|| Broj dana do Nove godine je: 307
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Greska: neispravan unos.
```

[Rešenje 2.5.27]

**Zadatak 2.5.28** Napisati funkciju `int broj_dana_izmedju(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati program koji učitava dva datuma u formatu `dd.mm.gggg` i na standardni izlaz ispisuje broj dana između ta dva datuma. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 12.3.2008.
Unesite drugi datum: 5.12.2008.
Broj dana izmedju dva datuma je: 268
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 26.9.1986.
Unesite drugi datum: 2.2.1701.
Broj dana izmedju dva datuma je: 104301
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 24.8.1998.
Unesite drugi datum: 12.10.2010.
Broj dana izmedju dva datuma je: 4440
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 24.8.1998.
Unesite drugi datum: 31.4.2004.
Greska: neispravan unos.
```

[Rešenje 2.5.28]

**Zadatak 2.5.29** Napisati funkciju `void romb(int n)` koja iscrtava romb čija je stranica dužine  $n$ . Napisati program koji učitava ceo pozitivan broj  $i$  i iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -5
Greska: neispravan unos.
```

[Rešenje 2.5.29]

**Zadatak 2.5.30** Napisati funkciju `void grafikon_h(int a, int b, int c, int d)` koja iscrtava horizontalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja  $i$  i iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 4 1 7 5
****
*
*****
****
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 5 2 2 10
*****
**
**
*****
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 8 -2 5 4
Greska: neispravan unos.
```

[Rešenje 2.5.30]

**Zadatak 2.5.31** Napisati funkciju `void grafikon_v(int a, int b, int c, int d)` koja iscrtava vertikalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 4 1 7 5
*
*
**
* **
* **
* **
****
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 5 2 2 4
*
* *
* *
****
****
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 8 -2 5 4
Greska: neispravan unos.
```

[Rešenje 2.5.31]

## 2.6 Rešenja

### Rešenje 2.5.1

```
1 #include <stdio.h>
3 /* Funkcija racuna minimum tri cela broja.
   Promenljive u listi argumenata funkcije (x, y i z), kao i one
   5 deklarisanе u samoj funkciji (minimum), lokalne su za tu
   funkciju, sto znaci da im se ne moze pristupiti nigde izvan
   7 funkcije min. */
int min(int x, int y, int z)
9 {
   /* Vrednost minimuma se postavlja na vrednost broja x. */
11  int minimum = x;
```

```

13  /* Vrsi se poredjenje sa druga dva broja i po potrebi
    azuriranje vrednosti minimuma. */
15  if (minimum > y)
        minimum = y;
17  if (minimum > z)
        minimum = z;
19
    /* Vrednost minimuma se vraca kao povratna vrednost funkcije. */
21  return minimum;
    }
23
    int main()
25  {
        /* Deklaracije potrebnih promenljivih. */
27        int x, y, z;

29        /* Ucitavaju se vrednosti tri broja. */
        printf("Unesite brojeve: ");
31        scanf("%d%d%d", &x, &y, &z);

33        /* Poziv funkcije i ispis rezultata. */
        printf("Minimum: %d\n", min(x, y, z));
35
        return 0;
37  }

```

### Rešenje 2.5.2

```

1  #include <stdio.h>
    #include <math.h>
3
    /* Funkcija vraca razlomljeni deo prosledjenog broja. */
5  float razlomljeni_deo(float x)
    {
7      /* Napomena: Funkcija fabs racuna apsolutnu vrednost realnog
        broja i njena deklaracija se nalazi u zaglavlju math.h.
        Funkcija abs racuna apsolutnu vrednost celog broja i njena
        deklaracija se nalazi u zaglavlju stdlib.h. */
11     x = fabs(x);

13     /* Razlomljeni deo broja se dobija tako sto se od samog broja
        oduzme njegov ceo deo. */
15     return x - (int) x;
    }
17
    int main()
19  {
        /* Deklaracija potrebnih promenljivih. */
21        float n;

```

## 2 Kontrola toka

---

```
23  /* Ucitava se ulazna vrednost. */
    printf("Unesite broj:");
25  scanf("%f", &n);

27  /* Ispis rezultata. */
    printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));
29
    return 0;
31 }
```

### Rešenje 2.5.3

```
1  #include <stdio.h>

3  /* Funkcija racuna zbir delilaca broja x. */
    int zbir_delilaca(int x)
5  {
    int i;

7
    /* Inicijalizacija zbira na 0. */
    int zbir = 0;

11  /* Svaki broj i izmedju 1 i sqrt(x) koji deli broj x se dodaje
    u zbir. Ako je u pitanju broj koji za koji vazi da je i*i
13     jednako x, onda se dodaje samo vrednost i, a ako nije, onda
    se pored vrednosti i dodaje i x/i.
15     Na primer, za x=6, kada je i=2, dodaju se i 2 i 6/2 = 3, a za
    x = 4, kada je i=2, dodaje se samo 2. */
17  for (i = 1; i*i <= x; i++)
    {
19      if (x % i == 0)
        {
21          zbir += i;
          if(i != x/i)
23              zbir += x/i;
        }
25  }

27  /* Povratna vrednost funkcije je dobijeni zbir. */
    return zbir;
29 }

31 int main()
    {
33  /* Deklaracija potrebnih promenljivih. */
    int k, i;

35
    /* Ucitava se broj k. */
    printf("Unesite broj k:");
37  scanf("%d", &k);

39 }
```

```

41  /* Vrsi se provera ispravnosti ulaznih podataka. */
    if (k <= 0)
    {
43      printf("Greska: neispravan unos.\n");
      return -1;
45    }

47    /* Za svaki broj od 1 do k se ispisuje zbir delilaca. */
    for (i = 1; i <= k; i++)
49      printf("%d ", zbir_delilaca(i));
    printf("\n");
51    return 0;
53  }

```

### Rešenje 2.5.4

```

1  #include <stdio.h>

3  /* Funkcija za dva neoznacena broja x i n utvrđuje da li je
   x neki stepen broja n. Ukoliko jeste, funkcija vraća izlozilac
5  stepena, a u suprotnom vraća -1. */
int je_stepen(unsigned int x, unsigned int n)
7  {
    /* Na pocetku, s = n^i = n^1 = n. */
    int i = 1;
    unsigned int s = n;

11    /* U svakoj iteraciji petlje, s se azurira tako da ima
    vrednost n^i. Postupak se ponavlja dok je s manji od x. */
13    while (s < x)
    {
15      s = s * n;
17      i++;
    }

19    /* Kako s ima vrednost n^i, ako vazi da je s jednako x, onda
    je bas brojac i trazeni izlozilac. */
21    if (s == x)
    {
23      return i;
    }

25    /* Ako nije, onda se vraća -1. */
    return -1;
27  }

29 int main()
    {
31      /* Deklaracija potrebnih promenljivih. */
      unsigned int x, n;
33      int st;

```

```
35  /* Ucitavaju se vrednosti x i n. */
    printf("Unesite dva broja: ");
37  scanf("%u%u", &x, &n);

39  /* Poziva se napisana funkcija. */
    st = je_stepen(x, n);

41

43  /* U zavisnosti od povratne vrednosti funkcije, vrši se
    ispis rezultata. */
    if (st != -1)
45     printf("Jeste: %u=%u^d\n", x, n, st);
    else
47     printf("Broj %u nije stepen broja %u.\n", x, n);

49  return 0;
}
```

### Rešenje 2.5.5

```
1  #include <stdio.h>

3  /* Funkcija racuna nzd(x,y) primenom Euklidovog algoritma. */
    int euklid(int x, int y)
5  {
        int ostatak;

7     /* Euklidov algoritam: trazi se nzd(x,y).
        Na primer nzd(12,18). Postupak koji se primenjuje je sledeci:
9         1. ostatak = x % y = 12 % 18 = 12.
           2. x postaje y => x = 18
           3. y postaje ostatak => y = 12
           =>
13        1. ostatak = x % y = 18 % 12 = 6
           2. x postaje y => x = 12
           3. y postaje ostatak => y = 6
           =>
17        1. ostatak = x % y = 12 % 6 = 0
           2. x postaje y => x = 6
           3. y postaje ostatak => y = 0
           => procedura se završava jer je y jednako 0, a
19        rezultat je poslednji ne-nula ostatak, tj. x.*/
        while (y)
21        {
23            ostatak = x % y;
25            x = y;
            y = ostatak;
27        }

29     /* Kao povratna vrednost funkcije se vraca x. */
        return x;
31 }
```



```
33 int main()
34 {
35     /* Deklaracija potrebnih promenljivih. */
36     int a, b;
37
38     /* Ucitavaju se vrednosti a i b. */
39     printf("Unesite dva cela broja:");
40     scanf("%d%d", &a, &b);
41
42     /* Ispis rezultata. */
43     printf("Najveci zajednicki delilac: %d\n", euklid(a, b));
44
45     return 0;
46 }
```

### Rešenje 2.5.6

```
1  #include <stdio.h>
2
3  /* Funkcija racuna zbir reciprocnih vrednosti brojeva
4     iz intervala [1,n]. */
5  float zbir_reciprocnih(int n)
6  {
7      float zbir = 0;
8      int i;
9
10     /* Za svako i izmedju 1 i n na zbir se dodaje vrednost 1/i.
11        Napomena: zbog celobrojnog deljenja mora da stoji 1.0/i. */
12     for (i = 1; i <= n; i++)
13         zbir += 1.0 / i;
14
15     /* Kao povratna vrednost funkcije se vraca izracunati zbir. */
16     return zbir;
17 }
18
19 int main()
20 {
21     /* Deklaracija potrebne promenljive. */
22     int n;
23
24     /* Ucitava se vrednost broja n. */
25     printf("Unesite broj n:\n");
26     scanf("%d", &n);
27
28     /* Vrsi se provera ispravnosti ulaza. */
29     if(n <= 0)
30     {
31         printf("Greska: neispravan unos.\n");
32         return -1;
33     }
34 }
```

## 2 Kontrola toka

---

```
35  /* Ispis rezultata. */
    printf("Zbir reciprocnih: %.2f\n", zbir_reciprocnih(n));
37
    return 0;
39 }
```

### Rešenje 2.5.7

```
1  #include <stdio.h>

3  /* Funkcija broji koliko puta se realan broj x javlja u nizu unetih
   brojeva. */
5  int prebrojavanje(float x)
   {
7      float y;
      int broj_pojavljivanja = 0;

9
      /* Brojevi se ucitavaju sve do unosa broja nula.
         Svaki put kada se unese broj koji je jednak broju x,
         brojac pojavljivanja se uveca za 1. */
11     printf("Unesite brojeve:\n");
      while(1)
13     {
15         scanf("%f", &y);

17         if(y == 0)
19             break;

21         if (x == y)
            broj_pojavljivanja++;
23     }

25     return broj_pojavljivanja;
   }

27
29 int main()
   {
31     /* Deklaracija potrebnih promenljivih. */
      float x;
      int rezultat;

33
      /* Ucitava se vrednost broja x. */
35     printf("Unesite broj x: ");
      scanf("%f", &x);

37
      /* Poziva se napisana funkcija i u promenljivoj rezultat se
         cuva njena povratna vrednost. */
39     rezultat = prebrojavanje(x);

41
      /* Ispis rezultata. */
43     printf("Broj pojavljivanja broja %.2f: %d\n", x, rezultat);
```

```
45     return 0;
}
```

### Rešenje 2.5.8

```
1  #include <stdio.h>
2  #include <math.h>
3
4  /* Funkcija vraća 1 ako je broj prost, a 0 u suprotnom. */
5  int prost(int x)
6  {
7      int i;
8
9      /* Brojevi 2 i 3 su prosti. */
10     if (x == 2 || x == 3)
11         return 1;
12
13     /* Parni brojevi nisu prosti. */
14     if (x % 2 == 0)
15         return 0;
16
17     /* Ako se nađje na broj koji deli broj x, onda broj x nije
18        prost. Provera se vrši za sve neparne brojeve između 3 i
19        sqrt(x), jer kada bi x imao parnog delioca, onda bi i broj 2
20        delio x, a taj uslov je već proveren. */
21     for (i = 3; i <= sqrt(x); i += 2)
22         if (x % i == 0)
23             return 0;
24
25     /* Ako nijedan od prethodnih uslova nije bio ispunjen, to znači
26        da nijedan broj ne deli x, pa je on prost. */
27     return 1;
28 }
29
30 /* Funkcija ispisuje prvih n prostih brojeva.
31    Ključna reč void označava da funkcija nema povratnu vrednost. */
32 void prvih_n_prostih(int n)
33 {
34     int broj_prostih = 0;
35     int k = 2;
36
37     /* Petlja se izvršava dok god se ne istampa n prostih brojeva. */
38     while(broj_prostih < n)
39     {
40         /* Ako se nađje na broj koji je prost, ispisuje se njegova
41            vrednost i uvećava se brojac. */
42         if(prost(k))
43         {
44             printf("%d ", k);
45             broj_prostih++;
46         }
47     }
48 }
```

```
    }
47
    /* Prelazi se na sledeci broj. */
49    k++;
    }
51    printf("\n");
}
53
/* Funkcija ispisuje sve proste brojeve cija je vrednost manja
55    od n. */
void prosti_brojevi_manji_od_n(int n)
57 {
    /* Ukoliko je n manje ili jednako 2, onda nema prostih brojeva
59        koji su manji od njega. U tom slucaju se ispisuje odgovarajuca
        poruka i naredbom return; se izlazi iz funkcije. */
61    if(n<=2)
    {
63        printf("ne postoje\n");
        return;
65    }

67    /* Za svaki broj k izmedju 2 i n-1 se vrši provera da li je prost
        i ako jeste, ispisuje se njegova vrednost. */
69    int k = 2;
    while(k < n)
71    {
        if(prost(k))
73            printf("%d ", k);
        k++;
75    }
    printf("\n");
77 }

79 int main()
{
81    /* Deklaracija potrebnih promenljivih. */
    int n;
83
    /* Ucitava se broj n. */
85    printf("Unesite broj n:");
    scanf("%d", &n);
87
    /* Vrši se provera ispravnosti ulaza. */
89    if(n <= 0)
    {
91        printf("Greska: neispravan unos.\n");
        return -1;
93    }

95    /* Ispis rezultata. */
    printf("Prvih n prostih: ");
97    prvih_n_prostih(n);
```

```

printf("Prosti manji od n: ");
99  prosti_brojevi_manji_od_n(n);

101 return 0;
}

```

### Rešenje 2.5.10

```

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     /* Deklaracija i inicijalizacija brojac. */
       int zbir_cifara = 0;
13     int broj_cifara = 0;

15     /* Uzima se apsolutna vrednost broja x kako bi program ispravno
       radio i za negativne brojeve. */
17     x = abs(x);

19     /* Dok god ima neobradjenih cifara, na zbir se dodaje poslednja
       cifra, brojac cifara se uvecava za 1 i sa broja x se uklanja
21     poslednja cifra. */
       while (x)
23     {
           zbir_cifara += x % 10;
25         broj_cifara++;
           x /= 10;
27     }

29     /* Kao povratna vrednost funkcije se vraca odgovarajuci
       kolicnik. */
31     return (float) zbir_cifara / broj_cifara;
   }

33
   int main()
35 {
       /* Deklaracija potrebne promenljive. */
37     int x;

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

43     /* Ispis rezultata. */

```

```
printf("%.3f\n", aritmeticka_sredina(x));  
45  
return 0;  
47 }
```

### Rešenje 2.5.11

```
1 #include<stdio.h>  
#include<stdlib.h>  
3  
/* Funkcija ispituje da li se cifra c nalazi u zapisu celog broja  
5 x. Vraca 1 ako je uslov ispunjen i 0 u suprotnom. */  
int sadrzi(int x, int c)  
7 {  
    /* Uzima se apsolutna vrednost broja x. */  
9     x = abs(x);  
  
11    /* Izdvaja se cifra po cifra broja x.  
    Ako se naidje na cifru cija je vrednost c, onda se kao  
13 rezultat funkcije vraca 1 (jer x sadrzi c). */  
    while (x)  
15    {  
        if (x % 10 == c)  
17            return 1;  
        x /= 10;  
19    }  
  
21    /* Ako se petlja zavrсила, znaci da se nijednom nije naislo  
    na cifru c, sto znaci da broj x ne sadrzi cifru c i kao  
23 povratna vrednost funkcije se vraca 0. */  
    return 0;  
25 }  
  
27 int main()  
{  
29     /* Deklaracija potrebnih promenljivih. */  
    int x, c;  
  
31  
    /* Ucitavaju se vrednosti x i c. */  
33    printf("Unesite broj i cifru:");  
    scanf("%d%d", &x, &c);  
35  
    /* Vrsi se provera ispravnosti ulaza. */  
37    if(c < 0 || c > 9)  
    {  
39        printf("Greska: neispravan unos.\n");  
        return -1;  
41    }  
  
43    /* U zavisnosti od povratne vrednosti funkcije, vrsi se ispis  
    odgovarajuce poruke. */
```

```

45     if (sadrzi(x, c))
46         printf("Cifra %d se nalazi u zapisu broja %d\n", c, x);
47     else
48         printf("Cifra %d se ne nalazi u zapisu broja %d\n", c, x);
49     return 0;
50 }

```

### Rešenje 2.5.12

```

1  #include<stdio.h>
2  #include<stdlib.h>
3
4  /* Funkcija odredjuje broj neparnih cifara u zapisu datog celog
5     broja. */
6  int broj_neparnih_cifara(int x)
7  {
8      int brojac_neparnih = 0;
9      char cifra;
10     x = abs(x);
11
12     while (x)
13     {
14         /* Izdvaja se poslednja cifra broja. */
15         cifra = x % 10;
16         /* Moze se izbeci koriscenje naredbe if pomocu narednog izraza.
17            Naime, vrednost izraza cifra%2 je 1 kada je cifra neparna,
18            odnosno 0 kada je parna. Tako ce na broj neparnih cifara
19            biti dodata jednica ako je cifra neparna, a ako je parna
20            bice dodata 0, sto jeste zeljeno ponasanje. */
21         brojac_neparnih += (cifra % 2);
22         x /= 10;
23     }
24
25     return brojac_neparnih;
26 }
27
28 int main()
29 {
30     /* Deklaracija potrebne promenljive. */
31     int x;
32
33     /* Ucitavaju se brojevi sve do unosa broja nula i vrsi se ispis
34        broja neparnih cifara za svaki ucitani broj. */
35     printf("Unesite cele brojeve:\n");
36     while(1)
37     {
38         scanf("%d", &x);
39         if(x == 0)
40             break;
41
42         printf("Broj neparnih cifara: %d\n", broj_neparnih_cifara(x));

```

```
43     }
44
45     return 0;
46 }
```

### Rešenje 2.5.13

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

/* Funkcija proverava da li su sve cifre broja x parne i vraca
   1 ako je uslov ispunjen i 0 ako nije. */
int sve_parne_cifre(int x)
{
    char cifra;
    x = abs(x);

    /* Ako se naidje na cifru koja nije parna, onda se kao povratna
       vrednost funkcije vraca 0. */
    while (x > 0)
    {
        cifra = x % 10;
        if (cifra % 2 == 1)
            return 0;
        x /= 10;
    }

    /* Ako se doslo do kraja petlje, znaci da se nije naislo ni na
       jednu neparnu cifru, sto znaci da su sve cifre parne i da
       treba da se vrati 1. */
    return 1;
}

/* Funkcija proverava da li su sve cifre broja x jednake i vraca
   1 ako jesu, a 0 u suprotnom. */
int sve_cifre_jednake(int x)
{
    char poslednja_cifra;
    x = abs(x);

    /* Izdvaja se poslednja cifra broja. */
    poslednja_cifra = x % 10;
    x /= 10;

    /* Za sve ostale cifre se proverava da li su jednake poslednjoj.
       Ako se naidje na neku koja nije, onda nisu sve cifre broja
       x jednake i kao povratna vrednost se vraca 0. */
    while (x)
    {
        if (x % 10 != poslednja_cifra)
            return 0;
    }
}
```



```

46     x /= 10;
47 }
48
49 /* Ako se stiglo do kraja petlje, znaci da su sve cifre broja
50    bile jednake poslednjoj cifri, pa se kao povratna vrednost
51    vraca 1. */
52 return 1;
53 }
54
55 int main()
56 {
57     /* Deklaracija potrebne promenljive. */
58     int x;
59
60     /* Ucitava se broj x. */
61     printf("Unesite broj:");
62     scanf("%d", &x);
63
64     /* U zavisnosti od povratne vrednosti napisanih funkcija
65        vrsi se ispis odgovarajucih poruka. */
66     if (sve_parne_cifre(x))
67         printf("Sve cifre broja su parne.\n");
68     else
69         printf("Broj sadrzi bar jednu neparnu cifru.\n");
70
71     if (sve_cifre_jednake(x))
72         printf("Cifre broja su jednake.\n");
73     else
74         printf("Cifre broja nisu jednake.\n");
75
76     return 0;
77 }

```

### Rešenje 2.5.14

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <math.h>
4
5  /* Funkcija uklanja cifru sa pozicije p iz broja n.
6     Cifra jedinica ima poziciju 1, desetica 2, itd.*/
7  int ukloni(int n, int p)
8  {
9     int znak, tezina_pozicije, levi_deo, desni_deo;
10
11     /* Pamti se znak broja. */
12     znak = n < 0 ? 1 : -1;
13
14     /* Uzima se apsolutna vrednost. */
15     n = abs(n);

```

```
17  /* Racuna se tezina prosledjene pozicije. */
    tezina_pozicije = pow(10, p-1);
19
    /* Broj se deli na dva dela - deo levo od cifre koja se izbacuje
    i deo desno od cifre koja se izbacuje. */
21  levi_deo = n/(10*tezina_pozicije);
23  desni_deo = n%tezina_pozicije;

25  /* Povratna vrednost funkcije se dobija spajanjem levog i desnog
    dela i mnozenjem sa znakom pocetnog broja. */
27  return znak * (levi_deo*10 + desni_deo);
}

29  int main()
31  {
    /* Deklaracija potrebnih promenljivih. */
33  int broj, p;

35  /* Ucitava se vrednost pozicije. */
    printf("Unesite poziciju: ");
37  scanf("%d", &p);

39  /* Vrsi se provera ispravnosti ulaza. */
    if(p <= 0)
41  {
        printf("Greska: neispravan unos.\n");
43        return -1;
    }

45  /* Ucitavaju se brojevi dok se ne unese nula i za svaki
    ucitani broj se ispisuje broj koji se dobije uklanjanjem
    cifre koja se nalazi na poziciji p. */
47  while (1)
    {
49        printf("Unesite broj: ");
51        scanf("%d", &broj);

53        if (broj == 0)
55            break;

57        printf("%d\n", ukloni(broj, p));
    }

59  return 0;
61 }
```

### Rešenje 2.5.15

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

```

3  /* Funkcija proverava da li se neka cifra nalazi u zapisu celog
5  broja i ako se nalazi vraca odgovarajucu poziciju (tj. njenu
   tezinu koja je neki stepen broja 10), a u suprotnom vraca -1.
7  Na primer, za broj = 1234 i cifra = 2, funkcija vraca 100. */
   int pozicija_cifre(int broj, int cifra)
9  {
   int tezina_pozicije = 1;
11
12  while(broj)
13  {
   if(broj%10 == cifra)
15     return tezina_pozicije;
16
17     tezina_pozicije *= 10;
   broj /= 10;
19 }
20
21 return -1;
22 }
23
24 /* Funkcija iz zapisa broja izbacuje cifru koja se nalazi
25 na prosledjenoj poziciji. Pozicija je stepen broja 10.
   Na primer, za x=1234 i pozicija = 10, treba da se izbaci 3.
27 levi_deo = 1234/(10*10) = 12
   desni_deo = 1234%10 = 4
28 Povratna vrednost je 12*10 + 4 = 124. */
   int izbaci_cifru(int broj, int pozicija)
31 {
   int levi_deo = broj/(pozicija*10);
32   int desni_deo = broj%pozicija;
   return levi_deo*10 + desni_deo;
35 }
36
37 /* Funkcija proverava da li su dva cela broja napisana pomocu istih
   cifara. Vraca 1 ako je uslov ispunjen, a 0 u suprotnom. */
38 int zapis(int x, int y)
39 {
40   int pozicija;
   x = abs(x);
42   y = abs(y);
43
44   while (x)
45   {
46     /* Proverava se da li y sadrzi poslednju cifru broja x. */
     pozicija = pozicija_cifre(y, x % 10);
47
48     /* Ako ne sadrzi, x i y se ne zapisuju pomocu istih cifara. */
49     if(pozicija == -1)
50     {
51       return 0;
52     }
53
     /* Ako sadrzi, iz x se izbacuje poslednja cifra, a iz y se

```

## 2 Kontrola toka

```
55     izbacuje ista ta cifra (koja se nalazi na pronadjenoj
        poziciji. */
57     x /= 10;
        y = izbaci_cifru(y, pozicija);
59 }

61 /* Na kraju petlje iz x su izbacene sve cifre, a vazi da su
        brojevi zapisani pomocu istih cifara samo ukoliko ni u y
63     nema preostalih cifara. */
        return y == 0;
65 }

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

71

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

75

        /* U zavisnosti od povratne vrednosti napisane funkcija,
77     ispisuje se odgovarajuca poruka. */
        if (zapis(x, y))
79             printf("Uslov je ispunjen.\n");
        else
81             printf("Uslov nije ispunjen.\n");

83     return 0;
}
```

### Rešenje 2.5.16

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

3

5  /* Funkcija proverava da li se cifre u zapisu broja nalaze u
        neopadajućem poretku. */
    int neopadajuće(int n)
7  {
        int tekuća_cifra, prethodna_cifra;
9         n = abs(n);

11        /* Izvan petlje se izdvaja poslednja cifra u zapisu broja da bi u
            petlji mogla da se poredi sa sledećom. */
13        prethodna_cifra = n % 10;
            n /= 10;

15

        /* U petlji se proverava poredak svake dve susedne cifre. Ukoliko
17        se detektuje da je poredak narusen, izlazi se iz funkcije i
            vraca se vrednost 0. */
```

```

19 while (n)
20 {
21     tekuca_cifra = n % 10;
22
23     if (tekuca_cifra > prethodna_cifra)
24         return 0;
25
26     /* Tekuca cifra postaje prethodna za narednu iteraciju. */
27     prethodna_cifra = tekuca_cifra;
28     n /= 10;
29 }
30
31 /* Nakon izlaska iz petlje povratna vrednost funkcije je 1 jer u
32    slucaju da je poredak u nekom trenutku narusen iz funkcije bi
33    se izaslo jos u petlji. */
34 return 1;
35 }
36
37 int main()
38 {
39     /* Deklaracija potrebne promenljive. */
40     int n;
41
42     /* Ucitava se vrednost broja n. */
43     printf("Unesite broj: ");
44     scanf("%d", &n);
45
46     /* U zavisnosti od povratne vrednosti napisane funkcije vrsi
47        se ispis odgovarajuce poruke. */
48     if (neopadajuce(n))
49         printf("Cifre su u neopadajucem poretku.\n");
50     else
51         printf("Cifre nisu u neopadajucem poretku.\n");
52
53     return 0;
54 }

```

### Rešenje 2.5.17

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  /* Funkcija proverava da li su cifre broja naizmenicno parne i
5     neparne. Ako je uslov ispunjen vraca 1, u suprotnom vraca 0. */
6  int par_nepar(int x)
7  {
8      int prethodna_cifra, tekuca_cifra;
9      x = abs(x);
10
11     /* Poslednja cifra broja se izdvaja van petlje da bi u petlji
12        moglo da se vrsi poredjenje. */

```

```
13  prethodna_cifra = x % 10;
    x /= 10;
15
16  while (x)
17  {
    tekuca_cifra = x % 10;
19
    /* Ukoliko su uzastopne cifre iste parnosti, uslov nije
21     ispunjen, rad petlje i funkcije se prekida i vraca se 0. */
    if (tekuca_cifra % 2 == prethodna_cifra % 2)
23         return 0;

    /* Tekuca cifra postaje prethodna cifra za narednu iteraciju. */
    prethodna_cifra = tekuca_cifra;
27     x /= 10;
    }
29
    /* Sve uzastopne cifre su razlicite parnosti jer ni jednom u
31     petlji uslov da su cifre iste parnosti nije bio ispunjen. */
    return 1;
33 }

35 int main()
36 {
37     /* Deklaracija potrebne promenljive. */
    int n;
39
    /* Ucitava se vrednost broja n. */
41     printf("Unesite broj n: ");
    scanf("%d", &n);
43
    /* U zavisnosti od povratne vrednosti napisane funkcije, vrsi
45     se ispis odgovarajuce poruke. */
    if (par_nepar(n))
47         printf("Broj ispunjava uslov.\n");
    else
49         printf("Broj ne ispunjava uslov.\n");

51     return 0;
}
```

### Rešenje 2.5.18

```
#include<stdio.h>
2 #include<math.h>
#include<stdlib.h>
4
/* Funkcija racuna broj cifara celog broja n. */
6 int broj_cifara(int n)
7 {
8     int brojac = 0;
```

```

n = abs(n);
10
if(n < 10)
12     return 1;

14 while(n)
{
16     brojac++;
    n /= 10;
18 }

20 return brojac;
}

22
/* Funkcija racuna broj koji se dobija rotacijom broja n za
24 jedno mesto ulevo. */
int rotacija(int n)
26 {
    int znak, prva_cifra, n_bez_prve_cifre, br_cifara;

28     znak = (n < 0) ? -1 : 1;
    n = abs(n);
30     br_cifara = broj_cifara(n);

32     /* Izdvajaju se prva cifra i deo broja bez prve cifre.
    Na primer: ako je n = 1234 onda je br_cifara = 4
    prva_cifra se dobija kao:
34     n / (10^(br_cifara-1)) = 1234/1000 = 1.
    n_bez_prve_cifre se dobija kao: n%1000 = 234. */
    int tezina_pozicije = pow(10, br_cifara-1);
    prva_cifra = n / tezina_pozicije;
40     n_bez_prve_cifre = n % tezina_pozicije;

42     /* Rezultat se dobija nadovezivanjem prve cifre na kraj i
    mnozenjem sa znakom pocetnog broja. */
44     return znak * (n_bez_prve_cifre*10 + prva_cifra);
}

46
int main()
48 {
    /* Deklaracija potrebne promenljive. */
50     int n;

52     /* Brojevi se ucitavaju sve do unosa broja nula i ispisuju
    se brojevi dobijeni kao rezultat izvršavanja funkcije rotacija
54     nad unetim brojevima. */
    while (1)
56     {
        printf("Unesite broj: ");
58         scanf("%d", &n);

60         if (n == 0)

```

```
        break;
62     printf("%d\n", rotacija(n));
64 }
66 return 0;
}
```

### Rešenje 2.5.19

```
1  #include<stdio.h>
3  /* Funkcija vraca zbir cifara datog broja x. */
4  int zbir_cifara(int x)
5  {
6      int zbir = 0;
7      while (x)
8      {
9          zbir += x%10;
10         x /= 10;
11     }
12     return zbir;
13 }
15 /* Funkcija vraca 1 ako je broj srecan, a 0 u suprotnom. */
16 int srecan(int x)
17 {
18     /* Dok god u broju x ima vise od 2 cifre, vrednost broja x se
19        zamenjuje sa zbirom njegovih cifara. Na primer, za pocetno
20        x = 7698, nakon prve iteracije x postaje 7+6+9+8 = 30, nakon
21        druge iteracije x postaje 3 + 0 = 3 i zatim se izlazi iz
22        petlje. */
23     while(x <= 10)
24         x = zbir_cifara(x);
25
26     /* Broj je srecan ako na kraju x ima vrednost 1. */
27     return (x == 1);
28 }
29
30 int main()
31 {
32     /* Deklaracija potrebnih promenljivih. */
33     int n, i;
34
35     /* Ucitava se vrednost broja n. */
36     printf("Unesite broj n: ");
37     scanf("%d", &n);
38
39     /* Vrsi se provera ispravnosti ulaza. */
40     if(n <= 0)
41     {
```



```
43     printf("Greska: neispravan unos.\n");
44     return -1;
45 }
46
47 /* Ispisuju se svi srecni brojevi manji ili jednaki n. */
48 printf("Srecni brojevi: ");
49 for (i = 1; i <= n; i++)
50     if (srecan(i))
51         printf("%d ", i);
52
53 printf("\n");
54 return 0;
55 }
```

## Rešenje 2.5.20

```
1  #include <stdio.h>
2  #include <math.h>
3  #include <stdlib.h>
4
5  /* Funkcija racuna broj cifara celog broja n. */
6  int broj_cifara(int n)
7  {
8      int brojac = 0;
9      n = abs(n);
10
11     if(n < 10)
12         return 1;
13
14     while(n)
15     {
16         brojac++;
17         n /= 10;
18     }
19
20     return brojac;
21 }
22
23 /* Funkcija proverava da li je broj Armstrongov. */
24 int armstrong(int x)
25 {
26     int suma = 0;
27     int n = broj_cifara(x);
28     int x_pocetno = x;
29
30     /* Racuna se suma n-tih stepena cifara broja x. */
31     while (x)
32     {
33         suma += pow(x % 10, n);
34         x /= 10;
35     }
```

```
37  /* Ako je suma jednaka pocetnoj vrednosti broja x, broj je
    Armstrongov, u suprotnom nije. */
39  return x_pocetno == suma;
    }

41
43  int main()
44  {
    /* Deklaracija potrebne promenljive. */
45  int x;

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

51  /* Proverava se da li je x Armstrongov broj i ispisuje se
    odgovarajuca poruka. */
53  if (armstrong(x))
    printf("Broj je Armstrongov.\n");
55  else
    printf("Broj nije Armstrongov.\n");

57  return 0;
59  }
```

### Rešenje 2.5.21

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

4 /* Funkcija racuna vrednost  $e^x$  kao parcijalnu sumu reda
    suma( $x^n/n!$ ), gde indeks n ide od 0 do beskonacno, pri cemu
6 se sumiranje vrši dok je razlika sabiraka u redu po apsolutnoj
    vrednosti manja od eps. */
8 double e_na_x(double x, double eps)
    {
10     double s = 1;
    double clan = 1;
12     int n = 1;

14     /* Parcijalnu suma se formira tako sto se u svakoj iteraciji
        petlje promenljivoj s doda jedan sabirak sume oblika  $(x^n)/n!$ 
16     koji se cuva u promenljivoj clan.

18     Svaki sabirak se dobija na osnovu prethodnog tako sto se
        prethodni pomnozi sa x i podeli sa n (n predstavlja redni broj
20     sabirka u sumi).

22     Prvi sabirak (kome odgovara  $n=0$ ) iznosi 1; zbog toga
        promenljive s i clan se inicijalizuju na vrednost 1.
24     */
    }
```

```

26     Sumiranje se sprovodi sve dok je sabirak po apsolutnoj
    vrednosti veci od trazene tacnosti eps. */
27 do
28 {
    clan = (clan * x) / n;
30     s += clan;
    n++;
32 } while (fabs(clan) > eps);

34     return s;
35 }

36 int main()
37 {
38     /* Deklaracija potrebnih promenljivih. */
40     double x, eps;

42     /* Ucitavavaju se vrednosti x i eps. */
    printf("Unesite broj x: ");
44     scanf("%lf", &x);
    printf("Unesite eps: ");
46     scanf("%lf", &eps);

48     /* Ispis rezultata. */
    printf("Rezultat: %f\n", e_na_x(x, eps));
50     return 0;
51 }

```

### Rešenje 2.5.22

```

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

4  /* Funkcija ispisuje vrednosti funkcije sin(x) u n ravnomerno
    rasporedjenih tacaka na intervalu [a,b]. */
6  void ispis(float a, float b, int n)
7  {
8      float i;
    float korak = (b - a) / (n - 1);
10
11     for (i = a; i <= b; i += korak)
12         printf("%.4f ", sin(i));

14     printf("\n");
15 }

16 int main()
17 {
18     /* Deklaracija potrebnih promenljivih. */
20     float a, b;
    int n;

```

```
22  /* Ucitavaju se granice intervala i vrsi se provera ispravnosti
24  ulaza. */
25  printf("Unesite dva realna broja:");
26  scanf("%f%f", &a, &b);
27  if(b >= a)
28  {
29      printf("Greska: neispravan unos.\n");
30      return -1;
31  }
32
33  /* Ucitava se broj n i vrsi se provera ispravnosti ulaza. */
34  printf("Unesite broj n:");
35  scanf("%d", &n);
36  if (n <= 1)
37  {
38      printf("Greska: neispravan unos.\n");
39      return -1;
40  }
41
42  /* Ispis rezultata. */
43  ispis(a, b, n);
44
45  return 0;
46  }
```

### Rešenje 2.5.23

```
1  #include <stdio.h>
2
3  /* Funkcija vraca karakter koji se u abecedi nalazi k mesta pre
4  datog karaktera c. */
5  char sifra(char c, int k)
6  {
7      /* Provera da li je karakter malo slovo. */
8      if (c >= 'a' && c <= 'z')
9      {
10         /* Ako karakter koji je k pozicija pre datog karaktera ispada
11         iz opsega malih slova. */
12         if (c - k < 'a')
13             /* Od k se oduzima rastojanje izmedju c i 'a' (jer je za
14             toliko karaktera vec vrateno u nazad), kako bi se odredilo
15             koliko preostali broj karaktera koji treba preskociti od
16             karaktera 'z'. */
17             return 'z' - (k - (c - 'a') - 1);
18         else
19             /* U suprotnom, karakter c-k ne ispada iz opsega malih slova,
20             te je dovoljno njega vratiti. */
21             return c - k;
22     }
23     else if (c >= 'A' && c <= 'Z')
```

```

25 {
    /* Postupak se ponavlja i za velika slova. */
    if (c - k < 'A')
27         return 'Z' - (k - (c - 'A') - 1);
    else
29         return c - k;
    }

31
    /* Ako nije ni malo ni veliko slovo, karakter se ne menja. */
33     return c;
    }

35
int main()
37 {
    /* Deklaracija potrebnih promenljivih. */
39     int k;
    char c;

41
    /* Ucitava se vrednost k. */
43     printf("Unesite broj k: ");
    scanf("%d", &k);

45
    /* Ucitavaju se karakteri sve do kraja ulaza i ispisuje se
47     njihova sifra. */
    printf("Unesite tekst (CTRL + D za prekid): ");
49     while ((c = getchar()) != EOF)
        putchar(sifra(c, k));

51
    return 0;
53 }

```

### Rešenje 2.5.25

```

1  #include<stdio.h>

3  /* Funkcija proverava da li je godina prestupna. */
int prestupna(int godina)
5  {
    if ((godina % 100 != 0 && godina % 4 == 0) || godina % 400 == 0)
7         return 1;
    else
9         return 0;
    }

11
    /* Funkcija odredjuje broj dana u datom mesecu. */
13 int broj_dana(int mesec, int godina)
    {
15     switch (mesec) {
        case 1:
17         case 3:
        case 5:

```

```
19     case 7:
20     case 8:
21     case 10:
22     case 12:
23         return 31;
24     case 4:
25     case 6:
26     case 9:
27     case 11:
28         return 30;
29     case 2:
30         if (prestupna(godina))
31             return 29;
32         else
33             return 28;
34     }
35     return -1;
36 }
37
38 /* Funkcija proverava da li je datum ispravan. Ako je datum
39    ispravan funkcija vraca 1, inace vraca 0. */
40 int ispravan(int dan, int mesec, int godina)
41 {
42     /* Ako je godina negativna, datum nije ispravan. */
43     if (godina < 0)
44         return 0;
45
46     /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */
47     if (mesec < 1 || mesec > 12)
48         return 0;
49
50     /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,
51        datum nije ispravan. */
52     if (dan < 1 || dan > broj_dana(mesec, godina))
53         return 0;
54
55     return 1;
56 }
57
58 /* Funkcija racuna sledeci dan. */
59 void sledeci_dan(int dan, int mesec, int godina)
60 {
61     /* Za kraj godine, odnosno za datum 31.12. sledeci datum je 1.1.
62        i godina se uvecava za jedan. */
63     if (mesec == 12 && dan == 31)
64         printf("1.1.%d.\n", godina + 1);
65     /* Ukoliko je dan jednak poslednjem danu u tom mesecu, odnosno
66        ako je jednak broju dana u tom mesecu, onda je sledeci datum
67        kada se mesec uveca za 1, a dan postane 1. Bitan je redosled
68        ovih naredbi. Ako bi ovo ispitivanje bilo prvo, onda bi se
69        mesec mogao uvecati na 13. sto ne bi bio ispravan datum. Zato
        se prvo proverava da li je kraj godine, pa tek onda da li je
```

```

71     kraj meseca. */
72     else if (dan == broj_dana(mesec, godina))
73         printf("1.%d.%d.\n", mesec + 1, godina);
74     /* Ako nije ni jedan od prethodna dva slucaja, onda se dan moze
75        uvecati na 1, bez bojazni da ce se prekoraciti broj dana u
76        datom mesecu. */
77     else
78         printf("%d.%d.%d.\n", dan + 1, mesec, godina);
79 }
80
81 int main()
82 {
83     /* Deklaracija potrebnih promenljivih. */
84     int dan, mesec, godina;
85
86     /* Ucitavaju se vrednosti dana, meseca i godine. */
87     printf("Unesite datum:");
88     scanf("%d.%d.%d.", &dan, &mesec, &godina);
89
90     /* Vrsi se provera ispravnosti datuma. */
91     if (!ispravan(dan, mesec, godina))
92     {
93         printf("Greska: neispravan unos.\n");
94         return -1;
95     }
96
97     /* Poziva se funkcija za ispis sledeceg dana. */
98     printf("Datum sledeceg dana je:");
99     sledeci_dan(dan, mesec, godina);
100
101     return 0;
102 }

```

### Rešenje 2.5.26

Za rešavanje ovog zadatka koristi se funkcija `od_nove_godine` koja je definisana u rešenju zadatka 2.5.28.

### Rešenje 2.5.27

Za rešavanje ovog zadatka koristi se funkcija `do_kraja_godine` koja je definisana u rešenju zadatka 2.5.28.

### Rešenje 2.5.28

```

1  #include<stdio.h>
2
3  /* Funkcija proverava da li je godina prestupna. */
4  int prestupna(int godina)
5  {

```

```
6     if ((godina % 100 != 0 && godina % 4 == 0) || godina % 400 == 0)
            return 1;
8     else
            return 0;
10 }

12 /* Funkcija odredjuje broj dana u datom mesecu. */
13 int broj_dana(int mesec, int godina)
14 {
15     switch (mesec) {
16     case 1:
17     case 3:
18     case 5:
19     case 7:
20     case 8:
21     case 10:
22     case 12:
23         return 31;
24     case 4:
25     case 6:
26     case 9:
27     case 11:
28         return 30;
29     case 2:
30         if (prestupna(godina))
31             return 29;
32         else
33             return 28;
34     }
35     return -1;
36 }

38 /* Funkcija proverava da li je datum ispravan. Ako je datum
39    ispravan funkcija vraca 1, inace vraca 0. */
40 int ispravan(int dan, int mesec, int godina)
41 {
42     /* Ako je godina negativna, datum nije ispravan. */
43     if (godina < 0)
44         return 0;

46     /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */
47     if (mesec < 1 || mesec > 12)
48         return 0;

50     /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,
51        datum nije ispravan. */
52     if (dan < 1 || dan > broj_dana(mesec, godina))
53         return 0;

54     return 1;
55 }
```



```
58 /* Funkcija odredjuje koliko dana je proteklo od pocetka godine. */
int od_nove_godine(int dan, int mesec, int godina)
60 {
    int suma_dana = 0, i;
62
    /* Za sve mesece pre datog datuma dodaje se broj dana za dati
64    mesec. */
    for (i = 1; i < mesec; i++)
66        suma_dana += broj_dana(mesec, godina);
68
    /* Na kraju se dodaje koliko je dana proteklo u datom mesecu, a
    to je zadato sa promenljivom dan. */
70    return suma_dana + dan;
72 }

/* Funkcija odredjuje koliko dana ima do kraja godine. */
74 int do_kraja_godine(int dan, int mesec, int godina)
{
76     int suma_dana = 0, i;
78
    /* Za sve mesece posle datog datuma dodaje se broj dana za dati
    mesec. */
80    for (i = mesec + 1; i <= 12; i++)
        suma_dana += broj_dana(mesec, godina);
82
    /* Na kraju se dodaje koliko je dana je ostalo u datom mesecu. */
84    return suma_dana + broj_dana(mesec, godina) - dan;
86 }

/* Funkcija vraca 1 ako je prvi datum pre drugog datuma. U
    suprotnom vraca 0. */
88 int prethodi(int dan1, int mesec1, int godina1, int dan2,
90             int mesec2, int godina2)
{
92     if (godina1 < godina2)
        return 1;
94     else if (godina1 > godina2)
        return 0;
96     else if (mesec1 < mesec2)
        return 1;
98     else if (mesec1 > mesec2)
        return 0;
100    else if (dan1 < dan2)
        return 1;
102    else
        return 0;
104 }

/* Funkcija vraca broj dana u datoj godini. */
106 int broj_dana_u_godini(int godina)
108 {
    if (prestupna(godina))
```

```
110     return 366;
111     else
112         return 365;
113 }
114
115 /* Funkcija racuna broj dana izmedju dva datuma. */
116 int broj_dana_izmedju(int dan1, int mesec1, int godina1, int dan2,
117                       int mesec2, int godina2) {
118     int pom, i;
119     int suma_dana = 0;
120
121     /* Vrsi se provera koji od datuma je ranije i ukoliko je to
122        potrebno, razmenjuju se tako da broj 1 ide uz prvi datum. */
123     if (!prethodi(dan1, mesec1, godina1, dan2, mesec2, godina2))
124     {
125         pom = dan1;
126         dan1 = dan2;
127         dan2 = pom;
128
129         pom = mesec1;
130         mesec1 = mesec2;
131         mesec2 = pom;
132
133         pom = godina1;
134         godina1 = godina2;
135         godina2 = pom;
136     }
137
138     /* Ako su godine razlicite. */
139     if (godina1 != godina2)
140     {
141         /* Za manji datum dodaje se broj dana do kraja godine. */
142         suma_dana = do_kraja_godine(dan1, mesec1, godina1);
143
144         /* Za sve godine koje su izmedju dve date godine dodaje se broj
145            dana u tim godinama. */
146         for (i = godina1 + 1; i < godina2; i++)
147             suma_dana += broj_dana_u_godini(i);
148
149         /* Za veci datum dodaje se broj dana od pocetka godine. */
150         suma_dana += od_nove_godine(dan2, mesec2, godina2);
151     }
152     /* Ako su godine iste, ali meseci razliciti. */
153     else if (mesec1 != mesec2)
154     {
155         /* Dodaje se broj dana do kraja prvog meseca. */
156         suma_dana = broj_dana(mesec1, godina1) - dan1;
157
158         /* Dodaje se broj dana za svaki mesec koji je izmedju dva data
159            meseca. Kako su godina1 i godina2 jednake svejedno je koja
160            od ove dve promenljive se koristi u pozivu funkcije. */
161         for (i = mesec1 + 1; i < mesec2; i++)
```

```

162     suma_dana += broj_dana(i, godina1);

164     /* Dodaje se broj dana od pocetka meseca. */
    suma_dana += dan2;
166 }
    /* Ako su i godine i meseci jednaki. */
168 else
    suma_dana = dan2 - dan1;
170
172     return suma_dana;
}

174 int main()
{
176     /* Deklaracija potrebnih promenljivih. */
    int dan1, mesec1, godina1, dan2, mesec2, godina2;
178
180     /* Ucitavaju se datumi. */
    printf("Unesite prvi datum:");
    scanf("%d.%d.%d.", &dan1, &mesec1, &godina1);
182
184     printf("Unesite drugi datum:");
    scanf("%d.%d.%d.", &dan2, &mesec2, &godina2);

186     /* Vrsi se provera ispravnosti unetih datuma. */
    if (!ispravan(dan1, mesec1, godina1)
188         || !ispravan(dan2, mesec2, godina2))
    {
190         printf("Greska: neispravan unos.\n");
        return -1;
192     }

194     /* Ispis rezultata. */
    printf("Broj dana izmedju dva datuma je: %d\n",
196         broj_dana_izmedju(dan1, mesec1, godina1, dan2, mesec2,
198                             godina2));

200     return 0;
}

```

### Rešenje 2.5.29

```

1  #include<stdio.h>

3  /* Funkcija iscrtava romb. */
void romb(int n)
5  {
    int i, j;
7
    /* Petlja iscrtava liniju po liniju romba. */
9    for (i = 0; i < n; i++)

```

```
11 {
    /* U svakoj liniji prvo se ispisuje n-i-1 razmaka. */
13     for (j = 0; j < n - i - 1; j++)
        printf(" ");

15     /* Ispisuje se n zvezdica. */
    for (j = 0; j < n; j++)
17         printf("*");

19     /* Ispisuje se novi red. */
    printf("\n");
21 }
23 }

25 int main()
26 {
    /* Deklaracija potrebne promenljive. */
27     int n;

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

33     /* Vrsi se provera ispravnosti ulaza. */
    if (n <= 0)
35     {
        printf("Greska: neispravan unos.\n");
37         return -1;
    }

39     /* Iscrtava se romb. */
41     romb(n);

43     return 0;
}
```

### Rešenje 2.5.30

```
1 #include<stdio.h>

3 /* Funkcija stampa n zvezdica za kojima sledi znak za novi red. */
void stampaj_zvezdice(int n)
5 {
    int i;
7     for (i = 0; i < n; i++)
        printf("*");

9     printf("\n");
11 }

13 /* Funkcija crta grafikon. */
```

```
void grafikon_h(int a, int b, int c, int d)
15 {
16     /* Prvo se ispisuje a zvezdica. */
17     stampaj_zvezdice(a);

18     /* Postupak se ponavlja za vrednosti b, c i d. */
19     stampaj_zvezdice(b);
21     stampaj_zvezdice(c);
22     stampaj_zvezdice(d);
23 }

24
25 int main()
26 {
27     /* Deklaracija potrebnih promenljivih. */
28     int a, b, c, d;

29     /* Ucitavaju se vrednosti a,b,c,d. */
31     printf("Unesite brojeve: ");
32     scanf("%d%d%d%d", &a, &b, &c, &d);

33     /* Vrsi se provera ispravnosti ulaza i ispisuje se rezultat. */
35     if (a < 0 || b < 0 || c < 0 || d < 0)
36     {
37         printf("Greska: neispravan unos.\n");
38         return -1;
39     }
40     else
41         grafikon_h(a, b, c, d);

42     return 0;
43 }
44 }
```

## Rešenje 2.5.31

```
1 #include<stdio.h>

2
3 /* Funkcija racuna najveći od 4 prosledjena broja. */
4 int maksimum(int a, int b, int c, int d)
5 {
6     int max;

7     max = a;
8     if (b > max)
9         max = b;
10    if (c > max)
11        max = c;
12    if (d > max)
13        max = d;

14    return max;
15 }
16 }
```

```
19  /* Pomocna funkcija za stampanje beline ili zvezdice. */
void stampaj_znak(int polje, int granica)
21  {
    if (polje < granica)
23      printf(" ");
    else
25      printf("*");
}

27  /* Funkcija iscrtava vertikalni grafikon. */
void grafikon_v(int a, int b, int c, int d)
29  {
    int i, max;

31      /* Na pocetku je potrebno pronaci najveću od ove cetiri
        vrednosti. */
33      max = maksimum(a, b, c, d);

35      /* Grafikon ukupno ima max horizontalnih linija. */
37      for (i = 0; i < max; i++)
39      {
        /* U svakoj od horizontalnih linija se nalazi po 4 polja: polje
41         za a,b,c i d uspravnu liniju. U svako od polja treba da se
        upise ili * ili belina, u zavisnosti od vrednosti i toga
43         koja linija se trenutno ispisuje. */

45         /* Stampa se znak za polje a. */
        stampaj_znak(i, max - a);

47         /* Stampa se znak za polje b. */
49         stampaj_znak(i, max - b);

51         /* Stampa se znak za polje c. */
        stampaj_znak(i, max - c);

53         /* Stampa se znak za polje d. */
55         stampaj_znak(i, max - d);

57         /* Na kraju svake horizontalne linije stampa se novi red. */
        printf("\n");
59     }
}

61  int main()
63  {
    /* Deklaracija potrebnih promenljivih. */
65     int a, b, c, d;

67     /* Ucitavaju se vrednosti cetiri broja. */
    printf("Unesite brojeve: ");
69     scanf("%d%d%d%d", &a, &b, &c, &d);
```

```
71  /* Vrsi se provera ispravnosti ulaza i poziva se funkcija za
    ispis grafikona. */
73  if (a < 0 || b < 0 || c < 0 || d < 0)
    {
75      printf("Greska: neispravan unos.\n");
      return -1;
77  }
    else
79      grafikon_v(a, b, c, d);
81  return 0;
}
```





# 3

## Predstavljanje podataka

### 3.1 Nizovi

**Zadatak 3.1.1** Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje:

- (a) elemente niza koji se nalaze na parnim pozicijama.
- (b) parne elemente niza.

Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### *Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
6
Unesite elemente niza:
1 8 2 -5 -13 75
Elementi niza na parnim pozicijama:
1 2 -13
Parni elementi niza:
8 2
```

#### *Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
3
Unesite elemente niza:
11 81 -63
Elementi niza na parnim pozicijama:
11 -63
Parni elementi niza:
```

#### *Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
-4
Greska: neispravan unos.
```

[Rešenje 3.1.1]

### 3 Predstavljanje podataka

---

**Zadatak 3.1.2** Napisati program koji učitava dimenziju niza, elemente niza i zatim menja uneti niz tako što kvadrira sve negativne elemente niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza:
12.34 -6 1 8 32.4 -16
12.34 36 1 8 32.4 256
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 9
Unesite elemente niza:
-8.25 6 17 2 -1.5 1 -7 2.65 -125.2
68.0625 6 17 2 2.25 1 49 2.65 15675.04
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza:
9.53 5 1 4.89
9.53 5 1 4.89
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 104
Greska: neispravan unos.
```

[Rešenje 3.1.2]

**Zadatak 3.1.3** Ako su  $a = (a_1, \dots, a_n)$  i  $b = (b_1, \dots, b_n)$  vektori dimenzije  $n$ , njihov skalarni proizvod se definiše kao  $a \cdot b = a_1 \cdot b_1 + \dots + a_n \cdot b_n$ . Napisati program koji računa skalarni proizvod dva vektora. Vektori se zadaju kao celobrojni nizovi sa najviše 100 elemenata. Program učitava dimenziju i elemente nizova, a na izlaz ispisuje vrednost skalarnog proizvoda. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju vektora:
5
Unesite koordinate vektora a:
8 -2 0 2 4
Unesite koordinate vektora b:
35 12 5 -6 -1
Skalarni proizvod: 240
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju vektora:
3
Unesite koordinate vektora a:
-1 0 1
Unesite koordinate vektora b:
5 5 5
Skalarni proizvod: 0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju vektora:
0
Greska: neispravan unos.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju vektora:
1
Unesite koordinate vektora a:
-1
Unesite koordinate vektora b:
1
Skalarni proizvod: -1
```

[Rešenje 3.1.3]

**Zadatak 3.1.4** Napisati program koji učitava dimenziju niza, elemente niza i ceo broj  $k$  i ispisuje indekse elemenata koji su deljivi sa  $k$ . Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 10 14 86 20
Unesite broj k: 5
0 3
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 6 14 8 9
Unesite broj k: 5
U nizu nema elemenata koji su
deljivi brojem 5.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza: 8 9 11 -4 8 11
Unesite broj k: 2
0 3 4
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza: 1 2 3 4 5 6
Unesite broj k: 0
Greska: neispravan unos.
```

[Rešenje 3.1.4]

**Zadatak 3.1.5** Autobusi su označeni rednim brojevima (počevši od 1) i u nizu se čuva vreme putovanja svakog autobusa u minutima. Međutim, zbog radova na putu između Požege i Užica, svi autobusi koji saobraćaju na tom potezu (autobusi označeni rednim brojevima od  $k$  do  $t$ ) saobraćaju  $m$  minuta duže. Napisati program koji učitava broj autobusa  $n$ ,  $n$  celih brojeva koji označavaju vreme putovanja tih autobusa i vrednosti  $k$ ,  $t$  i  $m$  i ispisuje vreme putovanja svih autobusa nakon unetih izmena. Maksimalan broj autobusa je 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj autobusa:
8
Unesite vreme putovanja:
24 78 13 124 56 90 205 45
Unesite vrednosti k, t i m:
3 6 23
Vreme putovanja nakon izmena:
24 78 36 147 79 113 205 45
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj autobusa:
8
Unesite vreme putovanja:
24 78 13 124 56 90 205 45
Unesite vrednosti k, t i m:
3 15 3
Greska: neispravan unos.
```

[Rešenje 3.1.5]

### 3 Predstavljanje podataka

---

**Zadatak 3.1.6** Napisati program koji za učitani ceo broj ispisuje broj pojavljivanja svake od cifara u zapisu tog broja. UPUTSTVO: Za evidenciju broja pojavljivanja svake cifre pojedinačno, koristiti niz.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj:
2355623
U zapisu broja 2355623, cifra 2 se pojaviljuje 2 puta
U zapisu broja 2355623, cifra 3 se pojaviljuje 2 puta
U zapisu broja 2355623, cifra 5 se pojaviljuje 2 puta
U zapisu broja 2355623, cifra 6 se pojaviljuje 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
```

[Rešenje 3.1.6]

**Zadatak 3.1.7** Napisati program koji učitava karaktere sve do unosa karaktera \* i ispisuje ih u redosledu suprotnom od redosleda čitanja. Maksimalan broj karaktera je 500.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: a
Unesite karakter: 8
Unesite karakter: 5
Unesite karakter: Y
Unesite karakter: I
Unesite karakter: o
Unesite karakter: ?
Unesite karakter: *
? o I Y 5 8 a
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: g
Unesite karakter: g
Unesite karakter: 2
Unesite karakter: 2
Unesite karakter: )
Unesite karakter: )
Unesite karakter: *
) ) 2 2 g g
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: U
Unesite karakter: 4
Unesite karakter: a
Unesite karakter: u
Unesite karakter: *
u a 4 U
```

[Rešenje 3.1.7]

**Zadatak 3.1.8** Napisati program koji učitava karaktere sve do kraja ulaza i izračunava koliko se puta u unetom tekstu pojavila svaka od cifara, svako malo slovo i svako veliko slovo. Ispisati broj pojavljivanja samo za karaktere koji su se u unetom tekstu pojavili barem jednom. UPUTSTVO: *Za evidenciju broja pojavljivanja cifara, malih i velih slova koristiti pojedinačne nizove.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
123 abcabcabc 123
Karakter 1 se pojavljuje 2 puta
Karakter 2 se pojavljuje 2 puta
Karakter 3 se pojavljuje 2 puta
Karakter a se pojavljuje 3 puta
Karakter b se pojavljuje 3 puta
Karakter c se pojavljuje 3 puta
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Programiranje 1 je zanimljivo!!
Karakter 1 se pojavljuje 1 puta
Karakter a se pojavljuje 3 puta
Karakter e se pojavljuje 2 puta
Karakter g se pojavljuje 1 puta
Karakter i se pojavljuje 3 puta
Karakter j se pojavljuje 3 puta
Karakter l se pojavljuje 1 puta
Karakter m se pojavljuje 2 puta
Karakter n se pojavljuje 2 puta
Karakter o se pojavljuje 2 puta
Karakter r se pojavljuje 3 puta
Karakter v se pojavljuje 1 puta
Karakter z se pojavljuje 1 puta
Karakter P se pojavljuje 1 puta
```

[Rešenje 3.1.8]

**Zadatak 3.1.9** Napisati program koji učitava jednu liniju teksta i ispisuje koliko puta se pojavilo svako od slova engleskog alfabeta u unetom tekstu. Ne praviti razliku između malih i velikih slova.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Tasi, tasi, TaNaNa i SVILENA marama.....
a:9 b:0 c:0 d:0 e:1 f:0 g:0 h:0 i:4 j:0 k:0 l:1 m:2
n:3 o:0 p:0 q:0 r:1 s:3 t:3 u:0 v:1 w:0 x:0 y:0 z:0
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Mihailo Petrovic Alas (6 Maj 1868 - 8 Jun 1943)
a:4 b:0 c:1 d:0 e:1 f:0 g:0 h:1 i:3 j:2 k:0 l:2 m:2
n:1 o:2 p:1 q:0 r:1 s:1 t:1 u:1 v:1 w:0 x:0 y:0 z:0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
```

```
Alan Matison Tjuring (London, 23. jun 1912 - Cesir, 7. jun 1954)
```

```
a:3 b:0 c:1 d:1 e:1 f:0 g:1 h:0 i:3 j:3 k:0 l:2 m:1
```

```
n:7 o:3 p:0 q:0 r:2 s:2 t:2 u:3 v:0 w:0 x:0 y:0 z:0
```

[Rešenje 3.1.9]

**Zadatak 3.1.10** Takmičari na Beogradskom maratonu su označeni rednim brojevima počevši od 0, a vreme za koje su istrčali maraton je izraženo u minutima. Ovi podaci zadati su nizom celih brojeva, pri čemu indeks niza označava redni broj takmičara, a vrednosti u nizu označavaju vreme trčanja. Napisati sledeće funkcije za obradu navedenih podataka:

- (a) `void ucitaj(int a[], int n)` koja učitava elemente niza  $a$  dimenzije  $n$ .
- (b) `void ispisi(int a[], int n)` koja ispisuje elemente niza  $a$  dimenzije  $n$ .
- (c) `int suma(int a[], int n)` koja računa i vraća ukupno vreme trčanja svih takmičara.
- (d) `float prosek(int a[], int n)` koja računa i vraća prosečno vreme (aritmetičku sredinu) trčanja takmičara.
- (e) `int maksimum(int a[], int n)` koja izračunava i vraća najduže vreme trčanja takmičara.
- (f) `int pozicija_minimum(int a[], int n)` koja vraća redni broj pobednika Beogradskog maratona, tj. onog takmičara koji je najkraće trčao. U slučaju da ima više takvih takmičara, vratiti onog sa najmanjim indeksom.

Napisati program koji učitava podatke o rezultatima takmičara na maratonu i ispisuje učitane podatke, ukupno, prosečno i maksimalno vreme trčanja, kao i redni broj pobednika maratona. Maksimalan broj takmičara je 1000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
```

```
Unesite dimenziju niza:
```

```
5
```

```
Unesite elemente niza: 140 126 170 220 130
```

```
Vreme trcanja takmicara: 140 126 170 220 130
```

```
Ukupno vreme: 786
```

```
Prosecno vreme trcanja: 157.20
```

```
Maksimalno vreme trcanja: 220
```

```
Indeks pobednika: 1
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
```

```
Unesite dimenziju niza:
```

```
-5
```

```
Greska: neispravan unos.
```

[Rešenje 3.1.10]

**Zadatak 3.1.11** Napisati funkciju koja izračunava broj elemenata celobrojnog niza koji su manji od poslednjeg elementa niza. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje broj elemenata koji zadovoljavaju pomenuti uslov. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 4
|| Unesite elemente niza: 11 2 4 9
|| 2
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 7
|| Unesite elemente niza: 7 2 1 14 65 2 8
|| 4
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 5
|| Unesite elemente niza: 25 18 29 30 14
|| 0
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza:
|| -45
|| Greska: neispravan unos.
```

[Rešenje 3.1.11]

**Zadatak 3.1.12** Napisati funkciju koja izračunava broj parnih elemenata celobrojnog niza koji prethode maksimalnom elementu niza. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje broj elemenata koji zadovoljavaju pomenuti uslov. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 4
|| Unesite elemente niza: 11 2 4 9
|| 0
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 7
|| Unesite elemente niza: 7 2 1 14 65 2 8
|| 2
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 5
|| Unesite elemente niza: 25 18 29 30 14
|| 1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 105
|| Greska: neispravan unos.
```

[Rešenje 3.1.12]

### 3 Predstavljanje podataka

---

**Zadatak 3.1.13** Napisati funkciju `int zbir(int a[], int n, int i, int j)` koja računa zbir elemenata niza celih brojeva  $a$  dužine  $n$  od pozicije  $i$  do pozicije  $j$ . Napisati program koji učitava dimenziju niza, elemente niza i vrednosti  $i$  i  $j$  i zatim ispisuje zbir u datom opsegu. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 5
Unesite elemente niza: 11 5 6 48 8
Unesite vrednosti za i i j: 0 2
Zbir je: 22
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 3
Unesite elemente niza: -2 8 1
Unesite vrednosti za i i j: 1 12
Greska: neispravan unos.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 7
Unesite elemente niza: -2 5 9 11 6 -3 -4
Unesite vrednosti za i i j: 2 5
Zbir: 23
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 9 5 7 6
Unesite vrednosti za i i j: 2 2
Zbir: 7
```

[Rešenje 3.1.13]

**Zadatak 3.1.14** Napisati funkciju `float zbir_pozitivnih(float a[], int n, int k)` koja izračunava zbir prvih  $k$  pozitivnih elemenata realnog niza  $a$  dužine  $n$ . Napisati program koji učitava dimenziju niza, elemente niza i broj  $k$  i zatim ispisuje zbir prvih  $k$  pozitivnih elemenata niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
2.34 1 -12.7 5.2 -8 -6.2 7 14.2
Unesite vrednost k: 3
Zbir je: 8.54
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 3
Unesite elemente niza:
-6.598 -8.14 -15
Unesite vrednost k: 4
Zbir je: 0.00
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 7
Unesite elemente niza:
-35.11 5.29 -1.98 12.1 12.2 -3.33 -4.17
Unesite vrednost k: 15
Zbir: 29.59
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 3
Unesite elemente niza:
-0.11 5.29 -4.17
Unesite vrednost k: -15
Greska: neispravan unos.
```

[Rešenje 3.1.14]



**Zadatak 3.1.15** Napisati funkciju koja menja niz tako što razmenjuje mesta najmanjem i najvećem elementu niza. Ukoliko se neki od njih javlja više puta, u obzir uzeti prvo pojavljivanje. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje izmenjeni niz. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 5
Unesite elemente niza: 8 -2 11 19 4
8 19 11 -2 4

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 10
Unesite elemente niza:
46 -2 51 8 -5 66 2 8 3 14
46 -2 51 8 66 -5 2 8 3 14

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 145
Greska: neispravan unos.

```

[Rešenje 3.1.15]

**Zadatak 3.1.16** Napisati program koji vrši pretragu niza nadmorskih visina.

- Napisati funkciju koja proverava da li niz sadrži zadatu nadmorsku visinu  $m$ . Povratna vrednost funkcije je 1 ako je vrednost sadržana u nizu ili 0 ako nije.
- Napisati funkciju koja vraća vrednost prve pozicije na kojoj se nalazi element koji ima nadmorsku visinu  $m$  ili  $-1$  ukoliko element nije u nizu.
- Napisati funkciju koja vraća vrednost poslednje pozicije na kojoj se nalazi element koji ima nadmorsku visinu  $m$  ili  $-1$  ukoliko element nije u nizu.

Program učitava podatke o nadmorskim visinama i ceo broj  $m$  i ispisuje da li u nizu postoji podatak o unetoj nadmorskoj visini. Ukoliko postoji, ispisuje i poziciju prvog i poslednjeg pojavljivanja vrednosti  $m$  u nizu. Pozicije se broje od 0. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
7
Unesite podatke:
800 1100 -200 1400 -200 1100 800
Unesite vrednost m:
1100
Nadmorska visina 1100 se nalazi medju podacima.
Pozicija prvog pojavljivanja: 1
Pozicija poslednjeg pojavljivanja: 5
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
-5
Greska: neispravan unos.
```

[Rešenje 3.1.16]

**Zadatak 3.1.17** Marko skuplja sličice za Svetsko prvenstvo u fudbalu. Marko je primetio da mu se neke sličice ponavljaju i rešio je da ih razmeni sa drugarima. Napisati funkciju `int duplikati(int a[], int n, int b[])` koja od niza *a* dimenzije *n* formira niz *b* koji sadrži sve elemente niza *a* koji se pojavljuju bar dva puta u nizu. Funkcija kao povratnu vrednost vraća dimenziju niza *b*. Napisati program koji učitava Markove sličice i ispisuje sve duplikate. Maksimalan broj elemenata niza je 600. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza a:
4 11 4 6 8 4 6 6
Elementi niza b: 4 6
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 13
Unesite elemente niza a:
8 26 7 2 1 1 7 2 2 2 7 5 1
Elementi niza b: 7 2 1
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 2
Unesite elemente niza a:
9 5
Elementi niza b:
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 0
Greska: neispravan unos.
```

[Rešenje 3.1.17]

**Zadatak 3.1.18** Palindrom je tekst koji se isto čita i sa leve i sa desne strane. Napisati funkciju koja proverava da li elementi niza karaktera čine palindrom (zanemariti velika/mala slova). Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje da li je uneti niz karaktera palindrom. Maksimalan

broj elemenata niza je 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
15
Unesite elemente niza:
AnaVoliMilovana
Niz jeste palindrom.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
26
Unesite elemente niza:
Zanimljivo je programirati!
Niz nije palindrom.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
1
Unesite elemente niza:
a
Niz jeste palindrom.
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
226
Greska: neispravan unos.
```

[Rešenje 3.1.18]

**Zadatak 3.1.19** Napisati funkciju koja proverava da li su elementi celobrojnog niza uređeni neopadajuće. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje da li je pomenuti uslov ispunjen. Maksimalan broj elemenata niza je 300. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
8
Unesite elemente niza:
-40 -8 -8 2 30 30 46 200
Niz jeste uredjen neopadajuće.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
4
Unesite elemente niza:
4 23 15 30
Niz nije uredjen neopadajuće.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
1
Unesite elemente niza:
5
Niz jeste uredjen neopadajuće.
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
304
Greska: neispravan unos.
```

[Rešenje 3.1.19]

### 3 Predstavljanje podataka

---

**Zadatak 3.1.20** Svaki indeks niza označava jedan dan u mesecu, a elementi celobrojnog niza predstavljaju broj artikala koji se prodao tog dana. Napisati funkciju koja računa najduži uzastopnu seriju dana za koju važi da broj prodatih artikala nije opao. Napisati program koji učitava broj dana u mesecu, broj prodatih artikala za svaki dan u mesecu i zatim ispisuje dužinu izračunate serije. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 30
Unesite broj prodatih artikala:
89 171 112 67 119 36 181 157
49 96 73 116 21 172
140 0 23 71 157 135 11 166 21
56 56 87 103 183 148 174
Duzina najduzeg neopadajuceg
prodavanja je 6.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 31
Unesite broj prodatih artikala:
215 223 262 95 18 116 334 97
146 146 19 314 270 115 21 40
253 27 210 68 96 175 41 242
98 163 8 218 107 102
Duzina najduzeg neopadajuceg
prodavanja je 3.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: -5
Greska: neispravan unos.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 31
Unesite broj prodatih artikala:
-215 223 262 95 18 116 334 97
146 146 19 314 -270 115 21 40
253 27 210 68 96 175 41 242
98 163 -8 218 107 102
Greska: neispravan unos.
```

[Rešenje 3.1.20]

**Zadatak 3.1.21** Napisati funkciju koja određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva. Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje dužinu najduže serije jednakih elemenata niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
9 -1 2 2 2 2 80 -200
Duzina najduze serije je 4.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
9 9 0 -3 -3 -3 -3 72
Duzina najduze serije je 4.
```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
1 2 3 4 5 6 7 8
Duzina najduze serije je 1.

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 108
Greska: neispravan unos.

```

[Rešenje 3.1.21]

**Zadatak 3.1.22** Napisati funkciju koja određuje da li se jedan niz javlja kao podniz uzastopnih elemenata drugog niza. Zadatak rešiti na dva načina:

- Tako da elementi jednog niza moraju da budu uzastopni u drugom nizu.
- Tako da elementi jednog niza ne moraju da budu uzastopni u drugom nizu, ali je redosled pojavljivanja isti.

Napisati program koji učitava dimenzije i elemente dva niza i zatim ispisuje da li je neki od pomenutih uslova ispunjen. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 4
Unesite elemente niza: 90 -22 15 14
Elementi drugog niza cine
uzastopni podniz prvog niza.
Elementi drugog niza cine
podniz prvog niza.

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 4
Unesite elemente niza: 2 7 15 7
Elementi drugog niza ne cine
uzastopni podniz prvog niza.
Elementi drugog niza cine
podniz prvog niza.

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 4
Unesite elemente niza: 90 -22 200 1
Elementi drugog niza ne cine
uzastopni podniz prvog niza.
Elementi drugog niza ne
cine podniz prvog niza.

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 1
Unesite elemente niza: 90
Elementi drugog niza cine
uzastopni podniz prvog niza.
Elementi drugog niza cine
podniz prvog niza.

```

[Rešenje 3.1.22]

### 3 Predstavljanje podataka

**Zadatak 3.1.23** Za celobrojni niz  $a$  dimenzije  $n$  kažemo da je *permutacija* ako sadrži sve brojeve od 1 do  $n$ .

- (a) Napisati funkciju `void brojanje(int a[], int b[], int n)` koja na osnovu celobrojnog niza  $a$  dimenzije  $n$  formira niz  $b$  tako što  $i$ -ti element niza  $b$  odgovara broju pojavljivanja vrednosti  $i$  u nizu  $a$ .
- (b) Napisati funkciju `int permutacija(int a[], int n)` koja proverava da li je zadati niz permutacija. Funkcija vraća vrednost 1 ako je svojstvo ispunjeno, odnosno 0 ako nije. UPUTSTVO: *Koristiti funkciju brojanje iz tačke (a).*

Napisati program koji učitava dimenziju niza i elemente niza i ispisuje da li je uneti niz permutacija. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 5
Unesite elemente niza:
1 5 4 3 2
Uneti niz je permutacija.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza:
2 3 3 1 1 5
Uneti niz nije permutacija.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 1
Unesite elemente niza:
1
Uneti niz je permutacija.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 101
Greska: neispravan unos.
```

[Rešenje 3.1.23]

**Zadatak 3.1.24** Napisati program koji učitava dva cela broja  $x$  i  $y$  i proverava da li se uneti brojevi zapisuju pomoću istih cifara. UPUTSTVO: *Zadatak rešiti korišćenjem nizova.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 251 125
Brojevi se zapisuju istim ciframa.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 8898 9988
Brojevi se ne zapisuju istim ciframa.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: -7391 1397
Brojevi se zapisuju istim ciframa.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: -1 1
Brojevi se zapisuju istim ciframa.
```

[Rešenje 3.1.24]

**Zadatak 3.1.25** Napisati sledeće funkcije koje vrše transformacije celobrojnog niza:

- (a) Napisati funkciju koja obrće elemente niza.
- (b) Napisati funkciju koja rotira niz ciklično za jedno mesto ulevo.
- (c) Napisati funkciju koja rotira niz ciklično za  $k$  mesta ulevo.

Napisati program koji učitava dimenziju niza, elemente niza i pozitivan ceo broj  $k$  i ispisuje niz koji se dobije nakon obrtanja početnog niza, niz koji se dobije rotiranjem tog niza za jedno mesto ulevo i niz koji se dobije dodatnim rotiranjem tog niza za  $k$  mesta ulevo. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
6
7 -3 11 783 26 -19
Elementi niza nakon obrtanja:
-17 28 785 13 -1 9
Elementi niza nakon rotiranja za 1 mesto ulevo:
28 785 13 -1 9 -17
Unesite jedan pozitivan ceo broj:
3
Elementi niza nakon rotiranja za 3 mesto ulevo:
-1 9 -17 28 785 13
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
252
Greska: neispravan unos.
```

[Rešenje 3.1.25]

**Zadatak 3.1.26** Napisati funkciju `void ukrsti(int a[], int b[], int n, int c[])` koja formira niz  $c$  koji se dobija naizmeničnim raspoređivanjem elemenata nizova  $a$  i  $b$ , tj.  $c = [a_0, b_0, a_1, b_1, \dots, a_{n-1}, b_{n-1}]$ . Napisati program koji učitava dimenziju i elemente dva niza i ispisuje niz koji se dobija ukrštanjem unetih nizova. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova:
5
Unesite elemente niza a:
2 -5 11 4 8
Unesite elemente niza b:
3 3 9 -1 17
Rezultujući niz:
2 3 -5 3 11 9 4 -1 8 17
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova:
105
Greska: neispravan unos.
```

[Rešenje 3.1.26]

**Zadatak 3.1.27** Napisati funkciju `void spoji(int a[], int b[], int n, int c[])` koja od nizova  $a$  i  $b$  dimenzije  $n$  formira niz  $c$  čija prva polovina odgovara elementima niza  $b$ , a druga polovina elementima niza  $a$ , tj.  $c = [b_0, b_1, \dots, b_{n-1}, a_0, a_1, \dots, a_{n-1}]$ . Napisati program koji učitava dimenziju i elemente dva niza i ispisuje niz koji se dobija spajanjem unetih nizova na pomenuti način. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 3
Unesite elemente niza a: 4 -8 32
Unesite elemente niza b: 5 2 11
5 2 11 4 -8 32
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 4
Unesite elemente niza a: 1 0 -1 0
Unesite elemente niza b: 5 5 5 3
5 5 5 3 1 0 -1 0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 145
Greska: neispravan unos.
```

[Rešenje 3.1.27]

\* **Zadatak 3.1.28** Napisati funkciju `void spoji_sortirano(int a[], int b[], int n, int c[])` koja od nizova  $a$  i  $b$  dimenzije  $n$  koji su uređeni neopadajuće po vrednosti formira niz  $c$  koji je uređen na isti način. Napisati program koji učitava dimenziju i elemente uređenih nizova  $a$  i  $b$  i ispisuje niz koji se dobije spajanjem ovih nizova na pomenuti način. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.



*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 5
Unesite elemente sortiranog niza:
2 11 28 40 63
Unesite elemente sortiranog niza:
-19 -5 5 11 52
-19 -5 2 5 11 11 28 40 52 63

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 3
Unesite elemente sortiranog niza:
-2 4 8
Unesite elemente sortiranog niza:
6 15 19
-2 4 6 8 15 19

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 145
Greska: neispravan unos.

```

[Rešenje 3.1.28]

**Zadatak 3.1.29** Napisati funkciju `void promeni_redosled(int a[], int n)` koja menja redosled elementima niza  $a$  dimenzije  $n$  tako da se parni elementi niza nalaze na početku niza, a neparni na kraju. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji je izmenjen na pomenuti način. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Nije dozvoljeno koristiti pomoćne nizove.*

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 10
-2 8 11 53 59 20 17 -8 3 14
Rezultujući niz:
14 142 -6 -278 28 34 33 -69 -9 9

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 10
9 142 -9 -278 -69 33 34 28 -6 14
Rezultujući niz:
-2 8 14 -8 20 59 17 53 3 11

```

[Rešenje 3.1.29]

**Zadatak 3.1.30** Napisati funkciju koja iz datog niza briše sve elemente koji su prosti brojevi. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem pomenutih elemenata. Zadatak rešiti uz korišćenje pomoćnog niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### 3 Predstavljanje podataka

---

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 5  
|| Unesite elemente niza: 11 5 6 48 8  
|| 6 48 8
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 4  
|| Unesite elemente niza: 11 5 19 21  
|| 21
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 5  
|| Unesite elemente niza: 12 18 9 31 7  
|| 12 18 9
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 3  
|| Unesite elemente niza: -31 11 -19
```

#### Primer 5

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 5  
|| Unesite elemente niza: -2 15 -11 8 7  
|| 15 8
```

[Rešenje 3.1.30]

**Zadatak 3.1.31** Napisati funkciju koja iz datog niza briše sve neparne elemente. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem neparnih elemenata. Zadatak rešiti bez korišćenja pomoćnog niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 4  
|| Unesite elemente niza: 8 9 15 12  
|| 8 12
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 6  
|| Unesite elemente niza: 21 5 3 22 19 188  
|| 22 188
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 4  
|| Unesite elemente niza: 133 129 121 101
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 8  
|| Unesite elemente niza:  
|| 15 -22 -23 13 18 46 14 -31  
|| -22 18 46 14
```

[Rešenje 3.1.31]

**Zadatak 3.1.32** Napisati funkciju koja iz datog niza briše sve elemente koji nisu deljivi svojom poslednjom cifrom. Izuzetak su elementi čija je poslednja cifra nula i koje zbog toga treba zadržati. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem pomenutih elemenata. Zadatak rešiti bez korišćenja pomoćnog niza. Maksimalan broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
9
Unesite elemente niza a:
173 -25 23 7 17 25 34 61 -4612
-25 7 25 61 -4612
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
0
Greška: neispravan unos.
```

[Rešenje 3.1.32]

**Zadatak 3.1.33** Napisati funkciju koja iz datog niza briše sve brojeve koji nisu deljivi svojim indeksom. Ne razmatrati da li je u novom nizu, nakon brisanja i pomeranja, element deljiv svojim indeksom. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobije brisanjem pomenutih elemenata. Zadatak rešiti bez korišćenja pomoćnog niza. Maksimalan broj elemenata niza je 700. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Nulti element niza treba zadržati jer nije dozvoljeno deljenje nulom.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 10
Unesite elemente niza:
4 2 1 6 7 8 10 2 16 3
Novi niz:
4 2 6 16
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 10
Unesite elemente niza:
-8 5 10 6 7 10 8 2 16 27
Novi niz:
-8 5 10 6 10 16 27
```

[Rešenje 3.1.33]

**Zadatak 3.1.34** Korišćenjem nizova moguće je predstaviti skupove podataka. Napisati program koji demonstrira osnovne operacije nad skupovima (uniju, presek i razliku). Pomoću dva niza predstaviti dva skupa celih brojeva. Maksimalan broj elemenata niza je 500. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 5
Unesite elemente niza a: 1 2 3 4 5
Unesite broj elemenata niza b: 3
Unesite elemente niza b: 5 4 9
Unija: 1 2 3 4 5 9
Presek: 4 5
Razlika: 1 2 3
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 3
Unesite elemente niza a: 11 4 -5
Unesite broj elemenata niza b: 2
Unesite elemente niza b: 18 9
Unija: 11 4 -5 18 9
Presek:
Razlika: 11 4 -5
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 6
Unesite elemente niza a: 12 7 9 12 5 1
Greska: skup ne moze imati duplikate.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: -2
Greska: neispravan unos.
```

[Rešenje 3.1.34]

**Zadatak 3.1.35** Prilikom ulaska u banku klijent dobija redni broj, a u nizu se čuva redosled opsluživanja klijenata. Tako, prvi klijent u nizu će biti prvi uslužen, a klijent koji je poslednji došao se nalazi na kraju niza. Redni brojevi se izdaju počevši od 1 svakog radnog dana, ali se niz za redosled stalno menja. Dodatno, postoje specijalni klijenti (npr. oni koji podižu stambeni kredit) koji mogu dobiti i negativan redni broj da bi se razlikovali od uobičajenih usluga koje banka omogućava. Pomozite radniku obezbeđenja da lakše prati redosled opsluživanja klijenata.

- (a) Napisati funkciju koja ubacuje datog klijenta sa rednim brojem  $x$  na kraj niza.
- (b) Napisati funkciju koja ubacuje datog klijenta sa rednim brojem  $x$  na početak niza (lica sa posebnim potrebama, trudnice, stara lica i ostale ugrožene kategorije).
- (c) Napisati funkciju koja ubacuje datog klijenta sa rednim brojem  $x$  na datu poziciju  $k$  (manje prioritetna lica, recimo službena lica ili roditelji sa decom, poziciju  $k$  bira radnik obezbeđenja).
- (d) Napisati funkciju koja izbacuje prvi element niza (usluženi klijent).
- (e) Napisati funkciju koja izbacuje poslednji element niza (klijent je odustao jer je shvatio da ima mnogo klijenata ispred njega).
- (f) Napisati funkciju koja izbacuje element sa date pozicije  $k$  (klijent je odustao jer je dugo čekao).

Napisati program koji testira rad funkcija. Maksimalan broj klijenata u jednom danu je 2000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite trenutni broj klijenata:
8
Unesite niz sa rednim brojevima klijenata:
2 5 -2 16 33 19 8 11
Unesite klijenta kojeg treba ubaciti u niz:
35
Niz nakon ubacivanja klijenta: 2 5 -2 16 33 19 8 11 35
Unesite prioritetnog klijenta kojeg treba ubaciti u niz:
36
Niz nakon ubacivanja klijenta: 36 2 5 -2 16 33 19 8 11 35
Unesite prioritetnog klijenta kojeg treba ubaciti u niz i njegovu poziciju:
-6 2
Niz nakon ubacivanja klijenta: 36 2 -6 5 -2 16 33 19 8 11 35
Niz nakon odlaska klijenta: 2 -6 5 -2 16 33 19 8 11 35
Niz nakon odlaska poslednjeg klijenta: 2 -6 5 -2 16 33 19 8 11
Unesite redni broj klijenta koji je napustio red:
-2
Niz nakon odlaska klijenta: 2 -6 5 16 33 19 8 11
```

[Rešenje 3.1.35]

## 3.2 Rešenja

### Rešenje 3.1.1

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

/* Predprocesorska direktiva kojom se definise maksimalan broj
   elemenata niza. */
#define MAKS 100

int main()
{
    /* Deklaracije potrebnih promenljivih. */
    int a[MAKS];
    int n, i;

    /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
       ulaza. */
    printf("Unesite dimenziju niza:\n");
    scanf("%d", &n);
    if (n <= 0 || n > MAKS)
    {
```

### 3 Predstavljanje podataka

```
20     printf("Greska: neispravan unos.\n");
    /* Za izlazak iz programa moze da se koristi i funkcija exit.
22     Argument EXIT_FAILURE oznacava da je doslo do neke greske
    pri izvršavanju programa. Deklaracija ove funkcije se nalazi
24     u zaglavlju stdlib.h. */
    exit(EXIT_FAILURE);
26 }

28 /* Ucitavaju se elementi niza. */
    printf("Unesite elemente niza:\n");
30     for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
32
34 /* Ispisuju se elementi niza na parnim pozicijama. */
    printf("Elementi niza na parnim pozicijama:\n");
    for (i = 0; i < n; i += 2)
36         printf("%d ", a[i]);
    printf("\n");
38
40 /* Ispisuju se parni elementi niza. */
    printf("Parni elementi niza:\n");
    for (i = 0; i < n; i++)
42         if (a[i] % 2 == 0)
            printf("%d ", a[i]);
44     printf("\n");

46 /* Kada se funkciji exit prosledi EXIT_SUCCESS to znaci da se
    program uspesno završio. Efekat je isti kao i da je navedeno
48     return 0; na ovom mestu. */
    exit(EXIT_SUCCESS);
50 }
```

#### Rešenje 3.1.2

```
1  #include <stdio.h>
    #include <stdlib.h>
3
    #define MAKS 100
5
6  int main()
7  {
    /* Deklaracija potrebnih promenljivih. */
9     float brojevi[MAKS];
    int n, i;
11
12     /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
    ulaza. */
13     printf("Unesite dimenziju niza: ");
15     scanf("%d", &n);
    if (n <= 0 || n > MAKS)
17     {
```

```

19     printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
21 }

23 /* Ucitavaju se elementi niza. */
    printf("Unesite elemente niza:\n");
    for (i = 0; i < n; i++)
25         scanf("%f", &brojevi[i]);

27 /* Ukoliko je i-ti element niza brojevi[i] negativan broj,
    kvadrira se tako sto se pomnozi sa samim sobom. */
29 for (i = 0; i < n; i++)
    if (brojevi[i] < 0)
31         brojevi[i] *= brojevi[i];

33 /* Ispisuje se novodobijeni niz. */
    for (i = 0; i < n; i++)
35         printf("%g ", brojevi[i]);
    printf("\n");
37
    exit(EXIT_SUCCESS);
39 }

```

### Rešenje 3.1.3

```

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

4 #define MAKS 100

6 int main()
{
8     /* Deklaracija potrebnih promenljivih. */
    int a[MAKS];
10    int b[MAKS];
    int n, i, skalarni_proizvod;

12    /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
    ulaza. */
14    printf("Unesite dimenziju vektora: ");
    scanf("%d", &n);
16    if (n <= 0 || n > MAKS)
    {
18        printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
20    }

22    /* Ucitavaju se koordinate vektora. */
    printf("Unesite koordinate vektora a: ");
24    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
26

```

### 3 Predstavljanje podataka

---

```
printf("Unesite koordinate vektora b: ");
28 for (i = 0; i < n; i++)
    scanf("%d", &b[i]);
30
/* Izracunava se skalarni proizvod po zadatoj formuli. */
32 skalarni_proizvod = 0;
for (i = 0; i < n; i++)
34     skalarni_proizvod += a[i] * b[i];

/* Ispis rezultata. */
36 printf("Skalarni proizvod: %d\n", skalarni_proizvod);
38
exit(EXIT_SUCCESS);
40 }
```

#### Rešenje 3.1.4

```
1 #include <stdio.h>
#include <stdlib.h>
3
#define MAKS 100
5
int main()
7 {
    /* Deklaracija potrebnih promenljivih. */
9     int brojevi[MAKS];
    int n, i, k, indikator;
11
    /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
13     ulaza. */
    printf("Unesite dimenziju niza: ");
15     scanf("%d", &n);
    if (n <= 0 || n > MAKS)
17     {
        printf("Greska: neispravan unos.\n");
19         exit(EXIT_FAILURE);
    }
21
    /* Ucitavaju se elementi niza. */
23     printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
25         scanf("%d", &brojevi[i]);

    /* Ucitava se broj k i proverava se njegova ispravnost. */
27     printf("Unesite broj k: ");
29     scanf("%d", &k);
    if (k == 0)
31     {
        printf("Greska: neispravan unos.\n");
33         exit(EXIT_FAILURE);
    }
}
```



```

35  /* Promenljiva koja cuva informaciju o tome da li je u nizu
37     postojao element koji je deljiv brojem k. Inicijalna vrednost
        je 0. */
39  indikator = 0;

41  /* Ukoliko je element niza deljiv brojem k, indikator se
        postavlja na 1 i ispisuje se indeks tog elementa. */
43  for (i = 0; i < n; i++)
44  {
45      if (brojevi[i] % k == 0)
46      {
47          indikator = 1;
48          printf("%d ", i);
49      }
50  }

51  /* Ukoliko je indikator jednak nuli to znaci da ne postoji
53     element u nizu koji je deljiv brojem k. */
54  if (indikator == 0)
55      printf("U nizu nema elemenata koji su deljivi brojem %d.\n", k);
56
57  exit(EXIT_SUCCESS);
}

```

### Rešenje 3.1.5

```

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

4  /* Indeksiranje autobusa pocinje od 1, pa zato maksimalna
        dimenzija niza mora biti 201, a ne 200. */
6  #define MAKS 201

8  int main()
9  {
10     /* Deklaracija potrebnih promenljivih. */
11     int n, niz[MAKS], i;
12     int k, t, m;

13
14     /* Ucitava se dimenzija niza. */
15     printf("Unesite broj autobusa: ");
16     scanf("%d", &n);

17
18     /* Vrsi se provera ispravnosti ulaza. */
19     if (n <= 0 || n > MAKS)
20     {
21         printf("Greska: neispravan unos.\n");
22         exit(EXIT_FAILURE);
23     }
24

```

### 3 Predstavljanje podataka

```
26  /* Ucitavaju se vremena putovanja. */
    printf("Unesite vreme putovanja:\n");
    for (i = 1; i <= n; i++)
28      scanf("%d", &niz[i]);

30  /* Ucitavaju se redni brojevi autobusa cije se vreme putovanja
    menja i vrednost kasnjenja. */
32  printf("Unesite vrednosti k, t i m:\n");
    scanf("%d%d%d", &k, &t, &m);

34

36  /* Vrsi se provera ispravnosti ulaza. */
    if (k <= 0 || k > n || t <= 0 || t > n || m < 0)
    {
38      printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
40  }

42  /* Azuriraju se vremena putovanja. */
    for (i = k; i <= t; i++)
44      niz[i] += m;

46  /* Ispis rezultata. */
    printf("Vreme putovanja nakon izmena:");
48    for (i = 1; i <= n; i++)
        printf("%d ", niz[i]);
50    printf("\n");

52    exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.6

```
1  #include<stdio.h>
    #include<stdlib.h>
3
5  #define BROJ_CIFARA 10

    int main()
7  {
    /* Deklaracije potrebnih promenljivih. */
9    int x, x_original, cifra, i;
    int brojac[BROJ_CIFARA];

11

13    /* Ucitava se ceo broj sa standardnog ulaza. */
    printf("Unesite ceo broj:\n");
    scanf("%d", &x);

15

17    /* Cuva se njegova x_originalna vrednost zbog finalnog ispisa. */
    x_original = x;
    x = abs(x);
19
```

```

21  /* Svaki element niza brojaci predstavlja brojac za jednu od
    cifara: brojac[0] predstavlja broj nula u zapisu broja x
    brojac[1] predstavlja broj jedinica u zapisu broja x ...
23  brojac[9] predstavlja broj devetki u zapisu broja x. */

25  /* Brojaci se na pocetku inicijalizuju nulama. */
    for (i = 0; i < BROJ_CIFARA; i++)
27      brojaci[i] = 0;

29  /* Sve dok ima cifara u zapisu broja x */
    do {
31      /* Izdvaja se krajnja desna cifara. */
        cifra = x % 10;

33      /* Uvecava se njen broj pojavljivanja. */
        brojaci[cifra]++;

35      /* Prelazi se na analiziranje sledece cifre. */
        x /= 10;
37    } while (x);

41  /* Ispisuju se informacije o ciframa koje se nalaze u zapisu
    broja x. */
    for (i = 0; i < BROJ_CIFARA; i++)
43    {
45        if (brojaci[i])
            {
47            printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n",
                    x_original, i, brojaci[i]);
49            }
        }

51    exit(EXIT_SUCCESS);
53 }

```

### Rešenje 3.1.7

```

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

3

5  #define MAKS 100

7  int main()
    {
9      /* Deklaracije potrebnih promenljivih. */
        char karakteri[MAKS];
        char c;
11     int i, n;

13     /* Ucitava se karakter po karakter sa standardnog ulaza sve dok
        se ne unese * ili se ne prekorači maksimalni broj karaktera. */

```

### 3 Predstavljanje podataka

---

```
15  for (i = 0; i < MAKS; i++)
16  {
17      printf("Unesite karakter: ");
18      scanf("%c", &c);
19      /* Cita se znak za novi red nakon unetog karaktera. */
20      getchar();
21
22      /* Ukoliko je unet karakter * izlazi se iz petlje. */
23      if (c == '*')
24          break;
25
26      /* Procitani karakter se smesta u niz. */
27      karakteri[i] = c;
28  }
29
30  /* Broj unetih karaktera nakon izlaska iz petlje je i. */
31  n = i;
32
33  /* Ispis karaktera u obrnutom redosledu. */
34  for (i = n-1; i >= 0; i--)
35      printf("%c ", karakteri[i]);
36  printf("\n");
37
38  exit(EXIT_SUCCESS);
39 }
```

#### Rešenje 3.1.8

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define BROJ_CIFARA 10
5  #define DUZINA_ALFABETA 26
6
7  /* Pomocna funkcija za ispis elemenata niza.
8   Vrednost n oznacava broj elemenata niza
9   (ima vrednost 10 ili 26).
10  Karakter c oznacava prvi karakter za datu kategoriju
11  ('a' za mala slova, 'A' za velika i '0' za cifre). */
12  void ispisi(int niz[], int n, char c)
13  {
14      int i;
15      for (i = 0; i < n; i++)
16      {
17          if (niz[i] != 0)
18              printf("Karakter %c se pojavljuje %d puta\n", c + i, niz[i]);
19      }
20  }
21
22  /* Funkcija inicijalizuje niz postavljajuci vrednosti svih
23   elemenata na nulu. */
```

```

24 void inicijalizuj(int niz[], int n)
25 {
26     int i;
27     for (i = 0; i < n; i++)
28         niz[i] = 0;
29 }
30
31 int main()
32 {
33     /* Deklaracije nizova brojaca za cifre, mala i velika slova. */
34     int cifre[BROJ_CIFARA];
35     int mala_slova[DUZINA_ALFABETA];
36     int velika_slova[DUZINA_ALFABETA];
37
38     /* Deklaracije pomocnih promenljivih. */
39     int c;
40
41     /* Brojaci se na pocetku inicijalizuju nulama. */
42     inicijalizuj(cifre, BROJ_CIFARA);
43     inicijalizuj(mala_slova, DUZINA_ALFABETA);
44     inicijalizuj(velika_slova, DUZINA_ALFABETA);
45
46     /* Ucitavaju se karakteri sve do kraja ulaza. */
47     while ((c = getchar()) != EOF)
48     {
49         if (c >= 'A' && c <= 'Z')
50         {
51             /* Ako je procitani karakter veliko slovo uvecava se broj
52              pojavljivanja odgovarajuceg velikog slova. Indeks velikog
53              slova u nizu se odredjuje oduzimanjem slova A.
54              Na taj nacin slovo 'A' ce imati indeks 0, slovo 'B' indeks
55              1, itd.*/
56             velika_slova[c - 'A']++;
57         }
58         else if (c >= 'a' && c <= 'z')
59         {
60             /* Ako je procitani karakter malo slovo uvecava se broj
61              pojavljivanja odgovarajuceg malog slova. */
62             mala_slova[c - 'a']++;
63         }
64         else if (c >= '0' && c <= '9')
65         {
66             /* Ako je procitani karakter cifra uvecava se broj
67              pojavljivanja odgovarajuce cifre. */
68             cifre[c - '0']++;
69         }
70     }
71
72     /* Ispisuju se trazene informacije. */
73     ispisi(cifre, BROJ_CIFARA, '0');
74     ispisi(mala_slova, DUZINA_ALFABETA, 'a');
75     ispisi(velika_slova, DUZINA_ALFABETA, 'A');

```

### 3 Predstavljanje podataka

---

```
76     exit(EXIT_SUCCESS);
78 }
```

#### Rešenje 3.1.9

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

#define DUZINA_ALFABETA 26

/* Pomocna funkcija za ispis elemenata niza. */
void ispisi(int niz[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        printf("%c:%d ", 'a' + i, niz[i]);
    putchar('\n');
}

/* Funkcija inicijalizuje niz postavljajuci vrednosti svih
   elemenata na nulu. */
void inicijalizuj(int niz[], int n)
{
    int i;
    for (i = 0; i < n; i++)
        niz[i] = 0;
}

int main()
{
    /* Deklaracija potrebnih promenljivih. */
    int mala_slova[DUZINA_ALFABETA];
    int c;

    /* Inicijalizacija niza brojaca na nule. */
    inicijalizuj(mala_slova, DUZINA_ALFABETA);

    /* 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']++;
    }

    /* Ispis rezultata. */
```

```
46     ispisi(mala_slova, DUZINA_ALFABETA);  
48     exit(EXIT_SUCCESS);  
}
```

### Rešenje 3.1.10

```
#include <stdio.h>  
2 #include <stdlib.h>  
  
4 #define MAKS 1000  
  
6 /* Funkcija ucitava elemente niza. */  
void ucitaj(int a[], int n)  
8 {  
    int i;  
10    printf("Unesite elemente niza: ");  
    for (i = 0; i < n; i++)  
12        scanf("%d", &a[i]);  
}  
  
14 /* Funkcija ispisi elemente niza. */  
16 void ispisi(int a[], int n)  
{  
18     int i;  
    for (i = 0; i < n; i++)  
20        printf("%d ", a[i]);  
    printf("\n");  
22 }  
  
24 /* Funkcija racuna sumu elemenata niza. */  
int suma(int a[], int n)  
26 {  
    int i;  
28     int suma_elementa = 0;  
    for (i = 0; i < n; i++)  
30        suma_elementa += a[i];  
    return suma_elementa;  
32 }  
  
34 /* Funkcija racuna proseku vrednost elemenata niza. */  
float prosek(int a[], int n)  
36 {  
    int suma_elementa = suma(a, n);  
38     return (float) suma_elementa / n;  
}  
  
40 /* Funkcija izracunava maksimum elemenata niza. */  
42 int maksimum(int a[], int n)  
{  
44     int najveci, i;
```

```
46     najveci = a[0];
47     for (i = 1; i < n; i++)
48         if (a[i] > najveci)
49             najveci = a[i];
50
51     return najveci;
52 }
53
54 /* Funkcija izracunava poziciju maksimalnog elementa u nizu. */
55 int pozicija_maksimuma(int a[], int n)
56 {
57     int najveci, pozicija_najveceg;
58     int i;
59
60     najveci = a[0];
61     pozicija_najveceg = 0;
62
63     for (i = 1; i < n; i++)
64     {
65         if (a[i] < najveci)
66         {
67             najveci = a[i];
68             pozicija_najveceg = i;
69         }
70     }
71
72     return pozicija_najveceg;
73 }
74
75 int main()
76 {
77     /* Deklaracija potrebnih promenljivih. */
78     int a[MAKS];
79     int n;
80
81     /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
82        ulaza. */
83     printf("Unesite dimenziju niza:");
84     scanf("%d", &n);
85     if (n <= 0 || n > MAKS)
86     {
87         printf("Greska: neispravan unos.\n");
88         exit(EXIT_FAILURE);
89     }
90
91     /* Ucitavaju se elementi niza. */
92     ucitaj(a, n);
93
94     /* Ispisuju se elementi niza. */
95     printf("Vreme trcanja takmicara: ");
96     ispisi(a, n);
```



```

98  /* Ispis ukupnog, prosecnog i maksimalnog vremena. */
    printf("Ukupno vreme: %d\n", suma(a, n));
100  printf("Prosecno vreme trcanja: %.2f\n", prosek(a, n));
    printf("Maksimalno vreme trcanja: %d\n", maksimum(a, n));

102

    /* Ispis indeksa pobednika. */
104  printf("Indeks pobednika: %d\n", pozicija_maksimuma(a, n));

106  exit(EXIT_SUCCESS);
}

```

### Rešenje 3.1.11

```

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

4  #define MAKS 100

6  /* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
8  {
    int i;
10  printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
12  {
        scanf("%d", &a[i]);
    }

14

    /* Funkcija prebrojavanje vraca broj elemenata niza koji su manji
16  od poslednjeg elementa. */
    int prebrojavanje(int a[], int n)
18  {
        int i;
20  int broj_manjih = 0;

22  /* Prebrojavaju se elementi niza za koje vazi da su manji od
        poslednjeg elementa (a[n-1]). Petlja ide od prvog do
24  predposlednjeg elementa. */
        for (i = 0; i < n - 1; i++)
26  {
            if (a[i] < a[n - 1])
28  {
                broj_manjih++;
            }
        }

30  return broj_manjih;
32  }

34  int main()
    {
36  /* Deklaracija potrebnih promenljivih. */
        int a[MAKS];

```

### 3 Predstavljanje podataka

---

```
38  int n;

40  /* Ucitava se dimenzija niza i vrsi se provera
    ispravnosti ulaza. */
42  printf("Unesite dimenziju niza: ");
    scanf("%d", &n);
44  if (n <= 0 || n > MAKS)
    {
46      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
48  }

50  /* Ucitavaju se elementi niza. */
    ucitaj(a, n);

52
    /* Ispis rezultata. */
54  printf("%d\n", prebrojavanje(a, n));

56  exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.12

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

3
    #define MAKS 100

5
    /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int a[], int n)
    {
9      int i;
        printf("Unesite elemente niza: ");
11     for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
13 }

15 /* Funkcija vraca poziciju najveceg elementa niza. */
    // int pozicija_najveceg(int a[], int n)
17 // {
    //     int i, pozicija = 0;
19 //     /* Prolazi se kroz niz i ako se naidje na element cija je
    //         vrednost veca od trenutno najveceg (a[pozicija]), vrsi
21 //         se azuriranje pozicije trenutno najveceg. */
    //     for(i=1; i<n; i++)
23 //         if(a[i] > a[pozicija])
    //             pozicija = i;
25 //
    //     return pozicija;
27 // }
```

```

29 /* Funkcija vraca broj parnih elemenata niza koji prethode
    maksimalnom elementu niza. */
31 // int prebrojavanje(int a[], int n)
    // {
33 //     int i;
    //     int pozicija_maksimuma = pozicija_najveceg(a,n);
35 //
    //     int broj_parnih = 0;
37 //     for (i = 0; i < pozicija_maksimuma; i++) {
    //         if (a[i] % 2 == 0) {
39 //             broj_parnih++;
    //         }
41 //     }
    //
43 //     return broj_parnih;
    // }

45 /* Zadatak se moze resiti i jednom prolaskom kroz niz.
47 Ideja je da se paralelno radi pretraga maksimalnog elementa
    i prebrojavanje parnih elemenata koji mu prethode.

49
    Ovo moze da se uradi sa dva brojacu parnih elemenata:
51 1. broj_parnih - brojac koji cuva broj parnih koji prethode
    trenutnom maksimumu.
53 2. broj_parnih_izmedju - brojac koji cuva broj parnih elemenata
    koji se nalaze iza trenutnog maksimuma

55
    Svaki put kada se maksimum azurira, na broj parnih se doda
57 broj parnih koji se prebrojao izmedju dva azuriranja,
    a broj_parnih_izmedju se vraca na nulu. */
59 int prebrojavanje_jednim_prolazom(int a[], int n)
    {
61     int i;
    int pozicija_maksimuma = 0;
63     int broj_parnih = 0;
    int broj_parnih_izmedju = 0;
65
    for (i = 0; i < n; i++)
67     {
        if(a[i] > a[pozicija_maksimuma])
69         {
            pozicija_maksimuma = i;
            broj_parnih += broj_parnih_izmedju;
            broj_parnih_izmedju = 0;
73         }

75         if (a[i] % 2 == 0)
            broj_parnih_izmedju++;
77     }

79     return broj_parnih;
    }

```

### 3 Predstavljanje podataka

---

```
81 int main()
82 {
83     /* Deklaracija potrebnih promenljivih. */
84     int a[MAKS];
85     int n;
86
87     /* Ucitava se dimenzija niza i vrsi se provera
88        ispravnosti ulaza. */
89     printf("Unesite dimenziju niza: ");
90     scanf("%d", &n);
91     if (n <= 0 || n > MAKS)
92     {
93         printf("Greska: neispravan unos.\n");
94         exit(EXIT_FAILURE);
95     }
96
97     /* Ucitavaju se elementi niza. */
98     ucitaj(a, n);
99
100
101     /* Ispis rezultata. */
102     /* I nacin:
103        printf("%d\n", prebrojavanje(a, n)); */
104
105     /* II nacin: jednim prolaskom kroz niz: */
106     printf("%d\n", prebrojavanje_jednim_prolazom(a, n));
107
108     exit(EXIT_SUCCESS);
109 }
```

#### Rešenje 3.1.13

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n)
8 {
9     int i;
10    printf("Unesite elemente niza: ");
11    for (i = 0; i < n; i++)
12        scanf("%d", &a[i]);
13 }
14
15 /* Funkcija racuna zbir elemenata niza od pozicije i do
16    pozicije j. */
17 int zbir(int a[], int i, int j)
18 {
19     int k;
```

```

21     int rezultat = 0;

23     /* Obilaze se elementi niza iz zadatog opsega. */
24     for (k = i; k <= j; k++) {
25         rezultat += a[k];
26     }

27     /* Vraca se izracunata vrednost. */
28     return rezultat;
29 }

31 int main()
32 {
33     /* Deklaracije potrebnih promenljivih. */
34     int n, i, j;
35     int a[MAKS];

37     /* Ucitava se dimenzija niza i vrsi se provera
38        ispravnosti ulaza. */
39     printf("Unesite dimenziju niza: ");
40     scanf("%d", &n);
41     if (n <= 0 || n > MAKS)
42     {
43         printf("Greska: neispravan unos.\n");
44         exit(EXIT_FAILURE);
45     }

47     /* Ucitavaju se elementi niza. */
48     ucitaj(a, n);

49     /* Ucitavaju se vrednosti granica i vrsi se provera
50        ispravnosti ulaza. */
51     printf("Unesite vrednosti za i i j: ");
52     scanf("%d%d", &i, &j);
53     if (i < 0 || j < 0 || i > n - 1 || j > n - 1 || i > j)
54     {
55         printf("Greska: neispravan unos.\n");
56         exit(EXIT_FAILURE);
57     }

59     /* Ispis rezultata. */
60     printf("Zbir je: %d", zbir(a, i, j));

63     exit(EXIT_SUCCESS);
64 }

```

### Rešenje 3.1.14

```

1 #include <stdio.h>
2 #include <stdlib.h>

```

### 3 Predstavljanje podataka

---

```
4 #define MAKS 100

6 /* Funkcija učitava elemente niza dimenzije n. */
void ucitaj(float a[], int n)
8 {
    int i;
10    printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
12        scanf("%f", &a[i]);
}

14 /* Funkcija racuna zbir prvih k pozitivnih elemenata niza. */
16 float zbir_pozitivnih(float a[], int n, int k)
{
18     int i;
    float zbir = 0;

20    /* Obilazi se element po element niza. Postupak se završava
22     * ukoliko se dodje do kraja niza ili ukoliko se sabere k
23     * pozitivnih elemenata. */
24    for (i = 0; i < n && k > 0; i++) {
        if (a[i] >= 0) {
26            zbir += a[i];
            /* Umanjuje se brojac pozitivnih elemenata. */
28            k--;
        }
30    }

32    return zbir;
}

34 int main()
36 {
    /* Deklaracija potrebnih promenljivih. */
38    int n, k;
    float a[MAKS];

40    /* Učitava se dimenzija niza i vrsi se provera
42     * ispravnosti ulaza. */
    printf("Unesite dimenziju niza: ");
44    scanf("%d", &n);
    if (n <= 0 || n > MAKS)
46    {
        printf("Greska: neispravan unos.\n");
48        exit(EXIT_FAILURE);
    }

50    /* Ucitavaju se elementi niza. */
52    ucitaj(a, n);

54    /* Učitava se broj k i vrsi se provera ispravnosti ulaza. */
    printf("Unesite vrednost k: ");
```

```

56     scanf("%d", &k);
    if (k < 0 || k > n)
58     {
        printf("Greska: neispravan unos.\n");
60         exit(EXIT_FAILURE);
    }

62     /* Ispis rezultata. */
64     printf("Zbir je: %.2f\n", zbir_pozitivnih(a, n, k));

66     exit(EXIT_SUCCESS);
}

```

### Rešenje 3.1.15

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 100
5
   /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int a[], int n)
   {
9      int i;
      printf("Unesite elemente niza: ");
11     for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
13 }

15 /* Funkcija ispisuje elemente niza dimenzije n. */
   void ispisi(int a[], int n)
17 {
    int i;
19     for (i = 0; i < n; i++)
        printf("%d ", a[i]);
21     printf("\n");
}

23 /* Funkcija razmenjuje najmanji i najveći element niza. */
25 void razmeni_min_max(int brojevi[], int n)
   {
27     int i;
    /* Najvecim, kao i najmanjim elementom niza proglašava se nulti
29     element niza. Pozicije najvećeg i najmanjeg elementa se
        postavljaju na 0. */
31     int najveći = brojevi[0];
    int najmanji = brojevi[0];
33     int pozicija_najveceg = 0;
    int pozicija_najmanjeg = 0;
35
    /* U prolazu kroz niz trazi se najveći i najmanji element i

```

### 3 Predstavljanje podataka

---

```
37     pamte se njihove pozicije. */
38     for (i = 1; i < n; i++)
39     {
40         if (brojevi[i] > najveci)
41         {
42             najveci = brojevi[i];
43             pozicija_najveceg = i;
44         }
45
46         if (brojevi[i] < najmanji)
47         {
48             najmanji = brojevi[i];
49             pozicija_najmanjeg = i;
50         }
51     }
52
53     /* Zamenjuju se elementi na pozicijama pozicija_najmanjeg i
54     pozicija_najveceg. */
55     brojevi[pozicija_najveceg] = najmanji;
56     brojevi[pozicija_najmanjeg] = najveci;
57 }
58
59 int main()
60 {
61     /* Deklaracija potrebnih promenljivih. */
62     int brojevi[MAKS];
63     int n;
64
65     /* Ucitava se dimenzija niza i vrsi se provera
66     ispravnosti ulaza. */
67     printf("Unesite dimenziju niza: ");
68     scanf("%d", &n);
69     if (n <= 0 || n > MAKS)
70     {
71         printf("Greska: neispravan unos.\n");
72         exit(EXIT_FAILURE);
73     }
74
75     /* Ucitavaju se elementi niza. */
76     ucitaj(brojevi, n);
77
78     /* Razmenjuju se najmanji i najveći element. */
79     razmeni_min_max(brojevi, n);
80
81     /* Ispisuje se rezultujući niz. */
82     ispisi(brojevi, n);
83
84     exit(EXIT_SUCCESS);
85 }
```

#### Rešenje 3.1.16



```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS 100
5
6  /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int a[], int n)
8  {
9      int i;
10     printf("Unesite podatke: ");
11     for (i = 0; i < n; i++)
12         scanf("%d", &a[i]);
13 }
14
15 /* Funkcija proverava da li niz sadrzi zadatu vrednost m. */
16 int sadrzi(int a[], int n, int m)
17 {
18     int i;
19     /* Prolazi se kroz sve elemente niza i ukoliko se naidje na
20        element cija je vrednost jednaka m, kao povratna vrednost
21        funkcije se vraca 1. */
22     for (i = 0; i < n; i++) {
23         if (a[i] == m)
24             return 1;
25     }
26
27     /* Ako se stigne do kraja niza i ne naidje na vrednost koja je
28        jednaka broju m, to znaci da se broj ne nalazi u nizu i da
29        funkcija treba da vrati 0. */
30     return 0;
31 }
32
33 /* Funkcija vraca indeks prvog pojavljivanja elementa m
34    u nizu a ili -1 ukoliko se m ne nalazi u nizu a. */
35 int prvo_pojavljivanje(int a[], int n, int m)
36 {
37     int i;
38     for (i = 0; i < n; i++)
39         if (a[i] == m)
40             return i;
41
42     /* Ako se stigne do kraja niza i ne naidje na vrednost koja je
43        jednaka broju m, to znaci da se broj ne nalazi u nizu i da
44        funkcija treba da vrati -1. */
45     return -1;
46 }
47
48 /* Funkcija vraca indeks poslednjeg pojavljivanja elementa m
49    u nizu a ili -1 ukoliko se m ne nalazi u nizu a. */
50 int poslednje_pojavljivanje(int a[], int n, int m)
51 {
```

### 3 Predstavljanje podataka

---

```
52  int i;

54  /* Polazi se od kraja niza i poredi se element po element sa
    zadatim brojem m. */
56  for (i = n - 1; i >= 0; i--)
    if (a[i] == m)
58      return i;

60  /* Ako se stigne do pocetka niza i ne naidje na vrednost koja je
    jednaka broju m, to znaci da se broj ne nalazi u nizu i da
62  funkcija treba da vrati -1. */
    return -1;
64 }

66 int main()
67 {
68     /* Deklaracije potrebnih promenljivih. */
69     int a[MAKS];
70     int n, m, i;

72     /* Ucitava se dimenzija niza i vrsi se provera
    ispravnosti ulaza. */
74     printf("Unesite dimenziju niza: ");
75     scanf("%d", &n);
76     if (n <= 0 || n > MAKS)
77     {
78         printf("Greska: neispravan unos.\n");
79         exit(EXIT_FAILURE);
80     }

82     /* Ucitavaju se elementi niza. */
83     ucitaj(a, n);

84

85     /* Ucitava se vrednost za pretragu. */
86     printf("Unesite vrednost m:");
87     scanf("%d", &m);

88

89     /* Ispisuju se rezultati pretrage. */
90     if (sadrzi(a, n, m))
91     {
92         printf("Nadmorska visina %d se nalazi medju podacima.\n", m);

93         i = prvo_pojavljivanje(a, n, m);
94         printf("Pozicija prvog pojavljivanja: %d\n", i);

95         i = poslednje_pojavljivanje(a, n, m);
96         printf("Pozicija poslednjeg pojavljivanja: %d\n", i);
97     }
98     else
99     {
100         printf("Nadmorska visina %d se ne nalazi medju podacima.\n", m);
101     }
102     exit(EXIT_SUCCESS);
```

104 | }

## Rešenje 3.1.17

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 600
5
   /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int a[], int n)
   {
9      int i;
      printf("Unesite elemente niza a: ");
11     for (i = 0; i < n; i++)
         scanf("%d", &a[i]);
13 }

15 /* Funkcija ispisuje elemente niza dimenzije n. */
   void ispisi(int niz[], int n)
17 {
      int i;
19     for (i = 0; i < n; i++)
         printf("%d ", niz[i]);
21     printf("\n");
   }

23
   /* Funkcija proverava da li niz a dimenzije n sadrzi zadatu
25     vrednost x. Pretraga se vrši od prosledjene pozicije. */
   int sadrzi(int niz[], int n, int od_pozicije, int x)
27 {
      int i;
29     for (i = od_pozicije; i < n; i++)
         if (niz[i] == x)
31         return 1;

33     return 0;
   }

35
   /* Funkcija formira niz b tako sto u njega ubacuje sve elemente
37     niza a koji se u tom nizu pojavljuju bar dva puta. */
   int duplikati(int a[], int n, int b[])
39 {
      /* Promenljiva j je brojac elemenata rezultujuceg niza. */
41     int i, j = 0;

43     /* Obilazi se element po element niza a.
      Trenutni element je duplikat ukoliko se javlja jos neki put u
45     nizu a. Dovoljno je gledati da li se nalazi iza tekuceg
      elementa jer ako se nalazi ispred, onda je on vec obradjen
47     (i duplikat je detektovan).

```

### 3 Predstavljanje podataka

```

49     Element a[i] se dodaje u niz duplikata ako vazi:
      1. a[i] je duplikat
      2. a[i] se ne nalazi u nizu duplikata
51     Provera sadrzi(a, n, i+1, a[i]) proverava prvi uslov.
      Provera !sadrzi(b, j, 0, a[i]) proverava drugi uslov. */
53     for (i = 0; i < n; i++)
    {
55         if (sadrzi(a, n, i+1, a[i]) && !sadrzi(b, j, 0, a[i]))
        {
57             b[j] = a[i];
              j++;
59         }
    }
61
62     /* Povratna vrednost funkcije je duzina niza b. */
63     return j;
64 }
65
66 int main()
67 {
68     /* Deklaracija potrebnih promenljivih. */
69     int a[MAKS], b[MAKS];
70     int n_a, n_b;
71
72     /* Ucitava se dimenzija niza i vrsi se provera
73        ispravnosti ulaza. */
74     printf("Unesite broj n: ");
75     scanf("%d", &n_a);
76     if (n_a <= 0 || n_a > MAKS)
77     {
78         printf("Greska: neispravan unos.\n");
79         exit(EXIT_FAILURE);
80     }
81
82     /* Ucitavaju se podaci o slicicama. */
83     ucitaj(a, n_a);
84
85     /* Niz b se popunjava duplikatima iz a. */
86     n_b = duplikati(a, n_a, b);
87
88     /* Ispis rezultata. */
89     printf("Elementi niza b: ");
90     ispisi(b, n_b);
91
92     exit(EXIT_SUCCESS);
93 }
```

#### Rešenje 3.1.18

```

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

```
4  #include <ctype.h>
6
8  #define MAKS 200
10
12 /* Funkcija ucitava elemente niza dimenzije n. */
14 void ucitaj(char niz[], int n)
16 {
18     int i;
19     /* Ucitava se niz od n karaktera. */
20     printf("Unesite elemente niza: ");
21     for (i = 0; i < n; i++)
22         scanf("%c", &niz[i]);
23 }
24
26 /* Funkcija proverava da li je niz karaktera palindrom. */
28 int je_palindrom(char niz[], int n)
30 {
31     int i;
32     /* U petlji se porede niz[0] i niz[n-1], zatim niz[1] i niz[n-2]
33        itd. Ako se naidje na par koji se razlikuje - niz nije
34        palindrom. */
35     for (i = 0; i < n / 2; i++)
36     {
37         if (tolower(niz[i]) != tolower(niz[n - 1 - i]))
38             return 0;
39     }
40
41     /* Izvrsila se cela petlja => niz je palindrom. */
42     return 1;
43 }
44
46 int main()
48 {
49     /* Deklaracije potrebnih promenljivih. */
50     char niz[MAKS];
51     int n;
52
53     /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
54        ulaza. */
55     printf("Unesite dimenziju niza: ");
56     scanf("%d", &n);
57     if (n <= 0 || n > MAKS)
58     {
59         printf("Greska: neispravan unos.\n");
60         exit(EXIT_FAILURE);
61     }
62
63     /* Preskace se novi red nakon unosa dimenzije. Ovo se radi
64        jer sledi ucitavanje karaktera i bez ove linije, prvi
65        karakter koji bi se upisao u niz bi bio novi red. */
66     getchar();
67 }
```

### 3 Predstavljanje podataka

---

```
/* Ucitavaju se elementi niza. */
56 ucitaj(niz, n);

/* Ispis rezultata. */
58 if(je_palindrom(niz, n))
60     printf("Niz jeste palindrom.\n");
    else
62     printf("Niz nije palindrom.\n");

64 exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.19

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

4 #define MAKS 300

6 /* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
8 {
    int i;
10     printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
12         scanf("%d", &a[i]);
}

14 /* Funkcija vraca 1 ako je niz uredjen neopadajuce,
    a nulu inace. */
16 int uredjen_neopadajuce(int niz[], int n)
18 {
    int i;
20     for (i = 0; i < n - 1; i++)
    {
22         if (niz[i] > niz[i + 1])
            return 0;
24     }

26     return 1;
}

28 int main()
30 {
    /* Deklaracija potrebnih promenljivih. */
32     int n, niz[MAKS];

34     /* Ucitava se dimenzija niza i vrsi se provera
        ispravnosti ulaza. */
36     printf("Unesite dimenziju niza: ");
    scanf("%d", &n);
```

```

38  if (n <= 0 || n > MAKS)
39  {
40      printf("Greska: neispravan unos.\n");
41      exit(EXIT_FAILURE);
42  }

44  /* Ucitavaju se elementi niza. */
45  ucitaj(niz, n);

46

47  /* Ispis rezultata. */
48  if(uredjen_neopadajuce(niz, n))
49      printf("Niz jeste uredjen neopadajuce.\n");
50  else
51      printf("Niz nije uredjen neopadajuce.\n");
52
53  exit(EXIT_SUCCESS);
54 }

```

### Rešenje 3.1.20

```

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

/* 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. */
#define MAKS_DANA 32

/* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
{
    int i;
    printf("Unesite broj prodatih artikala: ");
    for (i = 0; i < n; i++)
    {
        scanf("%d", &a[i]);
        if(a[i] < 0)
        {
            printf("Greska: neispravan unos.\n");
            exit(EXIT_FAILURE);
        }
    }
}

/* Funkcija racuna duzinu najduzeg neopadajuceg podniza niza a. */
int najduzi_neopadajuci(int a[], int n)
{
    int i;
    /* Na pocetku i duzina trenutne serije i duzina maksimalne serije
       se inicijalizuju na 1. */
    int duzina_trenutne_serije = 1;
    int duzina_najduze_serije = 1;

```

### 3 Predstavljanje podataka

```
32
33     for (i = 1; i < n; i++) {
34         /* Proverava se da li uzastopni elementi ispunjavaju
35            neopadajući uslov. Ako je to slučaj uvecava se dužina
36            serije, a ako nije, dužina trenutne serije se vraća na 1,
37            kako bi se ispravno računala dužina sledeće serije. */
38         if (a[i] >= a[i - 1])
39             dužina_trenutne_serije++;
40         else
41             dužina_trenutne_serije = 1;
42
43         /* Ukoliko je trenutna dužina serije veća od dužine do sada
44            najduže serije, parametar za dužinu najduže serije se
45            postavlja na novu, veću vrednost. */
46         if (dužina_trenutne_serije > dužina_najduže_serije)
47             dužina_najduže_serije = dužina_trenutne_serije;
48     }
49
50     return dužina_najduže_serije;
51 }
52
53 int main()
54 {
55     /* Deklaracija potrebnih promenljivih. */
56     int a[MAKS_DANA], n;
57
58     /* Učitava se dimenzija niza i vrsi se provera
59        ispravnosti ulaza. */
60     printf("Unesite dimenziju niza: ");
61     scanf("%d", &n);
62     if (n <= 0 || n > MAKS_DANA)
63     {
64         printf("Greska: neispravan unos.\n");
65         exit(EXIT_FAILURE);
66     }
67
68     /* Učitavaju se elementi niza. */
69     učitaj(a, n);
70
71     /* Ispis rezultata. */
72     printf("Dužina najdužeg neopadajućeg prodavanja je %d.\n",
73           najduzi_neopadajući(a, n));
74
75     exit(EXIT_SUCCESS);
76 }
```

#### Rešenje 3.1.21

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
```



```
5 #define MAKS 100
6
7 /* Funkcija ucitava elemente niza dimenzije n. */
8 void ucitaj(int a[], int n)
9 {
10     int i;
11     printf("Unesite elemente niza: ");
12     for (i = 0; i < n; i++)
13         scanf("%d", &a[i]);
14 }
15
16 /* Funkcija vraca duzinu najduze serije jednakih elemenata niza. */
17 int najduza_serija(int a[], int n)
18 {
19     int i;
20     /* Na pocetku i duzina trenutne serije i duzina maksimalne serije
21     se inicijalizuju na 1. */
22     int trenutna_serija = 1;
23     int najduza_serija = 1;
24
25     for (i = 1; i < n; i++) {
26         /* Proverava se da li su uzastopni elementi jednaki. Ako je to
27         slucaj uvecava se duzina serije. Ako uzastopni elementi nisu
28         jednaki serija je prekinuta i parametar za duzinu serije se
29         postavlja ponovo na 1 da bi mogla da se racuna duzina
30         sledece serije. */
31         if (a[i] == a[i - 1])
32             trenutna_serija++;
33         else
34             trenutna_serija = 1;
35
36         /* Ukoliko je trenutna duzina serije veca od duzine do sada
37         najduze serije, parametar za duzinu najduze serije se
38         postavlja na novu, vecu vrednost. */
39         if (trenutna_serija > najduza_serija)
40             najduza_serija = trenutna_serija;
41     }
42
43     return najduza_serija;
44 }
45
46 int main()
47 {
48     /* Deklaracija potrebnih promenljivih. */
49     int n, a[MAKS];
50
51     /* Ucitava se dimenzija niza i vrši se provera
52     ispravnosti ulaza. */
53     printf("Unesite dimenziju niza: ");
54     scanf("%d", &n);
55     if (n <= 0 || n > MAKS)
56     {
```

```
    printf("Greska: neispravan unos.\n");
57    exit(EXIT_FAILURE);
}

59    /* Ucitavaju se elementi niza. */
61    ucitaj(a, n);

63    /* Ispis rezultata. */
    printf("Duzina najduze serije je %d.\n", najduza_serija(a, n));
65    exit(EXIT_SUCCESS);
67 }
```

#### Rešenje 3.1.22

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

4  #define MAKS 100

6  /* Funkcija ucitava elemente niza dimenzije n. */
    void ucitaj(int niz[], int n)
8  {
    int i;
10     printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
12         scanf("%d", &niz[i]);
}

14

/* Resenje pod a. */
16 int podniz_uzastopnih(int a[], int n, int b[], int m)
{
18     int i, j;

20     /* Prolaze se elementi prvog niza. Svaki element prvog niza moze
        biti pocetak podniza, odnosno pocetak drugog niza. */
22     for (i = 0; i + m - 1 < n; i++)
    {
24         /* Prolaze se elementi drugog niza. Za svaki element niza b
            proverava se da li je jednak odgovarajucem elementu niza a.
            Za niz a razmatra se da li podniz pocinje od pozicije i. Tako
26             0-ti element niza b je na poziciji i, 1. element je na
            poziciji i+1, 2. na poziciji i+2, ..., j-ti na poziciji
28             i+j. Ako uslov nije ispunjen, petlja se prekida i proverava
            se da li na sledecoj poziciji u nizu a pocinje podniz. */
30         for (j = 0; j < m; j++)
            if (a[i + j] != b[j])
32                 break;
34         /* Ako petlja nije prekinuta nakon ispitivanja, brojac za niz b
            je jednak dimenziji niza b, odnosno svi elementi niza b se
            uzastopno nalaze u nizu a. */
36     }
```

```
38     if (j == m)
39         return 1;
40 }
41
42 /* Ukoliko niz b jeste uzastopni podniz uslov u petlji ce u nekom
43 trenutku biti ispunjen i iz petlje i funkcije ce se izaci sa
44 return naredbom. Ipak, ako se to nije desilo i dalje se
45 izvrsava funkcija, onda niz b nije uzastopni podniz. */
46 return 0;
47 }
48
49 /* Resenje pod b. */
50 int podniz(int a[], int n, int b[], int m)
51 {
52     int i, j;
53
54     /* Petljom se prolaze elementi niza a. */
55     for (i = 0, j = 0; i < n && j < m; i++)
56     {
57         /* Svaki put kada se naidje na element niza b, brojac za niz b
58 se uvecava i proverava se da li se sledeci element niza b
59 nalazi u nizu a. */
60         if (a[i] == b[j])
61             j++;
62     }
63
64     /* Ukoliko se pronadju svi elementi niza b u nizu a, onda je
65 brojac za niz b jednak dimenziji niza b. U tom slucaju se
66 vraca vrednost 1, odnosno da niz jeste podniz. */
67     return j == m;
68 }
69
70 int main()
71 {
72     /* Deklaracija potrebnih promenljivih. */
73     int n, a[MAKS];
74     int m, b[MAKS];
75
76     /* Ucitava se dimenzija niza i vrši se provera
77 ispravnosti ulaza. */
78     printf("Unesite dimenziju niza: ");
79     scanf("%d", &n);
80     if (n <= 0 || n > MAKS)
81     {
82         printf("Greska: neispravan unos.\n");
83         exit(EXIT_FAILURE);
84     }
85
86     /* Ucitavaju se elementi prvog niza. */
87     ucitaj(a, n);
88
89     /* Ucitava se dimenzija niza i vrši se provera
```

### 3 Predstavljanje podataka

```
    ispravnosti ulaza. */
90 printf("Unesite dimenziju niza: ");
   scanf("%d", &m);
92 if (m <= 0 || m > MAKS)
   {
94     printf("Greska: neispravan unos.\n");
     exit(EXIT_FAILURE);
96 }

98 /* Ucitavaju se elementi drugog niza. */
   ucitaj(b, m);

100
   /* a) */
102 if (podniz_uzastopnih(a, n, b, m))
     printf("Elementi drugog niza cine uzastopni podniz "
104           "prvog niza.\n");
   else
106     printf("Elementi drugog niza ne cine uzastopni podniz "
           "prvog niza.\n");
108
   /* b) */
110 if (podniz(a, n, b, m))
     printf("Elementi drugog niza cine podniz prvog niza.\n");
112 else
     printf("Elementi drugog niza ne cine podniz prvog niza.\n");
114
   exit(EXIT_SUCCESS);
116 }
```

#### Rešenje 3.1.23

```
1 #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 100
5
   /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int niz[], int n)
   {
9     int i;

11    /* Ucitavaju se elementi niza. */
     printf("Unesite elemente niza: ");
13     for (i = 0; i < n; i++)
     {
15         scanf("%d", &niz[i]);

17         /* Niz moze sadrzati elemente koji nisu u opsegu od 1 do n.
           U tom slucaju taj niz nije permutacija. */
19         if (niz[i] <= 0 || niz[i] > n)
         {
```

```
21     printf("Uneti niz nije permutacija.\n");
22     exit(EXIT_SUCCESS);
23 }
24 }
25 }
26
27 /* Funkcija prebrojava koliko puta se pojavljuje svaki element
28    niza a. */
29 void brojanje(int a[], int b[], int n)
30 {
31     int i;
32
33     /* Niz b se inicijalizuje nulama jer se za svaki element postavi
34        da se pojavljuje 0 puta u nizu a. */
35     for (i = 1; i <= n; i++)
36         b[i] = 0;
37
38     /* Prolazi se kroz niz a i za svaki element a[i] uvecava
39        se broj njegovog pojavljivanja u nizu b. Na primer, ako je
40        a[3] = 7, onda treba uvecati broj pojavljivanja broja 7, a to
41        je b[7]++, sto se krace moze zapisati kao b[a[3]]++.
42        Pretpostavlja se da je niz a dobro zadat, odnosno da su sve
43        njegove vrednosti u intervalu od 1 do n. */
44     for (i = 0; i < n; i++)
45         b[a[i]]++;
46 }
47
48 /* Funkcija proverava da li je niz a permutacija. */
49 int permutacija(int a[], int n)
50 {
51     /* Niz b moze imati index MAKS (jer niz b se posmatra od 1 do
52        MAKS), pa zato njegova dimenzija mora biti za jedan veca. */
53     int b[MAKS + 1];
54     int i;
55
56     /* Racuna se broj pojavljivanja svakog broja niza a. */
57     brojanje(a, b, n);
58
59     /* Ukoliko se svaki element niza a javlja tacno jednom u nizu a,
60        onda niz a jeste permutacija. Ovo svojstvo se proverava
61        koriscenjem dobijenog niza b. */
62     for (i = 1; i <= n; i++)
63     {
64         if (b[i] != 1)
65             return 0;
66     }
67
68     return 1;
69 }
70
71 int main()
72 {
```

### 3 Predstavljanje podataka

```
73  /* Deklaracija potrebnih promenljivih. */
    int a[MAKS], n;
75
    /* Ucitava se dimenzija niza i vrsi se provera
77     ispravnosti ulaza. */
    printf("Unesite dimenziju niza: ");
79     scanf("%d", &n);
    if (n <= 0 || n > MAKS)
81     {
        printf("Greska: neispravan unos.\n");
83         exit(EXIT_FAILURE);
    }
85
    /* Ucitavaju se elementi niza a. */
87     ucitaj(a, n);
89
    /* Ispis rezultta. */
    if(permutacija(a, n))
91         printf("Uneti niz je permutacija.\n");
    else
93         printf("Uneti niz nije permutacija.\n");
95     exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.24

```
1  #include <stdio.h>
    #include <stdlib.h>
3
    #define BROJ_CIFARA 10
5
    /* Funkcija inicijalizuje niz postavljajuci vrednosti svih
7     elemenata na nulu. */
    void inicijalizuj(int niz[], int n)
9    {
        int i;
11       for (i = 0; i < n; i++)
            niz[i] = 0;
13    }

15    /* Funkcija izdvaja cifru po cifru broja i uvecava odgovarajuci
17     element niza koji odgovara brojacu za tu cifru. Na primer,
    za broj=1123, po zavrsetku ove funkcije niz[1] ce imati vrednost
19     2 jer se cifra 1 pojavljuje 2 puta, niz[2] i niz[3] ce imati
    vrednost 1, a svi ostali elementi niza ce imati vrednost 0. */
    void analiza_cifara(int broj, int niz[])
21    {
        int c;
23
        /* Inicijalizacija svih brojaca na nule. */
```

```
25   inicijalizuj(niz, BROJ_CIFARA);

27   /* Uvecavanje odgovarajucih brojaca. */
   do {
29       c = broj % 10;
       niz[c]++;
31       broj /= 10;
   }
33   while (broj);
}

35
int main()
37 {
   /* Niz cifrex predstavlja brojace za cifre broja x. Niz cifrey
   predstavlja brojace za cifre broja y. */
39   int cifrex[BROJ_CIFARA], cifrey[BROJ_CIFARA];
41   int x, y, i, indikator;

43   /* Ucitavaju se brojevi x i y. */
   printf("Unesite dva broja: ");
45   scanf("%d%d", &x, &y);

47   /* Za slucaj da su unete vrednosti negativne, posmatra se njihova
   apsolutna vrednost. Ovo je opravdano iz razloga sto se brojevi
49   x i -x zapisuju istim ciframa. */
   x = abs(x);
51   y = abs(y);

53   /* Popunjavaju se nizovi sa brojacima cifara. */
   analiza_cifara(x, cifrex);
55   analiza_cifara(y, cifrey);

57   /* Promenljiva indikator sluzi za pracenje da li su oba broja
   sastavljena od istih cifara. */
59   indikator = 1;

61   for (i = 0; i < BROJ_CIFARA; i++) {
       /* Ako se broj pojavljivanja cifre i u zapisu broja x razlikuje
63       od broja pojavljivanja cifre i u zapisu broja y, brojevi se
       ne zapisuju istim ciframa. Zato se vrednost indikatora moze
65       postaviti na 0 i prekinuti dalje uporedjivanje broja
       pojavljivanja. */
67       if (cifrey[i] != cifrex[i]) {
           indikator = 0;
69       break;
       }
71   }

73   /* Ako je vrednost promenljive indikator ostala 1, to znaci da u
   petlji nije pronadjena cifra koja se ne pojavljuje isti broj
75   puta u zapisima brojeva x i y. Zato se moze zakljuciti da se
   brojevi zapisuju istim ciframa. */
```

### 3 Predstavljanje podataka

---

```
77     if (indikator)
78         printf("Brojevi se zapisuju istim ciframa.\n");
79     else
80         printf("Brojevi se ne zapisuju istim ciframa.\n");
81
82
83     exit(EXIT_SUCCESS);
84 }
```

#### Rešenje 3.1.25

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS 100
5
6  /* Funkcija učitava elemente niza dimenzije n. */
7  void učitaj(int a[], int n)
8  {
9      int i;
10     printf("Unesite elemente niza: ");
11     for (i = 0; i < n; i++)
12         scanf("%d", &a[i]);
13 }
14
15 /* Funkcija ispisuje elemente niza dimenzije n. */
16 void ispisi(int a[], int n)
17 {
18     int i;
19     for (i = 0; i < n; i++)
20         printf("%d ", a[i]);
21     printf("\n");
22 }
23
24 /* Funkcija obrće elemente niza. */
25 void obrni(int a[], int n)
26 {
27     int t, i, j;
28
29     /* Za niz a[0], a[1], ..., a[n-2], a[n-1] obrnuti niz je a[n-1],
30        a[n-2], ..., a[1], a[0]. Zato je potrebno razmeniti vrednosti
31        elemenata a[0] i a[n-1], a[1] i a[n-2], itd. i zaustaviti se
32        kada je vrednost indeksa prvog elementa veća od vrednosti
33        drugog elementa. */
34     for (i = 0, j = n - 1; i < j; i++, j--)
35     {
36         t = a[i];
37         a[i] = a[j];
38         a[j] = t;
39     }
40 }
```



```
42 /* Funkcija rotira niz ciklicno za jedno mesto u levo. */
void rotiraj1(int a[], int n)
44 {
    int i, prvi;

46     /* Izdvaja se prvi element niza. */
48     prvi = a[0];

50     /* Pomeraju se preostali elementi niza za jedno mesto u levo. */
    for (i = 0; i < n - 1; i++)
52         a[i] = a[i + 1];

54     /* Poslednjem elementu se dodeljuje sacuvana vrednost prvog
        elementa. */
56     a[n - 1] = prvi;
    }

58 /* Funkcija rotira niz ciklicno za k mesta u levo. */
void rotirajk(int a[], int n, int k)
60 {
    int i;

62     /* Odredjuje se vrednost broja k koja je u opsegu od 0 do n-1
        kako bi se izbegla suvisna prvieranja. */
64     k = k % n;

66     /* Niz se rotira za jednu poziciju ulevo k puta. */
    for (i = 0; i < k; i++)
68         rotiraj1(a, n);
70 }

72 int main()
74 {
    /* Deklaracija potrebnih promenljivih. */
76     int a[MAKS];
    int n, k;

78     /* Ucitava se dimenzija niza i vrsi se provera
        ispravnosti ulaza. */
80     printf("Unesite dimenziju niza: ");
82     scanf("%d", &n);
    if (n <= 0 || n > MAKS)
84     {
        printf("Greska: neispravan unos.\n");
86         exit(EXIT_FAILURE);
    }

88     /* Ucitavaju se elementi niza. */
90     ucitaj(a, n);

92     /* Obrtanje niza. */
```

### 3 Predstavljanje podataka

```
printf("Elementi niza nakon obrtanja:\n");
94 obrni(a, n);
   ispisi(a, n);

96
/* Rotiranje za jedno mesto u levo. */
98 printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
   rotiraj1(a, n);
100 ispisi(a, n);

/* Rotiranje za k mesta u levo. */
102 printf("Unesite jedan pozitivan ceo broj:");
104 scanf("%d", &k);
   if (k <= 0)
106 {
       printf("Greska: neispravan unos.\n");
108       exit(EXIT_FAILURE);
   }
110 rotirajk(a, n, k);
   printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n", k);
112 ispisi(a, n);

114 exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.26

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

4 #define MAKS 100

6 /* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int niz[], int n)
8 {
   int i;
10   for (i = 0; i < n; i++)
       scanf("%d", &niz[i]);
12 }

14 /* Funkcija ispisuje elemente niza. */
void ispisi(int niz[], int n)
16 {
   int i;
18   for (i = 0; i < n; i++)
       printf("%d ", niz[i]);
20   printf("\n");
   }

22
/* Funkcija formira niz c ukrstanjem nizova a i b. */
24 void ukrsti(int a[], int b[], int n, int c[])
{
}
```

```

26  int i, j;
    /* Formira se treci niz. Koriste se dva indeksa:
28     - indeks i pomocu kojeg se pristupa elementima nizova a i b i
       koji treba uvecati za 1 nakon svake iteracije
30     - indeks j pomocu kojeg se pristupa elementima rezultujuceg
       niza c; s obzirom da se u svakoj iteraciji u niz c smestaju
32     dva elementa, jedan iz niza a i jedan iz niza b, indeks j se
       uvecava za 2 nakon svake iteracije. */
34  for (i = 0, j = 0; i < n; i++, j += 2)
    {
36      c[j] = a[i];
      c[j + 1] = b[i];
38  }
}

40
42  int main()
43  {
    /* Deklaracija potrebnih promenljivih. */
44  int a[MAKS], b[MAKS], c[2 * MAKS];
    int n;

46
    /* Ucitava se dimenzija nizova i vrsi se provera
48     ispravnosti ulaza. */
    printf("Unesite dimenziju nizova: ");
50    scanf("%d", &n);
    if (n <= 0 || n > MAKS)
52    {
        printf("Greska: neispravan unos.\n");
54        exit(EXIT_FAILURE);
    }

56
    /* Ucitavaju se elementi nizova. */
58    printf("Unesite elemente niza a: ");
    ucitaj(a, n);
60    printf("Unesite elemente niza b: ");
    ucitaj(b, n);

62
    /* Formira se niz c. */
64    ukrsti(a, b, n, c);

66
    /* Ispisuju se elementi rezultujuceg niza. */
    printf("Rezultujuci niz:\n");
68    ispisi(c, 2*n);

70    exit(EXIT_SUCCESS);
}

```

### Rešenje 3.1.27

```

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

```

```
3  #define MAKS 100
5
6  /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int niz[], int n)
8  {
9      int i;
10     for (i = 0; i < n; i++)
11         scanf("%d", &niz[i]);
12 }
13
14 /* Funkcija ispisuje elemente niza dimenzije n. */
15 void ispisi(int niz[], int n)
16 {
17     int i;
18     for (i = 0; i < n; i++)
19         printf("%d ", niz[i]);
20     printf("\n");
21 }
22
23 /* Funkcija formira niz c nadovezivanjem nizova a i b. */
24 void spoji(int a[], int b[], int n, int c[])
25 {
26     int i;
27
28     /* Niz c ima 2*n elemenata: prvih n elemenata su elementi niza b,
29     a narednih n elemenata elementi niza a. Elementi niza b se
30     nalaze na pozicijama 0,1,2,...n-1, a elementi niza a na
31     pozicijama n,n+1,...2*n-1. Jednim prolaskom kroz petlju na
32     poziciju i u nizu c se postavlja element b[i] niza b, a na
33     poziciju n+i element a[i] niza a. */
34     for (i = 0; i < n; i++) {
35         c[i] = b[i];
36         c[n + i] = a[i];
37     }
38 }
39
40 int main()
41 {
42     /* Deklaracija potrebnih promenljivih. */
43     int a[MAKS], b[MAKS], c[2 * MAKS];
44     int n;
45
46     /* Ucitava se dimenzija nizova i vrsi se provera
47     ispravnosti ulaza. */
48     printf("Unesite dimenziju nizova: ");
49     scanf("%d", &n);
50     if (n <= 0 || n > MAKS)
51     {
52         printf("Greska: neispravan unos.\n");
53         exit(EXIT_FAILURE);
54     }
55 }
```

```

55  /* Ucitavaju se elementi nizova. */
57  printf("Unesite elemente niza a: ");
   ucitaj(a, n);
59  printf("Unesite elemente niza b: ");
   ucitaj(b, n);

61  /* Formira se niz c. */
63  spoji(a, b, n, c);

65  /* Ispisuju se elementi niza c. */
   ispisi(c, 2*n);
67
   exit(EXIT_SUCCESS);
69 }

```

### Rešenje 3.1.28

```

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

3
   #define MAKS 100

5
   /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int niz[], int n)
   {
9      int i;
      printf("Unesite elemente sortiranog niza:\n");
11     for (i = 0; i < n; i++)
         scanf("%d", &niz[i]);
13 }

15 /* Funkcija za ispis niza. */
   void ispisi(int niz[], int n)
17 {
      int i;
19     for (i = 0; i < n; i++)
         printf("%d ", niz[i]);
21     printf("\n");
   }

23
   int main()
25 {
      /* Deklaracija potrebnih promenljivih. */
27     int a[MAKS], b[MAKS], c[2 * MAKS];
      int n;
29     /* Brojac u petlji za elemente niza a. */
      int i = 0;
31     /* Brojac u petlji za elemente niza b. */
      int j = 0;
33     /* Brojac u petlji za elemente niza c. */

```

```
int k = 0;

35
/* Ucitava se dimenzija nizova i vrsi se provera
37   ispravnosti ulaza. */
printf("Unesite dimenziju nizova: ");
39 scanf("%d", &n);
if (n <= 0 || n > MAKS)
41 {
    printf("Greska: neispravan unos.\n");
43     exit(EXIT_FAILURE);
}

45
/* Ucitavaju se elementi nizova. */
47 ucitaj(a, n);
ucitaj(b, n);

49
/* Vrsi se spajanje nizova. */
51 while (i < n && j < n)
{
    53     /* Porede se elementi nizova a i b i u niz c upisuje se samo
        onaj koji je manji. Ako je upisan element iz niza a, onda se
        55     vrsi i uvecavanje brojaca i (prelazak na sledeci element
        niza a), a ako je upisan element iz niza b, onda se vrsi
        57     uvecavanje brojaca j (prelazak na sledeci element niza b). */
        if (a[i] < b[j])
        59         {
            c[k] = a[i];
            i++;
        }
        63     else
        {
            65         c[k] = b[j];
            j++;
        }
        67

        /* U nizu c na poziciju k je upisan ili a[i] ili b[j]. Brojac k
        se uvecava. */
        69         k++;
    }

    73
    /* Ukoliko je ostalo elemenata u nizu a, upisuju se u niz c. */
    75    while (i < n)
    {
        77        c[k] = a[i];
        k++;
        79        i++;
    }

    81
    /* Ukoliko je ostalo elemenata u nizu b, upisuju se u niz c. */
    83    while (j < n)
    {
        85        c[k] = b[j];
```

```

      k++;
87      j++;
    }

89      /* Ispis elemenata niza c cija dimenzija je zbir dimenzija nizova
91      a i b. */
    ispisi(c, 2 * n);
93
    exit(EXIT_SUCCESS);
95 }

```

### Rešenje 3.1.29

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 100
5
   /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int a[], int n)
   {
9      int i;
      printf("Unesite elemente niza: ");
11     for (i = 0; i < n; i++)
         scanf("%d", &a[i]);
13 }

15 /* Funkcija ispisuje elemente niza dimenzije n. */
   void ispisi(int niz[], int n)
17 {
      int i;
19     for (i = 0; i < n; i++)
         printf("%d ", niz[i]);
21     printf("\n");
   }

23
   /* Funkcija razmenjuje elemente niza tako da se na pocetku niza
25     nalaze svi parni elementi niza, nakon kojih slede svi neparni
       elementi niza. */
27 void promeni_redosled(int niz[], int n)
   {
29     int i = 0, j = n - 1;
       int pom;
31
       /* Krece se sa pocetka niza (po brojacu i) i sa kraja niza (po
33     brojacu j) i svaki put kada se naidje na elemente koji po
       parnosti ne odgovaraju delu niza u kome treba da budu, ti
35     elementi se zamene. */
       while (i < j && i < n && j >= 0)
37     {
         if (niz[i] % 2 != 0 && niz[j] % 2 == 0)

```

### 3 Predstavljanje podataka

```
39     {
40         pom = niz[i];
41         niz[i] = niz[j];
42         niz[j] = pom;
43     }

45     /* Ukoliko je element na poziciji i paran, prelazi se na
46        sledeci element niza, brojac i se uvecava. */
47     if (niz[i] % 2 == 0)
48         i++;

49     /* Ukoliko je element na poziciji j neparan, prelazi se na
50        sledeci element niza, brojac j se smanjuje. */
51     if (niz[j] % 2 != 0)
52         j--;
53 }
54
55 }

57 int main()
58 {
59     /* Deklaracija potrebnih promenljivih. */
60     int niz[MAKS];
61     int n;

62     /* Ucitava se dimenzija niza i vrsi se provera
63        ispravnosti ulaza. */
64     printf("Unesite dimenziju niza: ");
65     scanf("%d", &n);
66     if (n <= 0 || n > MAKS)
67     {
68         printf("Greska: neispravan unos.\n");
69         exit(EXIT_FAILURE);
70     }

71

72     /* Ucitavaju se elementi niza. */
73     ucitaj(niz, n);

74

75     /* Menja se niz na trazeni nacin. */
76     promeni_redosled(niz, n);

77

78     /* Ispis rezultata. */
79     printf("Rezultujuci niz:\n");
80     ispisi(niz, n);

81

82     exit(EXIT_SUCCESS);
83 }
```

#### Rešenje 3.1.30

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



```
4  #include <math.h>
6
8  #define MAKS 100
10
12 /* Funkcija ucitava elemente niza dimenzije n. */
14 void ucitaj(int a[], int n)
16 {
18     int i;
19     printf("Unesite elemente niza: ");
20     for (i = 0; i < n; i++)
21         scanf("%d", &a[i]);
22 }
24
26 /* Funkcija ispisuje elemente niza dimenzije n. */
28 void ispisi(int a[], int n)
30 {
32     int i;
33     for (i = 0; i < n; i++)
34         printf("%d ", a[i]);
35     printf("\n");
36 }
38
40 /* Funkcija vraca 1 ako je broj prost, a 0 u suprotnom. */
42 int prost(int x)
44 {
46     int i;
48
49     /* Brojevi 2 i 3 su prosti. */
50     if (x == 2 || x == 3)
51         return 1;
52
53     /* Parni brojevi nisu prosti. */
54     if (x % 2 == 0)
55         return 0;
56
57     /* Ako se naidje na broj koji deli broj x, onda broj x nije
58        prost. Provera se vrši za sve neparne brojeve izmedju 3 i
59        sqrt(x), jer kada bi x imao parnog delioca, onda bi i broj 2
60        delio x, a taj uslov je vec proveren. */
61     int koren_x = sqrt(x);
62     for (i = 3; i <= koren_x; i += 2)
63         if (x % i == 0)
64             return 0;
65
66     /* Ako nijedan od prethodnih uslova nije bio ispunjen, to znaci
67        da nijedan broj ne deli x, pa je on prost. */
68     return 1;
69 }
70
72 /* Funkcija od niza a formira niz b koji sadrzi sve elemente
73    niza a koji nisu prosti brojevi. Povratna vrednost funkcije
74    je broj elemenata niza b. */
```

```
int obrisi_proste(int a[], int n, int b[])
{
    int i, j;

    /* Kada se u nizu a naidje na prost element, on se upisuje u niz
       b i uvecava se brojac za niz b. */
    for (i = 0, j = 0; i < n; i++)
    {
        if (prost(a[i]) == 0)
        {
            b[j] = a[i];
            j++;
        }
    }

    return j;
}

int main()
{
    /* Deklaracije potrebnih promenljivih. */
    int a[MAKS], b[MAKS];
    int n_a, n_b;

    /* Ucitava se dimenzija niza i vrsi se provera
       ispravnosti ulaza. */
    printf("Unesite dimenziju niza: ");
    scanf("%d", &n_a);
    if (n_a <= 0 || n_a > MAKS)
    {
        printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
    }

    /* Ucitavaju se elementi niza. */
    ucitaj(a, n_a);

    /* Formira se niz b brisanjem prostih iz niza a. */
    n_b = obrisi_proste(a, n_a, b);

    /* Ispisuju se elementi niza b. */
    ispisi(b, n_b);

    exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.31

```
1 #include <stdio.h>
  #include <stdlib.h>
3
```

```
5  #define MAKS 100
6
7  /* Funkcija ucitava elemente niza dimenzije n. */
8  void ucitaj(int a[], int n)
9  {
10     int i;
11     printf("Unesite elemente niza: ");
12     for (i = 0; i < n; i++)
13         scanf("%d", &a[i]);
14 }
15
16 /* Funkcija ispisuje elemente niza dimenzije n. */
17 void ispisi(int niz[], int n)
18 {
19     int i;
20     for (i = 0; i < n; i++)
21         printf("%d ", niz[i]);
22     printf("\n");
23 }
24
25 /* Funkcija brise sve neparne elemente niza. */
26 int obrisi_neparne(int a[], int n)
27 {
28     int i, j;
29     /* Promenljiva j predstavlja brojac prve slobodne pozicije na koju
30     se moze upisati element niza koji treba da ostane u nizu. Kada
31     se naidje na element koji je paran, on se kopira na mesto a[j]
32     i poveca se vrednost brojaca j. Ukoliko se naidje na element
33     koji je neparan, njega treba preskociti. */
34     for (i = 0, j = 0; i < n; i++)
35     {
36         /* Ako je tekuci element niza a paran. */
37         if (a[i] % 2 == 0)
38         {
39             /* Premesta se na poziciju j. */
40             a[j] = a[i];
41
42             /* Vrednost brojaca j se priprema za narednu iteraciju. */
43             j++;
44         }
45         /* Ako je tekuci element niza a neparan, sa njim nista ne treba
46         raditi. */
47     }
48
49     /* Rezultujuci niz ima j elemenata. */
50     return j;
51 }
52
53 int main()
54 {
55     /* Deklaracija potrebnih promenljivih. */
56     int a[MAKS];
```

### 3 Predstavljanje podataka

---

```
int n;

57
/* Ucitava se dimenzija niza i vrsi se provera
59   ispravnosti ulaza. */
printf("Unesite dimenziju niza: ");
61 scanf("%d", &n);
if (n <= 0 || n > MAKS)
63 {
    printf("Greska: neispravan unos.\n");
65     exit(EXIT_FAILURE);
}

67
/* Ucitavaju se elementi niza. */
69 ucitaj(a, n);

71
/* Brisu se neparni elementi. */
n = obrisi_neparne(a, n);

73
/* Ispisuju se elementi modifikovanog niza a. */
75 ispisi(a, n);

77 exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.32

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

4 #define MAKS 100

6 /* Funkcija ucitava elemente niza dimenzije n. */
void ucitaj(int a[], int n)
8 {
    int i;
    printf("Unesite elemente niza: ");
    for (i = 0; i < n; i++)
12         scanf("%d", &a[i]);
}

14
/* Funkcija ispisuje elemente niza dimenzije n. */
16 void ispisi(int a[], int n)
{
    int i;
    for (i = 0; i < n; i++)
20         printf("%d ", a[i]);
    printf("\n");
22 }

24 /* Funkcija brise sve elemente niza koji nisu deljivi
    svojom poslednjom cifrom. Povratna vrednost funkcije je
```

```
26     broj elemenata rezultujućeg niza. */
int brisanje(int a[], int n)
28 {
    int i, j, poslednja_cifra;

30     /* Obilaze se svi elementi niza a. */
32     for (i = 0, j = 0; i < n; i++)
    {
34         /* Izdvaja se poslednja cifra tekućeg elementa. */
        poslednja_cifra = a[i] % 10;

36         /* Ako je poslednja cifra nula ili je element deljiv svojom
38            poslednjom cifrom, taj element se zadržava i smesta na
            poziciju j. */
40         if (poslednja_cifra == 0 || a[i] % poslednja_cifra == 0)
        {
42             a[j] = a[i];
            j++;
44         }
    }

46     return j;
48 }

50 int main()
{
52     /* Deklaracija potrebnih promenljivih. */
    int a[MAKS];
54     int n;

56     /* Učitava se dimenzija niza i vrsi se provera
        ispravnosti ulaza. */
58     printf("Unesite dimenziju niza: ");
    scanf("%d", &n);
60     if (n <= 0 || n > MAKS)
    {
62         printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
64     }

66     /* Učitavaju se elementi niza. */
    ucitaj(a, n);

68     /* Brisu se svi elementi koji nisu deljivi svojom poslednjom
70        cifrom. */
    n = brisanje(a, n);

72     /* Ispisuje se rezultujući niz. */
74     ispisi(a, n);

76     exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.1.33

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS 700
5
6  /* Funkcija ucitava elemente niza dimenzije n. */
7  void ucitaj(int a[], int n)
8  {
9      int i;
10     printf("Unesite elemente niza: ");
11     for (i = 0; i < n; i++)
12         scanf("%d", &a[i]);
13 }
14
15 /* Funkcija ispisuje elemente niza dimenzije n. */
16 void ispis(int a[], int n)
17 {
18     int i;
19     for (i = 0; i < n; i++)
20         printf("%d ", a[i]);
21     printf("\n");
22 }
23
24 /* Funkcija pomera za jedno mesto u levo elemente niza a pocevsi od
25    pozicije j. Element na poziciji j se brise i na njegovo mesto se
26    upisuje element na poziciji j+1, a u skladu sa tim svi ostali
27    elementi posle njega u nizu se pomeraju. */
28 void pomeri_za_jedno_mesto(int a[], int n, int j)
29 {
30     int i;
31     for (i = j; i < n; i++)
32         a[i] = a[i + 1];
33 }
34
35 /* Funkcija brise sve elemente niza koji nisu deljivi svojim
36    indeksom. Povratna vrednost funkcije je broj elemenata
37    rezultujuceg niza. */
38 int brisanje(int niz[], int n)
39 {
40     int i;
41
42     /* Potrebno je krenuti od poslednjeg elementa niza i petljom ici
43        ka pocetku niza (element na poziciji 0 se ne razmatra).
44        Proverava se da li je element potrebno obrisati i ako jeste
45        vrsi se pomeranje elemenata niza za jedno mesto u levo.
46        Prednost ovog resenja u odnosu na resenje kada se krene od
47        pocetka niza je u tome sto element koji se ispituje sigurno
48        nije promenio svoju poziciju usled pomeranja zbog brisanja.
49        Problem se moze resiti i koriscenjem pomocnog niza (uraditi za
50        vezbu). To resenje je efikasnije, ali trosi vise resursa. */
```

```

52     for (i = n - 1; i > 0; i--)
    {
54         if (niz[i] % i != 0)
        {
            pomeri_za_jedno_mesto(niz, n, i);
56             /* Nakon brisanja elementa, smanjuje se i dimenzija niza. */
            n--;
58         }
    }

60     return n;
62 }

64 int main()
    {
66         /* Deklaracija potrebnih promenljivih. */
        int n, niz[MAKS];

68         /* Ucitava se dimenzija niza i vrsi se provera
70            ispravnosti ulaza. */
        printf("Unesite dimenziju niza: ");
72         scanf("%d", &n);
        if (n <= 0 || n > MAKS)
74         {
            printf("Greska: neispravan unos.\n");
76             exit(EXIT_FAILURE);
        }

78         /* Ucitavaju se elementi niza. */
        ucitaj(niz, n);

80         /* Iz niza se brisu odgovarajuci elementi. */
        n = brisanje(niz, n);

82         /* Ispis novog niza. */
        printf("Novi niz:\n");
        ispisi(niz, n);

84         exit(EXIT_SUCCESS);
86     }
88 }
90

```

### Rešenje 3.1.34

```

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

3

   #define MAKS 500

5
   /* Funkcija vraca 1 ukoliko broj x postoji u nizu, 0 inace. */
7  int postoji(int niz[], int n, int x)
   {

```

```
9   int i;
10  for (i = 0; i < n; i++)
11      if (niz[i] == x)
12          return 1;
13
14  return 0;
15 }
16
17 /* Funkcija ucitava elemente niza dimenzije n. */
18 void ucitaj(int niz[], int n)
19 {
20     int i, element;
21     printf("Unesite elemente niza: ");
22     for (i = 0; i < n; i++)
23     {
24         scanf("%d", &element);
25         if(postoji(niz, i, element))
26         {
27             printf("Greska: skup ne moze imati duplikate.\n");
28             exit(EXIT_FAILURE);
29         }
30         niz[i] = element;
31     }
32 }
33
34 /* Funkcija ispisuje elemente niza dimenzije n. */
35 void ispisi(int niz[], int n)
36 {
37     int i;
38     for (i = 0; i < n; i++)
39         printf("%d ", niz[i]);
40     printf("\n");
41 }
42
43 int main()
44 {
45     /* Deklaracija potrebnih promenljivih. */
46     int a[MAKS], b[MAKS], unija[2 * MAKS], presek[MAKS],
47         razlika[MAKS];
48     int i, n_a, n_b, n_unija, n_presek, n_razlika;
49
50     /* Ucitava se dimenzija prvog niza i vrsi se provera
51        ispravnosti ulaza. */
52     printf("Unesite dimenziju niza: ");
53     scanf("%d", &n_a);
54     if (n_a <= 0 || n_a > MAKS)
55     {
56         printf("Greska: neispravan unos.\n");
57         exit(EXIT_FAILURE);
58     }
59
60     /* Ucitavaju se elementi niza. */
```



```
61  ucitaj(a, n_a);

63  /* Ucitava se dimenzija drugog niza i vrsi se provera
    ispravnosti ulaza. */
65  printf("Unesite dimenziju niza: ");
    scanf("%d", &n_b);
67  if (n_b <= 0 || n_b > MAKS)
    {
69      printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
71  }

73  /* Ucitavaju se elementi niza. */
    ucitaj(b, n_b);

75

77  /* Brojaci elemenata u nizovima unija, presek i razlika. */
    n_unija = 0;
    n_presek = 0;
79  n_razlika = 0;

81  for (i = 0; i < n_a; i++)
    {
83      /* Svi elementi niza a se dodaju u uniju. */
        unija[n_unija] = a[i];
85        n_unija++;

87      /* Ukoliko se element a[i] nalazi u nizu b i ne postoji u nizu
        presek, dodaje se presek i povecava se brojac elemenata u
89      nizu presek. */
        if (postoji(b, n_b, a[i]) == 1
91            && postoji(presek, n_presek, a[i]) == 0) {
            presek[n_presek] = a[i];
93            n_presek++;
        }

95

97      /* 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. */
99      if (postoji(b, n_b, a[i]) == 0
101          && postoji(razlika, n_razlika, a[i]) == 0) {
            razlika[n_razlika] = a[i];
            n_razlika++;
103        }
    }

105

107  /* Elemente niza b koji nisu uneti u uniju dodaju se u uniju. */
    for (i = 0; i < n_b; i++)
    {
109        if (postoji(unija, n_unija, b[i]) == 0)
        {
111            unija[n_unija] = b[i];
            n_unija++;
        }
    }
```

### 3 Predstavljanje podataka

---

```
113     }
114 }
115 /* Ispis rezultata. */
116 printf("Unija: ");
117 ispisi(unija, n_unija);
118
119 printf("Presek: ");
120 ispisi(presek, n_presek);
121
122 printf("Razlika: ");
123 ispisi(razlika, n_razlika);
124
125 exit(EXIT_SUCCESS);
126 }
```

#### Rešenje 3.1.35

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 2000
5
6 /* Funkcija ispisuje elemente niza dimenzije n. */
7 void ispisi(int niz[], int n)
8 {
9     int i;
10    for (i = 0; i < n; i++)
11        printf("%d ", niz[i]);
12    printf("\n");
13 }
14
15 /* Funkcija ubacuje element x na kraj niza. Vraca novu dimenziju
16    niza. */
17 int ubaci_na_kraj(int niz[], int n, int x)
18 {
19     if(n == MAKS)
20     {
21         printf("Greska: prekoracen je maksimalan broj elemenata niza.");
22         exit(EXIT_FAILURE);
23     }
24
25     niz[n] = x;
26     return n + 1;
27 }
28
29 /* Funkcija ubacuje element x na pocetak niza. Vraca novu dimenziju
30    niza. */
31 int ubaci_na_pocetak(int niz[], int n, int x)
32 {
33     if(n == MAKS)
34     {
```

```
35     printf("Greska: prekoracen je maksimalan broj elemenata niza.");
36     exit(EXIT_FAILURE);
37 }
38
39 int i;
40 /* Prvo se svi elementi niza pomere za jednu poziciju u desno da
41    bi se oslobodio prostor za prvi element niza. Poslednji
42    element niza se pomera sa pozicije (n-1) na poziciju (n).
43    Slicno se pomeraju i ostali elementi. */
44 for (i = n; i > 0; i--)
45     niz[i] = niz[i - 1];
46
47 /* Na prvu poziciju se upisuje novi element. Bitan je redosled
48    naredbi: ako bi prvo bio upisan novi element, a tek onda
49    izvršeno pomeranje, element na poziciji niz[0] bi bio obrisani
50    i ne bi mogao biti upisan na poziciju niz[1]. */
51 niz[0] = x;
52
53 return n + 1;
54 }
55
56 /* Funkcija ubacuje element x na neku poziciju u nizu. Vraca
57    novu dimenziju niza. */
58 int ubaci_na_poziciju(int niz[], int n, int x, int pozicija)
59 {
60     if(n == MAKS)
61     {
62         printf("Greska: prekoracen je maksimalan broj elemenata niza.");
63         exit(EXIT_FAILURE);
64     }
65
66     int i;
67     /* Prvo se svi elementi niza od pozicije do kraja pomere za jedno
68        mesto u desno da bi se oslobodio prostor za novi element niza.
69        */
70     for (i = n; i > pozicija; i--)
71         niz[i] = niz[i - 1];
72
73     /* Na poziciju se upisuje novi element. */
74     niz[pozicija] = x;
75
76     return n + 1;
77 }
78
79 /* Funkcija brise prvi element niza. Vraca novu dimenziju
80    niza. */
81 int brisi_prvog(int niz[], int n)
82 {
83     if(n == 0)
84     {
85         printf("Greska: nije moguće brisanje iz praznog niza.\n");
86         exit(EXIT_FAILURE);
87     }
88 }
```

### 3 Predstavljanje podataka

---

```
87  }

89  int i;
   /* Svi elementi niza pomeraju se za jedno mesto u levo. */
91  for (i = 0; i < n - 1; i++)
       niz[i] = niz[i + 1];
93
   return n - 1;
95 }

97 /* Funkcija brise poslednji element niza. Vraca novu
   dimenziju niza. */
99 int brisi_poslednjeg(int niz[], int n)
{
101     if(n == 0)
        {
103         printf("Greska: nije moguće brisanje iz praznog niza.\n");
        exit(EXIT_FAILURE);
105     }

107     /* Dovoljno je smanjiti dimenziju niza, elemente niza nije
        potrebno brisati. */
109     return n - 1;
}

111
113 /* Funkcija brise element x. Pretpostavlja se da element
   ima samo jedno pojavljivanje (za vezbu napisati funkciju koja
   brise sva pojavljivanja, ako ih ima vise). Vraca novu dimenziju
115 niza. */
int brisi_element(int niz[], int n, int x)
117 {
   int i, j;
119
   /* Prvo treba pronaci poziciju elementa u nizu. */
121   for (i = 0; i < n; i++)
       if (niz[i] == x)
123       break;

125   /* Provera da li element postoji u nizu. Ako je brojac stigao do
       kraja niza, onda element ne postoji u nizu. */
127   if (i == n) {
       printf("Klijent sa rednim brojem %d ne postoji u nizu.\n", x);
129       return n;
   }

131
   /* Ukoliko element postoji u nizu, svi elementi niza nakon njega
       se pomeraju za jedno mesto u levo. */
133   for (j = i; j < n - 1; j++)
       niz[j] = niz[j + 1];
135
137   return n - 1;
}
```

```
139 int main()
141 {
143     int n, niz[MAKS], i, klijent, pozicija;

145     /* Ucitava se dimenzija niza i vrsi se provera ispravnosti
        ulaza. */
147     printf("Unesite trenutni broj klijenata: ");
149     scanf("%d", &n);
151     if (n <= 0 || n > MAKS) {
153         printf("Greska: neispravan unos.\n");
155         exit(EXIT_FAILURE);
157     }

159     /* Ucitavaju se elementi niza. */
161     printf("Unesite klijenta kojeg treba ubaciti u niz: ");
163     scanf("%d", &klijent);
165     n = ubaci_na_kraj(niz, n, klijent);
167     printf("Niz nakon ubacivanja klijenta:\n");
169     ispis(niz, n);

171     /* Ubacivanje klijenta na pocetak. */
173     printf("Unesite prioritetnog klijenta kojeg treba
175         \"ubaciti u niz: ");
177     scanf("%d", &klijent);
179     n = ubaci_na_pocetak(niz, n, klijent);
181     printf("Niz nakon ubacivanja klijenta:\n");
183     ispis(niz, n);

185     /* Ubacivanje klijenta na zadatu poziciju. */
187     printf("Unesite prioritetnog klijenta kojeg treba ubaciti "
189         "u niz i njegovu poziciju:");
191     scanf("%d%d", &klijent, &pozicija);
193     if (pozicija < 0 || pozicija > n) {
195         printf("Greska: neispravan unos.\n");
197         exit(EXIT_FAILURE);
199     } else {
201         n = ubaci_na_poziciju(niz, n, klijent, pozicija);
203         printf("Niz nakon odlaska klijenta:\n");
205         ispis(niz, n);
207     }

209     /* Brisanje prvog klijenta. */
211     n = brisi_prvog(niz, n);
213     printf("Niz nakon odlaska klijenta:\n");
215     ispis(niz, n);
```

```
191  /* Brisanje poslednjeg klijenta. */
    n = brisi_poslednjeg(niz, n);
193  printf("Niz nakon odlaska klijenta:\n");
    ispis(niz, n);

195
    /* Brisanje klijenta sa datim rednim brojem. */
197  printf("Unesite redni broj klijenta koji je napustio red: ");
    scanf("%d", &klijent);
199  n = brisi_element(niz, n, klijent);
    printf("Niz nakon odlaska klijenta:\n");
201  ispis(niz, n);

203  exit(EXIT_SUCCESS);
}
```

## 3.3 Pokazivači

**Zadatak 3.3.1** Napisati funkciju `void uredi(int *pa, int *pb)` koja uređuje svoja dva celobrojna argumenta tako da se u prvom nalazi manja vrednost, a u drugom veća. Napisati program koji učitava dva cela broja i ispisuje uređene brojeve.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja:  2 5
|| Uredjene promenljive: 2, 5
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja:  11 -4
|| Uredjene promenljive: -4, 11
```

[Rešenje 3.3.1]

**Zadatak 3.3.2** Napisati funkciju `void rgb_u_cmy(int r, int g, int b, float* c, float* m, float* y)` koja datu boju u *rgb* formatu konvertuje u boju u *cmy* formatu po sledećim formulama:

$$c = 1 - r/255$$

$$m = 1 - g/255$$

$$y = 1 - b/255$$

Napisati program koji učitava boju u *rgb* formatu i ispisuje vrednosti unete boje u *cmy* formatu. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: Vrednosti boja u *rgb* formatu su u opsegu  $[0, 255]$ .

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite boju u rgb formatu: 56 111 24
|| cmy: (0.78, 0.56, 0.91)

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite boju u rgb formatu: 156 -90 5
|| Greska: neispravan unos.

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite boju u rgb formatu: 9 0 237
|| cmy: (0.96, 1.00, 0.07)

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite boju u rgb formatu: 300 11 27
|| Greska: neispravan unos.

```

[Rešenje 3.3.2]

**Zadatak 3.3.3** Napisati funkciju `int presek(float k1, float n1, float k2, float n2, float *px, float *py)` koja za dve prave date svojim koeficijentima pravca i slobodnim članovima određuje njihovu tačku preseka. Funkcija treba da vrati jedinicu ako se prave seku, a nulu ako nemaju tačku preseka (ako su paralelne). Napisati program koji učitava podatke o pravama i ukoliko prave imaju presek, ispisuje koordinate tačke preseka, a ako nemaju, ispisuje odgovarajuću poruku.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite k i n za prvu pravu: 4 5
|| Unesite k i n za drugu pravu: 11 -4
|| Prave se seku u tacki (1.29,10.14).

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite k i n za prvu pravu: 0.5 -4.7
|| Unesite k i n za drugu pravu: 0.5 9.1
|| Prave su paralelne.

```

[Rešenje 3.3.3]

**Zadatak 3.3.4** Napisati funkciju koja za dva cela broja izračunava njihov količnik i ostatak pri deljenju. Funkcija treba da vrati jedinicu ukoliko je uspešno izračunala vrednosti, a nulu ukoliko deljenje nije moguće. Napisati program koji učitava dva cela broja i ispisuje njihov količnik i ostatak pri deljenju. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 4 5
|| Kolicnik: 0
|| Ostatak: 4

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 4 0
|| Greska: neispravan unos.

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: -123 11
|| Kolicnik: -11
|| Ostatak: -2

```

[Rešenje 3.3.4]

### 3 Predstavljanje podataka

---

**Zadatak 3.3.5** Napisati funkciju koja za dužinu trajanja filma koje je dato u sekundama, određuje ukupno trajanje filma u satima, minutama i sekundama. Napisati program koji učitava trajanje filma u sekundama i ispisuje odgovarajuće vreme trajanja u formatu *broj\_sati:broj\_minuta:broj\_sekundi*. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Trajanje fima u sekundama: 5000  
1h:23m:20s
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Trajanje fima u sekundama: -300  
Greska: neispravan unos.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Trajanje fima u sekundama: 2500  
0h:41m:40s
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Trajanje fima u sekundama: 7824  
2h:10m:24s
```

[Rešenje 3.3.5]

**Zadatak 3.3.6** Napisati funkciju koja sa ulaza učitava karakter po karakter sve do kraja ulaza i prebrojava sva pojavljivanja karaktera tačka i sva pojavljivanja karaktera zarez. Napisati program koji za uneti tekst ispisuje koliko puta se pojavila tačka, a koliko puta se pojavio zarez.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
a.b.c.d  
a,b,,c,d,e  
Broj tacaka: 3  
Broj zareza: 5
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
.....789.....  
Broj tacaka: 10  
Broj zareza: 0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
sunce  
Broj tacaka: 0  
Broj zareza: 0
```

[Rešenje 3.3.6]

**Zadatak 3.3.7** Napisati funkciju `void par_nepar(int a[], int n, int parni[], int* np, int neparni[], int* nn)` koja razbija niz *a* na niz parnih i niz neparnih brojeva. Pokazivači *np* i *nn* redom treba da sadrže broj elemenata niza parnih tj. niza neparnih elemenata. Maksimalan broj elemenata niza je 50. Napisati program koji učitava dimenziju niza, a zatim i elemente niza i ispisuje odgovarajuće nizove parnih, odnosno neparnih elemenata unetog niza. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.



*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza:
1 8 9 -7 -16 24 77 4
Niz parnih brojeva: 8 -16 24 4
Niz neparnih brojeva: 1 9 -7 77

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
2 4 6 8 -11
Niz parnih brojeva: 2 4 6 8
Niz neparnih brojeva: -11

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 2
Unesite elemente niza:
-15 15
Niz parnih brojeva:
Niz neparnih brojeva: -15 15

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 0
Greska: neispravan unos.

```

[Rešenje 3.3.7]

**Zadatak 3.3.8** Napisati funkciju koja izračunava najmanji i najveći element niza realnih brojeva. Napisati program koji učitava niz realnih brojeva maksimalne dužine 50 i ispisuje vrednosti najmanjeg i najvećeg elementa niza, zaokružene na tri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
24.16 -32.11 999.25 14.25 11
Najmanji: -32.110
Najveci: 999.250

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza:
-5.126 -18.29 44 29.268
Najmanji: -18.290
Najveci: 44.000

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 1
Unesite elemente niza:
4.16
Najmanji: 4.160
Najveci: 4.160

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: -3
Greska: neispravan unos.

```

[Rešenje 3.3.8]

## 3.4 Rešenja

#### Rešenje 3.3.1

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Argumenti funkcije uredi_pogresno, promenljive a i b,
5 predstavljaju lokalne promenljive za ovu funkciju i prestaju
6 da postoje po zavrsetku funkcije. Zbog toga se efekti razmene
7 vrednosti promenljivih a i b u slucaju da je a>b ne vide u
8 glavnom programu.
9 void uredi_pogresno(int a, int b)
10 {
11     int pom;
12     if (a > b) {
13         pom = a;
14         a = b;
15         b = pom;
16     }
17 }
18 */
19
20 /* Argumenti funkcije uredi, promenljive pa i pb, takodje su
21 lokalne promenljive za ovu funkciju i prestaju da postoje kada
22 se funkcija zavrshi. Njima prosledjujemo adrese promenljivih a i
23 b koje zelimo da razmenimo u slucaju da je a>b.
24
25 Promenljivoj a pristupamo preko pokazivacke promenljive pa sa
26 *pa i slicno, promenljivoj b pristupamo sa *pb.
27
28 Vrednosti promenljivih *pa i *pb razmenjujemo kao i vrednosti
29 bilo koje dve celobrojne promenljive. */
30 void uredi(int *pa, int *pb)
31 {
32     int pom;
33     if (*pa > *pb)
34     {
35         pom = *pa;
36         *pa = *pb;
37         *pb = pom;
38     }
39 }
40
41 int main()
42 {
43     /* Deklaracija potrebnih promenljivih. */
44     int a, b;
45
46     /* Ucitavaju se vrednosti dva cela broja. */
47     printf("Unesite dva broja:");
48     scanf("%d%d", &a, &b);
49
50     /* Neispravan nacin:
```

```

51 uredi_pogresno(a, b);
    printf("Uredjene promenljive: %d, %d\n", a, b); */
53
    /* Funkcija uredi kao argumente prima dve pokazivacke
55    promenljive (int*,int*). Zbog toga joj je u pozivu funkcije
    neophodno proslediti adrese promenljivih koje zelimo da
57    uredimo rastuce, &a i &b. */
    uredi(&a, &b);
59    printf("Uredjene promenljive: %d, %d\n", a, b);
61
    exit(EXIT_SUCCESS);
}

```

### Rešenje 3.3.2

```

1  #include <stdio.h>
    #include <stdlib.h>
3
    #define MIN_RGB 0
5    #define MAKS_RGB 255
6
7    /* Funkcija koja vrsi konverziju boje iz rgb formata u cmy format.
    Kako se pomocu naredbe return ne moze vratiti vise od jedne
9    vrednosti, neophodno je da se promenljive cije se vrednosti
    racunaju prenesu preko pokazivaca. */
11 void rgb_u_cmy(int r, int g, int b, float* c, float* m, float* y)
    {
13     *c = 1 - r / 255.0;
        *m = 1 - g / 255.0;
15     *y = 1 - b / 255.0;
    }
17
    /* Funkcija koja proverava da li je vrednost boje u ispravnom
19    opsegu. */
    int ispravna_rgb_vrednost(int a)
21    {
        if (a < MIN_RGB || a > MAKS_RGB)
23         return 0;
        return 1;
25    }
26
27 int main()
    {
29     /* Deklaracija potrebnih promenljivih. */
        int r, g, b;
31     float c, m, y;
32
33     /* Ucitava se vrednost boje u rgb formatu. */
        printf("Unesite boju u rgb formatu: ");
35     scanf("%d%d%d", &r, &g, &b);
    }

```

```
37  /* Vrsi se provera ispravnosti ulaza. */
    if (!ispravna_rgb_vrednost(r) || !ispravna_rgb_vrednost(g) ||
39      !ispravna_rgb_vrednost(b))
    {
41      printf("Greska: neispravan unos.\n");
      exit(EXIT_FAILURE);
43  }

45  /* Konverzija boje i ispis rezultata. Funkciji se kao argumenti
      prosledjuju vrednosti brojeva r, g, i b, kao i adrese na koje
47      treba da se upisu izracunate c, m, y vrednosti. */
    rgb_u_cmy(r, g, b, &c, &m, &y);
49    printf("cmy: (%.2f,%.2f,%.2f)\n", c, m, y);

51    exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.3.3

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

4 /* Funkcija koja racuna presek pravih  $y = k_1 * x + n_1$  i
    $y = k_2 * x + n_2$ . Koordinate preseka (ako postoji) se upisuju
6   na adrese px i py. Kao povratna vrednost funkcije se vraca
   jedinica ukoliko presek postoji, a nula inace. */
8 int presek(float k1, float n1, float k2, float n2, float *px,
            float *py)
10 {
    /* Ako je koeficijent pravca jednak, prave su paralelne.
12     Povratna vrednost funkcije je 0, kao indikator da
       nema presecne tacke. */
14     if (k1 == k2)
        return 0;

16     /* Koordinate preseka se upisuju na adrese (px, py). */
18     *px = -(n1 - n2) / (k1 - k2);
        *py = k1 * (*px) + n1;

20     /* Funkcija vraca 1 kao indikator da je presek uspesno
       izracunat. */
22     return 1;
24 }

26 int main()
{
28     /* Deklaracije potrebnih promenljivih. */
        float k1, k2, n1, n2;
30     float x, y;

32     /* Ucitavaju se parametri za dve prave. */
```

```

34 printf("Unesite k i n za prvu pravu:");
scanf("%f%f", &k1, &n1);
36 printf("Unesite k i n za drugu pravu:");
scanf("%f%f", &k2, &n2);

38 /* Ispis rezultata. */
if (presek(k1, n1, k2, n2, &x, &y))
40     printf("Prave se seku u tacki (%.2f,%.2f).\n", x, y);
else
42     printf("Prave su paralelne.\n");

44 exit(EXIT_SUCCESS);
}

```

### Rešenje 3.3.4

```

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

4 /* Funkcija koja racuna kolicnik i ostatak pri deljenju a sa b.
   Ukoliko su ove vrednosti uspesno izracunate, funkcija vraca 1,
   a inace vraca nulu. */
6 int kolicnik_ostatak(int a, int b, int* pk, int* po)
8 {
    if(b == 0)
10         return 0;

12     *pk = a/b;
    *po = a%b;
14     return 1;
}

16 int main()
18 {
    /* Deklaracija potrebnih promenljivih. */
20     int a, b, kolicnik, ostatak;

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

26     /* Ispis rezultata. */
    if(kolicnik_ostatak(a, b, &kolicnik, &ostatak))
28     {
        printf("Kolicnik: %d\n", kolicnik);
30         printf("Ostatak: %d\n", ostatak);
    }
    else
32     {
34         printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
    }
}

```

### 3 Predstavljanje podataka

---

```
36     }
38     exit(EXIT_SUCCESS);
}
```

#### Rešenje 3.3.5

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  /* Funkcija koja dato trajanje izrazeno u ukupnom broju sekundi
5     konvertuje u trajanje koje je izrazeno u broju sati, minuta
6     i sekundi. */
7  void konverzija(int trajanje, int* psati, int* pminuti,
8                  int* psekunde)
9  {
10     *psati = trajanje / 3600;
11     trajanje -= *psati * 3600;
12
13     *pminuti = trajanje/60;
14     trajanje -= *pminuti * 60;
15
16     *psekunde = trajanje;
17 }
18
19 int main()
20 {
21     /* Deklaracija potrebnih promenljivih. */
22     int trajanje, sati, minuti, sekunde;
23
24     /* Ucitava se trajanje u sekundama. */
25     printf("Trajanje filma u sekundama: ");
26     scanf("%d", &trajanje);
27
28     /* Vrsi se provera ispravnosti ulaza. */
29     if(trajanje < 0)
30     {
31         printf("Greska: neispravan unos.\n");
32         exit(EXIT_FAILURE);
33     }
34
35     /* Racunanje rezultata. */
36     konverzija(trajanje, &sati, &minuti, &sekunde);
37
38     /* Ispis rezultata. */
39     printf("%dh:%dm:%ds\n", sati, minuti, sekunde);
40     exit(EXIT_FAILURE);
41 }
```

## Rešenje 3.3.6

```

1  #include <stdio.h>
   #include <stdlib.h>
3
4  /* Funkcija koja ucitava karakter po karakter sa ulaza i prebrojava
5     koliko puta se pojavio karakter '.' i koliko puta se pojavio
6     karakter ','. Ucitavanje se zaustavlja kada se dodje do kraja
7     ulaza (EOF-a). */
   void interpunkcija(int *br_tacaka, int *br_zareza)
9  {
10     /* Deklaracije i inicijalizacije pomocnih promenljivih. */
11     int tacke = 0, zarezi = 0;
12     char c;
13
14     /* Ucitavanje i prebrojavanje trazениh karaktera. */
15     while ((c = getchar()) != EOF)
16     {
17         if (c == '.')
18             tacke++;
19
20         if (c == ',')
21             zarezi++;
22     }
23
24     /* Smestanje rezultata na prosledjene adrese. */
25     *br_tacaka = tacke;
26     *br_zareza = zarezi;
27 }
28
29 int main()
30 {
31     /* Deklaracije potrebnih promenljivih. */
32     int br_tacaka, br_zareza;
33
34     /* Ucitavanje i obrada teksta. */
35     printf("Unesite tekst: \n");
36     interpunkcija(&br_tacaka, &br_zareza);
37
38     /* Ispis rezultata. */
39     printf("Broj tacaka: %d\n", br_tacaka);
40     printf("Broj zareza: %d\n", br_zareza);
41
42     exit(EXIT_SUCCESS);
43 }

```

## Rešenje 3.3.7

```

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

```

### 3 Predstavljanje podataka

---

```
3  #define MAKS 50
5
7  /* Funkcija koja od niza a formira dva niza: niz parnih elemenata
   niza a i niz neparnih elemenata niza a. Duzine rezultujucih
   nizova se upisuju na adrese np i nn. */
9  void par_nepar(int a[], int n, int parni[], int *np,
   int neparni[], int *nn)
11 {
12     int i, j, k;
13
14     /* Promenljiva i je brojac u originalnom nizu i on se uvecava u
15     svakoj iteraciji.
16     Promenljiva j je brojac za niz parnih brojeva i on treba da se
17     uveca svaki put kada se naidje na novi element ovog niza.
18     Promenljiva k je brojac za niz neparnih brojeva i on treba da
19     se uveca sveki put kada se naidje na novi element ovog niza. */
20     for (i = 0, j = 0, k = 0; i < n; i++)
21     {
22         if (a[i] % 2 == 0)
23         {
24             parni[j] = a[i];
25             j++;
26         }
27         else
28         {
29             neparni[k] = a[i];
30             k++;
31         }
32     }
33
34     /* Na kraju petlje, u promenljivoj j se nalazi podatak o broju
35     elementa niza parni[], a u promenljivoj k podatak o broju
36     elementa niza neparni[]. Ove vrednosti se upisuju na adrese
37     np i nn. */
38     *np = j;
39     *nn = k;
40 }
41
42 void ispisi(int niz[], int n)
43 {
44     int i;
45     for (i = 0; i < n; i++)
46         printf("%d ", niz[i]);
47     printf("\n");
48 }
49
50 int main()
51 {
52     /* Deklaracije potrebnih promenljivih. */
53     int n, n1, n2;
54     int a[MAKS], parni[MAKS], neparni[MAKS];
```



```

55     int i;

57     /* Ucitava se dimenzija niza. */
    printf("Unesite broj elemenata niza: ");
59     scanf("%d", &n);

61     /* Vrsi se provera ispravnosti ulaza. */
    if (n < 0 || n > MAKS)
63     {
        printf("Greska: neispravan unos.\n");
65         exit(EXIT_FAILURE);
    }

67     /* Ucitavaju se elementi niza. */
    printf("Unesite elemente niza: ");
69     for (i = 0; i < n; i++)
71         scanf("%d", &a[i]);

73     /* Nizovi parni[] i neparni[] se popunjavaju odgovarajucim
        vrednostima. */
75     par_nepar(a, n, parni, &n1, neparni, &n2);

77     /* Ispis niza parni[] koji ima n1 elemenata. */
    printf("Niz parnih brojeva: ");
79     ispisi(parni, n1);

81     /* Ispis niza neparni[] koji ima n2 elemenata. */
    printf("Niz neparnih brojeva: ");
83     ispisi(neparni, n2);

85     exit(EXIT_SUCCESS);
}

```

### Rešenje 3.3.8

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 50
5
   /* Funkcija koja racuna najmanji i najveći element niza a. */
7  void min_max(float a[], int n, float *najmanji, float *najveci)
   {
9      int i;

11     /* Vrednosti minimuma i maksimuma se inicijalizuju na vrednost
        prvog clana niza. */
13     *najmanji = a[0];
        *najveci = a[0];

15     /* U petlji se prolazi kroz ostale clanove niza i po potrebi se

```

```
17     vrsi azuriranje najmanje i najveće vrednosti. */
18     for (i = 1; i < n; i++)
19     {
20         if (a[i] > *najveci)
21             *najveci = a[i];
22
23         if (a[i] < *najmanji)
24             *najmanji = a[i];
25     }
26
27     /* Na kraju petlje, na adresama najmanji i najveći se nalaze
28        trazene vrednosti. */
29 }
30
31 int main()
32 {
33     /* Deklaracija potrebnih promenljivih. */
34     int i, n;
35     float a[MAKS], min, max;
36
37     /* Ucitava se dimenzija niza. */
38     printf("Unesite broj elemenata niza: ");
39     scanf("%d", &n);
40
41     /* Vrsi se provera ispravnosti ulaza. */
42     if (n < 0 || n > MAKS)
43     {
44         printf("Greska: neispravan unos.\n");
45         exit(EXIT_FAILURE);
46     }
47
48     /* Ucitavaju se elementi niza. */
49     printf("Unesite elemente niza:\n");
50     for (i = 0; i < n; i++)
51         scanf("%f", &a[i]);
52
53     /* Racunaju se vrednosti najmanjeg i najvećeg elementa. */
54     min_max(a, n, &min, &max);
55
56     /* Ispis rezultata. */
57     printf("Najmanji: %.3f\n", min);
58     printf("Najveci: %.3f\n", max);
59
60     exit(EXIT_SUCCESS);
61 }
```

## 3.5 Niske

**Zadatak 3.5.1** Napisati funkciju `void konvertuj(char s[])` koja menja nisku `s` tako što mala slova zamenjuje odgovarajućim velikim slovima, a velika

slova zamenjuje odgovarajućim malim slovima. Napisati program koji učitava nisku maksimalne dužine 10 karaktera i ispisuje konvertovanu nisku.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: BeoGrad
|| bEOgRAD
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: A+B+C
|| a+b+c
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 12345
|| 12345
```

[Rešenje 3.5.1]

**Zadatak 3.5.2** Napisati funkciju `void ubaci_zvezdice(char s[])` koja menja nisku `s` tako što u njoj svaki drugi karakter zameni zvezdicom. Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje izmenjenu nisku.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: *a*b*c*
|| Izmenjena niska: *****
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: zimA
|| Izmenjena niska: z*m*
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 123abc789
|| Izmenjena niska: 1*3*b*7*9
```

[Rešenje 3.5.2]

**Zadatak 3.5.3** Napisati program koji vrši poređenje niski. Napisati funkcije:

- `int jednake(char s1[], char s2[])` koja vraća jedinicu ako su  $s_1$  i  $s_2$  jednake niske, a nulu inače.
- `void u_velika_slova(char s[])` koja pretvara sva slova niske `s` u velika slova, a ostale karaktere ne menja.

Program učitava dve reči maksimalne dužine 20 karaktera i ispituje da li su unete reči jednake. Pri poređenju treba zanemariti razliku između malih i velikih slova.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite niske:
|| isPit2010
|| IsPit2010
|| Niske su jednake.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite niske:
|| Prog1
|| prog2
|| Niske nisu jednake.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite niske:
|| jun
|| JUNSKI
|| Niske nisu jednake.
```

[Rešenje 3.5.3]

### 3 Predstavljanje podataka

---

**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 završava samoglasnikom.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: vetar  
|| Niska se ne završava samoglasnikom.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: OLUJA  
|| Niska se završava samoglasnikom.
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: Programiranje1  
|| Niska se ne završava samoglasnikom.
```

[Rešenje 3.5.4]

**Zadatak 3.5.5** Napisati funkciju `int sadrzi_veliko(char s[])` koja proverava da li niska *s* sadrži veliko slovo. Napisati program koji za učitano nisku maksimalne dužine 20 karaktera proverava da li sadrži veliko slovo i ispisuje odgovarajuću poruku.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku:  
|| naocare  
|| Ne sadrzi veliko slovo.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku:  
|| DiopTrija0.75  
|| Sadrzi veliko slovo.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku:  
|| 21.06.2017.  
|| Ne sadrzi veliko slovo.
```

[Rešenje 3.5.5]

**Zadatak 3.5.6** Napisati program koji za učitano nisku *s* i karakter *c* ispituje da li se *c* pojavljuje u niski *s*. Ako se pojavljuje, program treba da ispiše indeks prvog pojavljivanja karaktera *c* u niski *s*, a u suprotnom -1. Pretpostaviti da niska može da ima najviše 20 karaktera.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
bazen
Unesite karakter:
z
2

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
lezaljka
Unesite karakter:
a
3

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
limunada
Unesite karakter:
b
-1

```

[Rešenje 3.5.6]

**Zadatak 3.5.7** Napisati funkciju `int podniska(char s[], char t[])` koja proverava da li je niska *t* podniska niske *s*. Napisati program koji učitava niske *s* i *t* maksimalne dužine 10 karaktera i ispisuje da li je niska *t* podniska niske *s*.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: abcde
Unesite nisku t: bcd
t je podniska niske s.

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: abcde
Unesite nisku t: bCd
t nije podniska niske s.

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: abcde
Unesite nisku t: def
t nije podniska niske s.

```

[Rešenje 3.5.7]

**Zadatak 3.5.8** Napisati funkciju `void skрати(char s[])` koja uklanja beline sa kraja date niske. Napisati program koji učitava liniju maksimalne dužine 100 karaktera i ispisuje učitano i izmenjenu nisku između zvezdica.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
rep belina
Ucitana niska:
*rep belina
Izmenjena niska:
*rep belina*

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
tri tabulatora na kraju
Ucitana niska:
*tri tabulatora na kraju
Izmenjena niska:
*tri tabulatora na kraju*

```

[Rešenje 3.5.8]

**Zadatak 3.5.9** Napisati funkciju `void ukloni_slova(char s[])` koja iz niske *s* uklanja sva mala i sva velika slova. Napisati program koji za učitano nisku maksimalne dužine 20 karaktera ispisuje odgovarajuću izmenjenu nisku.

### 3 Predstavljanje podataka

---

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
a1b2c3def
123
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
1+2=3
1+2=3
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
malaVELIKA
```

[Rešenje 3.5.9]

**Zadatak 3.5.10** Napisati funkciju `void ukloni(char *s)` koja iz niske uklanja sva slova iza kojih neposredno sledi slovo koje je u abecedi nakon njih, pri čemu se veličina slova zanemaruje. Napisati program koji učitava liniju teksta koja ima najviše 100 karaktera i ispisuje liniju koja se dobije nakon uklanjanja pomenutih karaktera.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Zdravo svima!
Zrvo vma!
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Danas je 10 stepeni.
Dns j 10 tpni.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Ima vetra, kise i hladnoce.
ma vtra, se i loe.
```

[Rešenje 3.5.10]

**Zadatak 3.5.11** Napisati program koji učitava nisku  $s$  maksimalne dužine 30 karaktera i formira nisku  $t$  trostrukim nadovezivanjem niske  $s$ .

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: dan
dandandan
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 3sesira
3sesira3sesira3sesira
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: a-b=5
a-b=5a-b=5a-b=5
```

[Rešenje 3.5.11]

**Zadatak 3.5.12** Napisati program koji za unetu reč maksimalne dužine 20 karaktera i pozitivan broj  $n$  manji od 10, formira rezultujuću reč tako što unetu reč kopira  $n$  puta, pri čemu se između svaka dva kopiranja umeće crtica. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: ana
Unesite broj n: 4
ana-ana-ana-ana

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 123
Unesite broj n: 1
123

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: x*y
Unesite broj n: 3
x*y-x*y-x*y

```

[Rešenje 3.5.12]

**Zadatak 3.5.13** Napisati funkciju `void kopiraj_n(char t[], char s[], int n)` koja kopira najviše  $n$  karaktera niske  $s$  u nisku  $t$ . Napisati program koji testira rad napisane funkcije. Pretpostaviti da je maksimalna dužina niske  $s$  20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: petar
Unesite broj n: 3
Rezultujuca niska: pet

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: gromobran
Unesite broj n: 4
Rezultujuca niska: grom

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: abc
Unesite broj n: 15
Rezultujuca niska: abc

```

[Rešenje 3.5.13]

**Zadatak 3.5.14** Napisati funkciju `void dupliranje(char t[], char s[])` koja na osnovu niske  $s$  formira nisku  $t$  tako što duplira svaki karakter niske  $s$ . Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje nisku koja se dobije nakon dupliranja karaktera.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: zima
zziiimmaa

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: A+B+C
AA++BB++CC

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: C
CC

```

[Rešenje 3.5.14]

**Zadatak 3.5.15** Napisati program koji učitava nisku cifara sa eventualnim vodećim znakom i pretvara je u ceo broj. NAPOMENA: *Pretpostaviti da je unos ispravan.*

### 3 Predstavljanje podataka

---

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| -1238
|| -1238
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| 73
|| 73
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| +1
|| 1
```

[Rešenje 3.5.15]

**Zadatak 3.5.16** Napisati program koji učitava ceo broj, pretvara ga u nisku i ispisuje dobijenu nisku.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj:
|| -6543
|| -6543
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj:
|| 84
|| 84
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj:
|| 5
|| 5
```

[Rešenje 3.5.16]

**Zadatak 3.5.17** Napisati funkciju `int heksadekadni_broj(char s[])` koja proverava da li je niskom `s` zadat korektan heksadekadni broj. Funkcija treba da vrati vrednost 1 ukoliko je uslov ispunjen, odnosno 0 ako nije. Napisati program koji za učitanu nisku maksimalne dužine 7 karaktera ispisuje da li je korektan heksadekadni broj. UPUTSTVO: *Heksadekadni broj je korektno zadat ako počinje prefiksom 0x ili 0X i ako sadrži samo cifre i mala ili velika slova A, B, C, D, E i F.*

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0x12EF
|| Korektan heksadekadni broj.
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0X22af
|| Korektan heksadekadni broj.
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0xErA9
|| Nekorektan heksadekadni broj.
```

[Rešenje 3.5.17]

**Zadatak 3.5.18** Napisati funkciju `int dekadna_vrednost(char s[])` koja izračunava dekadnu vrednost heksadekadnog broja zadatog niskom `s`. Napisati program koji za učitanu nisku maksimalne dužine 7 karaktera ispisuje odgovarajuću dekadnu vrednost. Pretpostaviti da je uneta niska korektan heksadekadni broj.



*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 0x2A34
10804
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 0Xff2
4082
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 0xE1A9
57769
```

[Rešenje 3.5.18]

**Zadatak 3.5.19** Napisati funkciju `int ucitaj_liniju(char s[], int n)` koja učitava liniju maksimalne dužine  $n$  u nisku  $s$  i vraća dužinu učitane linije. Napisati program koji učitava linije do EOF i ispisuje najdužu liniju i njenu dužinu. Ukoliko ima više linija maksimalne dužine, ispisati prvu. Pretpostaviti da svaka linija sadrži najviše 80 karaktera. NAPOMENA: *Linija može da sadrži blanko znakove, ali ne sadrži znak za novi red ili EOF.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite linije:
Dobar dan!
Kako ste, sta ima novo?
Ja sam dobro.
Kako ste, sta ima novo?
23
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite linije:
Prva linija
Druga linija
Trecia linija
Druga linija
12
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite linije:
Danas je lep dan.
Danas je lep dan.
17
```

[Rešenje 3.5.19]

\* **Zadatak 3.5.20** Napisati funkcije za rad sa rečenicama:

- `int procitaj_recenicu(char s[], int n)` koja učitava rečenicu i smešta je u nisku  $s$ . Funkcija vraća dužinu učitane rečenice. Učitavanje se završava nakon učitano g karaktera ., nakon  $n$  učitanih karaktera ili ako se dođe do kraja ulaza.
- `void prebroj(char s[], int *broj_malih, int *broj_velikih)` koja prebrojava mala i velika slova u niski  $s$ .

Napisati program koji učitava rečenice do kraja ulaza i ispisuje onu rečenicu kod koje je apsolutna razlika broja malih i velikih slova najveća. Pri učitavanju rečenica zanemariti sve beline koje se nalaze između dve rečenice. Pretpostaviti da jedna rečenica sadrži najviše 80 karaktera.

### 3 Predstavljanje podataka

---

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite tekst:
|| U ovom poglavlju se govori o niskama. Niske su nizovi karaktera ciji je
|| poslednji element terminalna nula.
|| U ovom zadatku je potrebno učitati recenice. Svaka recenica pocinje sa bilo
|| kojim karakterom koji nije belina. Na kraju recenice se nalazi tacka.
|| Niske su nizovi karaktera ciji je poslednji element terminalna nula.
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite tekst:
|| bbbAAA. AbAbAb. AbAbAbab.
|| AbAbAbab.
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite tekst:
|| Nije uneta nijedna recenica.
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite tekst:
|| AAAbbb. bbbAAA. AbAbAb.
|| AAAbbb
```

[Rešenje 3.5.20]

**Zadatak 3.5.21** Napisati funkciju `char* strchr_klon(char s[], char c)` koja vraća pokazivač na prvo pojavljivanje karaktera `c` u niski `s` ili `NULL` ukoliko se karakter `c` ne pojavljuje u niski `s`.<sup>1</sup> Napisati program koji za učitane niske maksimalne dužine 20 karaktera i karakter `c` ispisuje indeks prvog pojavljivanja karaktera `c` u okviru učitane niske ili `-1` ukoliko učitana niska ne sadrži uneti karakter.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: programiranje
|| Unesite karakter c: a
|| Pozicija: 5
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: 123456789
|| Unesite karakter c: y
|| Pozicija: -1
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: leto2017
|| Unesite karakter c: 0
|| Pozicija: 5
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: jedrilica
|| Unesite karakter c: I
|| Pozicija: -1
```

[Rešenje 3.5.21]

---

<sup>1</sup>Funkcija `strchr_klon` odgovara funkciji `strchr` čija se deklaracija nalazi u zaglavlju `string.h`. Slično važi i za ostale `klon` funkcije iz narednih zadataka.

**Zadatak 3.5.22** Napisati funkciju `int strspn_klon(char t[], char s[])` koja izračunava dužinu prefiksa niske  $t$  sastavljenog od karaktera niske  $s$ . Napisati program koji za učitane dve niske maksimalne dužine 20 karaktera ispisuje rezultat poziva napisane funkcije.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku t:  program
Unesite nisku s:  pero
3
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku t:  Barcelona
Unesite nisku s:  Brazil
3
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku t:  24.10.2017.
Unesite nisku s:  0123456789
2
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku t:  12345
Unesite nisku s:  9876543210
5
```

[Rešenje 3.5.22]

**Zadatak 3.5.23** Napisati funkciju `int strcspn_klon(char t[], char s[])` koja izračunava dužinu prefiksa niske  $t$  sastavljenog isključivo od karaktera koji se ne nalaze u niski  $s$ . Napisati program koji testira ovu funkciju za dve unete niske maksimalne dužine 100 karaktera.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku t:
programiranje
Unesite nisku s:
pero
0
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku t:
programiranje
Unesite nisku s:
analiza
5
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku t:
programiranje
Unesite nisku s:
1.10.
13
```

[Rešenje 3.5.23]

**Zadatak 3.5.24** Napisati funkciju `char* strstr_klon(char s[], char t[])` koja vraća pokazivač na prvo pojavljivanje niske  $t$  u niski  $s$  ili `NULL` ukoliko se niska  $t$  ne pojavljuje u niski  $s$ . Napisati program koji testira napisanu funkciju tako što učitava pet linija i ispisuje sve redne brojeve linija koje sadrže nisku *program*. Ukoliko ne postoji linija sa niskom *program*, ispisati odgovarajuću poruku. Pretpostaviti da je svaka linija maksimalne dužine 100 karaktera kao i da se linije numerišu od broja 1.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
tu program
c prog. jezik
c++ programskih jezik
Programski odbor
<b>program</b>
1 3 5
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
Programske paradigme
su predmet na
trecoj godini
programerskih
smerova.
4
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
U narednim
linijama
necemo navoditi
nisku koja se
trazi.
Nijedna linija ne sadrzi
nisku program.
```

[Rešenje 3.5.24]

**Zadatak 3.5.25** Napisati funkciju `int strcmp_klon(char s[], char t[])` koja vraća 0 ako su niske `s` i `t` jednake, neku pozitivnu vrednost ako je `s` leksikografski iza `t`, a neku negativnu vrednost inače. Napisati program koji učitava dve niske maksimalne dužine 20 karaktera i ako su različite, ispisuje učitane niske u rastućem leksikografskom poretku, a ako su jednake, ispisuje samo jednu nisku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku s:
Beograd
Unesite nisku t:
Amsterdam
Amsterdam
Beograd
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku s:
Beograd
Unesite nisku t:
Beograd
Beograd
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku s:
radnik
Unesite nisku t:
radnica
radnica
radnik
```

[Rešenje 3.5.25]

**Zadatak 3.5.26** Napisati funkciju `void obrni(char s[])` koja obrće nisku `s`. Napisati program koji obrće učitanu nisku maksimalne dužine 20 karaktera i ispisuje obrnutu nisku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
kisobran
narbosik
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Aleksandar
radnaskela
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
kajak
kajak
```

[Rešenje 3.5.26]

**Zadatak 3.5.27** Napisati funkciju `void rotiraj(char s[], int k)` koja rotira nisku *s* za *k* mesta ulevo. Napisati program koji učitava nisku maksimalne dužine 20 karaktera i nenegativan ceo broj *k* i ispisuje rotiranu nisku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| sveska
|| 2
|| eskasv
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| olovka
|| 6
|| olovka
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| rezac
|| 8
|| acrez
```

[Rešenje 3.5.27]

**Zadatak 3.5.28** Napisati program koji šifrira unetu nisku tako što svako slovo zamenjuje sledećim slovom abecede (slova 'z' i 'Z' zamenjuje redom sa 'a' i 'A'), a ostale karaktere ostavlja nepromenjene. Pretpostaviti da uneta niska nije duža od 20 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| bundeva
|| cvoefwb
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| zimzelen
|| ajnafmfo
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| Oktobar17
|| Plupcbs17
```

[Rešenje 3.5.28]

**Zadatak 3.5.29** Napisati funkciju `void sifruj(char rec[], char sifra[])` koja na osnovu date reči formira šifru tako što se svako slovo u reči zameni sa naredna tri slova u abecedi. Napisati program koji testira napisanu funkciju za reč maksimalne dužine 20 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| tam0
|| uvwbcdnopppqr
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| Zec
|| ABCfghdef
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| a+b=c
|| bcd+cde=def
```

[Rešenje 3.5.29]

**Zadatak 3.5.30** Napisati funkciju `void formiraj(char s1[], char s2[], char c1, char c2)` koja na osnovu niske  $s_1$  formira nisku  $s_2$  udvajanjem svih karaktera  $c_1$  u nisku  $s_1$  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
flooasster
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
bojica
Unesite prvi karakter:
b
Unesite drugi karakter:
a
bbojic
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
patentara
Unesite prvi karakter:
t
Unesite drugi karakter:
a
pttenttr
```

[Rešenje 3.5.30]

\* **Zadatak 3.5.31** Napisati program za rad sa brojevima zapisanim u različitim brojevnim sistemima.

- Napisati funkciju `unsigned int u_dekadni_sistem(char broj[], unsigned int osnova)` koja određuje dekadnu vrednost zapisa datog neoznačenog broja *broj* u datoj osnovi.
- Napisati funkciju `void iz_dekadnog_sistema(unsigned int broj, unsigned int osnova, char rezultat[])` koja datu dekadnu vrednost *broj* zapisuje u datoj osnovi *osnovi* i smešta rezultat u nisku *rezultat*. Pretpostaviti da je  $0 < b \leq 16$ .

Napisati program koji učitava broj  $n$  i osnove  $o_1$  i  $o_2$  i ispisuje dekadnu vrednost broja  $n$  u osnovi  $o_1$ , kao i vrednost koja se dobije kada se ta dekadna vrednost zapiše u osnovi  $o_2$ . Pretpostaviti da je ulaz ispravan i da će svi brojevi biti u opsegu tipa `unsigned`.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite n, o1 i o2: 10101011 2 16
Dekadna vrednost broja 10101011: 171
Vrednost broja 171 u osnovi 16: AB
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite n, o1 i o2: 1067 8 3
Dekadna vrednost broja 1067: 567
Vrednost broja 567 u osnovi 3: 210000
```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite n, o1 i o2: 1010111001010 2 3
Dekadna vrednost broja 1010111001010: 5578
Vrednost broja 5578 u osnovi 3: 21122121

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite n, o1 i o2: 111 3 5
Dekadna vrednost broja 111: 13
Vrednost broja 13 u osnovi 5: 23

```

[Rešenje 3.5.31]

## 3.6 Rešenja

## Rešenje 3.5.1

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <ctype.h>
4
5  /* Poslednji karakter svake niske je terminirajuca nula '\0',
6     specijalni karakter ciji je ASCII kod 0.
7
8     Ukoliko je pretpostavka da niska sadrzi najvise 10 karaktera,
9     neophodno je deklarirati niz od 11 karaktera, pri cemu se
10    dodatni izdvaja za terminirajuca nulu. */
11 #define MAKS_NISKA 11
12
13 /* Funkcija vrši konverziju svakog malog slova niske u odgovarajuće
14    veliko i obrnuto. Ostali karakteri ostaju nepromenjeni. */
15 void konvertuj(char s[])
16 {
17     int i;
18
19     /* Prolazi se kroz nisku, karakter po karakter, sve dok se ne
20        dodje do terminalne nule koja služi kao oznaka da se doslo
21        do kraja niske. */
22     for (i = 0; s[i] != '\0'; i++)
23     {
24         /* Svako malo slovo se pretvara u veliko i obrnuto. */
25         if (islower(s[i]))
26             s[i] = toupper(s[i]);
27         else if (isupper(s[i]))
28             s[i] = tolower(s[i]);
29     }
30
31     /* II način: Uslov u petlji može krace da se zapise sa s[i]
32        jer ASCII kod terminalne nule ima vrednost 0.
33     for (i = 0; s[i]; i++)
34     {

```

### 3 Predstavljanje podataka

---

```
35     if (islower(s[i]))
36         s[i] = toupper(s[i]);
37     else if (isupper(s[i]))
38         s[i] = tolower(s[i]);
39 } */
40 }
41
42 int main()
43 {
44     char s[MAKS_NISKA];
45     printf("Unesite nisku: ");
46
47     /* Za razliku od nizova koji se učitavaju i stampaju element po
48        element, niske se mogu učitati i odstampati pomoću jedne
49        scanf/printf naredbe korišćenjem specifikatora %s. */
50     scanf("%s", s);
51
52     /* Izmena niske. */
53     konvertuj(s);
54
55     /* Ispis rezultata. */
56     printf("%s\n", s);
57
58     exit(EXIT_SUCCESS);
59 }
```

#### Rešenje 3.5.2

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS_NISKA 21
5
6  /* Funkcija ubacuje zvezdice na svako drugo mesto niske s. */
7  void ubaci_zvezdice(char s[])
8  {
9      int i;
10
11     for(i=0; s[i] != '\0' && s[i+1] != '\0'; i+=2)
12         s[i+1] = '*';
13
14     /* II nacin:
15     for(i=0; s[i]; i++)
16         if(i%2 == 1)
17             s[i] = '*';
18     */
19 }
20
21 int main()
22 {
23     /* Deklaracija potrebnih promenljivih. */
```



```

    char s[MAKS_NISKA];

25
    /* Ucitavanje niske. */
27    printf("Unesite nisku: ");
    scanf("%s", s);

29
    /* Izmena niske. */
31    ubaci_zvezdice(s);

    /* Ispis rezultata. */
33    printf("Izmenjena niska: %s\n", s);
35
    exit(EXIT_SUCCESS);
37 }

```

### Rešenje 3.5.3

```

1  #include <stdio.h>
   #include <stdlib.h>
3  #include <ctype.h>

5  #define MAKS_NISKA 21

7  /* Funkcija pretvara sva slova niske s u velika slova. */
   void u_velika_slova(char s[])
9  {
    int i;

11     for(i=0; s[i]; i++)
13         s[i] = toupper(s[i]);
   }

15
   /* Funkcija vraća 1 ako su niske s1 i s2 jednake, a nulu inace. */
17 int jednake(char s1[], char s2[]){
    int i;

19
    /* Prolazi se kroz obe niske dok god ima neobradjenih karaktera
21     u bilo kojoj od njih. Ukoliko se naidje na karaktere koji
       su razliciti, kao povratna vrednost se vraća 0 jer u tom
23     slucaju niske nisu jednake. */
    for(i=0; s1[i] || s2[i]; i++)
25         if(s1[i] != s2[i])
           return 0;

27
    /* Ako se doslo do kraja petlje znaci da su se svi karakteri
29     poklopili, a da se pri tom doslo do kraja obe niske, tako da
       se kao povratna vrednost funkcije vraća 1 jer su niske
31     s1 i s2 jednake. */
    return 1;
33 }

```

### 3 Predstavljanje podataka

---

```
35 int main(){
    /* Deklaracija potrebnih promenljivih. */
37 char s1[MAKS_NISKA], s2[MAKS_NISKA];

    /* Ucitavaju se niske s1 i s2. */
39 printf("Unesite niske:\n");
41 scanf("%s%s", s1, s2);

    /* Kako bi se pri poredjenju zanemarila razlika izmedju malih i
       velikih slova, sva slova obe niske se pretvaraju u velika. */
43 u_velika_slova(s1);
45 u_velika_slova(s2);

    /* Ispis rezultata. */
47
49 if(jednake(s1, s2))
    printf("Niske su jednake.\n");
51 else
    printf("Niske nisu jednake.\n");
53
    exit(EXIT_FAILURE);
55 }
```

#### Rešenje 3.5.4

```
1  #include <stdio.h>
   #include <stdlib.h>
3  #include <ctype.h>
   #include <string.h>
5
   #define MAKS_NISKA 21
7
   /* Funkcija proverava da li je karakter c samoglasnik. */
9  int samoglasnik(char c)
   {
11     /* Karakter se pretvara u veliko slovo kako bi se izbegle posebne
       provere za mala i velika slova. */
13     c = toupper(c);

15     /* Samoglasnici su slova a, e, i, o i u */
    if (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U')
17         return 1;

19     return 0;
   }
21

   /* Funkcija proverava da li se niska s završava samoglasnikom. */
23  int samoglasnik_na_kraju(char s[])
   {
25     /* Funkcija strlen racuna duzinu date niske. Njena deklaracija se
       nalazi u zaglavlju string.h. */
27     int duzina = strlen(s);
```

```

29  /* Ako je niska prazna, ne završava se samoglasnikom. */
    if (duzina == 0)
31      return 0;

33  /* Proverava se da li je poslednji karakter niske samoglasnik. */
    return samoglasnik(s[duzina - 1]);
35 }

37 int main()
{
39  /* Deklaracija niske. */
    char s[MAKS_NISKA];

41  /* Učitava se niska. */
43  printf("Unesite nisku: ");
    scanf("%s", s);

45  /* Ispis rezultata. */
47  if (samoglasnik_na_kraju(s))
        printf("Niska se završava samoglasnikom.\n");
49  else
        printf("Niska se ne završava samoglasnikom.\n");
51  exit(EXIT_SUCCESS);
53 }

```

### Rešenje 3.5.5

```

#include <stdio.h>
2  #include <stdlib.h>
#include <ctype.h>

4  #define MAKS_NISKA 21

6  /* Funkcija proverava da li niska s sadrži bar jedno
   veliko slovo. */
8  int sadrzi_veliko(char s[])
10 {
    int i;

12    for(i=0; s[i]; i++)
14        if(isupper(s[i]))
            return 1;

16    return 0;

18  /* Cesta greska:
20    for(i=0; s[i]; i++)
        {
22        if(isupper(s[i]))

```

### 3 Predstavljanje podataka

---

```

        return 1;
24     else
        return 0;
26 }
    Cim se naidje na prvi karakter koji nije veliko
28 slovo, vraca se 0 kao oznaka da niska ne sadrzi
    veliko slovo, sto nije tacno. Nula moze da se vrati
30 tek kada se prodju svi karakteri niske s. */
}

32
33 int main()
34 {
    /* Deklaracija niske. */
36     char s[MAKS_NISKA];

    /* Ucitava se niska. */
38     printf("Unesite nisku:");
40     scanf("%s", s);

    /* Ispis rezultata. */
42     if(sadrzi_veliko(s))
44         printf("Sadrzi veliko slovo.\n");
    else
46         printf("Ne sadrzi veliko slovo.\n");

48     exit(EXIT_SUCCESS);
}

```

#### Rešenje 3.5.6

```

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

4  #define MAKS_NISKA 21

6  /* Funkcija vraca indeks prvog pojavljivanja karaktera c u okviru
    niske s. Ukoliko se ne pojavljuje, funkcija vraca -1. */
8  int pozicija(char s[], char c)
    {
10     int i;

12     for(i=0; s[i]; i++)
        if(s[i] == c)
14         return i;

16     return -1;
    }

18
19 int main()
20 {
    /* Deklaracija potrebnih promenljivih. */

```

```

22  char s[MAKS_NISKA];
    char c;

24

    /* Ucitavaju se niska i karakter. */
26  printf("Unesite nisku:");
    scanf("%s", s);
28  getchar();
    printf("Unesite karakter:");
30  c = getchar();

32  /* I nacin: */
    printf("%d\n", pozicija(s,c));
34

    /* II nacin:
36     Funkcija strchr(s,c) je funkcija koja vraca adresu prvog
        pojavljivanja karaktera c u niski s, ako se c pojavljuje u s,
38     a NULL inace.

40     Vrednost promenljive s je zapravo vrednost adrese prvog
        karaktera niske s.

42     Ako treba da se ispise indeks prvog pojavljivanja, to moze
44     da se uradi tako sto se od adrese koji je vratila funkcija
        strchr oduzme adresa prvog karaktera.

46     Na primer:
48     s = "koliba"      ==> s je adresa karaktera 'k'
        p = strchr(s, 'l') ==> p je adresa karaktera 'l'
50     |k|o|l|i|b|a|
        ^   ^
52     |   |
        s   p
54     Izraz p-s ima vrednost 2 (jer je rastojanje izmedju
        ove dve adrese 2).

56     Tip promenljive p je char* jer predstavlja adresu
58     jednog karaktera.

60  char *p = strchr(s, c);
    if (p != NULL)
62      printf("%d\n", p - s);
    else
64      printf("-1\n");
    */

66  exit(EXIT_SUCCESS);
68 }

```

### Rešenje 3.5.7

### 3 Predstavljanje podataka

---

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS_NISKA 11
5
6  /* Funkcija proverava da li je niska t podniska niske s. */
7  int podniska(char s[], char t[])
8  {
9      int i, j;
10
11     /* Spoljasnja petlja ide redom po niski s. */
12     for (i = 0; s[i] != '\0'; i++)
13     {
14         /* Unutrasnja petlja ide redom po niski t pomocu brojaca j
15          i proverava da li se cela niska t poklapa sa delom niske
16          s koji pocinje na poziciji i.
17
18          Cim se naidje na situaciju da se karakteri ne poklapaju,
19          izlazi se iz unutrasnje petlje. */
20         for (j = 0; t[j] != '\0'; j++)
21             if (s[i+j] != t[j])
22                 break;
23
24         /* Ako je unutrasnja petlja dosla do kraja niske t, to
25          znaci da su se svi karakteri iz t poklopili sa karakterima
26          iz s i t je podniska od s. */
27         if (t[j] == '\0')
28             return 1;
29     }
30
31     return 0;
32 }
33
34 int main()
35 {
36     /* Deklaracija niski s i t. */
37     char s[MAKS_NISKA], t[MAKS_NISKA];
38
39     /* Ucitavaju se niske s i t. */
40     printf("Unesite nisku s: ");
41     scanf("%s", s);
42     printf("Unesite nisku t: ");
43     scanf("%s", t);
44
45     /* Ispis rezultata. */
46     if (podniska(s, t))
47         printf("t je podniska niske s.\n");
48     else
49         printf("t nije podniska niske s.\n");
50
51     /* II nacin:
```

```

54     Funkcija strstr(t, s) proverava da li je t podniska od s
55     i kao povratnu vrednost vraća adresu prvog pojavljivanja t u s
56     ili NULL ukoliko se t ne pojavljuje u s.
57     Deklaracija ove funkcije se nalazi u zaglavlju string.h
58
59     char* p = strstr(t, s);
60     if(p == NULL)
61         printf("t nije podniska od s.\n");
62     else
63         printf("t je podniska od s.\n");
64     */
65
66     exit(EXIT_SUCCESS);
67 }

```

### Rešenje 3.5.8

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  #include <ctype.h>
5
6  #define MAKS_LINIJA 101
7
8  /* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
9   Funkcija ne smesta znak za novi red na kraj linije. */
10 void ucitaj_liniju(char s[], int n)
11 {
12     int i = 0;
13     int c;
14
15     /* Ucitava se karakter po karakter dok se ne unese novi red
16     ili oznaka za kraj ulaza ili dok se ne dostigne maksimalan
17     broj karaktera. */
18     while ((c = getchar()) != '\n' && i < n-1 && c != EOF)
19     {
20         s[i] = c;
21         i++;
22     }
23
24     /* Maksimalan broj karaktera za liniju je n-1 jer na kraju
25     treba ostaviti i jedno mesto za terminalnu nulu. */
26     s[i] = '\0';
27 }
28
29 /* Funkcija uklanja beline sa kraja niske s. */
30 void skрати(char s[])
31 {
32     int i;
33     /* Vrsi se prolazak kroz nisku sa desna na levo i trazi se pozicija
34     prvog karaktera koji nije belina.

```

```
35     Funkcija isspace proverava da li je dati karakter neka od belina
37     (blanko, tab ili novi red) i njena deklaracija se nalazi u
        zaglavlju ctype.h. */
39     for (i = strlen(s) - 1; i >= 0; i--)
        if (!isspace(s[i]))
41         break;

43     /* Nakon izlaska iz petlje, brojac i se nalazi na poziciji prvog
        karaktera sa desne strane koji nije belina. Iz tog razloga se
45     na poziciju i+1 upisuje terminalna nula kao oznaka da se sada
        tu nalazi kraj niske. */
47     s[i + 1] = '\0';
    }

49     int main()
51     {
        /* Deklaracija niske. */
53     char s[MAKS_LINIJA];

55     /* Ucitava se cela linija sa ulaza. */
        printf("Unesite nisku:\n");
57     ucitaj_liniju(s, MAKS_LINIJA);

59     /* NAPOMENA:
        Postoji vise nacina za ucitavanje cele linije sa standardnog
61     ulaza koriscenjem funkcija iz standardne c biblioteke. Jedan od
        njih je koriscenjem funkcije gets:
63     gets(s);
        Postoje razlozi zasto ova funkcija nije bezbedna za koriscenje
65     i oni ce biti objasnjeni u kasnijim poglavljima.
        U poglavlju "Datoteke" ce biti predstavljeni i bezbedni nacini
67     da se to uradi koriscenjem nekih drugih funkcija. */

69     /* Ispis rezultata. */
        printf("Ucitana niska:\n%s*\n", s);
71     skрати(s);
        printf("Izmenjena niska:\n%s*\n", s);

73     exit(EXIT_SUCCESS);
75 }
```

#### Rešenje 3.5.9

```
#include <stdio.h>
2 #include <stdlib.h>
#include <string.h>
4 #include <ctype.h>

6 #define MAKS_LINIJA 21
```



```

8  /* Funkcija uklanja sva slova iz niske s. */
void ukloni_slova(char s[])
10 {
    int i, j;

12     /* Prolazi se kroz nisku s karakter po karakter i vrsi se provera
14     da li trenutni karakter treba da se zadrzi. Karakter treba da
        se zadrzi ukoliko nije ni malo ni veliko slovo.

16     Brojac j služi da pamti gde se upisuje sledeći karakter koji
18     treba da se zadrži i svaki put kada se nađje na takav karakter,
        on se upisuje na poziciju j, a brojac j se uvećava. */
20     for(i=0, j=0; s[i]; i++){
        if(!islower(s[i]) && !isupper(s[i]))
22         {
            s[j] = s[i];
24             j++;
        }
26     }

28     /* Na kraju se na poziciji j upisuje i terminalna nula, kako bi se
        naznačilo da se kraj niske nalazi nakon poslednjeg zadržanog
30     karaktera. */
    s[j] = '\0';
32 }

34 int main()
{
36     /* Deklaracija niske. */
    char s[MAKS_LINIJA];

38     /* Učitava se niska s. */
40     printf("Unesite nisku:\n");
    scanf("%s", s);

42     /* Ispis rezultata. */
44     ukloni_slova(s);
    printf("%s\n", s);

46     exit(EXIT_SUCCESS);
48 }

```

### Rešenje 3.5.10

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <ctype.h>

5  #define MAKS_LINIJA 101

7  /* Funkcija učitava liniju maksimalne dužine n i upisuje je u s.

```

### 3 Predstavljanje podataka

---

```
    Funkcija ne smesta znak za novi red na kraj linije. */
9 void ucitaj_liniju(char s[], int n)
{
11     int i = 0, c;

13     while ((c = getchar()) != '\n' && i < n-1 && c != EOF)
    {
15         s[i] = c;
            i++;
17     }
    s[i] = '\0';
19 }

21 /* Pomocna funkcija koja proverava da li karakter c1 treba zadržati
    ako vazi da se iza njega nalazi c2. */
23 int treba_zadržati(char c1, char c2)
{
25     /* Ako neki od karaktera nije slovo, c1 se ne izbacuje. */
    if(!isalpha(c1) || !isalpha(c2))
27         return 1;

29     /* Oba karaktera se pretvaraju u veliko slovo kako bi se smanjio
        broj poredjenja. */
31     c1 = toupper(c1);
        c2 = toupper(c2);
33

35     /* c1 se zadržava ako se c2 ne nalazi iza njega u abecedi. */
    return c2 <= c1;
37 }

39 /* Funkcija uklanja sva slova za koja vazi da se neposredno
    nakon njih nalazi slovo koje je u abecedi iza njih. */
void ukloni(char s[])
41 {
    int i, j;
43     for(i=0, j=0; s[i]; i++){
        if(traba_zadržati(s[i], s[i+1]))
45         {
            s[j] = s[i];
47             j++;
        }
49     }
    s[j] = '\0';
51 }

53 int main()
{
55     /* Deklaracija niske. */
    char s[MAKS_LINIJA];

57     /* Ucitava se cela linija sa ulaza. */
59     printf("Unesite nisku:\n");
```

```

    učitaj_liniju(s, MAKS_LINIJA);

61     /* Ispis rezultata. */
63     ukloni(s);
    printf("%s\n", s);
65
    exit(EXIT_SUCCESS);
67 }

```

### Rešenje 3.5.11

```

1  #include <stdio.h>
   #include <stdlib.h>
3  #include <string.h>

5  #define MAKS_NISKA 31
   #define MAKS_REZULTAT 91

7  /* Niske se ne kopiraju naredbom dodele. Ukoliko je potrebno da neka
9   niska ima isti sadržaj kao i neka druga niska, može se koristiti
   funkcija strcpy(t, s) koja kopira karaktere niske s u nisku t
11  zajedno za terminirajućom nulom. Deklaracija ove funkcije se
   nalazi u zaglavlju string.h.

13
   Funkcija strcpy_klon predstavlja jednu implementaciju funkcije
15  strcpy. */
void strcpy_klon(char kopija[], char original[])
17 {
    int i;
19    for (i = 0; original[i]; i++)
        kopija[i] = original[i];

21    kopija[i] = '\0';
23 }

25 int main()
{
27     /* Deklaracija niski. */
    char s[MAKS_NISKA];
29    char t[MAKS_REZULTAT];

31    /* Učitava se niska. */
    printf("Unesite nisku: ");
33    scanf("%s", s);

35    /* Niska s se kopira u nisku t. */
    strcpy_klon(t, s);

37
    /* Funkcija strcat(s,t) nadovezuje karaktere niske s na kraj
39    niske t i novu nisku terminira karakterom '\0'. Deklaracija
    ove funkcije se nalazi u zaglavlju string.h. */

```

### 3 Predstavljanje podataka

---

```
41      /* Niska s se jos dva puta nadovezuje na nisku t. */
43      strcat(t, s);
44      strcat(t, s);
45
46      /* Ispis rezultata. */
47      printf("%s\n", t);
48
49      exit(EXIT_SUCCESS);
50  }
```

#### Rešenje 3.5.12

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define MAKS_NISKA 21
6  #define MAKS_N 10
7  /* Rezultat se dobija nadovezivanjem niske maksimalne duzine
8     MAKS_NISKA-1 i karaktera '-' najvise MAKS_N puta.
9     Odavde je maksimalna duzina rezultata:
10     (MAKS_NISKA - 1 + 1) * MAKS_N = MAKS_NISKA*MAKS_N.
11     Na ovo treba dodati jos jedan karakter zbog terminalne nule. */
12  #define MAKS_REZULTAT (MAKS_NISKA*MAKS_N + 1)
13
14  int main()
15  {
16      /* Deklaracija niski. */
17      char s[MAKS_NISKA];
18      char t[MAKS_REZULTAT];
19      int i, n;
20
21      /* Ucitava se niska. */
22      printf("Unesite nisku: ");
23      scanf("%s", s);
24
25      /* Ucitava se broj ponavljanja i vrsi se provera ispravnosti
26         ulaza. */
27      printf("Unesite broj n: ");
28      scanf("%d", &n);
29      if(n <= 0 || n > MAKS_N)
30      {
31          printf("Greska: neispravan unos.\n");
32          exit(EXIT_FAILURE);
33      }
34
35      /* Formira se rezultat. Prvi karakter rezultujuce niske se
36         postavlja na terminalnu nulu. Ovo se radi jer strcat
37         funkcionise tako sto krene od pocetka niske, ide do
38         terminalne nule i zatim pocevsi od tog mesta nadovezuje
```

```

40     nisku koja je prosledjena kao drugi argument. Na ovaj nacin
41     je obezbedjeno da ce prvi poziv funkcije strcat krenuti da
42     nadovezuje od pocetka niske t.
43     U petlji se na t nadovezuje prvo niska s, a zatim niska "-".
44     Ovo se ponavlja n-1 puta jer nakon poslednjeg nadovezivanja
45     niske s ne treba da se nadje "-". Iz tog razloga se po
46     zavrsetku petlje vrsi jos jedno nadovezivanje niske s, ali ne
47     i niske "-". */
48     t[0] = '\0';
49     for(i=0; i<n-1; i++)
50     {
51         strcat(t, s);
52         strcat(t, "-");
53     }
54     strcat(t, s);
55
56     /* Ispis rezultata. */
57     printf("%s\n", t);
58
59     exit(EXIT_SUCCESS);
60 }

```

### Rešenje 3.5.13

```

1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define MAKS_NISKA 21
6
7  /* Funkcija kopira prvih n karaktera niske s u nisku t. */
8  void kopiraj_n(char t[], char s[], int n)
9  {
10     int i;
11     /* Kopiranje se vrsi ili dok se ne dodje do terminalne nule u s
12     ili dok se ne prekopira n karaktera. */
13     for (i = 0; i < n && s[i] != '\0'; i++) {
14         t[i] = s[i];
15     }
16
17     /* Na kraju rezultujuce niske se upisuje terminalna nula. */
18     t[i] = '\0';
19 }
20
21 int main()
22 {
23     /* Deklaracije potrebnih promenljivih. */
24     int n;
25     char s[MAKS_NISKA], t[MAKS_NISKA];
26
27     /* Ucitava se niska. */

```

### 3 Predstavljanje podataka

```
28 printf("Unesite nisku: ");
   scanf("%s", s);

30

   /* Ucitava se broj n i vrsi se provera ispravnosti
   ulaza. */
32 printf("Unesite broj n: ");
   scanf("%d", &n);
34 if (n < 0 || n > MAKS_NISKA-1) {
36     printf("Greska: neispravan unos.\n");
       exit(EXIT_FAILURE);
38 }

40 /* Formira se rezultat. */
   kopiraj_n(t, s, n);

42

   /* II nacin:
   Koriscenjem funkcije strncpy(t, s, n), cija se deklaracija
   nalazi u zaglavlju string.h, kopira najvise n karaktera niske
   s u nisku t.
44     strncpy(t,s,n);
46     */

48     /* Ispis rezultata. */
52     printf("%s\n", t);

54     exit(EXIT_SUCCESS);
   }
```

#### Rešenje 3.5.14

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

3

   /* Duzina niske koja se ucitava, bez terminalne nule. */
5 #define MAKS_DUZINA 20

7 /* Duzine originalne i rezultujuce niske. */
   #define MAKS_NISKA (MAKS_DUZINA + 1)
9 #define MAKS_REZULTAT (2 * MAKS_DUZINA + 1)

11 /* Funkcija formira nisku t od niske s dupliranjem svakog
   karaktera. Npr. abc postaje aabbcc. */
13 void dupliranje(char t[], char s[])
   {
15     int i, j;

17     /* Brojac i oznacava tekucu poziciju u niski s, a
       brojac j oznacava tekucu poziciju u niski t. */
19     for (i = 0, j = 0; s[i] != '\0'; i++, j += 2)
       {
```

```

21     t[j] = s[i];
    t[j + 1] = s[i];
23 }

25 /* Upisuje se terminalna nula na kraj rezultujuće niske. */
    t[j] = '\0';
27 }

29 int main()
{
31     /* Deklaracija niski. */
    char s[MAKS_NISKA], t[MAKS_REZULTAT];
33
    /* Učitava se niska. */
35     printf("Unesite nisku: ");
    scanf("%s", s);
37
    /* Poziva se funkcija za dupliranje. */
39     dupliranje(t, s);

41     /* Ispis rezultata. */
    printf("%s\n", t);
43
    exit(EXIT_SUCCESS);
45 }

```

### Rešenje 3.5.15

```

#include <stdio.h>
2 #include <stdlib.h>
#include <ctype.h>
4
#define MAKS_NISKA 10
6
/* Funkcija formiraj_broj na osnovu niske b formira ceo broj
8   čiji je to zapis.

10   Ako su cifre broja a, b, c i d, tada broj možemo kreirati kao:
    a*103 + b*102 + c*101 + d*100.
12   Medjutim, efikasnije je koristiti Hornerovu sumu:
    10*(10*(10*(10*0 + a)+b)+c)+d. */
14 int formiraj_broj(char b[])
{
16     int i;
    int broj = 0, znak;
18
    /* Odredjivanje znaka broja i pozicije prve cifre. */
20     if(b[0] == '-')
    {
22         znak = -1;
        i = 1;

```

### 3 Predstavljanje podataka

---

```
24  }
    else if(b[0] == '+')
26  {
        znak = 1;
28      i = 1;
    }
30  else
    {
32      i = 0;
        znak = 1;
34  }

36  /* Prolazak kroz cifre broja i racunanje vrednosti broja
    koriscenjem Hornerove seme. Vrednost trenutne cifre se
38  dobija kada se od trenutnog karaktera (b[i]) oduzme
    karakter '0'.
40  Ako se naidje na karakter koji nije cifra, petlja se
    prekida. Na primer za b="123abc", rezultat treba da
42  bude 123. */
    for (; b[i] != '\0'; i++)
44    {
        if(isdigit(b[i]))
46        broj = broj * 10 + (b[i] - '0');
        else
48        break;
    }

50    return broj*znak;
52 }

54 int main()
    {
56    /* Deklaracija niske. */
        char s[MAKS_NISKA];

58    /* Broj se ucitava kao niska. */
60    scanf("%s", s);

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

64    /* II nacin:
        Koriscenjem funkcije atoi. Deklaracija ove funkcije se nalazi
        u zaglavlju stdlib.h.
66
68        printf("%d\n", atoi(s)); */

70    exit(EXIT_SUCCESS);
72 }
```

#### Rešenje 3.5.16



```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS_NISKA 10
5
6  /* Funkcija racuna broj cifara broja n. */
7  int broj_cifara(int n)
8  {
9      int i = 0;
10     do
11     {
12         i++;
13         n/=10;
14     }while(n);
15
16     return i;
17 }
18
19 /* Funkcija od prosledjenog broja formira nisku. */
20 void broj_u_nisku(int broj, char s[])
21 {
22     int n, cifra, i;
23
24     /* Promenljiva n cuva informaciju o duzini niske. Duzina niske
25      odgovara broju cifara prosledjenog broja. Ukoliko je broj
26      negativan, onda se duzina uvecava za 1 i na prvo mesto se
27      upisuje znak '-'. */
28     n = broj_cifara(broj);
29     if(broj < 0)
30     {
31         s[0] = '-';
32         n++;
33     }
34
35     /* U nastavku se radi sa apsolutnom vrednoscu broja. */
36     broj = abs(broj);
37
38     /* Cifre broja se upisuju u nisku s sa desna na levo. */
39     s[n] = '\0';
40     i = n-1;
41     do
42     {
43         /* Karakter koji odgovara trenutnoj cifri se dobija izrazom
44          '0' + cifra. Na primer, '0' + 5 je '5' jer se karakter '5'
45          nalazi 5 mesta nakon karaktera '0' u ASCII tablici. */
46         cifra = broj % 10;
47         broj = broj / 10;
48         s[i] = '0' + cifra;
49         i--;
50     } while(broj);
51 }
```

### 3 Predstavljanje podataka

---

```
52 int main()
53 {
54     /* Deklaracija broja i niske. */
55     int n;
56     char s[MAKS_NISKA];
57
58     /* Ucitava se broj. */
59     printf("Unesite ceo broj: ");
60     scanf("%d", &n);
61
62     /* Formira se niska. */
63     broj_u_nisku(n, s);
64
65     /* Ispis rezultata. */
66     printf("%s\n", s);
67
68     exit(EXIT_SUCCESS);
69 }
70
```

#### Rešenje 3.5.17

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_NISKA 8
6
7 /* Funkcija proverava da li je prosledjeni karakter ispravna
8    heksadekadna cifra. */
9 int hekza_cifra(char c)
10 {
11     c = toupper(c);
12
13     /* Cifra je ispravna ako je cifra ili ako je neko od slova:
14        A, B, C, D, E ili F. */
15     return isdigit(c) || (c >= 'A' && c <= 'F');
16 }
17
18 /* Funkcija proverava da li prosledjena niska s predstavlja
19    ispravan heksadekadni broj. */
20 int heksadekadni_broj(char s[])
21 {
22     int i;
23
24     /* Svaki heksadekasni broj pocinje sa 0x ili 0X. */
25     if (s[0] != '0' || toupper(s[1]) != 'X')
26         return 0;
27
28     /* Za svaki karakter niske s se proverava da li predstavlja
29        ispravnu heksadekadnu cifru. Ako se naidje na neku cifru koja
```

```

    ne zadovoljava taj uslov, onda se kao povratna vrednost
31   vraca nula. */
    for (i = 2; s[i]; i++)
33     if (!heksa_cifra(s[i]))
        return 0;
35
    /* Ako su sve cifre ispravne heksadekadne cifre, onda je i s
37     ispravan heksadekadni broj i funkcija vraca jedinicu. */
    return 1;
39 }

41 int main()
{
43     /* Deklaracija niske. */
    char s[MAKS_NISKA];

45     /* Ucitava se niska. */
47     printf("Unesite nisku: ");
    scanf("%s", s);

49     /* Ispis rezultata. */
51     if (heksadekadni_broj(s))
        printf("Korektan heksadekadni broj.\n");
53     else
        printf("Nekorektan heksadekadni broj.\n");
55
    exit(EXIT_SUCCESS);
57 }

```

### Rešenje 3.5.18

```

1  #include <stdio.h>
   #include <stdlib.h>
3  #include <string.h>
   #include <ctype.h>
5
   #define MAKS_NISKA 8
7
   /* Funkcija racuna dekadnu vrednost jedne heksadekadne cifre.
9     Ako je c broj, vrednost se dobija oduzimanjem '0'.
     Ako je c slovo, vrednost se dobija oduzimanjem 'A' i dodavanjem
11    10 (npr. vrednost karaktera 'B' je 10 + 'B' - 'A' = 11). */
int vrednost_heksa_cifre(char c)
13 {
    if(isdigit(c))
15     return c - '0';
    else
17     return 10 + toupper(c) - 'A';
}
19

/* Funkcija racuna dekadnu vrednost heksadekdnog broja. */

```

### 3 Predstavljanje podataka

```
21 int dekadna_vrednost(char s[])
22 {
23     int i, tezina_pozicije = 1, rezultat = 0;
24     int n = strlen(s);
25
26     /* Vrsi se prolazak kroz nisku sa desna na levo. Heksadekadna
27        cifra najveće težine se nalazi na poziciji n-1, a ona najveće
28        težine se nalazi na poziciji 2 (jer su prva dva karaktera 0x).
29
30        U svakoj iteraciji, na rezultat se dodaje vrednost tekuće
31        cifre, pomnožena sa vrednošću težine njene pozicije.
32        Na primer, za s = "0x1a8e", n=6
33        i = 5, rezultat += vrednost('e')*1 => rezultat += 11*1
34        i = 4, rezultat += vrednost('8')*16 => rezultat += 8*16
35        i = 3, rezultat += vrednost('a')*256 => rezultat += 10*256
36        i = 2, rezultat += vrednost('1')*4096 => rezultat += 1*4096 */
37     for (i = n - 1; i >= 2; i--)
38     {
39         rezultat += tezina_pozicije * vrednost_heksa_cifre(s[i]);
40         tezina_pozicije *= 16;
41     }
42
43     return rezultat;
44 }
45
46 int main()
47 {
48     /* Deklaracija niske. */
49     char s[MAKS_NISKA];
50
51     /* Ucitava se niska. */
52     printf("Unesite nisku: ");
53     scanf("%s", s);
54
55     /* Ispis rezultata. */
56     printf("%d\n", dekadna_vrednost(s));
57
58     exit(EXIT_SUCCESS);
59 }
```

#### Rešenje 3.5.19

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_LINIJA 81
6
7 /* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
8    Funkcija ne smesta znak za novi red na kraj linije. */
9 int ucitaj_liniju(char s[], int n)
```

```

11  {
12      int i = 0;
13      int c;
14
15      /* Ucitava se karakter po karakter dok se ne unese novi red
16         ili oznaka za kraj ulaza ili dok se ne dostigne maksimalan
17         broj karaktera. */
18      while ((c = getchar()) != '\n' && i < n-1 && c != EOF)
19      {
20          s[i] = c;
21          i++;
22      }
23
24      /* Maksimalan broj karaktera za liniju je n-1 jer na kraju
25         treba ostaviti i jedno mesto za terminalnu nulu. */
26      s[i] = '\0';
27
28      return i;
29  }
30
31  int main()
32  {
33      /* Deklaracija potrebnih promenljivih. */
34      char linija[MAKS_LINIJA], najduza_linija[MAKS_LINIJA];
35      int duzina_najduze = 0, duzina;
36
37      /* U petlji se ucitavaju linije sve dok se ne unese prazna linija.
38         Ukoliko se unese linija koja je duza od trenutno najduze,
39         vrsi se azuriranje duzine najduze linije, kao i same linije. */
40      while ((duzina = ucitaj_liniju(linija, MAKS_LINIJA)) > 0)
41      {
42          if (duzina_najduze < duzina)
43          {
44              duzina_najduze = duzina;
45              strcpy(najduza_linija, linija);
46          }
47      }
48
49      /* Ispis rezultata. */
50      if(duzina_najduze == 0)
51          printf("Nije uneta nijedna linija.\n");
52      else
53          printf("%s\n%d\n", najduza_linija, duzina_najduze);
54
55      exit(EXIT_SUCCESS);
56  }

```

### Rešenje 3.5.20

```

1  #include <stdio.h>
2  #include <stdlib.h>

```

```
1  #include <string.h>
4  #include <ctype.h>
6
6  #define MAKS_RECENICA 81
8
8  /* Funkcija ucitava recenicu maksimalne duzine n. */
10 int ucitaj_recenicu(char s[], int n)
10 {
12     int i = 0, c;
14
14     /* Preskacu se beline sa pocetka ako ih ima. Po zavrsetku ove
16     petlje u c se nalazi prvi sledeci karakter koji nije belina. */
16     do{
18         c = getchar();
18     } while(isspace(c));
20
20     /* Ako je taj karakter EOF, zavrшава se ucitavanje. */
22     if(c == EOF)
24         return 0;
26
26     /* U nisku se smesta karakter, prelazi se na sledeci karakter i
28     postupak se ponavlja sve dok se ne unese tacka, EOF ili dok
30     se ne popuni maksimalan broj karaktera koje recenica moze
32     da sadrzi. */
32     do{
34         s[i] = c;
34         i++;
36         c = getchar();
36     } while (c != '.' && i < n-2 && c != EOF);
38
38     /* Ako je poslednji uneti karakter EOF, zavrшава se ucitavanje. */
40     if(c == EOF)
42         return 0;
44
44     /* Na kraju svake recenice stoji tacka za kojom sledi '\0'. */
46     s[i] = '.';
48     s[i+1] = '\0';
50
50     return i+1;
52 }
54
54 /* Funkcija prebrojava mala i velika slova. */
56 void prebroj(char s[] , int* broj_malih, int* broj_velikih)
56 {
58     int i, mala = 0, velika = 0;
60
60     for(i=0; s[i]; i++)
62     {
64         if(islower(s[i]))
66             mala++;
68         else if(isupper(s[i]))
70             velika++;
72     }
74 }
```

```

    }

56     *broj_malih = mala;
58     *broj_velikih = velika;
    }

60 int main()
62 {
    /* Deklaracija potrebnih promenljivih. */
64     char recenica[MAKS_RECENICA];
    char rezultujuca_recenica[MAKS_RECENICA];
66     int najveca_razlika = -1, trenutna_razlika;
    int mala, velika;
68     int ucitana_bar_jedna = 0;

70     /* U petlji se ucitavaju recenice sve dok se ne unese EOF. */
    while (ucitaj_recenicu(recenica, MAKS_RECENICA) > 0) {
72
74         /* Prebrojavaju se mala i velika slova. */
        prebroj(recenica, &mala, &velika);

76         /* Racuna se njihova apsolutna razlika. */
        trenutna_razlika = abs(mala-velika);

78         /* Ako je razlika veca od trenutno najvece, azurira se vrednost
80            najvece razlike i pamti se trenutna recenica. */
        if(trenutna_razlika > najveca_razlika){
82            najveca_razlika = trenutna_razlika;
            strcpy(rezultujuca_recenica, recenica);
84        }

86        /* Indikator koji oznacava da se petlja bar jednom izvrsila, tj.
            da korisnik nije odmah zadao EOF. */
88        ucitana_bar_jedna = 1;
    }

90    /* Ispis rezultata. */
92    if(ucitana_bar_jedna)
        printf("%s\n", rezultujuca_recenica);
94    else
        printf("Nije uneta nijedna recenica. ");
96    exit(EXIT_SUCCESS);
98 }

```

### Rešenje 3.5.21

```

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

4  #define MAKS_NISKA 21

```

```
6  /* Funkcija vraca adresu prvog pojavljivanja karaktera c u niski s
   ili NULL ukoliko se c ne pojavljuje u s.
8
   Trazeni rezultat moze se dobiti koriscenjem funkcije strchr
10  cija se deklaracija nalazi u zaglavlju string.h.
   Funkcija strchr_klon predstavlja jednu mogucu implementaciju
12  ove funkcije. */
char* strchr_klon(char s[], char c)
14 {
   int i;
16
   /* Za svaki karakter se proverava da li je jednak karakteru c.
   Ako se naidje na takav karakter, kao povratna vrednost funkcije
18   se vraca njegova adresa (&s[i]). */
   for(i=0; s[i]; i++)
20     if(s[i] == c)
22       return &s[i];

   /* Ako je petlja zavrшена, znaci da nije pronadjen karakter koji
24   je jednak karakteru c i kao povratna vrednost funkcije se vraca
   NULL pokazivac kao oznaka da se c ne nalazi u s. */
26   return NULL;
28 }

int main()
30 {
   /* Deklaracije potrebnih promenljivih. */
32   char s[MAKS_NISKA];
   char c;
34

   /* Ucitava se niska s. */
36   printf("Unesite nisku s: ");
   scanf("%s", s);
38

   /* Preskace se novi red koji je unet nakon niske s i
   ucitava se karakter c. */
40   getchar();
   printf("Unesite karakter c: ");
42   scanf("%c", &c);

   /* Racunanje i ispis rezultata. */
44   char* p = strchr_klon(s, c);
   if(p == NULL)
46     printf("Pozicija: -1\n");
   else
48     printf("Pozicija: %ld\n", p-s);

   exit(EXIT_SUCCESS);
50
52 }
54
```



## Rešenje 3.5.22

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define MAKS_NISKA 21
6
7  /* Funkcija racuna duzinu prefiksa niske t koji se moze
8   zapisati pomocu karaktera niske s.
9   Na primer, t="programiranje", s="grupacija", rezultat je 2 jer
10   niska s sadrzi prva dva karaktera niske t, ali ne i treci.
11
12   Trazeni rezultat moze se dobiti koriscenjem funkcije strstr
13   cija se deklaracija nalazi u zaglavlju string.h.
14   Funkcija strstr_klon predstavlja jednu mogucu implementaciju
15   ove funkcije. */
16 int strstr_klon(char t[], char s[])
17 {
18     int i, brojac = 0;
19
20     /* Ide se redom po karakterima niske t i za svaki karakter se
21     vrši provera da li se on nalazi u zapisu niske s. Za ovo se
22     koristi funkcija strchr. Ako se nalazi, uvecava se brojac,
23     a ako se ne nalazi, prekida se petlja. */
24     for(i=0; t[i]; i++)
25     {
26         if(strchr(s, t[i]) != NULL)
27             brojac++;
28         else
29             break;
30     }
31
32     return brojac;
33 }
34
35 int main()
36 {
37     /* Deklaracije potrebnih promenljivih. */
38     char s[MAKS_NISKA];
39     char t[MAKS_NISKA];
40
41     /* Ucitavaju se niske. */
42     printf("Unesite nisku t: ");
43     scanf("%s", t);
44     printf("Unesite nisku s: ");
45     scanf("%s", s);
46
47     /* Racunanje i ispis rezultata. */
48     printf("%d\n", strstr_klon(t,s));
49
50     exit(EXIT_SUCCESS);
```

```
}  
}
```

#### Rešenje 3.5.23

Rešenje ovog zadatka se svodi na rešenje zadatka 3.5.22, uz razliku da se ovde prebrojavaju karakteri koji se ne nalaze u zapisu niske s.

#### Rešenje 3.5.24

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4
5  #define MAKS_NISKA 101
6
7  /* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
   Funkcija ne smesta znak za novi red na kraj linije. */
8  void ucitaj_liniju(char s[], int n)
9  {
10     int i = 0, c;
11
12     while ((c = getchar()) != '\n' && i < n-1 && c != EOF) {
13         s[i] = c;
14         i++;
15     }
16
17     s[i] = '\0';
18 }
19
20 /* Funkcija vraca pokazivac na prvo pojavljivanje niske
   t u okviru niske s ili NULL ukoliko se t ne nalazi u s.
21
22 Trazeni rezultat moze se dobiti koriscenjem funkcije strstr
23 cija se deklaracija nalazi u zaglavlju string.h.
24 Funkcija strstr_klon predstavlja jednu mogucu implementaciju
25 ove funkcije. */
26 char* strstr_klon(char s[], char t[])
27 {
28     int i, j;
29
30     /* Spoljasnja petlja ide redom po niski s. */
31     for (i = 0; s[i] != '\0'; i++) {
32         /* Unutrasnja petlja ide redom po niski t pomocu brojaca j
33            i proverava da li se cela niska t poklapa sa delom niske
34            s koji pocinje na poziciji i.
35
36            Cim se naidje na situaciju da se karakteri ne poklapaju,
37            izlazi se iz unutrasnje petlje. */
38         for (j = 0; t[j] != '\0'; j++)
39             if (s[i+j] != t[j])
40                 continue;
41         return s+i;
42     }
43     return NULL;
44 }
```

```

43     break;
44
45     /* Ako je untrasnja petlja dosla do kraja niske t, to
46        znaci da su se svi karakteri iz t poklopili sa karakterima
47        iz s i t je podniska od s.
48        Kao povratna vrednost se vraca mesto gde t pocinje u s. */
49     if (t[j] == '\0')
50         return &s[i];
51 }
52
53 return NULL;
54 }
55
56 int main()
57 {
58     /* Deklaracije potrebnih promenljivih. */
59     char linija[MAKS_NISKA];
60     int i, bar_jedna = 0;
61
62     /* Ucitavanje linija i ispis rednih brojeva linija
63        koje sadrze rec "program". */
64     for(i=1; i<=5; i++)
65     {
66         ucitaj_liniju(linija, MAKS_NISKA);
67         if(strstr_klon(linija, "program") != NULL){
68             printf("%d ", i);
69             bar_jedna = 1;
70         }
71         /* II nacin: koriscenjem funkcije strstr cija se deklaracija
72            nalazi u zaglavlju string.h:
73            if(strstr(linija, "program") != NULL){
74                printf("%d ", i);
75                bar_jedna = 1;
76            } */
77     }
78     printf("\n");
79
80     /* Ako indikator bar_jedna i dalje ima vrednost 0, znaci da
81        nije uneta nijedna linija koja sadrzi rec "program". */
82     if(!bar_jedna)
83     {
84         printf("Nijedna linija ne sadrzi nisku program.\n");
85     }
86
87     exit(EXIT_SUCCESS);
88 }

```

### Rešenje 3.5.25

```

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

```

### 3 Predstavljanje podataka

---

```
3  #define MAKS_NISKA 21
5
7  /* Funkcija poredi dve niske i vraca nulu ukoliko su jednake,
   nesto pozitivno ukoliko je niska s1 leksikografski iza s2,
   a neku negativnu vrednost inace.
9
11     Trazeni rezultat moze se dobiti koriscenjem funkcije strcmp
   cija se deklaracija nalazi u zaglavlju string.h.
   Funkcija strcmp_klon predstavlja jednu mogucu implementaciju
13     ove funkcije. */
15 int strcmp_klon(char s1[], char s2[])
16 {
17     int i;
18
19     /* Prolazi se kroz obe niske dok god se odgovarajuci karakteri
   poklapaju. Ako se u ovom prolasku desi da je petlja dosla
   do kraja obe niske, onda su one jednake i kao povratna vrednost
21     funkcije se vraca 0. */
23     for (i = 0; s1[i] == s2[i]; i++)
24         if (s1[i] == '\0')
25             return 0;
26
27     /* Ako niske nisu jednake, znaci da je brojac i stao na prvom
   mestu gde se niske s1 i s2 razlikuju. Posto funkcija treba da
   vrati pozitivnu vrednost ako je niska s1 laksikografski iza
29     s2, a negativnu u suprotnom, ovo moze biti realizovano vracanjem
   razlike ASCII kodova.
   Na primer: s1 = "pero", s2 = "program"
   Nakon petlje, brojac i ima vrednost 1 (jer je tu prva razlika).
33     Kao povratna vrednost se vraca s1[1] - s2[1] = 'e' - 'r' = -13
   sto kao negativna vrednost govori da se s1 nalazi
35     leksikografski ispred s2. */
37     return s1[i] - s2[i];
38 }
39
41 int main()
42 {
43     /* Deklaracije potrebnih promenljivih. */
44     char s[MAKS_NISKA];
45     char t[MAKS_NISKA];
46     int rezultat;
47
48     /* Ucitavaju se niske s i t. */
49     printf("Unesite nisku s: ");
50     scanf("%s", s);
51     printf("Unesite nisku t: ");
52     scanf("%s", t);
53
54     /* Vrsi se poredjenje niski i ispisuje se rezultat. */
55     rezultat = strcmp_klon(s,t);
```

```

55  /* II nacin: Koriscenjem funkcije strcmp cija se deklaracija
    nalazi u zaglavlju string.h:
57  rezultat = strcmp(s, t); */

59  if(rezultat == 0)
    printf("%s\n", s);
61  else if(rezultat < 0)
    printf("%s\n%s\n", s, t);
63  else
    printf("%s\n%s\n", t, s);

65  exit(EXIT_SUCCESS);
67  }

```

### Rešenje 3.5.26

```

1  #include <stdio.h>
   #include <stdlib.h>
3  #include <string.h>

5  #define MAKS_NISKA 21

7  /* Funkcija obrce nisku s. */
   void obrni(char s[])
9  {
    int i, j;
11   int n = strlen(s);
    char c;

13   /* Brojac i ide od prvog karaktera niske s, a brojac j
    od poslednjeg i dok god se ne sretnu, vrsi se zamena
    karaktera koji se nalaze na njihovim pozicijama. */
17   for(i=0, j=n-1; i<j; i++,j--)
    {
19       c = s[i];
        s[i] = s[j];
21       s[j] = c;
    }
23 }

25 int main()
   {
27   /* Deklaracija niske. */
    char s[MAKS_NISKA];

29   /* Ucitava se niska. */
31   printf("Unesite nisku: ");
    scanf("%s", s);

33   /* Racunanje i ispis rezultata. */
35   obrni(s);

```

### 3 Predstavljanje podataka

---

```
printf("%s\n", s);
37 exit(EXIT_SUCCESS);
39 }
```

#### Rešenje 3.5.27

```
#include <stdio.h>
2 #include <stdlib.h>
#include <string.h>
4
#define MAKS_NISKA 21
6
/* Funkcija rotira nisku za jedno mesto ulevo. */
8 void rotiraj1(char s[], int n)
{
10     int i;
    /* Pamti se prvi karakter. */
12     char prvi = s[0];

    /* Svaki sledeci karakter se pomera za jedno mesto ulevo. */
14     for(i=0; i<n-1; i++)
16         s[i] = s[i+1];

    /* Prvi karakter se upisuje na kraj niske. */
18     s[n-1] = prvi;
20 }

/* Funkcija rotira nisku s za k mesta ulevo. */
22 void rotiraj(char s[], int k){
24     int i;
    int n = strlen(s);

26     for(i=0; i<k; i++)
28         rotiraj1(s, n);
}

30
int main()
32 {
    /* Deklaracija potrebnih promenljivih. */
34     char s[MAKS_NISKA];
    int k;

36
    /* Ucitavaju se niska i vrednost k. */
38     printf("Unesite nisku: ");
    scanf("%s", s);
40     printf("Unesite k: ");
    scanf("%d", &k);
42
    /* Vrsi se provera ispravnosti ulaza. */
44     if(k < 0)
```

```
46     {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
48 }

50 /* Racunanje i ispis rezultata. */
    rotiraj(s, k);
52     printf("%s\n", s);

54     exit(EXIT_SUCCESS);
}
```

### Rešenje 3.5.28

```
#include <stdio.h>
2  #include <stdlib.h>
    #include <string.h>
4  #include <ctype.h>

6  #define MAKS_NISKA 21

8  /* Funkcija svako slovo niske s menja sa slovom
    koje se u ASCII tablici nalazi neposredno iza njega.
10     Specijalan slucaj je slovo z koje treba da se zameni
    sa slovom a. Ostali karakteri ostaju nepromenjeni. */
12 void sifruj(char s[])
    {
14     int i;

16     for(i=0; s[i]; i++)
    {
18         if(isalpha(s[i]))
        {
20             if(s[i] == 'z')
                s[i] = 'a';
22             else if(s[i] == 'Z')
                s[i] = 'A';
24             else
                s[i] = s[i] + 1;
26         }
    }
28 }

30 int main()
    {
32     /* Deklaracija niske. */
    char s[MAKS_NISKA];

34     /* Ucitava se niska. */
    printf("Unesite nisku: ");
36     scanf("%s", s);
```

### 3 Predstavljanje podataka

---

```
38      /* Racunanje i ispis rezultata. */
40      sifruj(s);
      printf("%s\n", s);
42
      exit(EXIT_SUCCESS);
44 }
```

#### Rešenje 3.5.29

```
#include <stdio.h>
2 #include <stdlib.h>
#include <ctype.h>
4
#define MAKS_DUZINA 20
6 #define MAKS_NISKA (MAKS_DUZINA + 1)
#define MAKS_REZULTAT (3*MAKS_DUZINA + 1)
8
/* Pomocna funkcija koja za prosledjeno slovo
10 vraca slovo koje ide posle njega. */
char sledeci(char c)
12 {
    if(c == 'z')
14         return 'a';

    if(c == 'Z')
16         return 'A';

    return c+1;
18
20 }

22 /* Funkcija od niske s formira rezultujucu nisku koja se dobija
    na sledeci nacin:
24 1. ako je s[i] slovo, onda se u rezultujucu nisku upisuju naredna
    tri slova alfabeta (kada se stigne do kraja alfabeta, ide se u
26 krug, tj. nakon slova z sledi slovo a)
    2. ako s[i] nije slovo, samo se s[i] prepisuje u rezultat. */
28 void sifruj(char s[], char rezultat[])
{
30     int i, j;

32     /* Brojac i se koristi za nisku s, a brojac j za
        rezultujucu nisku. */
34     for(i=0, j=0; s[i]; i++)
    {
36         if(isalpha(s[i]))
        {
38             /* Ako je s[i] slovo, onda se u rezultat upisuju 3 slova koja
                slede nakon njega. */
40             rezultat[j] = sledeci(s[i]);
            rezultat[j+1] = sledeci(rezultat[j]);
```



```

42     rezultat[j+2] = sledeci(rezultat[j+1]);
43     j += 3;
44 }
45 else
46 {
47     /* Ako s[i] nije slovo, onda se samo prepisuje u rezultat. */
48     rezultat[j] = s[i];
49     j++;
50 }
51 }
52
53 /* Na kraj rezultata se dopisuje terminalna nula. */
54 rezultat[j] = '\0';
55 }
56
57 int main()
58 {
59     /* Deklaracija niske. */
60     char s[MAKS_NISKA];
61     char rezultat[MAKS_REZULTAT];
62
63     /* Ucitava se niska. */
64     printf("Unesite nisku: ");
65     scanf("%s", s);
66
67     /* Racunanje i ispis rezultata. */
68     sifruj(s, rezultat);
69     printf("%s\n", rezultat);
70
71     exit(EXIT_SUCCESS);
72 }

```

### Rešenje 3.5.30

```

#include <stdio.h>
2 #include <stdlib.h>
#include <ctype.h>
4
#define MAKS_DUZINA 20
6 #define MAKS_NISKA (MAKS_DUZINA + 1)
#define MAKS_REZULTAT (2*MAKS_DUZINA + 1)
8
/* Funkcija od niske s formira rezultujucu nisku na sledeci nacin:
10 1. Svi karakteri niske s koji su jednaki c1 se dupliraju.
11 2. Svi karakteri niske s koji su jednaki c2 se brisu.
12 3. Ostali karakteri se samo prepisuju. */
void formiraj(char s[], char rezultat[], char c1, char c2)
14 {
15     int i, j;
16
17     /* Brojac i se koristi za nisku s, a brojac j za

```

```
18     rezultujucu nisku. */
19     for(i=0, j=0; s[i]; i++)
20     {
21         if(s[i] == c1)
22         {
23             /* Ako je s[i] jednako c1, duplira se u rezultatu. */
24             rezultat[j] = s[i];
25             rezultat[j+1] = s[i];
26             j += 2;
27         }
28         else if(s[i] != c2)
29         {
30             /* Ako s[i] razlicito od c2, upisuje se u rezultat. */
31             rezultat[j] = s[i];
32             j++;
33         }
34     }
35
36     /* Na kraj rezultata se dopisuje terminalna nula. */
37     rezultat[j] = '\0';
38 }
39
40 int main()
41 {
42     /* Deklaracija potrebnih promenljivih. */
43     char s[MAKS_NISKA];
44     char rezultat[MAKS_REZULTAT];
45     char c1, c2;
46
47     /* Ucitavaju se niska i karakteri. */
48     printf("Unesite nisku: ");
49     scanf("%s", s);
50     getchar();
51     printf("Unesite prvi karakter: ");
52     scanf("%c", &c1);
53     getchar();
54     printf("Unesite drugi karakter: ");
55     scanf("%c", &c2);
56
57     /* Vrsi se provera ispravnosti ulaza. */
58     if(c1 == c2)
59     {
60         printf("Greska: neispravan unos.\n");
61         exit(EXIT_FAILURE);
62     }
63
64     /* Racunanje i ispis rezultata. */
65     formiraj(s, rezultat, c1, c2);
66     printf("%s\n", rezultat);
67
68     exit(EXIT_SUCCESS);
69 }
```

## Rešenje 3.5.31

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>
5
6 #define MAKS_NISKA 20
7
8 /* Pomocna funkcija koja racuna dekadnu vrednost prosledjenog
   karaktera ('1' ima vrednost 1, 'C' ima vrednost 12). */
10 unsigned vrednost_cifre(char c)
11 {
12     c = toupper(c);
13     if(isdigit(c))
14         return c - '0';
15     else
16         return c - 'A' + 10;
17 }
18
19 /* Funkcija racuna dekadnu vrednost neonacenog broja
   zapisanog u datoj osnovi. */
20 unsigned int u_dekadni_sistem(char broj[], unsigned int osnova)
21 {
22     int i, n = strlen(broj);
23     int rezultat = 0;
24     int tezina_pozicije = 1;
25
26     for(i=n-1; i>=0; i--)
27     {
28         rezultat += vrednost_cifre(broj[i])*tezina_pozicije;
29         tezina_pozicije *= osnova;
30     }
31
32     return rezultat;
33 }
34
35 /* Funkcija obrce nisku s. */
36 void obrni(char s[])
37 {
38     int i, j;
39     int n = strlen(s);
40     char c;
41
42     for(i=0, j=n-1; i<j; i++,j--){
43         c = s[i];
44         s[i] = s[j];
45         s[j] = c;
46     }
47 }
48
49 /* Pomocna funkcija koja dekadnu vrednost cifre pretvara u
```

```
    odgovarajuci karakter (12 u 'C', 5 u '5', itd.). */
52 char ostatak_u_char(int ostatak)
{
54     if(ostatak < 10)
        return '0' + ostatak;
56     else
        return 'A' + ostatak - 10;
58 }

60 /* Funkcija datu dekadnu vrednost broja prebacuje u broj u
    datoj osnovi. */
62 void iz_dekadnog_sistema(unsigned int broj, unsigned int osnova,
                           char rezultat[])
64 {
    int i = 0;
66     int ostatak;

68     do{
        ostatak = broj % osnova;
70         broj = broj / osnova;
        rezultat[i] = ostatak_u_char(ostatak);
72         i++;
    }while(broj);

74     rezultat[i] = '\0';

76     obrni(rezultat);
78 }

80 int main()
{
82     char broj[MAKS_NISKA];
    char broj2[MAKS_NISKA];
84     unsigned int osnova1, osnova2;

86     printf("Unesite n, o1 i o2: ");
    scanf("%s%u%u", broj, &osnova1, &osnova2);
88

    unsigned dekadna_vrednost = u_dekadni_sistem(broj, osnova1);
90     printf("Dekadna vrednost broja %s: %u\n", broj, dekadna_vrednost);

92     iz_dekadnog_sistema(dekadna_vrednost, osnova2, broj2);
    printf("Vrednost broja %u u osnovi %u: %s\n", dekadna_vrednost,
        osnova2, broj2);
94     exit(EXIT_SUCCESS);
96 }
```

## 3.7 Višedimenzionni nizovi

**Zadatak 3.7.1** Napisati program koji učitava i zatim ispisuje vrednosti učitane matrice. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ , a da se sa ulaza najpre učitavaju dva cela broja  $m$  i  $n$ , a potom i elementi matrice celih brojeva dimenzije  $m \times n$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Matrica je:
1 2 3 4
5 6 7 8
9 10 11 12
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Matrica je:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
500 3
Greska: neispravan unos.
```

[Rešenje 3.7.1]

**Zadatak 3.7.2** Napisati program koji za učitane celobrojnu matricu dimenzije  $m \times n$  izračunava njenu Euklidsku normu. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. UPUTSTVO: *Euklidska norma matrice je kvadratni koren sume kvadrata svih elemenata matrice.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Euklidska norma je: 25.495
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Euklidska norma je: 15.875
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
500 3
Greska: neispravan unos.
```

[Rešenje 3.7.2]

**Zadatak 3.7.3** Napisati funkcije za rad sa celobrojnim matricama:

### 3 Predstavljanje podataka

---

- (a) `void ucitaj(int a[][MAKS], int n, int m)` kojom se učitavaju vrednosti matrice celih brojeva  $a$  dimenzije  $m \times n$ ,
- (b) `void ispisi(int a[][MAKS], int n, int m)` kojom se ispisuju vrednosti matrice  $a$  dimenzije  $m \times n$ .

Napisati program koji najpre učitava, a zatim i ispisuje vrednosti učitane matrice. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Matrica je:
1 2 3 4
5 6 7 8
9 10 11 12
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Matrica je:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
500 3
Greska: neispravan unos.
```

[Rešenje 3.7.3]

**Zadatak 3.7.4** Napisati funkciju `void transponovana(int a[][MAKS], int m, int n, int b[][MAKS])` koja određuje matricu  $b$  koja je dobijena transponovanjem matrice  $a$ . Napisati program koji za učitane matricu celih brojeva<sup>2</sup> ispisuje odgovarajuću transponovanu matricu. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

---

<sup>2</sup>Pod pojmom *učitati matricu* ili *za datu matricu* uvek se podrazumeva da se prvo unose dimenzije matrice, a potom i sama matrica.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Transponovana matrica je:
1 5 9
2 6 10
3 7 11
4 8 12

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Transponovana matrica je:
1 5 7 1 0
1 0 8 2 1
2 2 9 4 1

```

## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
500 3
Greska: neispravan unos.

```

[Rešenje 3.7.4]

**Zadatak 3.7.5** Napisati funkciju `void razmeni(int a[][MAKS], int m, int n, int k, int t)` u kojoj se razmenjuju elementi  $k$ -te i  $t$ -te vrste matrice  $a$  dimenzije  $m \times n$ . Napisati program koji za učitanoj 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  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Unesite indekse vrsta:
0 2
9 10 11 12
5 6 7 8
1 2 3 4

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Unesite indekse vrsta:
1 3
1 1 2
1 2 4
7 8 9
5 0 2
0 1 1

```

## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Unesite indekse vrsta:
-1 50
Greska: neispravan unos.

```

[Rešenje 3.7.5]

**Zadatak 3.7.6** Napisati program koji za učitanoj matricu celih brojeva ispisuje indekse onih elemenata matrice koji su jednaki zbiru svih svojih susednih

### 3 Predstavljanje podataka

```

- - - - - s b s -
- s s s - - s s s -
- s a s - - - - -
- s s s - - - - -
- - - - - - - s s
- - - - - - - s c

```

Slika 3.1: *Susedni elementi u matrici.*

elemenata. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

UPUTSTVO: *Broj susednih elemenata matrice zavisi od položaja elementa u matrici. Na slici 3.1 su slovom s obeleženi susedni elementi matrice za elemente a, b i c.*

#### Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
4 5
Unesite elemente matrice:
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
Indeksi elemenata koji su
jednaki zbiru suseda su:
1 1
3 1
3 4

```

#### Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
7 10 12 20
-1 -3 1 7
0 -4 7 2 0
Indeksi elemenata koji su
jednaki zbiru suseda su:
0 3
1 2

```

#### Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
5 -3
Greska: neispravan unos.

```

[Rešenje 3.7.6]

**Zadatak 3.7.7** Napisati funkciju koja formira niz  $b_0, b_1, \dots, b_n$  od matrice tako što element niza  $b_i$  izračunava kao srednju vrednost elemenata  $i$ -te vrste matrice. Napisati program koji za učitane matricu celih brojeva ispisuje dobijeni niz. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.



## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
4 5
Unesite elemente matrice:
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
Dobijeni niz je:
1.6 3.6 0.6 1.4

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 4
Unesite elemente matrice:
7 10 12 20
-1 -3 1 7
0 -47 2 0
Dobijeni niz je:
12.25 1 -11.25

```

## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
51 13
Greska: neispravan unos.

```

[Rešenje 3.7.7]

**Zadatak 3.7.8** Relacija se može predstaviti kvadratnom matricom nula i jedinica na sledeći način: element  $i$  je u relaciji sa elementom  $j$  ukoliko se u preseku  $i$ -te vrste i  $j$ -te kolone nalazi jedinica, a nije u relaciji ukoliko se tu nalazi nula. Napisati funkcije:

- (a) `int reflektivna(int a[][MAKS], int n)` kojom se za relaciju zadatom matricom  $a$  dimenzije  $n \times n$  ispituje da li je reflektivna;
- (b) `int simetricna(int a[][MAKS], int n)` kojom se za relaciju zadatom matricom  $a$  dimenzije  $n \times n$  ispituje da li je simetrična;
- (c) `int tranzitivna(int a[][MAKS], int n)` kojom se za relaciju zadatom matricom  $a$  ispituje dimenzije  $n \times n$  da li je tranzitivna;
- (d) `int ekvivalencija(int a[][MAKS], int n)` kojom se za relaciju koja je zadata matricom  $a$  dimenzije  $n \times n$  ispituje da li je relacija ekvivalencije.

Napisati program koji za učitane dimenziju  $n$  i kvadratnu matricu dimenzije  $n \times n$  ispisuje osobine odgovarajuće relacije. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$  i da matrica za vrednosti elemenata može imati samo nule i jedinice. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
4
Unesite elemente matrice:
1 0 0 0
0 1 1 0
0 0 1 0
0 0 0 0
Relacija nije refleksivna.
Relacija nije simetrična.
Relacija jeste tranzitivna.
Relacija nije ekvivalencija.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
4
Unesite elemente matrice:
1 1 0 0
1 1 1 0
0 0 1 0
0 0 0 1
Relacija jeste refleksivna.
Relacija jeste simetrična.
Relacija nije tranzitivna.
Relacija nije ekvivalencija.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
54
Greska: neispravan unos.
```

[Rešenje 3.7.8]

**Zadatak 3.7.9** Data je kvadratna matrica dimenzije  $n \times n$ .

- Napisati funkciju `float trag(float a[] [MAKS], int n)` koja računa trag matrice, odnosno zbir elemenata na glavnoj dijagonali matrice.
- Napisati funkciju `float suma_sporodna(float a[] [MAKS], int n)` koja računa zbir elemenata na sporednoj dijagonali matrice.
- Napisati funkciju `float suma_iznad(float a[] [MAKS], int n)` koja određuje sumu elemenata iznad glavne dijagonale.
- Napisati funkciju `float suma_ispod(float a[] [MAKS], int n)` koja određuje sumu elemenata ispod sporedne dijagonale matrice.

Napisati program koji za učitanoj 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  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 4
Unesite elemente matrice:
6 12.08 -1 20.5
8 90 -33.4 19.02
7.02 5 -20 14.5
8.8 -1 3 -22.8
Trag je 53.20.
Suma na sporednoj dijagonali je 0.90.
Suma iznad glavne dijagonale je 31.70.
Suma ispod sporedne dijagonale je -1.82.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 5
Unesite elemente matrice:
1 2 3 5 5
7 8 9 0 1
6 4 3 2 2
8 9 1 3 4
0 3 1 8 6
Trag je 21.00.
Suma na sporednoj dijagonali je 17.00.
Suma iznad glavne dijagonale je 33.00.
Suma ispod sporedne dijagonale je 24.00.
```

[Rešenje 3.7.9]

**Zadatak 3.7.10** Kvadratna matrica je donje trougaona ako se u gornjem trouglu (iznad glavne dijagonale, ne uključujući dijagonalu) nalaze sve nule. Napisati program koji za učitane kvadratnu matricu proverava da li je ona donje trougaona i ispisuje odgovarajuću poruku. Pretpostaviti da je maksimalna dimenzija matrice  $100 \times 100$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 5
Unesite elemente matrice:
-1 0 0 0 0
2 10 0 0 0
0 1 5 0 0
7 8 20 14 0
-23 8 5 1 11
Matrica jeste donje trougaona.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 3
Unesite elemente matrice:
2 -2 1
1 2 2
2 1 -2
Matrica nije donje trougaona.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 200
Greska: neispravan unos.
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 4
Unesite elemente matrice:
2 0 0 0
7 80 0 0
-9 4 4 0
14 23 -8 1
Matrica jeste donje trougaona.
```

[Rešenje 3.7.10]

**Zadatak 3.7.11** Napisati program koji za učitane celobrojnu kvadratnu matricu ispisuje redni broj kolone koja ima najveći zbir elemenata. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
3
Unesite elemente matrice:
1 2 3
7 3 4
5 3 1
Indeks kolone je: 0
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
4
Unesite elemente matrice:
7 8 9 10
7 6 11 4
3 1 2 -2
8 3 9 9
Indeks kolone je: 2
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
104
Greska: neispravan unos.
```

[Rešenje 3.7.11]

**Zadatak 3.7.12** Napisati program koji za učitane 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  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
3
Unesite elemente matrice:
2 3.2 4
7 8.8 1
2.3 1 1
Razlika je: 2.20
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
5
Unesite elemente matrice:
2.3 1 12 8 -20
4 -8.2 7 14.5 19
1 -2.5 9 11 33
3 4.3 -5.7 2 8
9 56 1.08 7 5.5 19.01
Razlika je:-30.38
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
52
Greska: neispravan unos.
```

[Rešenje 3.7.12]

**Zadatak 3.7.13** Napisati program koji za učitane celobrojnu matricu dimenzije  $m \times n$  i uneta dva broja  $p$  i  $k$  ( $p \leq m$ ,  $k \leq n$ ) ispisuje sume svih podmatrica dimenzije  $p \times k$  unete matrice. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: 3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Unesite dva cela broja: 3 3
Sume podmatrica su: 54 63
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: 3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Unesite dva cela broja: 2 3
Sume podmatrica su: 24 30 48 54
```

### Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: 5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Unesite dva cela broja: 2 2
Sume podmatrica su: 7 5 20 19 18 23 4 8

```

### Primer 4

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: -3 200
Greska: neispravan unos.

```

[Rešenje 3.7.13]

**Zadatak 3.7.14** Napisati program koji za učitano celobrojnu kvadratnu matricu ispituje da li su njeni elementi po kolonama, vrstama i dijagonalama (glavnoj i sporednoj) sortirani strogo rastuće. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 3
Unesi elemente matrice:
1 2 3
4 5 6
7 8 9
Elementi su sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi su sortirani po dijagonalama.

```

### Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 2
Unesi elemente matrice:
6 9
4 10
Elementi nisu sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi nisu sortirani po dijagonalama.

```

### Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 4
Unesi elemente matrice:
5 5 7 9
6 10 11 13
8 12 14 15
13 15 16 20
Elementi su sortirani po kolonama.
Elementi nisu sortirani po vrstama.
Elementi su sortirani po dijagonalama.

```

### Primer 4

```

INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 1
Unesi elemente matrice:
5
Elementi su sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi su sortirani po dijagonalama.

```

[Rešenje 3.7.14]

### 3 Predstavljanje podataka

**Zadatak 3.7.15** Napisati program koji za učitane celobrojnu kvadratnu matricu ispituje da li su zbirovi elemenata njenih kolona uređeni u strogo rastućem poretku. Pretpostaviti da je maksimalna dimenzija matrice  $10 \times 10$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 4
Unesite elemente matrice:
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
Sume nisu uredjenje strogo rastuce.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 3
Unesite elemente matrice:
1 2 3
4 5 6
7 8 9
Sume jesu uredjenje strogo rastuce.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 3
Unesite elemente matrice:
2 -2 1
1 2 2
2 1 -2
Sume nisu uredjenje strogo rastuce.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 5
Unesite elemente matrice:
-1 0 3 0 20
0 0 0 10 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
Sume jesu uredjenje strogo rastuce.
```

[Rešenje 3.7.15]

**Zadatak 3.7.16** Matrica je *ortonormirana* ako je vrednost skalarnog proizvoda svakog para različitih vrsta jednak nuli, a vrednost skalarnog proizvoda vrste sa samom sobom jednak jedinici. Napisati program koji za unetu celobrojnu kvadratnu matricu proverava da li je ortonormirana. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Skalarni proizvod vektora*  $a = (a_1, a_2, \dots, a_n)$  i  $b = (b_1, b_2, \dots, b_n)$  je  $a_1 \cdot b_1 + a_2 \cdot b_2 + \dots + a_n \cdot b_n$ .

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 4
Unesite elemente matrice:
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
Matrica jeste ortonormirana.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 3
Unesite elemente matrice:
1 2 3
4 5 6
7 8 9
Matrica nije ortonormirana.
```

## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 3
Unesite elemente matrice:
2 -2 1
1 2 2
2 1 -2
Matrica nije ortonormirana.

```

## Primer 4

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 5
Unesite elemente matrice:
-1 0 0 0 0
0 0 0 1 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
Matrica jeste ortonormirana.

```

[Rešenje 3.7.16]

**Zadatak 3.7.17** Kvadratna matrica je *magični kvadrat* ako je suma elemenata u svakoj koloni i svakoj vrsti jednaka. Napisati program koji proverava da li je data celobrojna kvadratna matrica magični kvadrat i ispisuje odgovarajuću poruku na standardni izlaz. Pretpostaviti da je maksimalna dimenzija matrice je  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 4
Unesite elemente matrice:
1 5 3 1
2 1 2 5
3 2 2 3
4 2 3 1
Matrica jeste magicni kvadrat.

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 3
Unesite elemente matrice:
1 2 3
4 5 6
-1 3 3
Matrica nije magicni kvadrat.

```

[Rešenje 3.7.17]

\* **Zadatak 3.7.18** Napisati program koji učitava celobrojnu kvadratnu matricu i ispisuje elemente matrice u grupama koje su paralelne sa njenom sporednom dijagonalom, počevši od gornjeg levog ugla. Pretpostaviti da je maksimalna dimenzija matrice  $100 \times 100$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### 3 Predstavljanje podataka

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
3
Unesite elemente matrice:
1 2 3
4 5 6
7 8 9
Ispis je:
1
2 4
3 5 7
6 8
9
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
5
Unesite elemente matrice:
7 -8 1 2 3
90 11 0 5 4
12 -9 14 23 8
80 6 88 17 62
-22 10 44 57 -200
Ispis je:
7
-8 90
1 11 12
2 0 -9 80
3 5 14 6 -22
4 23 88 10
8 17 44
62 57
-200
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
-5
Greska: neispravan unos.
```

[Rešenje 3.7.18]

\* **Zadatak 3.7.19** Napisati funkciju `void mnozenje(int a[][MAKS], int m, int n, int b[][MAKS], int k, int t, int c[][MAKS])` koja računa matricu  $c$  kao proizvod matrica  $a$  i  $b$ . Dimenzija matrice  $a$  je  $n \times m$ , a dimenzija matrice  $b$  je  $k \times t$ . Napisati program koji ispisuje proizvod učitanih matrica. Pretpostaviti da je maksimalna dimenzija matrica  $50 \times 50$ . Ukoliko množenje matrica nije moguće ili je došlo do greške prilikom unosa podataka ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice A: 3 4
Unesite elemente matrice A:
1 2 8 9
-4 5 2 3
7 6 4 10
Unesite dimenzije matrice B: 4 2
Unesite elemente matrice B:
11 5
6 7
8 9
0 -3
Rezultat mnozenja je:
87 64
2 24
145 83
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice A: 5 2
Unesite elemente matrice A:
1 7
9 0
-10 2
92 3
14 -8
Unesite dimenzije matrice B: 2 4
Unesite elemente matrice B:
7 8 9 10
-11 2 34 78
Rezultat mnozenja je:
-70 22 247 556
63 72 81 90
-92 -76 -22 56
611 742 930 1154
186 96 -146 -484
```



*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice A: 3 4
Unesite elemente matrice A:
1 2 8 9
-4 5 2 3
7 6 4 10
Unesite dimenzije matrice B: 5 2
Množenje matrica nije moguće.

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice A: -3 4
Greska: neispravan unos.

```

[Rešenje 3.7.19]

\* **Zadatak 3.7.20** Element matrice naziva se *sedlo* ako je istovremeno najmanji u svojoj vrsti, a najveći u svojoj koloni. Napisati program koji ispisuje indekse i vrednosti onih elemenata matrice realnih brojeva koji su sedlo. Maksimalna dimenzija matrice je  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
2 3
Unesite elemente matrice:
1 2 3
0 5 6
Sedlo: 0 0 1

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 3
Unesite elemente matrice:
10 3 20
15 5 100
30 -1 200
Sedlo: 1 1 5

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice:
3 -3
Greska: neispravan unos.

```

[Rešenje 3.7.20]

\* **Zadatak 3.7.21** Napisati program koji ispisuje elemente matrice celih brojeva u spiralmnom redosledu počevši od gornjeg levog ugla krećući se u smeru suprotnom od smera kazaljke na satu. Maksimalna dimenzija matrice je  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

### 3 Predstavljanje podataka

---

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice:
3 3
Unesite elemente matrice:
1 2 3
4 5 6
7 8 9
Ispis je:
1 2 3 6 9 8 7 4 5
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju matrice: 5 7
Unesite elemente matrice:
7 -8 1 2 3 -54 87
90 11 0 5 4 9 18
12 -9 14 23 8 -22 74
80 6 88 17 62 38 41
-22 10 44 57 -200 39 55
Ispis je:
7 -8 1 2 3 -54 87 18 74 41 55
39 -200 57 44 10 -22 80 12 90
11 0 5 4 9 -22 38 62 17 88 6
-9 14 23 8
```

[Rešenje 3.7.21]

\* **Zadatak 3.7.22** Matrica  $a$  se sadrži u matrici  $b$  ukoliko postoji podmatrica matrice  $b$  identična matrici  $a$ . Napisati program koji za dve učitane matrice celih brojeva proverava da li se druga matrica sadrži u prvoj učitanoj matrici. Maksimalna dimenzija obe matrice je  $50 \times 50$ . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: 3 4
Unesite elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Unesite dimenzije matrice: 2 2
Unesite elemente matrice:
2 3
4 10
Druga matrica je sadržana u prvoj matrici.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: 3 4
Unesite elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Unesite dimenzije matrice: 2 2
Unesite elemente matrice:
2 8
6 4
Druga matrica nije sadržana
u prvoj matrici.
```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: 5 5
Unesite elemente matrice:
7 -8 1 2 3
90 11 0 5 4
12 -9 14 23 8
80 6 88 17 62
-22 10 44 57 -200
Unesite dimenzije matrice: 3 4
Unesite elemente matrice:
90 11 0 5
12 -9 14 23
80 6 88 17
Druga matrica je sadržana u prvoj matrici.

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije matrice: 5 5
Unesite elemente matrice:
7 -8 1 2 3
90 11 0 5 4
12 -9 14 23 8
80 6 88 17 62
-22 10 44 57 -200
Unesite dimenzije matrice: 53 4
Greska: neispravan unos.

```

[Rešenje 3.7.22]

## 3.8 Rešenja

## Rešenje 3.7.1

```

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

#define MAKS 50

int main()
{
    /* Deklaracije potrebnih promenljivih. */
    int a[MAKS][MAKS];
    int m, n;
    int i, j;

    /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
    printf("Unesite dimenzije matrice: ");
    scanf("%d%d", &m, &n);
    if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
    {
        printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
    }

    /* Ucitavanje elemenata matrice. */
    printf("Unesite elemente matrice:\n");
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)

```

### 3 Predstavljanje podataka

```
26     scanf("%d", &a[i][j]);

28     /* Ispis elemenata matrice. */
    for (i = 0; i < m; i++)
    {
30         for (j = 0; j < n; j++)
32             printf("%d ", a[i][j]);
        printf("\n");
34     }

36     return 0;
}
```

#### Rešenje 3.7.2

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

4 #define MAKS 50

6 int main()
8 {
    /* Deklaracije potrebnih promenljivih. */
10     int a[MAKS][MAKS];
    int m, n;
12     int suma = 0;
    int i, j;

14     /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
16     printf("Unesite dimenzije matrice: ");
    scanf("%d%d", &m, &n);
18     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
        printf("Greska: neispravan unos.\n");
20         exit(EXIT_FAILURE);
    }

22     /* Ucitavanje elemenata matrice. */
24     printf("Unesite elemente matrice:\n");
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
26             scanf("%d", &a[i][j]);

28     /* Racunanje sume kvadrata svih elemenata. */
30     for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
32             suma += a[i][j] * a[i][j];

34     /* Ispis rezultata. */
    printf("Euklidska norma je %.3lf.\n", sqrt(suma));
36 }
```

```
    return 0;
38 }
```

### Rešenje 3.7.3

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

4 #define MAKS 50

6 /* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
8 {
    int i, j;

10    printf("Unesite elemente matrice:\n");
12    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
14        scanf("%d", &a[i][j]);
16 }

18 /* Funkcija ispisuje elemente matrice dimenzije m*n. */
void ispisi(int a[][MAKS], int m, int n)
{
20    int i, j;

22    for (i = 0; i < m; i++)
    {
24        for (j = 0; j < n; j++)
            printf("%d ", a[i][j]);
26        printf("\n");
    }
28 }

30 int main()
{
32    /* Deklaracije potrebnih promenljivih. */
    int a[MAKS][MAKS];
34    int m, n;

36    /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
    printf("Unesite dimenzije matrice: ");
38    scanf("%d%d", &m, &n);
    if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
40    {
        printf("Greska: neispravan unos.\n");
42        exit(EXIT_FAILURE);
    }

44    /* Ucitavanje elemenata matrice. */
46    ucitaj(a, m, n);
```

```
48  /* Ispis učitane matrice. */
    ispisi(a, m, n);
50
    return 0;
52 }
```

#### Rešenje 3.7.4

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

4  #define MAKS 50

6  /* Funkcija učitava elemente matrice dimenzije m*n. */
void učitaj(int a[][MAKS], int m, int n)
8  {
    int i, j;

10     printf("Unesite elemente matrice:\n");
    for (i = 0; i < m; i++)
12         for (j = 0; j < n; j++)
            scanf("%d", &a[i][j]);
14 }

16 /* Funkcija ispisuje elemente matrice dimenzije m*n. */
18 void ispisi(int a[][MAKS], int m, int n)
{
20     int i, j;

    for (i = 0; i < m; i++) {
22         for (j = 0; j < n; j++)
            printf("%d ", a[i][j]);
24         printf("\n");
    }
26 }

28 /* Funkcija formira maticu t transponovanjem matrice a. */
30 void transponovana(int a[][MAKS], int m, int n, int t[][MAKS])
{
32     int i, j;

    for (i = 0; i < m; i++)
34         for (j = 0; j < n; j++)
            t[j][i] = a[i][j];
36 }

38 int main()
40 {
    /* Deklaracije potrebnih promenljivih. */
42     int a[MAKS][MAKS], t[MAKS][MAKS];
```

```

44     int m, n;

46     /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
46     printf("Unesite dimenzije matrice: ");
46     scanf("%d%d", &m, &n);
48     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
48     {
50         printf("Greska: neispravan unos.\n");
50         exit(EXIT_FAILURE);
52     }

54     /* Ucitavanje elemenata matrice. */
54     ucitaj(a, m, n);

56     /* Formiranje transponovane matrice. */
58     transponovana(a, m, n, t);

60     /* Ispis rezultata. */
60     ispisi(t, n, m);

62     return 0;
64 }

```

### Rešenje 3.7.5

```

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

4  #define MAKS 50

6  /* Funkcija ucitava elemente matrice dimenzije m*n. */
6  void ucitaj(int a[][MAKS], int m, int n)
8  {
10     int i, j;

10     printf("Unesite elemente matrice:\n");
12     for (i = 0; i < m; i++)
12         for (j = 0; j < n; j++)
14             scanf("%d", &a[i][j]);
14 }

16 /* Funkcija ispisuje elemente matrice dimenzije m*n. */
18 void ispisi(int a[][MAKS], int m, int n)
18 {
20     int i, j;

22     for (i = 0; i < m; i++)
22     {
24         for (j = 0; j < n; j++)
24             printf("%d ", a[i][j]);
26         printf("\n");
26     }

```

```
    }
28 }

30 /* Funkcija razmenjuje elemente k-te i t-te vrste. */
void razmeni(int a[][MAKS], int m, int n, int k, int t)
32 {
    int j, pom;

34     for (j = 0; j < n; j++)
    {
36         pom = a[k][j];
38         a[k][j] = a[t][j];
        a[t][j] = pom;
40     }
    }

42
44 int main()
{
    /* Deklaracije potrebnih promenljivih. */
46     int a[MAKS][MAKS];
48     int m, n;
    int k, t;

50     /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
    printf("Unesite dimenzije matrice: ");
52     scanf("%d%d", &m, &n);
    if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
54     {
        printf("Greska: neispravan unos.\n");
56         exit(EXIT_FAILURE);
    }

58
    /* Ucitavanje elemenata matrice. */
60     ucitaj(a, m, n);

62     /* Ucitavanje indeksa vrsta i provera ispravnosti ulaza. */
    printf("Unesite indekse vrsta: ");
64     scanf("%d%d", &k, &t);
    if (k < 0 || k >= m || t < 0 || t >= m)
66     {
        printf("Greska: neispravan unos.\n");
68         exit(EXIT_FAILURE);
    }

70
    /* Razmena k-te i t-te vrste. */
72     razmeni(a, m, n, k, t);

74     /* Ispis rezultata. */
    ispisi(a, m, n);
76
    return 0;
78 }
```



## Rešenje 3.7.6

```

1  #include <stdio.h>
2  #include <stdlib.h>

4  #define MAKS 50

6  /* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
8  {
    int i, j;

10     printf("Unesite elemente matrice:\n");
12     for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
14         scanf("%d", &a[i][j]);
    }

16 int main()
18 {
    /* Deklaracije potrebnih promenljivih. */
20     int a[MAKS][MAKS];
    int m, n, i, j, suma_suseda;
22     int k, t;

24     /* Ucitavanje dimenzija matrice i proveru ispravnosti ulaza. */
    printf("Unesite dimenzije matrice: ");
26     scanf("%d%d", &m, &n);
    if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
28     {
        printf("Greska: neispravan unos.\n");
30         exit(EXIT_FAILURE);
    }

32     /* Ucitavanje elemenata matrice. */
34     ucitaj(a, m, n);

36     /* Izracunavanje i ispis rezultata. */
    printf("Indeksi elemenata koji su jednaki zbiru suseda su:\n");
38     for (i = 0; i < m; i++)
    {
40         for (j = 0; j < n; j++)
        {
42             suma_suseda = 0;

44             /* Vrsi se racunanje sume elemenata podmatrice velicine 3*3
               ciji je centralni element a[i][j]. Pri racunanju ove sume
46             vodi se racuna da se ne izadje iz okvira matrice a. */
            for (k = i - 1; k <= i + 1; k++)
48                 for (t = j - 1; t <= j + 1; t++)
                    if (k >= 0 && k < m && t >= 0 && t < n)
50                     suma_suseda += a[k][t];
        }
    }

```

### 3 Predstavljanje podataka

```
52      /* Od ukupne sume se oduzima tekuci element kako bi se dobio
53         zbir elemenata koji su njegovi susedi. */
54      suma_suseda -= a[i][j];

56      /* Ukoliko je suma suseda jednaka tekucem elementu, ispisuju
57         se indeksi tekuceg elementa matrice. */
58      if (suma_suseda == a[i][j])
59          printf("%d %d\n", i, j);
60  }
61  }
62  return 0;
63 }
```

#### Rešenje 3.7.7

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(int a[][MAKS], int m, int n)
{
    int i, j;

    printf("Unesite elemente matrice:\n");
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
            scanf("%d", &a[i][j]);
}

/* Funkcija formira niz b tako sto element b[i] ima vrednost
   prosečne vrednosti i-te vrste matrice. */
void kreiraj_niz(int a[][MAKS], int m, int n, double b[])
{
    int i, j, suma;

    for (i = 0; i < m; i++)
    {
        suma = 0;
        for (j = 0; j < n; j++)
            suma += a[i][j];

        b[i] = (double) suma / n;
    }
}

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

```

36  int a[MAKS][MAKS];
    double b[MAKS];
38  int m, n, i;

40  /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */
    printf("Unesite dimenzije matrice: ");
42  scanf("%d%d", &m, &n);
    if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
44  {
        printf("Greska: neispravan unos.\n");
46  exit(EXIT_FAILURE);
    }

48  /* Ucitavanje elemenata matrice. */
50  ucitaj(a, m, n);

52  /* Formira se niz b. */
    kreiraj_niz(a, m, n, b);
54
56  /* Ispis rezultata. */
    printf("Dobijeni niz je:\n");
    for (i = 0; i < m; i++)
58  {
        printf("%g ", b[i]);
        printf("\n");
60  }

62  return 0;
}

```

### Rešenje 3.7.8

```

1  #include <stdio.h>
2  #include <stdlib.h>

4  #define MAKS 50

6  /* Funkcija ucitava elemente matrice dimenzije n*n. */
    void ucitaj(int a[][MAKS], int n)
8  {
    int i, j;

10  printf("Unesite elemente matrice:\n");
    for (i = 0; i < n; i++)
12  {
        for (j = 0; j < n; j++)
14  {
            scanf("%d", &a[i][j]);
        }
    }

16  /* Relacija je refleksivna ukoliko je za svako i, a[i][i] = 1.
    Funkcija proverava da li je relacija zadata matricom a
    refleksivna i vraca 1 ukoliko jeste, a 0 inace. */
20  int refleksivna(int a[][MAKS], int n)
    {

```

```
22  int i;
24  for (i = 0; i < n; i++)
    if (a[i][i] != 1)
26      return 0;
28  return 1;
29 }
30
31 /* Relacija je simetricna ukoliko za svaki par i, j vazi da je
32    a[i][j] = a[j][i]. Funkcija proverava da li je relacija zadata
    matricom a simetricna i vraca 1 ukoliko jeste, a 0 inace. */
34 int simetricna(int a[][MAKS], int n)
35 {
36     int i, j;
38     for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
40             if (a[i][j] != a[j][i])
                return 0;
42     return 1;
43 }
44
45 /* Relacija je tranzitivna ukoliko za svaku trojku i, j, k vazi da
46    ako je a[i][j] = 1 i a[j][k] = 1, onda je i a[i][k] = 1.
47    Funkcija proverava da li je relacija zadata matricom a
48    tranzitivna i vraca 1 ukoliko jeste, a 0 inace. */
50 int tranzitivna(int a[][MAKS], int n)
51 {
52     int i, j, k;
54     for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
56             for (k = 0; k < n; k++)
                if (a[i][j] == 1 && a[j][k] == 1 && a[i][k] == 0)
58                 return 0;
60     return 1;
61 }
62
63 /* Relacija je relacija ekvivalencije ukoliko je refleksivna,
64    tranzitivna i simetricna. Funkcija proverava da li je relacija
    zadata matricom a relacija ekvivalencije i vraca 1 ukoliko
66    jeste, a 0 inace. */
68 int ekvivalencija(int a[][MAKS], int n)
69 {
70     if (refleksivna(a, n) && simetricna(a, n) && tranzitivna(a, n))
71         return 1;
72     return 0;
73 }
```

```

74 int main()
75 {
76     /* Deklaracije potrebnih promenljivih. */
77     int a[MAKS][MAKS];
78     int n;
79
80     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
81     printf("Unesite dimenziju matrice: ");
82     scanf("%d", &n);
83     if (n <= 0 || n > MAKS)
84     {
85         printf("Greska: neispravan unos.\n");
86         exit(EXIT_FAILURE);
87     }
88
89     /* Ucitavanje elemenata matrice. */
90     ucitaj(a, n);
91
92     /* Racunanje i ispis rezultata. */
93     if (refleksivna(a, n))
94         printf("Relacija jeste refleksivna.\n");
95     else
96         printf("Relacija nije refleksivna.\n");
97
98     if (simetricna(a, n))
99         printf("Relacija jeste simetricna.\n");
100    else
101        printf("Relacija nije simatrica.\n");
102
103    if (tranzitivna(a, n))
104        printf("Relacija jeste tranzitivna.\n");
105    else
106        printf("Relacija nije tranzitivna.\n");
107
108    if (ekvivalencija(a, n))
109        printf("Relacija jeste ekvivalencija.\n");
110    else
111        printf("Relacija nije ekvivalencija.\n");
112
113    return 0;
114 }

```

### Rešenje 3.7.9

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije n*n. */

```

```
void ucitaj(float a[][MAKS], int n)
8 {
    int i, j;

10     printf("Unesite elemente matrice:\n");
12     for (i = 0; i < n; i++)
        for (j = 0; j < n; j++)
14         scanf("%f", &a[i][j]);
}

16 /* Funkcija racuna trag matrice. */
18 float trag(float a[][MAKS], int n)
{
20     float suma = 0;
    int i;

22     for (i = 0; i < n; i++)
24         suma += a[i][i];

26     return suma;
}

28 /* Funkcija racuna sumu elemenata koji se nalaze na sporednoj
30    dijagonali matrice. */
32 float suma_sporedna(float a[][MAKS], int n)
{
    float suma = 0;
    int i;

34     for (i = 0; i < n; i++)
36         suma += a[i][n - i - 1];

38     return suma;
40 }

42 /* Funkcija racuna sumu elemenata koji se nalaze iznad glavne
    dijagonale matrice. */
44 float suma_iznad(float a[][MAKS], int n)
{
46     float suma = 0;
    int i, j;

48     for (i = 0; i < n; i++)
50         for (j = i + 1; j < n; j++)
            suma += a[i][j];

52     return suma;
54 }

56 /* Funkcija racuna sumu elemenara koji se nalaze ispod sporedne
    dijagonale matrice. */
58 float suma_ispod(float a[][MAKS], int n)
```

```

60     float suma = 0;
61     int i, j;
62
63     for (i = 0; i < n; i++)
64         for (j = n - i - 1; j > i; j--)
65             suma += a[i][j];
66
67     return suma;
68 }
69
70 int main()
71 {
72     /* Deklaracije potrebnih promenljivih. */
73     float a[MAKS][MAKS];
74     int n;
75
76     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
77     printf("Unesite dimenziju matrice: ");
78     scanf("%d", &n);
79     if (n <= 0 || n > MAKS)
80     {
81         printf("Greska: neispravan unos.\n");
82         exit(EXIT_FAILURE);
83     }
84
85     /* Ucitavanje elemenata matrice. */
86     ucitaj(a, n);
87
88     /* Ispis rezultata. */
89     printf("Trag je %.2f.\n", trag(a, n));
90     printf("Suma na sporednoj dijagonali je %.2f.\n",
91           suma_sporedna(a, n));
92     printf("Suma iznad glavne dijagonale je %.2f.\n",
93           suma_iznad(a, n));
94     printf("Suma ispod sporedne dijagonale je %.2f.\n",
95           suma_ispod(a, n));
96
97     return 0;
98 }

```

### Rešenje 3.7.10

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS 50
5
6  /* Funkcija ucitava elemente matrice dimenzije n*n. */
7  void ucitaj(int a[][MAKS], int n)
8  {

```

```
10     int i, j;

12     printf("Unesite elemente matrice:\n");
13     for (i = 0; i < n; i++)
14         for (j = 0; j < n; j++)
15             scanf("%d", &a[i][j]);
16 }

17 /* Funkcija proverava da li je matrica donje trougaona i vraca
18    jedinicu ukoliko jeste, a nulu inace. */
19 int donje_trougaona(int a[][MAKS], int n)
20 {
21     int i, j;

22     /* Prolazi se kroz sve elemente iznad glavne dijagonale i ukoliko
23        se naidje na element koji je razlicit od nule, onda matrica
24        nije donje trougaona. */
25     for (i = 0; i < n; i++)
26         for (j = i + 1; j < n; j++)
27             if (a[i][j] != 0)
28                 return 0;

29     /* Ukoliko su svi elementi iznad glavne dijagonale nule, matrica
30        jeste donje trougaona. */
31     return 1;
32 }

33 int main()
34 {
35     /* Deklaracije potrebnih promenljivih. */
36     int a[MAKS][MAKS];
37     int n;

38     /* Ucitavanje dimenzije matrice i proveru ispravnosti ulaza. */
39     printf("Unesite dimenziju matrice: ");
40     scanf("%d", &n);
41     if (n <= 0 || n > MAKS)
42     {
43         printf("Greska: neispravan unos.\n");
44         exit(EXIT_FAILURE);
45     }

46     /* Ucitavanje elemenata matrice. */
47     ucitaj(a, n);

48     /* Ispis rezultata. */
49     if (donje_trougaona(a, n))
50         printf("Matrica jeste donje trougaona.\n");
51     else
52         printf("Matrica nije donje trougaona.\n");

53     return 0;
54 }
```



```
}

```

### Rešenje 3.7.11

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS 50
5
6  /* Funkcija ucitava elemente matrice dimenzije n*n. */
7  void ucitaj(int a[][MAKS], int n)
8  {
9      int i, j;
10
11      printf("Unesite elemente matrice:\n");
12      for (i = 0; i < n; i++)
13          for (j = 0; j < n; j++)
14              scanf("%d", &a[i][j]);
15  }
16
17  int main()
18  {
19      /* Deklaracije potrebnih promenljivih. */
20      int a[MAKS][MAKS];
21      int n, i, j;
22      int maksimalni_zbir, trenutni_zbir = 0, indeks_kolone;
23
24      /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
25      printf("Unesite dimenziju matrice: ");
26      scanf("%d", &n);
27      if (n <= 0 || n > MAKS)
28      {
29          printf("Greska: neispravan unos.\n");
30          exit(EXIT_FAILURE);
31      }
32
33      /* Ucitavanje elemenata matrice. */
34      ucitaj(a, n);
35
36      /* Maksimalni zbir se inicijalizuje na vrednost zbira prve
37         kolone. U ovom slucaju bi bilo pogresno da se maksimalni zbir
38         inicijalizuje na nulu jer moze da se desi da su svi elementi
39         matrice negativni. Drugi nacin da se ispravno inicijalizuje
40         maksimalni zbir jeste da mu se dodeli vrednost konstante
41         INT_MIN cija se definicija nalazi u zaglavlju limits.h. */
42      for (i = 0; i < n; i++)
43          trenutni_zbir += a[i][0];
44
45      maksimalni_zbir = trenutni_zbir;
46      indeks_kolone = 0;

```

### 3 Predstavljanje podataka

---

```
48  /* Racuna se zbir svake sledece kolone i azurira se vrednost
    maksimalnog zbira. */
50  for (j = 1; j < n; j++) {
    /* Racuna se zbir kolone j. */
52    trenutni_zbir = 0;
    for (i = 0; i < n; i++)
54      trenutni_zbir += a[i][j];

56    /* Ukoliko je taj zbir veci od trenutno maksimalnog zbira,
       azurira se vrednost maksimalnog zbira i pamti se tekuca
58    kolona. */
    if (trenutni_zbir > maksimalni_zbir) {
60      maksimalni_zbir = trenutni_zbir;
      indeks_kolone = j;
62    }
  }
64

  /* Ispis rezultata. */
66  printf("Indeks kolone je: %d\n", indeks_kolone);

68  return 0;
}
```

#### Rešenje 3.7.12

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

4  #define MAKS 50

6  /* Funkcija ucitava elemente matrice dimenzije n*n. */
void ucitaj(float a[][MAKS], int n)
8  {
    int i, j;

10    printf("Unesite elemente matrice:\n");
    for (i = 0; i < n; i++)
12      for (j = 0; j < n; j++)
14        scanf("%f", &a[i][j]);
  }

16
int main()
18 {
    /* Deklaracije potrebnih promenljivih. */
20    float a[MAKS][MAKS];
    int n, i, j;
22    float gornji_trougao = 0, donji_trougao = 0;

24    /* Ucitavanje dimenzije matrice i proveru ispravnosti ulaza. */
    printf("Unesite dimenziju matrice: ");
26    scanf("%d", &n);
```

```

28     if (n <= 0 || n > MAKS)
    {
        printf("Greska: neispravan unos.\n");
30         exit(EXIT_FAILURE);
    }

32     /* Ucitavanje elemenata matrice. */
34     ucitaj(a, n);

36     /* Racuna se suma gornjeg trougla. */
    for (i = 0; i < n / 2; i++)
38         for (j = i + 1; j < n - i - 1; j++)
            gornji_trougao += a[i][j];

40     /* Racuna se suma donjeg trougla. */
42     for (i = n / 2; i < n; i++)
        for (j = n - i; j < i; j++)
44            donji_trougao += a[i][j];

46     /* Ispis rezultata. */
    printf("Razlika je: %.2f\n", gornji_trougao - donji_trougao);
48
    return 0;
50 }

```

### Rešenje 3.7.13

```

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

3
   #define MAKS 50

5
   /* Funkcija ucitava elemente matrice dimenzije m*n. */
7  void ucitaj(int a[][MAKS], int m, int n)
   {
9      int i, j;

11     printf("Unesite elemente matrice:\n");
    for (i = 0; i < m; i++)
13         for (j = 0; j < n; j++)
            scanf("%d", &a[i][j]);
15 }

17 int main()
   {
19     /* Deklaracije potrebnih promenljivih. */
    int a[MAKS][MAKS];
21     int n, i, j, m, x, y, p, k;
    int suma;

23     /* Ucitavanje dimenzija matrice i provera ispravnosti ulaza. */

```

### 3 Predstavljanje podataka

```
25 printf("Unesite dimenzije matrice: ");
   scanf("%d%d", &m, &n);
27 if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
   {
29     printf("Greska: neispravan unos.\n");
       exit(EXIT_FAILURE);
31 }

33 /* Ucitavanje elemenata matrice. */
   ucitaj(a, m, n);
35
   /* Ucitavanje dimenzija p i k i proverava ispravnosti ulaza. */
37 printf("Unesite dva cela broja: ");
   scanf("%d%d", &p, &k);
39 if (p <= 0 || p > m || k <= 0 || k > n)
   {
41     printf("Greska: neispravan unos.\n");
       exit(EXIT_FAILURE);
43 }

45 /* Racunanje i ispis rezultata. */
   printf("Sume podmatrice su: ");
47 for (i = 0; i <= m - p; i++)
   {
49     for (j = 0; j <= n - k; j++)
       {
51         /* Za svaku poziciju (i,j), racuna se suma podmatrice
           dimenzije p*k, ciji je gornji levi ugao a[i][j]. */
53         suma = 0;
           for (x = 0; x < p; x++)
               for (y = 0; y < k; y++)
                   suma += a[i + x][j + y];
57
           printf("%d ", suma);
59     }
       }
61 printf("\n");

63 return 0;
}
```

#### Rešenje 3.7.14

```
1 #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 50
5
   /* Funkcija ucitava elemente matrice dimenzije n*n. */
7 void ucitaj(int a[][MAKS], int n)
   {
```

```
9   int i, j;

11  printf("Unesite elemente matrice:\n");
   for (i = 0; i < n; i++)
13     for (j = 0; j < n; j++)
       scanf("%d", &a[i][j]);
15 }

17 /* Funkcija proverava da li je kolona j sortirana rastuce i vraca
   jedinicu ukoliko jeste, a nulu inace. */
19 int sortirana_kolona(int a[][MAKS], int n, int j)
   {
21   int i;

23   for (i = 0; i < n - 1; i++)
       if (a[i][j] >= a[i + 1][j])
25     return 0;

27   return 1;
   }

29 /* Funkcija proverava da li je svaka kolona matrice sortirana
   rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
31 int sortirani_po_kolonama(int a[][MAKS], int n)
   {
33   int j;

35   for (j = 0; j < n; j++)
37     if (!sortirana_kolona(a, n, j))
       return 0;

39   return 1;
41 }

43 /* Funkcija proverava da li je i-ta vrsta sortirana rastuce i vraca
   jedinicu ukoliko jeste, a nulu inace. */
45 int sortirana_vrsta(int a[][MAKS], int n, int i)
   {
47   int j;

49   for (j = 0; j < n - 1; j++)
       if (a[i][j] >= a[i][j + 1])
51     return 0;

53   return 1;
   }

55 /* Funkcija proverava da li je svaka vrsta matrice sortirana
   rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
57 int sortirani_po_vrstama(int a[][MAKS], int n)
   {
59   int i;
```

### 3 Predstavljanje podataka

---

```
61     for (i = 0; i < n; i++)
63         if (!sortirana_vrsta(a, n, i))
64             return 0;
65
66     return 1;
67 }
68
69 /* Funkcija proverava da li je glavna dijagonala matrice sortirana
70    rastuce i vraća jedinicu ukoliko jeste, a nulu inace. */
71 int sortirana_glavna(int a[][MAKS], int n)
72 {
73     int i;
74
75     for (i = 0; i < n - 1; i++)
76         if (a[i][i] >= a[i + 1][i + 1])
77             return 0;
78
79     return 1;
80 }
81
82 /* Funkcija proverava da li je sporedna dijagonala matrice
83    sortirana rastuce i vraća jedinicu ukoliko jeste, a nulu inace. */
84 int sortirana_sporedna(int a[][MAKS], int n)
85 {
86     int i;
87
88     for (i = 0; i < n - 1; i++)
89         if (a[i][n - i - 1] >= a[i + 1][n - i - 2])
90             return 0;
91
92     return 1;
93 }
94
95 /* Funkcija proverava da li su obe dijagonale matrice sortirane
96    rastuce i vraća jedinicu ukoliko jesu, a nulu inace. */
97 int sortirani_po_dijagonalama(int a[][MAKS], int n)
98 {
99     return sortirana_glavna(a, n) && sortirana_sporedna(a, n);
100 }
101
102 int main()
103 {
104     /* Deklaracije potrebnih promenljivih. */
105     int a[MAKS][MAKS];
106     int n;
107
108     /* Ucitavanje dimenzije matrice i proveru ispravnosti ulaza. */
109     printf("Unesite dimenziju matrice: ");
110     scanf("%d", &n);
111     if (n <= 0 || n > MAKS)
112     {
```

```

113     printf("Greska: neispravan unos.\n");
114     exit(EXIT_FAILURE);
115 }

117 /* Ucitavanje elemenata matrice. */
118 ucitaj(a, n);

119 /* Ispis rezultata. */
121 if (sortirani_po_kolonama(a, n))
122     printf("Elementi su sortirani po kolonama.\n");
123 else
124     printf("Elementi nisu sortirani po kolonama.\n");

125 if (sortirani_po_vrstama(a, n))
126     printf("Elementi su sortirani po vrstama.\n");
127 else
128     printf("Elementi nisu sortirani po vrstama.\n");

129 if (sortirani_po_dijagonalama(a, n))
130     printf("Elementi su sortirani po dijagonalama.\n");
131 else
132     printf("Elementi nisu sortirani po dijagonalama.\n");

133 return 0;
134 }

```

### Rešenje 3.7.15

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS 10
5
6  /* Funkcija ucitava elemente matrice dimenzije n*n. */
7  void ucitaj(int a[][MAKS], int n)
8  {
9      int i, j;
10
11     printf("Unesite elemente matrice:\n");
12     for (i = 0; i < n; i++)
13         for (j = 0; j < n; j++)
14             scanf("%d", &a[i][j]);
15 }
16
17 /* Funkcija racuna sumu elemenata kolone j. */
18 int suma_kolone(int a[][MAKS], int n, int j)
19 {
20     int suma = 0, i;
21
22     for (i = 0; i < n; i++)
23         suma += a[i][j];

```

```
25     return suma;
26 }
27
28 /* Funkcija proverava da li su sume kolona uredjene rastuce i vraca
29    jedinicu ako jesu, a nulu inace. */
30 int uredjene_suma(int a[][MAKS], int n)
31 {
32     int prethodna_suma, trenutna_suma;
33     int j;
34
35     /* Prva suma se inicijalizuje na sumu prve kolone. */
36     prethodna_suma = suma_kolone(a, n, 0);
37
38     for (j = 1; j < n; j++)
39     {
40         /* Racuna se suma trenutne kolone. */
41         trenutna_suma = suma_kolone(a, n, j);
42
43         /* Ukoliko je ta suma manja ili jednaka prethodnoj, poredak
44            suma nije rastuci. */
45         if (trenutna_suma <= prethodna_suma)
46             return 0;
47
48         /* Suma trenutne kolone postaje suma prethodne kolone za
49            narednu iteraciju. */
50         prethodna_suma = trenutna_suma;
51     }
52
53     return 1;
54 }
55
56 int main()
57 {
58     /* Deklaracije potrebnih promenljivih. */
59     int a[MAKS][MAKS];
60     int n;
61
62     /* Ucitavanje dimenzije matrice i proveru ispravnosti ulaza. */
63     printf("Unesite dimenziju matrice: ");
64     scanf("%d", &n);
65     if (n <= 0 || n > MAKS)
66     {
67         printf("Greska: neispravan unos.\n");
68         exit(EXIT_FAILURE);
69     }
70
71     /* Ucitavanje elemenata matrice. */
72     ucitaj(a, n);
73
74     /* Ispis rezultata. */
75     if (uredjene_suma(a, n))
```



```

    printf("Sume jesu uredjenje strogo rastuce.\n");
77 else
    printf("Sume nisu uredjenje strogo rastuce.\n");
79
    return 0;
81 }

```

### Rešenje 3.7.16

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 200
5
   /* Funkcija ucitava elemente matrice dimenzije n*n. */
7  void ucitaj(int a[][MAKS], int n)
   {
9      int i, j;

11     printf("Unesite elemente matrice:\n");
    for (i = 0; i < n; i++)
13         for (j = 0; j < n; j++)
            scanf("%d", &a[i][j]);
15 }

17 /* Funkcija racuna skalarni proizvod i-te i j-te vrste matrice. */
   int skalarni_proizvod(int a[][MAKS], int n, int i, int j)
19 {
    int suma = 0, k;

21     for (k = 0; k < n; k++)
23         suma += a[i][k] * a[j][k];

25     return suma;
   }

27 /* Matrica je ortonormirana ukoliko je skalarni proizvod svakog
29    para razlicitih vrsta jednak nuli, a skalarni proizvod svake
    vrste same sa sobom jednak jedinici. Funkcija proverava da li
31    je matrica ortonormirana i vraca jedinicu ukoliko jeste, a nulu
    inace. */
33 int ortonormirana(int a[][MAKS], int n)
   {
35     int i, j;

37     /* Za svaki par vrsta se racuna skalarni proizvod i proverava da
        li je uslov ispunjen. Ukoliko nije, kao povratna vrednost
39     funkcije se vraca nula. */
    for (i = 0; i < n; i++)
41     {
        for (j = i; j < n; j++)

```

### 3 Predstavljanje podataka

```
43     {
44         /* Provera za slucaj kada se racuna skalarni proizvod vrste
45            same sa sobom. */
46         if (i == j && skalarni_proizvod(a, n, i, i) != 1)
47             return 0;
48
49         /* Provera za par razlicitih vrsta. */
50         if (i != j && skalarni_proizvod(a, n, i, j) != 0)
51             return 0;
52     }
53 }
54
55 /* Ako je izvorsavanje stiglo do kraja petlje, znaci da je uslov
56    ispunjen za sve vrste, tj. da je matrica ortonormirana. */
57 return 1;
58 }
59
60 int main()
61 {
62     /* Deklaracije potrebnih promenljivih. */
63     int a[MAKS][MAKS];
64     int n;
65
66     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
67     printf("Unesite dimenziju matrice: ");
68     scanf("%d", &n);
69     if (n <= 0 || n > MAKS)
70     {
71         printf("Greska: neispravan unos.\n");
72         exit(EXIT_FAILURE);
73     }
74
75     /* Ucitavanje elemenata matrice. */
76     ucitaj(a, n);
77
78     /* Ispis rezultata. */
79     if (ortonormirana(a, n))
80         printf("Matrica jeste ortonormirana.\n");
81     else
82         printf("Matrica nije ortonormirana.\n");
83
84     return 0;
85 }
```

#### Rešenje 3.7.17

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
```

```
/* Funkcija učitava elemente matrice dimenzije n*n. */
7 void ucitaj(int a[][MAKS], int n)
{
9     int i, j;

11     printf("Unesite elemente matrice:\n");
    for (i = 0; i < n; i++)
13         for (j = 0; j < n; j++)
            scanf("%d", &a[i][j]);
15 }

/* Funkcija racuna sumu kolone j. */
17 int suma_kolone(int a[][MAKS], int n, int j)
19 {
    int i, suma = 0;

21     for (i = 0; i < n; i++)
23         suma += a[i][j];

25     return suma;
}

/* Funkcija racuna sumu i-te vrste. */
27 int suma_vrste(int a[][MAKS], int n, int i)
29 {
    int j, suma = 0;

31     for (j = 0; j < n; j++)
33         suma += a[i][j];

35     return suma;
37 }

/* Funkcija proverava da li elementi matrice predstavljaju magicni
   kvadrat. */
39 int magicni_kvadrat(int a[][MAKS], int n)
41 {
43     /* Da bi matrica bila magicni kvadrat, sume svih vrsta i kolona
       treba da budu jednake. Suma se zato inicijalizuje na sumu prve
45     kolone. */
    int suma = suma_kolone(a, n, 0);
47     int i, j;

49     /* Proverava se da li su sume ostalih kolona jednake izracunatoj
       sumi. Ukoliko se naidje na kolonu koja ne zadovoljava ovaj
51     uslov, matrica nije magicni kvadrat. */
    for (j = 1; j < n; j++)
53         if (suma_kolone(a, n, j) != suma)
            return 0;

55     /* Proverava se i da li su sume svih vrsta jednake izracunatoj
       sumi. Ukoliko se naidje na vrstu koja ne zadovoljava ovaj
57     uslov, matrica nije magicni kvadrat. */
}
```

### 3 Predstavljanje podataka

```
        uslov, matrica nije magicni kvadrat. */
59     for (i = 0; i < n; i++)
        if (suma_vrste(a, n, i) != suma)
61         return 0;

63     /* Ako sve vrste i kolone imaju jednake sume, matrica je magicni
        kvadrat. */
65     return 1;
}

67
68 int main()
69 {
    /* Deklaracije potrebnih promenljivih. */
71     int a[MAKS][MAKS];
72     int n;
73
74     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
75     printf("Unesite dimenziju matrice: ");
76     scanf("%d", &n);
77     if (n <= 0 || n > MAKS)
78     {
79         printf("Greska: neispravan unos.\n");
80         exit(EXIT_FAILURE);
81     }

83     /* Ucitavanje elemenata matrice. */
84     ucitaj(a, n);
85
86     /* Ispis rezultata. */
87     if (magicni_kvadrat(a, n))
88         printf("Matrica jeste magicni kvadrat.\n");
89     else
90         printf("Matrica nije magicni kvadrat.\n");
91
92     return 0;
93 }
```

#### Rešenje 3.7.18

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

4 #define MAKS 100

6 /* Funkcija ucitava elemente matrice dimenzije n*n. */
7 void ucitaj(int a[][MAKS], int n)
8 {
9     int i, j;
10
11     printf("Unesite elemente matrice:\n");
12     for (i = 0; i < n; i++)
```

```

14     for (j = 0; j < n; j++)
        scanf("%d", &a[i][j]);
16 }
17
18 int main()
19 {
20     /* Deklaracije potrebnih promenljivih. */
21     int a[MAKS][MAKS];
22     int n;
23     int i, j, k;
24
25     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
26     printf("Unesite dimenziju matrice: ");
27     scanf("%d", &n);
28     if (n <= 0 || n > MAKS)
29     {
30         printf("Greska: neispravan unos.\n");
31         exit(EXIT_FAILURE);
32     }
33
34     /* Ucitavanje elemenata matrice. */
35     ucitaj(a, n);
36
37     /* Petlja kojom se ispisuju dijagonale iznad sporedne dijagonale,
38        ukljucujuci i sporednu dijagonalu.
39        Npr. za n=4, indeksi elemenata u matrici su:
40        (0,0) (0,1) (0,2) (0,3)
41        (1,0) (1,1) (1,2) (1,3)
42        (2,0) (2,1) (2,2) (2,3)
43        (3,0) (3,1) (3,2) (3,3)
44        Dakle, ispis elemenata ide u sledecem redosledu:
45        (0,0)
46        (0,1) (1,0)
47        (0,2) (1,1) (2,0)
48        (0,3) (1,2) (2,1) (3,0)
49        Za k-ti ispis vazi da indeksi vrsta imaju vrednosti od nula do
50        k, a indeksi kolona od k do nula. */
51     for (k = 0; k < n; k++)
52     {
53         /* Indeks kolone se inicijalizuje na k. */
54         j = k;
55         /* Indeks vrste se inicijalizuje na 0. */
56         i = 0;
57
58         /* Ispisuju se odgovarajuci elementi, indeks vrste se povecava,
59            a indeks kolone se smanjuje. */
60         while (j >= 0)
61         {
62             printf("%d ", a[i][j]);
63             i++;
64             j--;
65         }
66     }

```

### 3 Predstavljanje podataka

```
        printf("\n");
66    }

68    /* Petlja kojom se ispisuju dijagonale ispod sporedne dijagonale.
    Npr. za n=4, indeksi elemenata u matrici su:
70    (0,0) (0,1) (0,2) (0,3)
    (1,0) (1,1) (1,2) (1,3)
72    (2,0) (2,1) (2,2) (2,3)
    (3,0) (3,1) (3,2) (3,3)
74    Dakle, ispis elemenata ide u sledecem redosledu:
    (1,3) (2,2) (3,1)
76    (2,3) (3,2)
    (3,3)
78    Za k-ti ispis vazi da indeksi vrsta imaju vrednosti od k do
    n-1, a indeksi kolona od n-1 do 1. */
80    for (k = 1; k < n; k++)
    {
82        i = k;
        j = n - 1;
84
        while (i < n)
86        {
            printf("%d ", a[i][j]);
88            i++;
            j--;
90        }
        printf("\n");
92    }

94    return 0;
}
```

#### Rešenje 3.7.19

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS 50
5
   /* Funkcija ucitava elemente matrice dimenzije m*n. */
7  void ucitaj(int a[][MAKS], int m, int n)
   {
9      int i, j;

11     printf("Unesite elemente matrice:\n");
     for (i = 0; i < m; i++)
13         for (j = 0; j < n; j++)
             scanf("%d", &a[i][j]);
15 }

17 /* Funkcija ispisuje elemente matrice dimenzije m*n. */
```

```

19 void ispisi(int a[][MAKS], int m, int n)
{
21     int i, j;

23     for (i = 0; i < m; i++)
    {
25         for (j = 0; j < n; j++)
            printf("%d ", a[i][j]);
27         printf("\n");
    }
29 }

/* Funkcija vrši množenje matrica a i b i rezultat smesta u matricu
31 c. */
void mnozenje(int a[][MAKS], int m, int n,
33             int b[][MAKS], int k, int t, int c[][MAKS])
{
35     int i, j, w;

37     for (i = 0; i < m; i++)
    {
39         for (j = 0; j < t; j++)
            {
41             /* Element c[i][j] se dobija kao skalarni proizvod i-te vrste
                matrice a i j-te kolone matrice b. */
43             c[i][j] = 0;
                for (w = 0; w < k; w++)
45                 c[i][j] += a[i][w] * b[w][j];
            }
47         }
49     }

int main()
51 {
    /* Deklaracije potrebnih promenljivih. */
53     int a[MAKS][MAKS], b[MAKS][MAKS], c[MAKS][MAKS];
    int m, n;
55     int k, t;

57     /* Ucitavanje dimenzija prve matrice i provera ispravnosti
        ulaza. */
59     printf("Unesite dimenzije matrice A: ");
    scanf("%d%d", &m, &n);
61     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
    {
63         printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
65     }

67     /* Ucitavanje elemenata prve matrice. */
    ucitaj(a, m, n);
69

```

### 3 Predstavljanje podataka

---

```
71  /* Ucitavanje dimenzija druge matrice i provera ispravnosti
    ulaza. */
73  printf("Unesite dimenzije matrice B: ");
75  scanf("%d%d", &k, &t);
    if (k <= 0 || k > MAKS || t <= 0 || t > MAKS)
77  {
        printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
    }
79
    /* Provera da li se odgovarajuće dimenzije matrica poklapaju. */
81  if (n != k)
    {
83      printf("Množenje matrica nije moguće.\n");
        exit(EXIT_FAILURE);
85  }

87  /* Ucitavanje elemenata druge matrice. */
    ucitaj(b, k, t);
89
    /* Racunanje proizvoda. */
91  mnozenje(a, m, n, b, k, t, c);

93  /* Ispis rezultata. */
    printf("Rezultat množenja je:\n");
95  ispisi(c, m, t);

97  return 0;
}
```

#### Rešenje 3.7.20

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

4 #define MAKS 50

6 /* Funkcija ucitava elemente matrice dimenzije m*n. */
void ucitaj(double a[][MAKS], int m, int n)
8 {
    int i, j;
10
    printf("Unesite elemente matrice:\n");
12    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
14        scanf("%lf", &a[i][j]);
}

16 int main()
18 {
    /* Deklaracije potrebnih promenljivih. */
```



```
20 double a[MAKS][MAKS];
21 int m, n, k, i, j;
22
23 int indeks_kolone;
24 double maks_kolone, min_vrstte;
25
26 /* Ucitavanje dimenzija prve matrice i provera ispravnosti ulaza.
27 */
28 printf("Unesite dimenzije matrice: ");
29 scanf("%d%d", &m, &n);
30 if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
31 {
32     printf("Greska: neispravan unos.\n");
33     exit(EXIT_FAILURE);
34 }
35
36 /* Ucitavanje elemenata prve matrice. */
37 ucitaj(a, m, n);
38
39 /* Pronalazak elemenata koji su sedlo. */
40 for (i = 0; i < m; i++)
41 {
42     /* Pronalazi se najmanji element u tekucoj vrsti. Pamti se
43        kolona kojoj taj element pripada. */
44     min_vrstte = a[i][0];
45     indeks_kolone = 0;
46
47     for (j = 1; j < n; j++)
48     {
49         if (a[i][j] < min_vrstte)
50         {
51             min_vrstte = a[i][j];
52             indeks_kolone = j;
53         }
54     }
55
56     /* Pronalazi se najveći element u zapamcenoj koloni. */
57     maks_kolone = a[0][indeks_kolone];
58
59     for (k = 1; k < m; k++)
60         if (a[k][indeks_kolone] > maks_kolone)
61             maks_kolone = a[k][indeks_kolone];
62
63     /* Element je sedlo ukoliko je on istovremeno najmanji u svojoj
64        vrsti i najveći u svojoj koloni. */
65     if (min_vrstte == maks_kolone)
66         printf("Sedlo: %d %d %g\n", i, indeks_kolone, min_vrstte);
67 }
68
69 return 0;
70 }
```

#### Rešenje 3.7.21

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije m*n. */
7 void ucitaj(int a[][MAKS], int m, int n)
8 {
9     int i, j;
10
11     printf("Unesite elemente matrice:\n");
12     for (i = 0; i < m; i++)
13         for (j = 0; j < n; j++)
14             scanf("%d", &a[i][j]);
15 }
16
17 int main()
18 {
19     /* Deklaracije potrebnih promenljivih. */
20     int a[MAKS][MAKS];
21     int m, n, brojac, i, j, pravac;
22     int gornja_granica, donja_granica, leva_granica, desna_granica;
23
24     /* Ucitavanje dimenzija matrice i proveru ispravnosti ulaza. */
25     printf("Unesite dimenzije matrice: ");
26     scanf("%d%d", &m, &n);
27     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
28     {
29         printf("Greska: neispravan unos.\n");
30         exit(EXIT_FAILURE);
31     }
32
33     /* Ucitavanje elemenata matrice. */
34     ucitaj(a, m, n);
35
36     /* Ciklicni ispis elemenata matrice:
37     Npr. za n=4, indeksi elemenata u matrici su:
38     (0,0) (0,1) (0,2) (0,3)
39     (1,0) (1,1) (1,2) (1,3)
40     (2,0) (2,1) (2,2) (2,3)
41     (3,0) (3,1) (3,2) (3,3)
42     Ispis treba da ide sledecim redosledom:
43     1. krece se sa leva na desno (0,0) (0,1) (0,2) (0,3)
44     2. zatim se ide na dole (1,3) (2,3) (3,3)
45     3. zatim na levo (3,2) (3,1) (3,0)
46     4. zatim na gore (2,0) (1,0)
47     (ovde se staje jer je (0,0) vec ispisano) i prelazi se opet
48     na levo. Koraci 1-4 se ponavljaju dok god se ne ispisu svi
49     elementi. Ideja je da kada se ispisu elementi
50     prve vrste (kada se ide sa leva na desno), da se pomeri
```

```
51     "gornja granica ispisa" za 1, kako bi se naznacilo da je taj
    red vec ispisan. Slicno, kada se vrsi ispis odozgo na dole,
53     uspesno je ispisana jedna kolona pa je potrebno pomeriti
    "desnu granicu ispisa" za jedan u levo. Kada se ispise jedna
55     vrsta sa desna na levo, vrsi se pomeranje donje granice ispisa
    za jedan na gore. Slicno, kada se ispise jedna kolona odozdo
57     na gore, pomera se leva granica ispisa za jedan u desno. */
    gornja_granica = 0;
59     donja_granica = m - 1;
    leva_granica = 0;
61     desna_granica = n - 1;

63     /* Promenljiva pravac govori u kom smeru ispis ide. */
    pravac = 1;

65     /* Promenljive i i j su indeksi elementa koji se ispisuje. */
67     i = 0;
    j = 0;

69     for (brojac = 0; brojac < m * n; brojac++) {
71         printf("%d ", a[i][j]);

73         switch (pravac) {
            /* Ako je pravac = 1, trenutni smer ispisa je sa leva na
75             desno. */
            case 1:
77                 /* Ako je ispisan element na desnoj granici, onda se menja
                    pravac ispisa. */
79                 if (j == desna_granica)
                    {
81                     /* Prelazi se na pravac odozgo na dole. */
                        pravac = 2;
83                     /* Pomera se gornja granica za jedan na dole. */
                        gornja_granica++;
85                     /* Pomera se vrednost vrste za jedan na dole. */
                        i++;
87                 }
                else
89                 {
                    /* Ako jos uvek nije ispisan element na desnoj granici,
91                     vrsi se pomeranje na sledeci element u trenutnoj vrsti. */
                        j++;
93                 }
                break;

95                 /* Ako je pravac = 2, trenutni smer ispisa je odozgo na dole.
                    Slicno kao i u prethodnom slucaju, ako se dodje do donje
97                     granice, menja se pravac i pomera se desna granica za
                    jedno mesto u levo. A ako nije, samo se prelazi na narednu
99                     vrstu. */
            case 2:
101                if (i == donja_granica)
```

### 3 Predstavljanje podataka

---

```
103     {
104         pravac = 3;
105         desna_granica--;
106         j--;
107     }
108     else
109     {
110         i++;
111     }
112     break;
113
114     /* Ako je pravac = 3, trenutni smer ispisa je sa desna na
115        levo. Slicno kao i u prethodnom slucaju, ako se dodje do
116        leve granice, menja se pravac i pomera se donja granica za
117        jedno mesto na gore. A ako nije, samo se prelazi na
118        narednu kolonu. */
119     case 3:
120         if (j == leva_granica)
121         {
122             pravac = 4;
123             donja_granica--;
124             i--;
125         }
126         else
127         {
128             j--;
129         }
130         break;
131
132     /* Ako je pravac = 4, trenutni smer ispisa je odozdo na gore.
133        Slicno kao i u prethodnim slucajevima, ako se dodje do
134        gornje granice, menja se pravac i pomera se leva granica
135        za jedno mesto u desno. A ako nije, samo se prelazi na
136        narednu vrstu. */
137     case 4:
138         if (i == gornja_granica)
139         {
140             pravac = 1;
141             leva_granica++;
142             j++;
143         }
144         else
145         {
146             i--;
147         }
148     }
149 }
150
151 return 0;
152 }
```

## Rešenje 3.7.22

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAKS 50
5
6  /* Funkcija ucitava elemente matrice dimenzije m*n. */
7  void ucitaj(int a[][MAKS], int m, int n)
8  {
9      int i, j;
10
11      printf("Unesite elemente matrice:\n");
12      for (i = 0; i < m; i++)
13          for (j = 0; j < n; j++)
14              scanf("%d", &a[i][j]);
15  }
16
17  /* Funkcija proverava da li je matrica b podmatrica matrice a i
18     vraca jedinicu ukoliko jeste, a nulu inace. */
19  int podmatrica(int a[][MAKS], int m, int n,
20                int b[][MAKS], int k, int t)
21  {
22      int i, j, x, y;
23      int jeste_podmatrica;
24
25      for (i = 0; i <= m - k; i++)
26      {
27          for (j = 0; j <= n - t; j++)
28          {
29              /* Za svaku poziciju (i,j) se proverava da li je podmatrica
30                 dimenzije k*t ciji je gornji levi ugao a[i][j] jednaka
31                 matrici b. */
32              jeste_podmatrica = 1;
33              for (x = 0; x < k && jeste_podmatrica; x++)
34                  for (y = 0; y < t && jeste_podmatrica; y++)
35                      if (a[i + x][j + y] != b[x][y])
36                          jeste_podmatrica = 0;
37
38              if (jeste_podmatrica)
39                  return 1;
40          }
41      }
42
43      return 0;
44  }
45
46  int main()
47  {
48      /* Deklaracije potrebnih promenljivih. */
49      int a[MAKS][MAKS], b[MAKS][MAKS];
50      int m, n;
```

```
int k, t;

52
/* Ucitavanje dimenzija prve matrice i provera ispravnosti
54   ulaza. */
printf("Unesite dimenzije matrice A: ");
56 scanf("%d%d", &m, &n);
if (n <= 0 || n > MAKS || m <= 0 || m > MAKS)
58 {
    printf("Greska: neispravan unos.\n");
60    exit(EXIT_FAILURE);
}

62
/* Ucitavanje elemenata prve matrice. */
64 ucitaj(a, m, n);

66
/* Ucitavanje dimenzija druge matrice i provera ispravnosti
   ulaza. */
68 printf("Unesite dimenzije matrice B: ");
scanf("%d%d", &k, &t);
70 if (k <= 0 || k > MAKS || t <= 0 || t > MAKS)
{
72     printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
74 }

76
/* Ucitavanje elemenata druge matrice. */
ucitaj(b, k, t);

78
/* Ispis rezultata. */
80 if (podmatrica(a, m, n, b, k, t))
    printf("Druga matrica je sadrzana u prvoj matrici.\n");
82 else
    printf("Druga matrica nije sadrzana u prvoj matrici.\n");
84
86 return 0;
}
```

## 3.9 Strukture

**Zadatak 3.9.1** Definirati strukturu kojom se opisuje kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja. Napisati program koji za učitana dva kompleksna broja ispisuje vrednost zbira, razlike, proizvoda i količnika. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite realni i imaginarni deo prvog broja:  1 2
Unesite realni i imaginarni deo drugog broja: -2 3
Zbir: -1.00+5.00*i
Razlika: 3.00-1.00*i
Proizvod: -8.00-1.00*i
Kolicnik: 0.31-0.54*i

```

[Rešenje 3.9.1]

**Zadatak 3.9.2** Definirati strukturu kojom se opisuje razlomak. Napisati funkcije koje izračunavaju zbir i proizvod dva razlomka. Napisati program koji za uneti ceo broj  $n$  i unetih  $n$  razlomaka ispisuje njihov ukupan zbir i proizvod. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj razlomaka:  5
Unesite razlomke:
1 2
7 8
3 4
5 6
2 9
Suma svih razlomaka je 229/72.
Proizvod svih razlomaka je 35/576.

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj razlomaka:  10
Unesite razlomke:
4 3
12 25
3 8
1 3
8 9
2 3
5 6
-24 50
7 18
-7 19
Suma svih razlomaka je 6089/1368.
Proizvod svih razlomaka je 1568/577125.

```

[Rešenje 3.9.2]

**Zadatak 3.9.3** Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vocka* sadrži ime voćke i količinu vitamina C u miligramima (realan broj). Napisati funkcije:

- `int ucitaj(Vocka niz[])` koja učitava voćke sa standardnog ulaza sve do unosa reči KRAJ i kao povratnu vrednost vraća broj učitanih voćki;
- `Vocka vocka_sa_najvise_vitamina(Vocka niz[], int n)` koja pronalazi voćku koja ima najviše C vitamina.

Napisati program koji učitava podatke o voćkama i ispisuje ime voćke sa najviše vitamina C. Pretpostaviti da broj voćki neće biti veći od 50, kao i da je ime voćke

### 3 Predstavljanje podataka

---

niska od najviše 20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime vočke i njenu količinu vitamina C: jabuka 4.6
Unesite ime vočke i njenu količinu vitamina C: limun 83.5
Unesite ime vočke i njenu količinu vitamina C: kivi 71
Unesite ime vočke i njenu količinu vitamina C: banana 8.7
Unesite ime vočke i njenu količinu vitamina C: pomorandža 70.8
Unesite ime vočke i njenu količinu vitamina C: KRAJ
Voce sa najvise C vitamina je: limun
```

[Rešenje 3.9.3]

**Zadatak 3.9.4** Definirati strukturu **Grad** koja sadrži ime grada i prosečnu temperaturu u toku decembra (realan broj). Napisati funkcije:

- (a) `void ucitaj(Grad gradovi[], int n)` koja učitava podatke o gradovima sa standardnog ulaza.
- (b) `void ispisi(Grad gradovi[], int n)` koja ispisuje podatke o gradovima koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni celzijusa.

Napisati program koji učitava imena  $n$  gradova i njihove prosečne temperature, a zatim ispisuje imena gradova sa idealnom temperaturom za klizanje. Pretpostaviti da je maksimalan broj gradova 50 i da je maksimalna dužina imena grada 20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj gradova: 4
Unesite grad i temperaturu: Beograd 7
Unesite grad i temperaturu: Uzice 1.5
Unesite grad i temperaturu: Subotica 4
Unesite grad i temperaturu: Zrenjanin 9
Gradovi sa idealnom temperaturom za klizanje u decembru:
Beograd
Subotica
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj gradova: 2
Unesite grad i temperaturu: Varsava 11
Unesite grad i temperaturu: Prag 2
Gradovi sa idealnom temperaturom za klizanje u decembru:
```



[Rešenje 3.9.4]

**Zadatak 3.9.5** Definisati strukturu `ParReci` koja sadrži reč na srpskom jeziku i odgovarajući prevod na engleski jezik. Napisati program koji do kraja ulaza učitava sve parove reči, a potom za rečenicu koja se zadaje u jednoj liniji ispisati prevod. Ako je reč u rečenici nepoznata umesto nje ispisati odgovarajući broj zvezdica. Pretpostaviti da je maksimalna dužina reči 50 karaktera, maksimalan broj parova reči 100, a maksimalna dužina rečenice 100 karaktera.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
zima winter
godina year
sreca happiness
programiranje programming
caj tea
Unesite recenicu za prevod:
piti caj zimi je sreca
**** tea **** ** happiness
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
zima winter
pas dog
sreca happiness
prijatelj friend
solja cup
covek man
Unesite recenicu za prevod:
pas je covekov najbolji prijatelj
dog is ***** best friend
```

[Rešenje 3.9.5]

**Zadatak 3.9.6** *Cenoteka* pomaže kupcima da pronađu najpovoljniju cenu za proizvod koji žele da kupe. Napisati program koji učitava najpre broj različitih prodavnica, a zatim i podatke o ceni traženog artikla u različitim prodavnicama. Jedan red ulaza sadrži naziv prodavnice i cenu u toj prodavnici (realan broj). Korisnik zadaje željenu cenu proizvoda, a program ispisuje imena svih onih prodavnica u kojima je cena proizvoda manja ili jednaka željenoj. Pretpostaviti da je maksimalan broj prodavnica 50, a maksimalna dužina naziva prodavnice 20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj prodavnica: 5
idea 58.9
mavi 58.2
roda 55.1
tempo 54.5
interex 57.99
Unesite zeljenu cenu: 57.0
Povoljne prodavnice su:
roda
tempo
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj prodavnica: 4
dm 43.2
lily 45.99
benu_apoteke 43.99
sephora 50.99
Unesite zeljenu cenu: 47.00
Povoljne prodavnice su:
dm
lily
benu_apoteke
```

[Rešenje 3.9.6]

### 3 Predstavljanje podataka

**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). Napisati program koji za decu datog pola i broja godina ispisuje na tri decimale prosečnu ocenu obdaništa. Pretpostaviti da je maksimalan broj dece u obdaništu 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj dece: 5
Unesite podatke za svako dete, pol,
broj godina i ocenu:
m 3 5
z 3 4
m 4 2
m 5 4
m 3 4
Unesite pol i broj godina: m 3
Prosečna ocena je: 4.500.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj dece: 10
Unesite podatke za svako dete, pol,
broj godina i ocenu:
m 3 5
z 4 4
m 5 4
z 4 3
z 3 2
z 4 5
m 6 5
z 4 4
z 4 5
m 6 3
Unesite pol i broj godina: z 4
Prosečna ocena je: 4.200.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj dece: 15
Unesite podatke za svako dete, pol,
broj godina i ocenu:
m 3 2
z 7 5
Greska: neispravan broj godina.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj dece: 2
Unesite podatke za svako dete, pol,
broj godina i ocenu:
m 3 2
z 3 5
Unesite pol i broj godina: h 5
Greska: neispravan pol.
```

[Rešenje 3.9.7]

**Zadatak 3.9.8** Definirati strukturu kojom se opisuje student. Student je zadat svojim imenom i prezimenom, 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 smjera. Potom ispisati podatke za studenta koji ima najveći prosek. Ako ima više takvih studenata ispisati ih sve. Pretpostaviti da je maksimalan broj studenata 2000, a maksimalna dužina imena i prezimena po 30 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj studenata: 5
Unesite podatke o studentima:
0. student: Kocic Marija R 9.14
1. student: Tanja Mratinkovic R 7.88
2. student: Mihailo Simic N 8.44
3. student: Milena Medar I 9.14
4. student: Ljubica Mihic N 9.00
Unesite smer: R
Studenti sa R smeru:
Kocic Marija
Tanja Mratinkovic
-----
Svi studenti koji imaju maksimalni prosek:
Kocic Marija, R, 9.14
Milena Medar, I, 9.14

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj studenata: 4
Unesite podatke o studentima:
0. student: Djordje Lazarevic N 9.05
1. student: Minja Peric W 7.70
Greska: neispravan unos smeru.

```

[Rešenje 3.9.8]

**Zadatak 3.9.9** Definisati strukturu Djak koja sadrži ime đaka i 9 ocena (ocene su celi brojevi od 1 do 5). Napisati program koji učitava podatke o đacima sve do kraja ulaza i na standardni izlaz ispisuje prvo imena nedovoljnih đaka, a zatim imena odličnih đaka. Đak je nedovoljan ako ima barem jednu jedinicu, a odličan ako ima prosek ocena veći ili jednak 4.5. Pretpostaviti da je maksimalna dužina imena djaka 20 karaktera, kao i da je maksimalan broj djaka 30. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite podatke o djaku: Maja 4 5 2 3 4 4 3 3 4
Unesite podatke o djaku: Nikola 5 4 5 5 5 4 4 5 5
Unesite podatke o djaku: Jasmina 2 2 1 1 2 3 3 1 3
Unesite podatke o djaku: Pera 5 4 5 3 5 5 1 5 5
Unesite podatke o djaku: Pavle 4 3 2 4 3 2 4 3 2
Unesite podatke o djaku:

NEDOVOLJNI: Jasmina Pera
ODLICNI: Nikola

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite podatke o djaku: Uros 3 4 2 3 4 2 3 4 4
Unesite podatke o djaku: Nebojsa 4 5 5 5 4 5 5 5 5
Unesite podatke o djaku: Sreten 2 3 2 4 5 4 4 4 2
Unesite podatke o djaku:

NEDOVOLJNI:
ODLICNI: Nebojsa

```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite podatke o djaku:  Mirko 2 3 4 4 4 3 3 3 4
Unesite podatke o djaku:  Mihailo 2 3 10 5 5 2 3 4 2
Greska: neispravna ocena.
```

[Rešenje 3.9.9]

**Zadatak 3.9.10** Definisati strukturu `Osoba` kojom se opisuje jedan unos u imenik. Za svaku osobu su dati podaci: ime, prezime i email adresa. Napisati program koji učitava ceo broj  $n$ , a zatim podatke o  $n$  osoba. Ispisati imena i prezimena svih osoba koje imaju gmail adresu (čija se email adresa završava sa @gmail.com). Pretpostaviti da je maksimalan broj osoba 50, kao i da je maksimalna dužina imena osobe 20 karaktera, prezimena 30 karaktera, a email adrese 50 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Može se smatrati da je svaka email adresa dobro zadata i sadrži samo jedno pojavljivanje znaka @.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj osoba: 3
Unesite podatke o osobama:
ime, prezime i email.
Dusko Dugousko dusko@yahoo.com
Pink Panther panter@gmail.com
Pera Detlic pd@gmail.com
Vlasnici gmail naloga su:
Pink Panther
Pera Detlic
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj osoba: 3
Unesite podatke o osobama:
ime, prezime i email.
Homer Simpson homer@yahoo.com
Mardz Simpson mardz@matf.bg.ac.rs
Nema vlasnika gmail naloga.
```

[Rešenje 3.9.10]

\* **Zadatak 3.9.11** Napisati program koji izračunava prosečnu cenu jedne potrošačke korpe. Potrošačka korpa se sastoji od broja kupljenih artikala i niza kupljenih artikala. Svaki artikal određen je svojim nazivom, količinom i cenom. Program treba da učitava broj potrošača  $n$ , 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. Pretpostaviti da je maksimalan broj potrosackih korpi 100, maksimalan broj artikala u korpi 20 i da naziv svakog artikla sadrži maksimalno 30 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj potrosackih korpi: 3
Unesite podatke o korpi:
Broj artikala: 4
Unesite artikal, naziv, kolicinu i cenu: jabuke 10 22.4
Unesite artikal, naziv, kolicinu i cenu: dezodorans 1 120.99
Unesite artikal, naziv, kolicinu i cenu: C_supa 3 36.56
Unesite artikal, naziv, kolicinu i cenu: sunka 1 230.99
Unesite podatke o korpi:
Broj artikala: 2
Unesite artikal, naziv, kolicinu i cenu: Jafa_keks 55.78
Unesite artikal, naziv, kolicinu i cenu: Najlepse_zelje 62.99
Unesite podatke o korpi:
Broj artikala: 3
Unesite artikal, naziv, kolicinu i cenu: prasak_zav_1 1199.99
Unesite artikal, naziv, kolicinu i cenu: omeksivac 1 279.99
Unesite artikal, naziv, kolicinu i cenu: protiv_kamenca 1 699.99

Korpa 0:
jabuke 10 22.40
dezodorans 1 120.99
C_supa 3 36.56
sunka 1 230.99
-----
ukupno: 685.66

Korpa 1:
Jafa_keks 55 0.78
Najlepse_zelje 62 0.99
-----
ukupno: 104.28

Korpa 2:
prasak_zav_1 1 1199.99
omeksivac 1 279.99
protiv_kamenca 1 699.99
-----
ukupno: 2179.97

Prosečna cena potrosacke korpe: 989.97

```

[Rešenje 3.9.11]

**Zadatak 3.9.12** Definirati strukturu *Lopta* sa poljima *poluprecnik* (ceo broj u centimetrima) i *boja* (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Napisati funkcije

- `void ucitaj(Lopta niz[], int n)` koja učitava podatke o  $n$  lopti u niz.
- `double ukupna_zapremina(Lopta niz[], int n)` koja računa ukupnu zapreminu svih lopti.

### 3 Predstavljanje podataka

---

- (c) `int broj_crvenih(Lopta niz[], int n)` koja prebrojava koliko ima crvenih lopti u nizu.

Napisati program koji učitava informacije o  $n$  lopti i ispisuje ukupnu zapreminu i broj crvenih lopti. Pretpostaviti da je maksimalan broj lopti 50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 4
Unesite dalje poluprecnike i boje lopti
(1-plava, 2-zuta, 3-crvena, 4-zelena):
1.lopta: 4 1
2.lopta: 1 3
3.lopta: 2 3
4.lopta: 10 4
Ukupna zapremina: 4494.57
Broj crvenih lopti: 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 8
Unesite dalje poluprecnike i boje lopti
(1-plava, 2-zuta, 3-crvena, 4-zelena):
1. lopta: 2 1
2. lopta: 30 3
3. lopta: 7 3
4. lopta: 4 1
5. lopta: 5 2
6. lopta: 6 2
7. lopta: 12 3
8. lopta: 14 2
Ukupna zapremina: 134996.34
Ukupno crvenih lopti: 3
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 8
Unesite dalje poluprecnike i boje lopti
(1-plava, 2-zuta, 3-crvena, 4-zelena):
1. lopta: 1 2
2. lopta: 2 10
Greska: neispravan unos.
```

[Rešenje 3.9.12]

**Zadatak 3.9.13** Napisati program za predstavljanje poligona i izračunavanje njegovog obima i dužine stranica.

- (a) Definisati strukturu `Tacka` kojom se opisuje tačka Dekartovske ravni čije su  $x$  i  $y$  koordinate podaci tipa `double`.
- (b) Definisati funkciju `double rastojanje(const Tacka* a, const Tacka* b)` koja izračunava rastojanje između dve tačke.
- (c) Definisati funkciju `int ucitaj_poligon(Tacka* tacke, int n)` koja učitava maksimalno  $n$  puta po dve vrednosti tipa `double` (koje predstavljaju koordinate temena poligona) i upisuje ih u zadati niz tačaka. Funkcija vraća broj uspešno učitanih tačaka.

- (d) Definirati funkciju `double obim(Tacka* poligon, int n)` koja izračunava obim poligona sa  $n$  tačaka u zadatom nizu UPUTSTVO: *Prilikom računanja obima ne zaboraviti stranicu koja spaja poslednje i prvo teme..*
- (e) Definirati funkciju `double maksimalna_stranica(Tacka* poligon, int n)` koja izračunava dužinu najduže stranice poligona sa  $n$  tačaka u zadatom nizu.
- (f) Napisati funkciju `double povrsina_trougla(const Tacka* A, const Tacka* B, const Tacka* C)` za računanje površine trougla.
- (g) Napisati funkciju `double povrsina(Tacka* poligon, int n)` za računanje površine konveksnog poligona. uputstvoZadatak se može rešiti korišćenjem funkcije `povrsina_trougla`.

Napisati program koji učitava poligon sa maksimalno  $n$  temena i za učitani poligon ispisuje na tri decimale obim, maksimalnu dužinu stranice i površinu. Pretpostaviti da je uneti poligon konveksan. Poligon mora imati barem tri temena. Pretpostaviti da je maksimalan broj temena 1000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite maksimalan broj tacaka poligona: 10
0 0
0 6
3 3
Obim poligona je 14.485.
Duzina maksimalne stranice je 6.000.
Povrsina poligona je 9.000.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite maksimalan broj tacaka poligona: 10
0 0
12 0
13 2
16 5
20 10
18 15
15 20
10 20
8 15
3 4
Obim poligona je 63.566.
Duzina maksimalne stranice je 12.083.
Povrsina poligona je 247.500.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite maksimalan broj tacaka poligona: 4
0 0
Greska: poligon mora imati bar tri tacke.
```

[Rešenje 3.9.13]

### 3 Predstavljanje podataka

\* **Zadatak 3.9.14** Definirati strukturu **Izraz** kojom se opisuje numerički izraz nad celim brojevima koji se sastoji od dva celobrojna operanda i numeričke operacije (sabiranje, oduzimanje, množenje ili celobrojno deljenje) nad celim brojevima.

- (a) Napisati funkciju koja ispituje da li je dati izraz korektno zadat i vraća jedinicu ako jeste, a nulu inače. Podrazumeva se da je izraz korektno zadat ako operacija odgovara  $+$ ,  $-$ ,  $*$  ili  $/$  i u slučaju deljenja drugi operand je različit od 0.
- (b) Napisati funkciju koja za dati izraz određuje vrednost izraza.
- (c) Napisati funkciju koja učitava izraze. Funkcija treba da učitava sa standardnog ulaza  $n$  izraza koji su zadati prefiksno — prvo operacija, a potom dva operanda.

Napisati program koji učitava prirodan broj  $n$ , 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. Pretpostaviti da je maksimalan broj izraza 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj izraza: 4
Unesite izraze u prefiksnoj notaciji:
+ 10 4
- 9 2
* 11 2
/ 7 3
Maksimalna vrednost izraza: 22
Izrazi čija je vrednost manja
od polovine maksimalne vrednosti:
9 - 2 = 7
7 / 3 = 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj izraza: 10
Unesite izraze u prefiksnoj notaciji:
+ 10 2
- -678 34
* 77 2
+ 1000 -23
+ 102 4
- 200 23
/ 67 12
/ 1000 2
* 44 6
/ 13 1
Maksimalna vrednost izraza: 977
Izrazi čija je vrednost manja
od polovine maksimalne vrednosti:
10 + 2 = 12
-678 - 34 = -712
77 * 2 = 154
102 + 4 = 106
200 - 23 = 177
67 / 12 = 5
44 * 6 = 264
13 / 1 = 13
```



*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj izraza: 3
Unesite izraze u prefiksnoj notaciji:
* 1 2
/ 3 0
Greska: deljenje nulom.

```

[Rešenje 3.9.14]

\* **Zadatak 3.9.15** Definirati strukturu kojom se opisuje polinom. Polinom je dat svojim stepenom i realnim koeficijentima.

- Napisati funkciju koja sa standardnog ulaza učitava polinome sve do kraja ulaza. Polinomi su zadati stepenom i koeficijentima. Funkcija kao povratnu vrednost vraća broj učitanih polinoma.
- Napisati funkciju koja ispisuje polinom u obliku  $k_0 \pm k_1 * x \pm k_2 * x^2 \pm k_3 * x^3 \pm \dots \pm k_n * x^n$  (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.
- Napisati funkciju koja za dati polinom određuje njegov integral.

Napisati program koji učitava polinome do kraja ulaza i za svaki učitani polinom određuje i ispisuje integral tog polinoma. Pretpostaviti da je maksimalan broj polinoma 100, a maksimalan stepen polinoma 10. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite stepen: 3
Unesite koeficijente polinoma:
1 0 3 1
Unesite stepen: 4
Unesite koeficijente polinoma:
7 9 4 0 4
Unesite stepen:

Integrali su:
1.00*x + 1.00*x^3 + 0.25*x^4
7.00*x + 4.50*x^2 + 1.33*x^3 + 0.80*x^5

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite stepen: 3
Unesite koeficijente polinoma:
1 0 -4 1
Unesite stepen: 2
Unesite koeficijente polinoma:
1 2 -3
Unesite stepen: 1
Unesite koeficijente polinoma:
0 -1
Unesite stepen:

Integrali su:
1.00*x -1.33*x^3 + 0.25*x^4
1.00*x + 1.00*x^2 -1.00*x^3
-0.50*x^2

```

## 3.10 Rešenja

### Rešenje 3.9.1

```
1 #include <stdio.h>
3 /* Struktura koja opisuje kompleksni broj. */
4 typedef struct
5 {
6     float re;
7     float im;
8 } KompleksanBroj;
9
10 /* Kada se neka promenljiva zadaje kao argument funkcije, obicno
11 se prenosi po vrednosti (bez pokazivaca), ako se ona nece
12 menjati u funkciji ili po adresi (preko pokazivaca), ako ce se
13 njena vrednost promeniti u funkciji.
14
15 Prilikom poziva funkcije, za svaki argument funkcije kreira se
16 promenljiva koja predstavlja lokalnu kopiju argumenta i koja
17 prestaje da postoji po zavrsetku funkcije. S obzirom da se
18 strukture sastoje od vise polja, zauzimaju vise memorije nego
19 nestrukturne promenljive. Zbog toga je za njihovo kopiranje
20 potrebno vise vremena i vise memorijskih resursa nego za
21 kopiranje nestrukturnih promenljivih.
22
23 Da bi program bio efikasniji, korisno je da se struktura uvek
24 prenosi po adresi (preko pokazivaca), bez obzira da li ce se
25 ona u toj funkciji menjati ili ne. Pokazivac na strukturu
26 zauzima manje memorije nego sama struktura pa je izrada njegove
27 kopije brza, a kopija pokazivaca uzima manji memorijski prostor
28 nego kopija strukture.
29
30 Kada se strukturna promenljiva prenosi u funkciju po adresi
31 (preko pokazivaca), tada postoji mogucnost da se njena polja
32 menjaju u funkciji. Ukoliko to nije potrebno, uz argument se
33 dodaje kljucna rec const. Na taj nacin, u slucaju pokusaja
34 izmene strukturne promenljive koja je prosledjena kao const,
35 kompajler ce prijaviti gresku. Na ovaj nacin se obezbedjuje
36 da promenljiva koja je preneti po adresi ne bude cak ni slucajno
37 izmenjena u funkciji.
38
39 Funkcija izracunava zbir kompleksnih brojeva. */
40 KompleksanBroj saberi(const KompleksanBroj * a,
41                       const KompleksanBroj * b)
42 {
```

```

43     KomplexsanBroj c;
44     c.re = a->re + b->re;
45     c.im = a->im + b->im;
46     return c;
47 }

49 /* Funkcija izracunava razliku kompleksnih brojeva. */
KompleksanBroj oduzmi(const KompleksanBroj * a,
51                     const KompleksanBroj * b)
{
53     KompleksanBroj c;
54     c.re = a->re - b->re;
55     c.im = a->im - b->im;
56     return c;
57 }

59 /* Funkcija izracunava proizvod kompleksnih brojeva. */
KompleksanBroj pomnozi(const KompleksanBroj * a,
61                      const KompleksanBroj * b)
{
63     KompleksanBroj c;
64     c.re = a->re * b->re - a->im * b->im;
65     c.im = b->re * a->im + a->re * b->im;
66     return c;
67 }

69 /* Funkcija izracunava kolicnik kompleksnih brojeva. */
KompleksanBroj podeli(const KompleksanBroj * a,
71                     const KompleksanBroj * b)
{
73     KompleksanBroj c;
74     c.re = (a->re * b->re + a->im * b->im) /
75           (b->re * b->re + b->im * b->im);
76     c.im = (b->re * a->im - a->re * b->im) /
77           (b->re * b->re + b->im * b->im);
78     return c;
79 }

81 int main()
{
83     /* Deklaracije potrebnih promenljivih. */
84     KompleksanBroj a, b;
85     KompleksanBroj c;

87     /* Ucitavanje kompleksnih brojeva. */
88     printf("Unesite realni i imaginarni deo prvog broja: ");
89     scanf("%f%f", &a.re, &a.im);

91     printf("Unesite realni i imaginarni deo drugog broja: ");
92     scanf("%f%f", &b.re, &b.im);

93     /* Ispis zbira. */

```

### 3 Predstavljanje podataka

```
95  c = saberi(&a, &b);
    /* Ukoliko je imaginarni deo negativan, njegov zapis vec
97     ukljucuje znak, u suprotnom, broj je oblika a + b*i. */
    printf("Zbir: %.2f%c%.2f*i\n",
99         c.re, c.im > 0 ? '+' : '-', c.im);

101 /* Ispis razlike. */
    c = oduzmi(&a, &b);
103 printf("Razlika: %.2f%c%.2f*i\n",
        c.re, c.im > 0 ? '+' : '-', c.im);

105 /* Ispis proizvoda. */
    c = pomnozi(&a, &b);
107 printf("Proizvod: %.2f%c%.2f*i\n",
        c.re, c.im > 0 ? '+' : '-', c.im);

109 /* Ispis kolicnika. */
    if (b.re != 0 || b.im != 0)
111     {
113         c = podeli(&a, &b);
115         printf("Kolicnik: %.2f%c%.2f*i\n",
            c.re, c.im > 0 ? '+' : '-', c.im);
117     }
    else
119         printf("Kolicnik ne postoji.\n");

121 return 0;
}
```

#### Rešenje 3.9.2

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

3
   typedef struct {
5       int brojilac;
       int imenilac;
7   } Razlomak;

9   /* Funkcija Euklidovim algoritmom racuna najveći zajednički
   delilac brojeva a i b. */
11  int nzd(int a, int b)
   {
13     int ostatak;

15     while (b != 0) {
        ostatak = a % b;
17         a = b;
        b = ostatak;
19     }
}
```

```
21     return a;
22 }
23
24 /* Funkcija vraca razlomak koji se dobija deljenjem imenioca i
25    brojioca sa njihovim najvećim zajedničkim deliocem. */
26 void uprosti(Razlomak * r)
27 {
28     int nzd_razlomka = nzd(r->brojilac, r->imenilac);
29     r->brojilac /= nzd_razlomka;
30     r->imenilac /= nzd_razlomka;
31 }
32
33 /* Funkcija racuna zbir razlomaka a i b. */
34 Razlomak saberi(const Razlomak* a, const Razlomak* b)
35 {
36     Razlomak c;
37
38     c.brojilac = a->brojilac * b->imenilac + b->brojilac * a->imenilac;
39     c.imenilac = a->imenilac * b->imenilac;
40     uprosti(&c);
41
42     return c;
43 }
44
45 /* Funkcija racuna proizvod razlomaka a i b. */
46 Razlomak pomnozi(const Razlomak* a, const Razlomak* b)
47 {
48     Razlomak c;
49
50     c.brojilac = a->brojilac * b->brojilac;
51     c.imenilac = a->imenilac * b->imenilac;
52     uprosti(&c);
53
54     return c;
55 }
56
57 int main()
58 {
59     /* Deklaracije potrebnih promenljivih. */
60     int n, i;
61     Razlomak suma, proizvod, r;
62
63     /* Ucitavanje broja razlomaka i provera ispravnosti ulaza. */
64     printf("Unesite broj razlomaka: ");
65     scanf("%d", &n);
66     if (n <= 0)
67     {
68         printf("Greska: neispravan unos.\n");
69         exit(EXIT_FAILURE);
70     }
71
72     /* Inicijalizacija sume i proizvoda. */
```

### 3 Predstavljanje podataka

```
73 suma.brojilac = 0;
   suma.imenilac = 1;
75 proizvod.brojilac = 1;
   proizvod.imenilac = 1;
77
   /* Ucitavanje razlomaka i racunanje rezultata. */
79 printf("Unesite razlomke:\n");
   for (i = 0; i < n; i++)
81 {
       scanf("%d%d", &r.brojilac, &r.imenilac);
83
       suma = saberi(&suma, &r);
85       proizvod = pomnozi(&proizvod, &r);
   }
87
   /* Ispis rezultata. */
89 printf("Suma svih razlomaka je %d/%d.\n", suma.brojilac,
        suma.imenilac);
91 printf("Proizvod svih razlomaka je %d/%d.\n", proizvod.brojilac,
        proizvod.imenilac);
93
   return 0;
95 }
```

#### Rešenje 3.9.3

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

4 #define MAKS_IME 21
   #define MAKS_VOCKI 50
6
   /* Struktura koja opisuje vocku. */
8 typedef struct
   {
10     char ime[MAKS_IME];
       float vitamin;
12 } Vocka;

14 /* Funkcija ucitava podatke o vockama u niz struktura.
   Kao povratnu vrednost vraca broj ucitanih vocki. */
16 int ucitaj(Vocka niz[])
   {
18     int i=0;
       char ime[MAKS_IME];
20
       /* Vocke se ucitavaju sve dok se ne unese rec "KRAJ". */
22     do
       {
24         printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
```

```

26     /* Ucitavanje imena vocke. */
    scanf("%s", ime);
28     if (strcmp(ime, "KRAJ") == 0)
        break;
30     strcpy(niz[i].ime, ime);

32     /* Ucitavanje kolicine vitamina C. */
    scanf("%f", &niz[i].vitamin);
34
    i++;
36 }
    while (i < MAKS_VOCKI);
38
    return i;
40 }

42 /* Funkcija pronalazi vocku sa najvise C vitamina. */
Vocka vocka_sa_najvise_vitamina(Vocka niz[], int n)
44 {
    /* Pronalazak pozicije vocke sa najvise vitamina c. */
46     int maks_i = 0, i;
    for (i = 1; i < n; i++)
48         if (niz[i].vitamin > niz[maks_i].vitamin)
            maks_i = i;

50     /* Kao povratna vrednost se vraca vocka na poziciji maks_i. */
52     return niz[maks_i];
}

54
55 int main()
56 {
    /* Deklaracije potrebnih promenljivih. */
58     Vocka vocke[MAKS_VOCKI], najzdravija;
    int n;

60
    /* Ucitavanje ulaza. */
62     n = ucitaj(vocke);

64
    /* Ispis rezultata. */
    najzdravija = vocka_sa_najvise_vitamina(vocke, n);
66     printf("Voce sa najvise C vitamina je: %s\n", najzdravija.ime);

68     return 0;
}

```

### Rešenje 3.9.4

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS_IME 21

```

### 3 Predstavljanje podataka

---

```
5 #define MAKS_GRADOVA 50
6 #define DONJA_GRANICA 3
7 #define GORNJA_GRANICA 8
8
9 typedef struct
10 {
11     char ime_grada[MAKS_IME];
12     float temperatura;
13 } Grad;
14
15 /* Funkcija ucitava podatke o gradovima u niz. */
16 void ucitaj(Grad gradovi[], int n)
17 {
18     int i;
19     for (i = 0; i < n; i++)
20     {
21         printf("Unesite grad i temperaturu: ");
22         scanf("%s %f", gradovi[i].ime_grada, &gradovi[i].temperatura);
23     }
24 }
25
26 /* Funkcija ispisuje gradove sa idealnom temperaturom za klizanje
27    u decembru. */
28 void ispisi(Grad gradovi[], int n)
29 {
30     int i;
31
32     printf("Gradovi sa idealnom temperaturom za "
33           "klizanje u decembru:\n");
34     for (i = 0; i < n; i++)
35     {
36         if (gradovi[i].temperatura >= DONJA_GRANICA &&
37             gradovi[i].temperatura <= GORNJA_GRANICA)
38         {
39             printf("%s\n", gradovi[i].ime_grada);
40         }
41     }
42 }
43
44 int main()
45 {
46     /* Deklaracije potrebnih promenljivih. */
47     int n;
48     Grad gradovi[MAKS_GRADOVA];
49
50     /* Ucitavanje broja gradova i proveru ispravnosti ulaza. */
51     printf("Unesite broj gradova: ");
52     scanf("%d", &n);
53     if (n <= 0 || n > MAKS_GRADOVA)
54     {
55         printf("Greska: neispravan unos.\n");
56         exit(EXIT_FAILURE);
57     }
58 }
```



```

57     }

59     /* Ucitavanje podataka o gradovima. */
    ucitaj(gradovi, n);

61     /* Ispis rezultata. */
63     ispisi(gradovi, n);

65     return 0;
}

```

### Rešenje 3.9.5

```

#include <stdio.h>
#include <string.h>

#define MAKS_REC 21
#define MAKS_BROJ_REC 100

typedef struct
{
    char sr[MAKS_REC];
    char en[MAKS_REC];
} ParReci;

/* Funkcija ucitava parove reci u recnik. */
int ucitaj(ParReci recnik[])
{
    int i = 0;
    char sr[MAKS_REC];
    char en[MAKS_REC];

    /* Ucitavaju se parovi reci sa standardnog ulaza sve do kraja
       ulaza. */
    while (scanf("%s %s", sr, en) != EOF)
    {
        if (i == MAKS_BROJ_REC)
            break;

        strcpy(recnik[i].sr, sr);
        strcpy(recnik[i].en, en);

        i++;
    }

    return i;
}

/*
Funkcija u recniku koji sadrzi n reci trazi prevod reci rec
i upisuje ga u prevod. Ukoliko se rec ne nalazi u recniku,

```

### 3 Predstavljanje podataka

---

```
    prevod se sastoji od zvezdica pri cemu broj zvezdica odgovara
40    duzini nepoznate reci. */
void pronadji_prevod(ParReci recnik[], int n, char rec[],
42                      char prevod[])
{
44    int i;

46    /* Pretraga reci. */
    for (i = 0; i < n; i++)
48    {
        if (strcmp(recnik[i].sr, rec) == 0)
50        {
            strcpy(prevod, recnik[i].en);
52            return;
        }
54    }

56    /* Ukoliko rec nije pronadjena, formira se prevod reci koji se
        sastoji od zvezdica. */
58    for (i = 0; rec[i]; i++)
        prevod[i] = '*';
60    prevod[i] = '\0';
}

62
int main()
64 {
    /* Deklaracije potrebnih promenljivih. */
66    ParReci recnik[MAKS_BROJ_REC];
    int n;
68    char rec[MAKS_REC];
    char prevod[MAKS_REC];
70    char c;

72    /* Ucitavaju se parovi reci u recnik. */
    n = ucitaj(recnik);

74

76    /* Ucitava se recenica i ispisuje se njen prevod. */
    printf("Unesite recenicu za prevod: \n");
    do {
78        /* Ucitava se rec po rec date recenice i pronalazi se njen
            prevod. */
80        scanf("%s", rec);
        pronadji_prevod(recnik, n, rec, prevod);
82        printf("%s ", prevod);

84        /* Ukoliko je karakter iza reci znak za novi red, onda se
            prekida sa unosom, a ako nije ucitava se sledeca rec. */
86        c = getchar();
    } while (c != '\n');

88    putchar('\n');
90
```

```

    return 0;
92 }

```

### Rešenje 3.9.6

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS_PODATAKA 50
5  #define MAKS_NAZIV 21
7
   typedef struct
   {
9     char naziv_prodavnice[MAKS_NAZIV];
     double cena_artikla;
11  } Artikal;

13  /* Funkcija ucitava podatke o ceni artikla u razlicitim
   prodavnicama. */
15  void ucitaj(Artikal niz[], int n)
   {
17     int i;
     for (i = 0; i < n; i++)
19     {
         scanf("%s%lf", niz[i].naziv_prodavnice, &niz[i].cena_artikla);
21         if (niz[i].cena_artikla <= 0)
         {
23             printf("Greska: neispravan unos.\n");
             exit(EXIT_FAILURE);
25         }
     }
27 }

29  /* Funkcija ispisuje imena svih prodavnica u kojima je cena
   artikla manja ili jednaka zeljenoj ceni. */
31  void ispisi(Artikal niz[], int n, double zeljena_cena)
   {
33     int i;
     printf("Povoljne prodavnice su:\n");
35     for (i = 0; i < n; i++)
         if (niz[i].cena_artikla <= zeljena_cena)
37         printf("%s\n", niz[i].naziv_prodavnice);
   }
39

   int main()
41  {
   /* Deklaracije potrebnih promenljivih. */
43     Artikal niz[MAKS_PODATAKA];
     double zeljena_cena;
45     int n;

```

### 3 Predstavljanje podataka

---

```
47  /* Ucitavanje broja prodavnica i provera ispravnosti ulaza. */
    printf("Unesite broj prodavnica: ");
49  scanf("%d", &n);
    if (n <= 0 || n > MAKS_PODATAKA)
51  {
        printf("Greska: neispravan unos.\n");
53      exit(EXIT_FAILURE);
    }
55
    /* Ucitavanje podataka o cenama. */
57  ucitaj(niz, n);

59  /* Ucitavanje zeljene cene. */
    printf("Unesite zeljenu cenu: ");
61  scanf("%lf", &zeljena_cena);

63  /* Ispis rezultata. */
    ispisi(niz, n, zeljena_cena);
65
    return 0;
67 }
```

#### Rešenje 3.9.7

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

4  #define MAKS_DECE 200

6  typedef struct
    {
8      char pol;
        int broj_godina;
10     int ocena;
    } Dete;

12
    /* Funkcija ucitava podatke o deci i proverava ispravnost unetih
14     podataka. */
    void ucitaj(Dete niz[], int n)
16  {
        char blanko;
18     int i;
        printf("Unesite podatke za svako dete, pol, broj godina i "
20             "ocenu:\n");
        for (i = 0; i < n; i++)
22     {
            scanf("%c%c%d", &blanko, &niz[i].pol, &niz[i].broj_godina,
24                  &niz[i].ocena);

26     /* Ispitivanje pogresnog unosa. */
        if (niz[i].pol != 'm' && niz[i].pol != 'z')
```

```
28     {
30         printf("Greska: neispravan pol.\n");
31         exit(EXIT_FAILURE);
32     }
33     if (niz[i].broj_godina > 6 || niz[i].broj_godina < 3)
34     {
35         printf("Greska: neispravan broj godina.\n");
36         exit(EXIT_FAILURE);
37     }
38     if (niz[i].ocena < 1 || niz[i].ocena > 5)
39     {
40         printf("Greska: neispravna ocena.\n");
41         exit(EXIT_FAILURE);
42     }
43 }
44
45 int main()
46 {
47     /* Deklaracija potrebnih promenljivih. */
48     int n, i, broj_godina;
49     Dete niz[MAKS_DECE];
50     char blanko, pol;
51     int suma, broj_dece;
52
53     /* Ucitavanje broja dece i provera ispravnosti ulaza. */
54     printf("Unesite broj dece: ");
55     scanf("%d", &n);
56     if (n <= 0 || n > MAKS_DECE)
57     {
58         printf("Greska: neispravan unos.\n");
59         exit(EXIT_FAILURE);
60     }
61
62     /* Ucitavanje podataka o deci. */
63     ucitaj(niz, n);
64
65     /* Ucitavanje trazениh podataka. */
66     printf("Unesite pol i broj godina: ");
67     scanf("%c%c%d", &blanko, &pol, &broj_godina);
68
69     /* Ispitivanje ispravnosti unetih podataka. */
70     if (pol != 'm' && pol != 'z')
71     {
72         printf("Greska: neispravan pol.\n");
73         exit(EXIT_FAILURE);
74     }
75     if (broj_godina > 6 || broj_godina < 3)
76     {
77         printf("Greska: neispravan broj godina.\n");
78         exit(EXIT_FAILURE);
79     }
80 }
```

### 3 Predstavljanje podataka

```
80  /* Racuna se prosecna ocena dece ciji se pol i broj godina
82     poklapaju sa unetim. */
    suma = 0;
84     broj_dece = 0;
    for (i = 0; i < n; i++)
86     {
        if (niz[i].pol == pol && niz[i].broj_godina == broj_godina)
88         {
            suma += niz[i].ocena;
            broj_dece++;
90         }
92     }

94     /* Ispis rezultata. */
    if (broj_dece == 0)
96         printf("Ne postoje deca sa takvim karakteristikama.\n");
    else
98         printf("Prosecna ocena je: %.3lf.\n", (double)suma / broj_dece);
100    return 0;
}
```

#### Rešenje 3.9.8

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

4 #define MAKS_STUDENATA 2000
#define MAKS_NISKA 31

6 typedef struct Student {
8     char ime[MAKS_NISKA];
    char prezime[MAKS_NISKA];
10     char smer;
    float prosek;
12 } Student;

14 /* Funkcija ucitava podatke o studentima u niz. */
void ucitaj(Student niz[], int n)
16 {
    int i;

18     printf("Unesite podatke o studentima:\n");
    for (i = 0; i < n; i++)
20     {
        printf("%d. student: ", i);
        scanf("%s %s %c %f", niz[i].ime, niz[i].prezime,
24             &niz[i].smer, &niz[i].prosek);

26         if (niz[i].smer != 'R' && niz[i].smer != 'I' &&
```

```
28     niz[i].smer != 'V' && niz[i].smer != 'N' &&
    niz[i].smer != 'T' && niz[i].smer != 'O')
30 {
    printf("Greska: neispravan unos smer.\n");
    exit(EXIT_FAILURE);
32 }
34 }

36 /* Funkcija ispisuje podatke o studentu. */
void ispisi(const Student * s)
38 {
    printf("%s %s, %c, %.2f\n", s->ime, s->prezime, s->smer,
40         s->prosek);
}

42 /* Funkcija racuna najveći prosek. */
float najveći_prosek(Student studenti[], int n)
44 {
    float maks_prosek;
    int i;

46     maks_prosek = studenti[0].prosek;
    for (i = 1; i < n; i++)
48         if (maks_prosek < studenti[i].prosek)
50             maks_prosek = studenti[i].prosek;

52     return maks_prosek;
54 }

56 int main()
58 {
    /* Deklaracija potrebnih promenljivih. */
    Student studenti[MAKS_STUDENATA];
    int n, i;
    float maks_prosek;
    char smer;

62     /* Ucitavanje broja studenata i provera ispravnosti ulaza. */
    printf("Unesite broj studenata: ");
    scanf("%d", &n);
    if (n < 0 || n > MAKS_STUDENATA)
64     {
        printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
70     }

72     /* Ucitavanje podataka o studentima. */
    ucitaj(studenti, n);

74     /* Ucitavanje smer. Pre smer se preskace novi red koji je unet
    nakon podataka o poslednjem studentu. */
76 }
```

### 3 Predstavljanje podataka

```
printf("Unesite smer: ");
80 getchar();
scanf("%c", &smer);
82 if (smer != 'R' && smer != 'I' && smer != 'V' && smer != 'N' &&
    smer != 'T' && smer != 'O')
84 {
    printf("Greska: neispravan unos smer.\n");
86     exit(EXIT_FAILURE);
}

88 /* Ispis studenata sa unetog smer. */
90 printf("Studenti sa %c smer:\n", smer);
92 for (i = 0; i < n; i++)
    if (studenti[i].smer == smer)
        printf("%s %s\n", studenti[i].ime, studenti[i].prezime);
94 printf("-----\n");

96 /* Racunanje najveceg proseka. */
maks_prosek = najveći_prosek(studenti, n);

98 /* Ispis svih studenata sa najvećim prosekom. */
100 printf("Svi studenti koji imaju maksimalni prosek:\n");
102 for (i = 0; i < n; i++)
    if (studenti[i].prosek == maks_prosek)
        ispisi(&studenti[i]);
104
106 return 0;
}
```

#### Rešenje 3.9.9

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

4 #define MAKS_IME 21
#define BROJ_OCENA 9
6 #define MAKS_DJAKA 30

8 typedef struct
{
10     char ime[MAKS_IME];
    int ocena[BROJ_OCENA];
12 }
Djak;

14 /* Funkcija proverava ispravnost date ocene. */
16 void proveraj_ocenu(int ocena)
{
18     if (ocena < 1 || ocena > 5)
    {
20         printf("Greska: neispravna ocena.\n");
    }
```



```
    exit(EXIT_FAILURE);
22 }
23 }
24
25 /* Funkcija ucitava podatke o djacima u niz. */
26 int ucitaj(Djak niz[])
27 {
28     int i,j;
29
30     while (i < MAKS_DJAKA)
31     {
32         printf("Unesite podatke o djaku: ");
33         /* Ucitavanje imena. */
34         if(scanf("%s", niz[i].ime) == EOF)
35             break;
36
37         /* Ucitavanje ocena. */
38         for (j = 0; j < BROJ_OCENA; j++)
39         {
40             scanf("%d", &niz[i].ocena[j]);
41             provera_ocene(niz[i].ocena[j]);
42         }
43
44         i++;
45     }
46
47     return i;
48 }
49
50 /* Funkcija racuna prosechnu ocenu datog djaka. */
51 float prosecna_ocena(const Djak* djak)
52 {
53     int j;
54     float suma = 0;
55     for (j = 0; j < BROJ_OCENA; j++)
56         suma += djak->ocena[j];
57
58     return suma / BROJ_OCENA;
59 }
60
61 int main()
62 {
63     /* Deklaracija potrebnih promenljivih. */
64     Djak niz[MAKS_DJAKA];
65     int i = 0, n, j;
66     float prosek;
67
68     /* Ucitavanje podataka o djacima. */
69     n = ucitaj(niz);
70
71     /* Ispisivanje imena nedovoljnih učenika. */
72     printf("\n\nNEDOVOLJNI: ");
```

### 3 Predstavljanje podataka

---

```
    for (i = 0; i < n; i++)
74      for (j = 0; j < 9; j++)
          if (niz[i].ocena[j] == 1) {
76              printf("%s ", niz[i].ime);
                  break;
78          }
    printf("\n");
80
    /* Ispisivanje imena odlicnih ucenika. */
82    printf("ODLICNI: ");
    for (i = 0; i < n; i++)
84    {
        prosek = prosecna_ocena(&niz[i]);
86        if (prosek >= 4.5)
            printf("%s ", niz[i].ime);
88    }
    printf("\n");
90
    return 0;
92 }
```

#### Rešenje 3.9.10

```
1  #include <stdio.h>
   #include <string.h>
3  #include <stdlib.h>

5  #define MAKS_IME 21
   #define MAKS_PREZIME 31
7  #define MAKS_EMAIL 51
   #define MAKS_OSoba 50
9
   typedef struct
11 {
    char ime[MAKS_IME];
13    char prezime[MAKS_PREZIME];
    char email[MAKS_EMAIL];
15 } Osoba;

17 /* I nacin:
   Funkcija proverava da li se prosledjeni email završava sa
19    "gmail.com" koriscenjem funkcije strtok. */
int gmail(char email[])
21 {
    /* Funkcija strtok "deli" nisku u podniske tako sto ih razdvaja
23    na mestu na kom se nalazi prosledjeni delimiter (u ovom slucaju
    je to "@").
25    Na primer, ukoliko je email="pera.peric@gmail.com", funkcija
    deli ovu nisku na "pera.peric" i "gmail.com". */
27    char *deo = strtok(email, "@");
```

```

29  /* Kada se funkcija sledeci put pozove i pri tom pozivu se kao
    prvi argument navede NULL, tada funkcija vraca sledeci token u
    nizu,
31  a to je u ovom slucaju "gmail.com". */
    deo = strtok(NULL, "");
33
    /* Ako se email zavrшава na "gmail.com", funkcija vraca 1, a
35  u suprotnom 0. */
    return strcmp(deo, "gmail.com") == 0;
37 }

39 // /* II nacin:
    // Funkcija proverava da li se prosledjeni email zavrшава sa
41 // "gmail.com" koriscenjem funkcije strchr. */
    // int gmail2(char email[])
    // {
43 //     /* Pronalazi se pokazivac na znak @. */
45 //     char* desni_deo = strchr(email, '@');
    //
47 //     /* Poredi se niska koja pocinje jedan karakter posle @ sa
    // niskom "gmail.com". */
49 //     return strcmp(desni_deo+1, "gmail.com") == 0;
    // }

51 int main()
53 {
    /* Deklaracije potrebnih promenljivih. */
55     int n, i, indikator;
    Osoba osobe[MAKS_OSOBA];

57
    /* Ucitavanje broja osoba i provera ispravnosti ulaza. */
59     printf("Unesite broj osoba: ");
    scanf("%d", &n);
61     if (n < 0 || n >= MAKS_OSOBA)
    {
63         printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
65     }

67
    /* Ucitavanje podataka o osobama. */
    printf("Unesite podatke o osobama, ime, prezime i email.\n");
69     for (i = 0; i < n; i++)
        scanf("%s%s%s", osobe[i].ime, osobe[i].prezime, osobe[i].email);

71
    /* Ispis rezultata. */
73     for (i = 0; i < n; i++)
    {
75         if (gmail(osobe[i].email))
        {
77             if (!indikator)
            {
79                 /* U ovu granu ce se uci samo kada se naidje na prvog

```

### 3 Predstavljanje podataka

---

```

        vlasnika gmail naloga. */
81     printf("Vlasnici gmail naloga su:\n");
        indikator = 1;
83     }
    printf("%s %s\n", osobe[i].ime, osobe[i].prezime);
85 }
}

/* Ukoliko se nije naislo ni na jednog vlasnika gmail naloga,
89     promenljiva indikator ce ostati 0 i u tom slucaju se ispsuje
    odgovarajuca poruka. */
91     if(!indikator)
        printf("Nema vlasnika gmail naloga.\n");
93
95     return 0;
}
```

#### Rešenje 3.9.11

```

1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAKS_ARTIKALA 20
5  #define MAKS_KORPI 100
   #define MAKS_NAZIV 31
7
   typedef struct
9  {
        char naziv[MAKS_NAZIV];
11     int kolicina;
        float cena;
13 } Artikal;

15 typedef struct
   {
17     int broj_artikala;
        Artikal artikli[MAKS_ARTIKALA];
19 } Korpa;

21 /* Funkcija ucitava jedan artikal i proverava ispravnost
    ucitanih podataka. */
23 void ucitaj_artikal(Artikal * a)
   {
25     printf("Unesite artikal, naziv, kolicinu i cenu: ");
        scanf("%s%d%f", a->naziv, &a->kolicina, &a->cena);
27
        if (a->kolicina <= 0)
29     {
            printf("Greska: neispravan unos kolicine (%d).\n", a->kolicina);
31     }
        exit(EXIT_FAILURE);
    }
```

```
33     if (a->cena < 0)
34     {
35         printf("Greska: neispravan unos cene (%f).\n", a->cena);
36         exit(EXIT_FAILURE);
37     }
38 }
39
40
41 /* Funkcija ucitava podatke o jednoj potrosackoj korpi. */
42 void ucitaj_korpu(Korpa * k)
43 {
44     int i;
45     printf("Unesite podatke o korpi: \n");
46
47     /* Ucitavanje broja artikala u korpi. */
48     printf("Broj artikala: ");
49     scanf("%d", &k->broj_artikala);
50     if (k->broj_artikala <= 0)
51     {
52         printf("Greska: neispravan unos broja artikala (%d).\n",
53             k->broj_artikala);
54         exit(EXIT_FAILURE);
55     }
56
57     /* Ucitavanje podataka o svakom artiklu. */
58     for (i = 0; i < k->broj_artikala; i++)
59         ucitaj_artikal(&k->artikli[i]);
60 }
61
62 /* Funkcija ucitava podatke o n potrosackih korpi. */
63 void ucitaj_niz_korpi(Korpa korpe[], int n)
64 {
65     int i;
66     for (i = 0; i < n; i++)
67         ucitaj_korpu(&korpe[i]);
68 }
69
70 /* Funkcija racuna ukupan racun za datu korpu. */
71 float izracunaj_racun(const Korpa * k)
72 {
73     int i;
74     float racun = 0;
75
76     for (i = 0; i < k->broj_artikala; i++)
77         racun += k->artikli[i].kolicina * k->artikli[i].cena;
78
79     return racun;
80 }
81
82 /* Funkcija ispisuje racun za datu korpu.*/
83 void ispisi_racun(const Korpa * k)
84 {
85 }
```

### 3 Predstavljanje podataka

---

```
85     int i;
86     for (i = 0; i < k->broj_artikala; i++)
87         printf("\t%s %d %.2f\n", k->artikli[i].naziv,
88             k->artikli[i].kolicina, k->artikli[i].cena);
89     printf("-----\n");
90     printf("\tukupno: %.2f\n", izracunaj_racun(k));
91 }

93 /* Funkcija ispisuje racune za sve potrosacke korpe u nizu. */
94 void ispisi_racune_za_korpe(Korpa korpe[], int n)
95 {
96     int i;
97     for (i = 0; i < n; i++)
98     {
99         printf("\nKorpa %d:\n", i);
100         ispisi_racun(&korpe[i]);
101     }
102 }

103 /* Funkcija racuna prosechnu cenu potrosacke korpe za dati
104    niz potrosackih korpi. */
105 float prosek(Korpa korpe[], int n)
106 {
107     int i;
108     float prosecna_cena = 0;
109
110     for (i = 0; i < n; i++)
111         prosecna_cena += izracunaj_racun(&korpe[i]);
112
113     return prosecna_cena / n;
114 }

115

117 int main()
118 {
119     /* Deklaracije potrebnih promenljivih. */
120     int n;
121     Korpa korpe[MAKS_KORPI];

122
123     /* Ucitavanje broja potrosackih korpi i provera ispravnosti
124        ulaza. */
125     printf("Unesite broj potrosackih korpi:");
126     scanf("%d", &n);
127     if (n < 0 || n > MAKS_KORPI)
128     {
129         printf("Greska: neispravan unos.\n");
130         exit(EXIT_FAILURE);
131     }

132
133     /* Ucitavanje podataka o potrosackim korpama. */
134     ucitaj_niz_korpi(korpe, n);
135
136     /* Ispis svih racuna. */
```

```
137     ispisi_racune_za_korpe(korpe, n);

139     /* Ispis prosečne cene potrosacke korpe. */
    printf("Prosečna cena potrosacke korpe: %.2f\n",
141           prosek(korpe, n));

143     return 0;
}
```

### Rešenje 3.9.12

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

5  #define MAKS 50

7  typedef struct {
   int poluprecnik;
9   enum { plava, zuta, crvena, zelena } boja;
} Lopta;

11 /* Funkcija racuna zapreminu lopte. */
13 float zapremina(const Lopta* l)
{
15     return pow(l->poluprecnik, 3) * 4 / 3 * M_PI;
}

17 /* Funkcija racuna zbir zapremina svih lopti u nizu. */
19 float ukupna_zapremina(Lopta lopte[], int n)
{
21     int i;
   float ukupno = 0;

23     for (i = 0; i < n; i++)
25         ukupno += zapremina(&lopte[i]);

27     return ukupno;
}

29 /* Funkcija broji lopte cija je boja jednaka boji koja je
   prosledjena kao argument funkcije. */
31 int broj_lopti_u_boji(Lopta lopte[], int n, unsigned boja)
33 {
   int br = 0;
35     int i;

37     for (i = 0; i < n; i++)
39         if (lopte[i].boja == boja)
            br++;
}
```

### 3 Predstavljanje podataka

```
41     return br;
42 }
43
44 int main()
45 {
46     /* Deklaracije potrebnih promenljivih. */
47     Lopta lopte[MAKS];
48     int n;
49     int i;
50     unsigned boja;
51
52     /* Ucitavanje broja lopti i provera ispravnosti ulaza. */
53     printf("Unesite broj lopti: ");
54     scanf("%d", &n);
55     if (n < 0 || n > MAKS)
56     {
57         printf("Greska: neispravan unos.\n");
58         exit(EXIT_FAILURE);
59     }
60
61     /* Ucitavanje lopti u niz. */
62     printf("Unesite dalje poluprecnike i boje lopti "
63           "(1-plava, 2-zuta, 3-crvena, 4-zelena):\n");
64     for (i = 0; i < n; i++)
65     {
66         printf("%d. lopta: ", i + 1);
67         scanf("%d%u", &lopte[i].poluprecnik, &boja);
68         if(boja < 1 || boja>4)
69         {
70             printf("Greska: neispravan unos.\n");
71             exit(EXIT_FAILURE);
72         }
73         lopte[i].boja = boja;
74     }
75
76     /* Ispis rezultata. */
77     printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));
78     printf("Ukupno crvenih lopti: %d\n",
79           broj_lopti_u_boji(lopte, n, crvena));
80
81     return 0;
82 }
```

#### Rešenje 3.9.13

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 #define MAKS_TACAKA 1000
6
```



```
typedef struct
8 {
    int x, y;
10 } Tacka;

12 /* Funkcija racuna rastojanje izmedju dve tacke. */
double rastojanje(const Tacka* a, const Tacka* b)
14 {
    return sqrt(pow(a->x - b->x, 2) + pow(a->y - b->y, 2));
16 }

18 /* Funkcija ucitava tacke poligona. */
int ucitaj_poligon(Tacka poligon[], int maks_tacaka)
20 {
    int i = 0;
22
    while (scanf("%d%d", &poligon[i].x, &poligon[i].y) != EOF)
24     {
        i++;
26         if (i >= maks_tacaka)
            break;
28     }

30     return i;
32 }

34 /* Funkcija racuna obim poligona. */
double obim_poligona(Tacka poligon[], int n)
36 {
    double obim = 0;
    int i;
38
    for (i = 0; i < n - 1; i++)
40         obim += rastojanje(&poligon[i], &poligon[i + 1]);

42     obim += rastojanje(&poligon[n - 1], &poligon[0]);

44     return obim;
46 }

48 /* Funkcija racuna najduzu stranicu poligona. */
double maksimalna_stranica(Tacka poligon[], int n)
50 {
    double maks = rastojanje(&poligon[0], &poligon[n - 1]);
    double stranica;
    int i;
52
    for (i = 0; i < n - 1; i++) {
54         stranica = rastojanje(&poligon[i], &poligon[i + 1]);
56         if (stranica > maks)
            maks = stranica;
58     }
}
```

```
60     return maks;
61 }
62
63 /* Funkcija racuna povrsinu trougla cija su temena A, B i C. */
64 double povrsina_trougla(const Tacka* A, const Tacka* B, const Tacka*
65     C)
66 {
67     double a = rastojanje(B, C);
68     double b = rastojanje(A, C);
69     double c = rastojanje(A, B);
70     double s = (a + b + c) / 2;
71
72     return sqrt(s * (s - a) * (s - b) * (s - c));
73 }
74
75 /* Funkcija racuna povrsinu poligona. */
76 double povrsina_poligona(Tacka * poligon, int n)
77 {
78     double P = 0;
79     int i;
80
81     for (i = 1; i < n - 1; i++)
82         P += povrsina_trougla(&poligon[0], &poligon[i], &poligon[i + 1]);
83
84     return P;
85 }
86
87 int main()
88 {
89     /* Deklaracije potrebnih promenljivih. */
90     int maks_tacaka, n;
91     Tacka poligon[MAKS_TACAKA];
92
93     /* Ucitavanje maksimalnog broja tacaka i provera ispravnosti. */
94     printf("Uneti maksimalan broj tacaka poligona: ");
95     scanf("%d", &maks_tacaka);
96     if (maks_tacaka < 3 || maks_tacaka > MAKS_TACAKA)
97     {
98         printf("Greska: neispravan unos.\n");
99         exit(EXIT_FAILURE);
100     }
101
102     /* Ucitavanje poligona. */
103     n = ucitaj_poligon(poligon, maks_tacaka);
104     if (n < 3)
105     {
106         printf("Greska: poligon mora imati bar tri tacke.\n");
107         exit(EXIT_FAILURE);
108     }
109
110     /* Ispis rezultata. */
```

```

110     printf("Obim poligona je %.3lf.\n",
            obim_poligona(poligon, n));
112     printf("Duzina maksimalne stranice je %.3lf.\n",
            maksimalna_stranica(poligon, n));
114     printf("Povrsina poligona je %.3lf.\n",
            povrsina_poligona(poligon, n));
116
117     return 0;
118 }

```

### Rešenje 3.9.14

```

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

#define MAKS 1000

typedef struct
{
    char o;
    int x;
    int y;
} Izraz;

/* Funkcija proverava da li je izraz ispravno zadat. */
int korektan_izraz(const Izraz* izraz)
{
    if (izraz->o != '+' && izraz->o != '-' &&
        izraz->o != '*' && izraz->o != '/')
    {
        printf("Greska: neispravna operacija.\n");
        return 0;
    }

    if (izraz->o == '/' && izraz->y == 0)
    {
        printf("Greska: deljenje nulom.\n");
        return 0;
    }

    return 1;
}

/* Funkcija ucitava n izraza sa standardnog ulaza. */
void ucitaj(Izraz izrazi[], int n)
{
    int i;

    printf("Unesite izraze u prefiksnoj notaciji:\n");
    for (i = 0; i < n; i++)
    {

```

### 3 Predstavljanje podataka

---

```
40     scanf("%c%d%d", &izrazi[i].o, &izrazi[i].x, &izrazi[i].y);
41     /* Preskace se novi red koji se nalazi nakon izraza, kako bi
42        naredni izraz bio ispravno ucitan. */
43     getchar();
44
45     /* Provera ispravnosti ucitanog izraza. */
46     if (!korektan_izraz(&izrazi[i]))
47     {
48         printf("Greska: neispravan unos.\n");
49         exit(EXIT_FAILURE);
50     }
51 }
52 }
53
54 /* Funkcija racuna vrednost izraza. */
55 int vrednost(const Izraz* izraz)
56 {
57     switch (izraz->o)
58     {
59         case '+':
60             return izraz->x + izraz->y;
61         case '-':
62             return izraz->x - izraz->y;
63         case '*':
64             return izraz->x * izraz->y;
65         case '/':
66             return izraz->x / izraz->y;
67         default:
68             printf("Greska: neispravna operacija.\n");
69             exit(EXIT_FAILURE);
70     }
71 }
72
73 /* Funkcija racuna najveću vrednost izraza. */
74 int najveća_vrednost(Izraz izrazi[], int n)
75 {
76     int i;
77     int maks_vrednost, tr_vrednost;
78
79     maks_vrednost = vrednost(&izrazi[0]);
80
81     for (i = 1; i < n; i++)
82     {
83         tr_vrednost = vrednost(&izrazi[i]);
84         if (tr_vrednost > maks_vrednost)
85             maks_vrednost = tr_vrednost;
86     }
87
88     return maks_vrednost;
89 }
90
91 int main()
```

```

92 {
    /* Deklaracije potrebnih promenljivih. */
94     int n;
    Izraz izrazi[MAKS];
96     int maks, trenutna_vrednost;
    float polovina;
98     int i;

    /* Ucitavanje broja izraza i provera ispravnosti ulaza. */
    printf("Unesite broj izraza: ");
102     scanf("%d", &n);
    if (n < 0 || n > MAKS)
104     {
        printf("Greska: neispravan unos.\n");
106         exit(EXIT_FAILURE);
    }

108     /* Preskace se belina koja se unosi nakon broja izraza.
    Ovaj korak je neophodan jer se izraz zadaje u formatu
    <operacija> <operand> <operand>
112     A <operacija> je tipa char i kada bi ovaj korak bio
    izostavljen, ta belina bi bila ucitana kao <operacija>
114     za prvi izraz. */
    getchar();
116     ucitaj(izrazi, n);

118     /* Pronalazak polovine maksimalne vrednosti. */
    maks = najveca_vrednost(izrazi, n);
120     printf("Maksimalna vrednost izraza:%d\n", maks);
    polovina = maks / 2.0;

122     /* Ispis rezultata. */
    printf("Izrazi cija je vrednost manja od polovine maksimalne "
124           "vrednosti:\n");
    for (i = 0; i < n; i++)
126     {
        trenutna_vrednost = vrednost(&izrazi[i]);
        if (trenutna_vrednost < polovina)
130         {
            printf("%d %c %d = %d\n", izrazi[i].x, izrazi[i].o,
132                   izrazi[i].y, trenutna_vrednost);
        }
134     }

136     return 0;
}

```

### Rešenje 3.9.15

```

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

```

```
1  #include <math.h>
2
3
4
5  #define MAKS_STEPEN 10
6  #define MAKS_POLINOMA 100
7
8  typedef struct
9  {
10     int stepen;
11     float koef[MAKS_STEPEN + 1];
12 } Polinom;
13
14 /* Funkcija učitava podatke o polinomima. */
15 int učitaj(Polinom niz[])
16 {
17     int i=0,j;
18
19     while(i<MAKS_POLINOMA)
20     {
21         printf("Unesite stepen: ");
22         if (scanf("%d", &(niz[i].stepen)) == EOF)
23             break;
24
25         if (niz[i].stepen > MAKS_STEPEN || niz[i].stepen < 0)
26         {
27             printf("Greska: neispravan unos stepena.\n");
28             exit(EXIT_FAILURE);
29         }
30
31         printf("Unesite koeficijente polinoma:\n");
32         for (j = 0; j <= niz[i].stepen; j++)
33             scanf("%f", &(niz[i].koef[j]));
34
35         i++;
36     }
37     return i;
38 }
39
40 /* Prvi monom je specijalan jer se ispred njega ne vrsi
41    eksplicitan ispis znaka.
42    Na primer, za polinom  $x + 3x^2$ , prvi monom je  $x$ .
43    Svakom sledecem monomu (u ovom slucaju samo  $3x^2$ )
44    u ispisu prethodi znak (+ ili -).
45    Funkcija ispisuje prvi monom. */
46 void ispis_prvog_monoma(float koef, int stepen)
47 {
48     printf("%.2f", koef);
49
50     if (stepen == 1)
51         printf("*x ");
52     else if (stepen > 1)
53         printf("*x^%d ", stepen);
54 }
```

```
56 /* Funkcija ispisuje monom koji nije prvi. */
void ispis_monoma(float koef, int stepen)
58 {
    /* Monomi ciji je koeficijent nula se ne ispisuju. */
60     if (koef != 0)
    {
62         /* Ispis znaka. */
        if (koef > 0)
64             printf("+ ");
        else
66             printf("- ");

68         /* Ispis koeficijenta. */
        printf("%.2f", fabs(koef));

70         /* Ispis ostatka. */
72         if (stepen == 1)
            printf("*x ");
74         else if (stepen > 1)
            printf("*x^%d ", stepen);
76     }
    }

78 /* Funkcija ispisuje ceo polinom p. */
80 void ispis(const Polinom * p)
{
82     int i;

84     /* Vrsi se ispis prvog monoma. Posto je moguće da prvi monom
        ima koeficijent 0, trazi se prvi monom sa ne-nula
86     koeficijentom. */
    for(i=0; i <= p->stepen; i++)
88     {
        if(p->koef[i] != 0)
90     {
            ispis_prvog_monoma(p->koef[i], i);
92             break;
        }
94     }

96     /* Ispis ostalih monoma. Nastavlja se od mesta gde se stalo u
        prethodnoj petlji i iz tog razloga je preskocen korak
98     inicijalizacije brojaca i. */
    for (; i <= p->stepen; i++)
100        ispis_monoma(p->koef[i], i);

102    printf("\n");
}

104 /* Funkcija racuna integral polinoma p. */
106 void integral(const Polinom * p, Polinom * integ)
```

### 3 Predstavljanje podataka

---

```
{
108     int i;

110     integ->stepen = p->stepen + 1;
    integ->koef[0] = 0;

112     for (i = 1; i <= integ->stepen; i++)
114         integ->koef[i] = (float) p->koef[i - 1] / i;
}

116 int main()
118 {
    /* Deklaracija potrebnih promenljivih. */
120     Polinom polinomi[MAKS_POLINOMA], integ;
    int n, i;

122     /* Ucitavanje polinoma. */
124     n = ucitaj(polinomi);

126     /* Ispis integrala. */
    printf("\n\nIntegrali su:\n");
128     for (i = 0; i < n; i++)
    {
130         integral(&polinomi[i], &integ);
        ispis(&integ);
132     }

134     return 0;
}
```



## 4

# Ulaz i izlaz programa

## 4.1 Argumenti komandne linije

**Zadatak 4.1.1** Napisati program koji ispisuje broj navedenih argumenata komandne linije, a zatim i same argumenate i njihove redne brojeve.

*Primer 1*

```
POKRETANJE: ./a.out d1.txt 10 13.5 d2.txt
IZLAZ:
Broj argumenata je 5:
0: ./a.out
1: d1.txt
2: 10
3: 13.5
4: d2.txt
```

*Primer 2*

```
POKRETANJE: ./a.out
IZLAZ:
Broj argumenata je 1:
0: ./a.out
```

[Rešenje 4.1.1]

**Zadatak 4.1.2** Napisati program koji ispisuje zbir numeričkih argumenata komandne linije. UPUTSTVO: *Koristiti funkciju atoi.*

*Primer 1*

```
POKRETANJE: ./a.out 5 ana 9 -2 11 aca 4 2
IZLAZ:
Zbir numerickih argumenata: 29
```

*Primer 2*

```
POKRETANJE: ./a.out a1 b1 c1 d1
IZLAZ:
Zbir numerickih argumenata: 0
```

### Primer 3

```
POKRETANJE: ./a.out 33 1 @matf 44
IZLAZ:
Zbir numerickih argumenata: 78
```

### Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
Zbir numerickih argumenata: 0
```

[Rešenje 4.1.2]

**Zadatak 4.1.3** Napisati program koji na osnovu broja  $n$ , koji se zadaje kao argument komandne linije, ispisuje cele brojeve iz intervala  $[-n, n]$ . U slučaju greške ispisati odgovarajuću poruku o grešci.

### Primer 1

```
POKRETANJE: ./a.out 2
IZLAZ:
-2 -1 0 1 2
```

### Primer 2

```
POKRETANJE: ./a.out
IZLAZ:
Greska: neispravan poziv.
```

### Primer 3

```
POKRETANJE: ./a.out 0
IZLAZ:
0
```

[Rešenje 4.1.3]

**Zadatak 4.1.4** Napisati program koji ispisuje argumente komandne linije koji počinju karakterom @.

### Primer 1

```
POKRETANJE: ./a.out @marija @milena #zvezda
IZLAZ:
Argumenti koji pocinju sa @:
@marija @milena
```

### Primer 2

```
POKRETANJE: ./a.out bundeva pomorandza
IZLAZ:
Nema argumenata koji pocinju sa @.
```

### Primer 3

```
POKRETANJE: ./a.out sanke @zapad zujanje
IZLAZ:
Argumenti koji pocinju sa @:
@zapad
```

### Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
Nema argumenata koji pocinju sa @.
```

[Rešenje 4.1.4]

**Zadatak 4.1.5** Napisati program koji ispisuje broj argumenata komandne linije koji sadrže karakter @.

## 4.1 Argumenti komandne linije

---

### Primer 1

```
POKRETANJE: ./a.out pera@gmail.com www.math.rs
IZLAZ:
1
```

### Primer 2

```
POKRETANJE: ./a.out math.rs pera@math.rs
IZLAZ:
1
```

### Primer 3

```
POKRETANJE: ./a.out japan caj
IZLAZ:
0
```

### Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
0
```

[Rešenje 4.1.5]

**Zadatak 4.1.6** Napisati program koji proverava da li se među zadatim argumentima komandne linije nalaze barem dva ista.

### Primer 1

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt ulaz.txt
IZLAZ:
Medju argumentima ima istih.
```

### Primer 2

```
POKRETANJE: ./a.out srce pik tref tref
IZLAZ:
Medju argumentima ima istih.
```

### Primer 3

```
POKRETANJE: ./a.out Riba ribi grize rep.
IZLAZ:
Medju argumentima nema istih.
```

### Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
Medju argumentima nema istih.
```

[Rešenje 4.1.6]

**Zadatak 4.1.7** Napisati program koji ispisuje sve opcije koje su navedene u komandnoj liniji. UPUTSTVO: *Opcije su karakteri koji se nalaze nakon karaktera -.*

### Primer 1

```
POKRETANJE: ./a.out -rf in.txt
IZLAZ:
Opcije su: r f
```

### Primer 2

```
POKRETANJE: ./a.out
IZLAZ:
Medju argumentima nema opcija.
```

### Primer 3

```
POKRETANJE: ./a.out ulaz.txt
IZLAZ:
Medju argumentima nema opcija.
```

### Primer 4

```
POKRETANJE: ./a.out in.txt -l -n 10 -fi out.txt
IZLAZ:
Opcije su: l n f i
```

[Rešenje 4.1.7]

## 4.2 Rešenja

### Rešenje 4.1.1

```
1 #include <stdio.h>
3 /* Argumenti komandne linije cuvaju se u nizu niski. Svaki element
   tog niza odgovara jednom argumentu komandne linije, pri cemu
5 prvi element predstavlja naziv programa koji se pokrece.
   Celobrojna promenljiva argc predstavlja ukupan broj argumenata
7 komandne linije ukljucujuci i argument koji odgovara nazivu
   programa, a promenljiva argv pomenuti niz niski koji sadrzi
9 same argumente. */
int main(int argc, char *argv[])
11 {
    int i;
13
    /* Ispis broja argumenata komandne linije. */
15    printf("Broj argumenata je: %d\n", argc);
17
    /* Ispis svakog od navedenih argumenata. */
    for (i = 0; i < argc; i++)
19        printf("%d: %s\n", i, argv[i]);
21
    return 0;
}
```

### Rešenje 4.1.2

```
#include <stdio.h>
2 #include <stdlib.h>
  #include <ctype.h>
4
/* Funkcija proverava da li prosledjenu nisku cine samo
6 karakteri koji su cifre. */
int samo_cifre(char arg[])
```

```

8 {
    int i;
10   for(i=0; arg[i]; i++)
        if(!isdigit(arg[i]))
12         return 0;

14   return 1;
}

16
17 int main(int argc, char *argv[])
18 {
    /* Deklaracija potrebnih promenljivih. */
20   int i, suma = 0;

22   /* Kako su argumenti komandne linije niske, potrebno ih je
    konvertovati u brojeve. Za ovo je moguće koristiti funkciju
24   atoi: atoi("567") ima vrednost 567.
    Treba voditi racuna:
26   atoi("abc") ima vrednost 0, ali atoi("12abc") ima vrednost 12.
    Dakle ova funkcija se zaustavlja u trenutku kada se u okviru
28   niske nađe na prvi karakter koji nije cifra. Iz tog razloga
    je potrebno proveriti da li dati argument sadrži samo cifre. */
30   for (i = 1; i < argc; i++)
        if(samo_cifre(argv[i]))
32         suma += atoi(argv[i]);

34   /* Ispis rezultata. */
    printf("Zbir numerickih argumenata: %d\n", suma);
36
    return 0;
38 }

```

### Rešenje 4.1.3

```

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

4 int main(int argc, char *argv[])
{
6   /* Deklaracije potrebnih promenljivih. */
    int n, i;

8   /* Proverava se broj argumenata komandne linije. */
10   if (argc != 2)
    {
12       printf("Greska: neispravan poziv.\n");
        exit(EXIT_FAILURE);
14   }

16   /* Ucitava se broj n i uzima se njegova apsolutna vrednost. */
    n = atoi(argv[1]);

```

## 4 Ulaz i izlaz programa

```
18  n = abs(n);

20  /* Ispis rezultata. */
    for (i = (-1) * n; i <= n; i++)
22      printf("%d ", i);
    printf("\n");

24      return 0;
26 }
```

### Rešenje 4.1.4

```
#include <stdio.h>

2  int main(int argc, char *argv[])
4  {
    /* Deklaracija potrebnih promenljivih. */
6    int i, indikator = 0;

8    /* Ispisuju se svi argumenti komandne linije ciji
       je prvi karakter znak '@'.
       Ako se program pokrene sa ./a.out @pera mika @zika
10    argv[0] je ./a.out i on se preskace.
       argv[1] je @pera, a prvi karakter je onda argv[1][0].
       Dakle, za argv[i] treba proveravati da li je
12    argv[i][0] jednak karakteru '@'. */
14    for (i = 1; i < argc; i++)
16    {
        if (argv[i][0] == '@')
18        {
            /* Promenljiva indikator služi da detektuje da li postoji
               bar jedna niska koja pocinje sa @. Ukoliko se naidje na
20            prvu takvu nisku, ispisuje se trazena poruka i indikator
               se postavlja na 1.*/
22            if(!indikator)
            {
24                printf("Argumenti koji pocinju sa @:\n");
26                indikator = 1;
            }
            printf("%s ", argv[i]);
28        }
30    }

32    /* Ukoliko je indikator i dalje 0, znaci da nijedan argument ne
       pocinje karakterom @. */
34    if(!indikator)
        printf("Nema argumenata koji pocinju sa @.");
36    printf("\n");

38    return 0;
}
```

## Rešenje 4.1.5

```
1 #include <stdio.h>
2 #include <string.h>
3
4 int main(int argc, char *argv[])
5 {
6     /* Deklaracija potrebnih promenljivih. */
7     int i, br = 0;
8
9     /* Prebrojavaju se argumenti koji sadrže karakter @. */
10    for (i = 1; i < argc; i++)
11        if (strchr(argv[i], '@') != NULL)
12            br++;
13
14    /* Ispis rezultata. */
15    printf("%d\n", br);
16
17    return 0;
18 }
```

## Rešenje 4.1.6

```
1 #include <stdio.h>
2 #include <string.h>
3
4 int main(int argc, char *argv[])
5 {
6     /* Deklaracije potrebnih promenljivih. */
7     int i, j;
8
9     /* Ukoliko ima jedan ili nijedan argument, onda ne može da bude
10     duplikata. */
11    if (argc < 2)
12    {
13        printf("Medju argumentima nema istih.\n");
14        return 0;
15    }
16
17    /* Za svaki argument komandne linije se proverava da li postoji
18    neki od argumenata koji mu je jednak. */
19    for (i = 0; i < argc; i++)
20    {
21        /* Za fiksirano argv[i] se vrši provera svih argumenata koji se
22        nalaze nakon njega. */
23        for (j = i+1; j < argc; j++)
24            if (strcmp(argv[i], argv[j]) == 0)
```

```
26     {
27         printf("Medju argumentima ima istih.\n");
28         return 0;
29     }
30 }
31
32 /* Ukoliko se prethodna petlja zavrсила, a nije se izaslo iz
33    programa, znaci da medju argumentima nema istih. */
34 printf("Medju argumentima nema istih.\n");
35
36 return 0;
37 }
```

### Rešenje 4.1.7

```
#include <stdio.h>
2
3 int main(int argc, char* argv[])
4 {
5     /* Deklaracije potrebnih promenljivih. */
6     int i, j, indikator = 0;
7
8     /* Prolazi se kroz sve argumente komandne linije. */
9     for(i=1; i < argc; i++)
10    {
11        /* Ukoliko argument pocinje karakterom '-', znaci da se navode
12           opcije. */
13        if(argv[i][0] == '-')
14        {
15            /* Ukoliko je u pitanju prvi niz opcija, ispisuje se
16               odgovarajuca poruka i indikator se postavlja na 1. */
17            if(!indikator)
18            {
19                printf("Opcije su: ");
20                indikator = 1;
21            }
22
23            /* Ispisuju se sve opcije, tj. svi karakteri argumenta argv[i]
24               koji se nalaze nakon '-'. */
25            for(j=1; argv[i][j]; j++)
26                printf("%c ", argv[i][j]);
27        }
28    }
29
30    /* Ukoliko indikator nakon petlje ima vrednost 0, znaci da nije
31       navedena nijedna opcija. */
32    if(!indikator)
33        printf("Medju argumentima nema opcija.\n");
34    else
35        printf("\n");
36 }
```



```

38     return 0;
    }

```

## 4.3 Datoteke

**Zadatak 4.3.1** Napisati program koji prepisuje sadržaj datoteke *ulaz.txt* u datoteku *izlaz.txt* karakter po karakter. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```

ULAZ.TXT
Danas je 21. mart.
To je prvi dan proleca.

IZLAZ.TXT
Danas je 21. mart.
To je prvi dan proleca.

```

### Primer 2

```

ULAZ.TXT
Ispit iz Programiranja 1 je
zakazan za 10. jun.

IZLAZ.TXT
Ispit iz Programiranja 1 je
zakazan za 10. jun.

```

### Primer 3

```

ULAZ.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
datoteke ulaz.txt

```

[Rešenje 4.3.1]

**Zadatak 4.3.2** Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```

ULAZ.TXT
Volim programiranje.

IZLAZ.TXT
Vipgmae

```

### Primer 2

```

ULAZ.TXT
abcdefghi
123456789

IZLAZ.TXT
adg
147

```

### Primer 3

```

ULAZ.TXT
U Beogradu ce biti
suncan i lep
dan.

IZLAZ.TXT
Ueruei
nn pa

```

[Rešenje 4.3.2]

**Zadatak 4.3.3** Napisati program koji šifrira sadržaj datoteke *podaci.txt* tako što svako slovo ciklično zamenjuje njegovim prethodnikom suprotne veličine i upisuje u datoteku *sifra.txt*. Na primer, **b** se zamenjuje sa **A**, **B** sa **a**, **a** sa **Z**, **A** sa **z**, itd. Ostali karakteri ostaju nepromenjeni. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```

PODACI.TXT
Matematicki fakultet
Studentski trg 16
Beograd

SIFRA.TXT
1ZSDLZSHBJH EZJTKSDS
rSTCDMSRJH sQF 16
aDNFQZC

```

### Primer 2

```

PODACI.TXT
a=x+y;
x=b+5;

SIFRA.TXT
Z=W+X;
W=A+5;

```

### Primer 3

```

PODACI.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
datoteke podaci.txt.

```

[Rešenje 4.3.3]

**Zadatak 4.3.4** Napisati program koji za dve datoteke čija se imena unose sa standardnog ulaza, radi sledeće:

- za svaku cifru u prvoj datoteci, u drugu datoteku upisuje 0
- za svako slovo u prvoj datoteci, u drugu datoteku upisuje 1
- za sve ostale karaktere u prvoj datoteci, u drugu datoteku upisuje 2

Maksimalna dužina naziva datoteka je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite ime prve datoteke:
prva.dat
Unesite ime druge datoteke:
druga.dat

PRVA.DAT
abc.123.[]
567.ABC.

DRUGA.DAT
111200022220002111222

```

### Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite ime prve datoteke:
ulaz.txt
Unesite ime druge datoteke:
izlaz.txt

ULAZ.TXT
18. februar 2019.

IZLAZ.TXT
11220000000211112

```

### Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite ime prve datoteke:
in.dat
Unesite ime druge datoteke:
out.dat

IN.DAT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
prve datoteke.

```

[Rešenje 4.3.4]

**Zadatak 4.3.5** Sa standardnog ulaza učitavaju se imena dve datoteke i jedan karakter koji označava opciju. Napisati program koji prepisuje sadržaj prve datoteke u drugu na sledeći način:

- ukoliko je zadata opcija u, sva mala slova zamenjuje velikim

- ukoliko je zadata opcija 1, sva velika slova zamenjuje malim

Maksimalna dužina naziva datoteka je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite imena datoteka i opciju:
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:
Unesite imena datoteka i opciju:
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:
Unesite imena datoteka i opciju:
primer.c prazna.txt V

PRIMER.C
#include <stdio.h>
int main()
{
}

PRAZNA.TXT

IZLAZ ZA GREŠKE:
Greska: neispravan unos.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite imena datoteka i opciju:
primer.c prazna.txt V

PRIMER.C NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
prve datoteke.
```

[Rešenje 4.3.5]

**Zadatak 4.3.6** Napisati program koji prebrojava mala slova u datoteci *podaci.txt* i dobijeni rezultat ispisuje na standardni izlaz. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
PODACI.TXT
Matematicki fakultet
Studentski trg 16
Beograd

IZLAZ:
Broj malih slova je: 35
```

### Primer 2

```
PODACI.TXT
PrograMiranje

IZLAZ:
Broj malih slova je: 11
```

### Primer 3

```
PODACI.TXT
MATEMATIKA
12+34=46

IZLAZ:
Broj malih slova je: 0
```

[Rešenje 4.3.6]

**Zadatak 4.3.7** Napisati program koji u datoteci čije se ime unosi sa standardnog ulaza prebrojava koliko se puta svaka cifra pojavljuje i na standardni izlaz ispisuje cifru sa najvećim brojem pojavljivanja. Ukoliko ima više takvih cifara, ispisati sve. Ukoliko datoteka ne sadrži nijednu cifru, ispisati odgovarajuću poruku. Maksimalna dužina naziva datoteka je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
ulaz.txt

ULAZ.TXT
danas je lep dan
i ja zelim
da postanem programer

IZLAZ:
Datoteka ne sadrzi cifre.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
prva.dat

PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120

IZLAZ:
0
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
primer.c

PRIMER.C
1 22 333.444

IZLAZ:
3 4
```

[Rešenje 4.3.7]

**Zadatak 4.3.8** Napisati program koji u datoteci čije je ime dato kao argument komandne linije proverava da li su zagrade pravilno uparene. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```

POKRETANJE: ./a.out zgrade.txt

ZAGRADE.TXT
ab( cd) ..
((3+4)*5+1)*9

IZLAZ:
Zgrade jesu uparene.

```

*Primer 2*

```

POKRETANJE: ./a.out primer2.dat

PRIMER2.DAT
(7+8
nisu(
uparene

IZLAZ:
Zgrade nisu uparene.

```

*Primer 3*

```

POKRETANJE: ./a.out primer3.dat

PRIMER3.DAT
)) 7 + 6 ((

IZLAZ:
Zgrade nisu uparene.

```

*Primer 4*

```

POKRETANJE: ./a.out

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.

```

[Rešenje 4.3.8]

**Zadatak 4.3.9** Napisati program koji prebrojava slova i cifre u datoteci.

- a) Napisati funkciju `int unesi_skup(char s[], FILE* f)` kojom se unosi skup elemenata iz datoteke `f`. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiđe na znak za novi red ili nedozvoljeni karakter za skup. Funkcija vraća broj elemenata skupa koji su uspešno učitani.
- b) Napisati funkciju `void prebroj(char s[], int *br_slova, int *br_cifara)` kojom se određuje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu. Napisati program koji koristeći prethodne funkcije prebrojava cifre i slova u datoteci čije se ime unosi kao argument komandne linije i ispisuje dobijene vrednosti na standardni izlaz. Pretpostaviti da je maksimalan broj elemenata skupa 1000. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

## 4 Ulaz i izlaz programa

---

Primer 1

```
POKRETANJE: ./a.out skup.txt
SKUP.TXT
abc56ighj9012hjFGHH
IZLAZ:
broj slova: 13
broj cifara: 6
```

Primer 2

```
POKRETANJE: ./a.out skup2.txt
SKUP2.TXT
ovdeimamo$dolar
IZLAZ:
broj slova: 9
broj cifara: 0
```

Primer 3

```
POKRETANJE: ./a.out skup3.txt
SKUP3.TXT
broJ3
broj5
IZLAZ:
broj slova: 4
broj cifara: 1
```

Primer 4

```
POKRETANJE: ./a.out skup4.txt
SKUP4.TXT
11.2.2019.
IZLAZ:
broj slova: 0
broj cifara: 2
```

Primer 5

```
POKRETANJE: ./a.out skup5.txt
SKUP5.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

Primer 6

```
POKRETANJE: ./a.out
IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.9]

**Zadatak 4.3.10** Napisati program koji sa standardnog ulaza učitava reč  $s$  i u datoteku *rotacije.txt* upisuje sve njene rotacije. Maksimalna dužina reči je 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite rec: abcde
ROTACIJE.TXT
abcde
bcdea
cdeab
deabc
eabcd
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite rec: 1234
ROTACIJE.TXT
1234
2341
3412
4123
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite rec: a=3*x+5;
ROTACIJE.TXT
a=3*x+5;
=3*x+5;a
3*x+5;a=
*x+5;a=3
x+5;a=3*
+5;a=3*x
5;a=3*x+
;a=3*x+5
```

[Rešenje 4.3.10]

**Zadatak 4.3.11** Sa standardnog ulaza se učitava ime datoteke i nenegativan ceo broj  $k$ . Napisati program koji učitava reči iz datoteke, i svaku pročitano

reč rotira za  $k$  mesta u levo i tako dobijenu reč upisuje u datoteku čije je ime *rotirano.txt*. Maksimalna dužina naziva datoteke je 20 karaktera. Može se pretpostaviti da datoteka sadrži samo slova i beline i da je maksimalna dužina jedne reči u datoteci 100 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke: ulaz.txt
Unesite broj k: 3

ULAZ.TXT
jedan dva
tri cetiri

ROTIRANO.TXT
anjed dva tri iricet
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke: in.dat
Unesite broj k: 5

IN.DAT
Popodne ce biti kise

ROTIRANO.TXT
nePopod ec itib isek
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke: input.txt
Unesite broj k: 0

INPUT.TXT
Popodne ce
biti kise

ROTIRANO.TXT
Popodne ce biti kise
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke: tekst.dat
Unesite broj k: 7

TEKST.DAT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

[Rešenje 4.3.11]

**Zadatak 4.3.12** Napisati program koji iz datoteke *razno.txt* u datoteku *palindromi.txt* prepisuje sve palindrome. Maksimalna dužina reči je 200 karaktera, a maksimalan broj reči nije poznat. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku. UPUTSTVO: *Reč je palindrom ako se čita isto sa leve i desne strane bez obzira na veličinu slova.*

*Primer 1*

```
RAZNO.TXT
Ana i melem su primeri palindroma.

PALINDROMI.TXT:
Ana i melem
```

*Primer 2*

```
RAZNO.TXT
jabuka neven pomorandza kuk
Oko kapAk pero radar caj

PALINDROMI.TXT:
neven kuk
Oko kapAk radar
```

## 4 Ulaz i izlaz programa

---

### Primer 3

```
RAZNO.TXT
  ovde nema palindroma
PALINDROMI.TXT:
```

### Primer 4

```
RAZNO.TXT
  Ana voli Milovana.
PALINDROMI.TXT:
  Ana
```

[Rešenje 4.3.12]

**Zadatak 4.3.13** U datoteci čije se ime zadaje sa standardnog ulaza nalazi se broj  $n$  ( $n \leq 256$ ), a zatim i  $n$  reči dužine najviše 50 karaktera. Napisati program koji učitava reči iz datoteke u niz i:

- (a) ispisuje ga na standardni izlaz
- (b) iz niza uklanja sve duplikate i upisuje transformisani niz u datoteku *rez.txt*

Pretpostaviti da je maksimalna dužina naziva datoteke 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  Unesite ime datoteke: dat1.txt

DAT1.TXT
  12 sunce kisa oblacno mraz oblacno
  mraz oblacno oblacno sveze mecava
  mecava mecava

IZLAZ:
  sunce kisa oblacno mraz oblacno mraz
  oblacno oblacno sveze mecava mecava mecava

REZ.TXT:
  sunce kisa oblacno mraz sveze mecava
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  Unesite ime datoteke: dat2.txt

DAT2.TXT
  14
  so secer supa so ljuto secer kiselo slatko
  ljuto
  paprika, ljuta paprika, ljuto dete

IZLAZ:
  so secer supa so ljuto secer kiselo slatko
  ljuto paprika, ljuta paprika, ljuto dete

REZ.TXT:
  so secer supa ljuto kiselo slatko
  paprika, ljuta dete
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
  Unesite ime datoteke: dat3.txt

DAT3.TXT
  4 abc 1234 (5+3)*12.4 11-k

IZLAZ:
  abc 1234 (5+3)*12.4 11-k

REZ.TXT:
  abc 1234 (5+3)*12.4 11-k
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
  Unesite ime datoteke: dat4.txt

DAT4.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
  Greska: neuspesno otvaranje
  ulazne datoteke.
```



[Rešenje 4.3.13]

**Zadatak 4.3.14** U datoteci čije se ime zadaje kao prvi argument komandne linije nalazi se ceo pozitivan broj  $n$ , a zatim i  $n$  celih brojeva. Napisati program koji na standardni izlaz ispisuje koliko  $k$ -tocifrenih brojeva postoji u datoteci, pri čemu se pozitivan ceo broj  $k$  zadaje kao drugi argument komandne linije. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
POKRETANJE: ./a.out ulaz.txt 2
ULAZ.TXT
6
15
193
-27
9790
35
1
IZLAZ:
Broj 2-cifrenih brojeva: 3
```

*Primer 2*

```
POKRETANJE: ./a.out in.dat 5
ULAZ.TXT
4
15
193
-27
9790
IZLAZ:
Broj 5-cifrenih brojeva: 0
```

*Primer 3*

```
POKRETANJE: ./a.out in.txt 3
IN.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

*Primer 4*

```
POKRETANJE: ./a.out in.txt
IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.14]

**Zadatak 4.3.15** Napisati program koji na standardni izlaz ispisuje maksimum brojeva iz datoteke *brojevi.txt*. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
BROJEVI.TXT
2.36 -16.11 5.96 8.88
-265.31 54.96 38.4
IZLAZ:
Najveci broj je: 54.96
```

*Primer 2*

```
BROJEVI.TXT
10.5 183.111 -90.2 3.167
IZLAZ:
Najveci broj je: 183.111
```

*Primer 3*

```
BROJEVI.TXT
-62.7 -190.2 -2.3 -1000
-198.25 -8
IZLAZ:
Najveci broj je: -2.3
```

[Rešenje 4.3.15]

**Zadatak 4.3.16** Prvi red datoteke *matrice.txt* sadrži dva cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice  $A$ . Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji pronalazi sve elemente matrice  $A$  koji su jednaki zbiru svih svojih susednih elemenata i štampa ih u obliku (broj vrste, broj kolone, vrednost elementa). Pretpostaviti da je sadržaj datoteke ispravan. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

|                                                                                                             |                                                                                          |                                                                               |
|-------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| <p><i>Primer 1</i></p> <pre> MATRICE.TXT 3 4 1 2 3 4 7 2 15 -3 -1 3 1 3  IZLAZ: (1, 0, 7) (1, 2, 15) </pre> | <p><i>Primer 2</i></p> <pre> MATRICE.TXT 2 2 1 1 -2 2  IZLAZ: (0, 0, 1) (0, 1, 1) </pre> | <p><i>Primer 3</i></p> <pre> MATRICE.TXT 1 4 9 3 5 2  IZLAZ: (0, 2, 5) </pre> |
|-------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------|

[Rešenje 4.3.16]

**Zadatak 4.3.17** Prvi red datoteke *ulaz.txt* sadrži dva cela broja između 2 i 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice  $A$ . Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji nalazi i štampa sve četvorke oblika  $(A(i,j), A(i+1,j), A(i,j+1), A(i+1,j+1))$  u kojima su svi elementi međusobno različiti. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

|                                                                                                                                               |                                                                                                            |                                                                      |
|-----------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|
| <p><i>Primer 1</i></p> <pre> ULAZ.TXT 3 4 1 2 3 4 7 2 15 -3 -1 3 1 3  IZLAZ: (3, 15, 4, -3) (7, -1, 2, 3) (2, 3, 15, 1) (15, 1, -3, 3) </pre> | <p><i>Primer 2</i></p> <pre> MATRICE.TXT 1 4 9 3 5 2  IZLAZ ZA GREŠKE: Greska: neispravna dimenzija </pre> | <p><i>Primer 3</i></p> <pre> MATRICE.TXT 2 2 1 1 -2 2  IZLAZ: </pre> |
|-----------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|

[Rešenje 4.3.17]

**Zadatak 4.3.18** U datoteci *tacke.txt* se nalazi broj tačaka, a zatim u posebnim linijama za svaku tačku njene  $x$  i  $y$  koordinate. Napisati program koji u

datoteku *rastojanja.txt* upisuje rastojanje svake od učitanih tačaka od koordinatnog početka, a na standardni izlaz koordinate tačke koja je od njega najudaljenija. Koristiti strukturu *Tacka* sa poljima  $x$  i  $y$ , kao i funkciju kojom se računa rastojanje tačke od koordinatnog početka. Maksimalan broj tačaka u datoteci je 50. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
TACKE.TXT
4
11 -2
3 5
8 -8
0 4

RASTOJANJA.TXT
11.18
5.83
11.31
4.00

IZLAZ:
Najudaljenija tačka: (8, -8)
```

*Primer 2*

```
TACKE.TXT
-2
0 0
9 -8

IZLAZ ZA GREŠKE:
Greska: neispravan broj tacaka.
```

[Rešenje 4.3.18]

**Zadatak 4.3.19** Definisati strukturu kojom se opisuje trodimenzioni vektor sa celobrojnim koordinatama  $x$ ,  $y$  i  $z$ . U datoteci *vektori.txt* nalazi se nepoznati broj vektora (najviše 200). Napisati program koji učitava vektore iz ove datoteke i na standardni izlaz ispisuje koordinate vektora sa najvećom dužinom. Dužina vektora se izračunava po formuli:

$$|v| = \sqrt{x^2 + y^2 + z^2}$$

U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
VEKTORI.TXT
2
4 -1 7
3 1 2

IZLAZ:
4 -1 7
```

*Primer 2*

```
VEKTORI.TXT
670

IZLAZ ZA GREŠKE:
Greska: neispravan broj
vektora.
```

*Primer 3*

```
VEKTORI.TXT
3
0 0 0
0 1 0
1 0 0

IZLAZ:
0 1 0
```

### Primer 4

```
VEKTORI.TXT
4
3 0 1
4 5 2
1 0 0
2 -1 2

IZLAZ:
4 5 2
```

### Primer 5

```
VEKTORI.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

### Primer 6

```
VEKTORI.TXT
1
1 1 1

IZLAZ:
1 1 1
```

[Rešenje 4.3.19]

**Zadatak 4.3.20** Definisati strukturu **Pravougaonik** koja sadrži dužine stranica i ime pravougaonika. Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava podatke o pravougaoncima (nepoznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine među pravougaoncima koji nisu kvadrati. Maksimalan broj pravougaonika je 200, maksimalna dužina imena pravougaonika je 4. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
POKRETANJE: ./a.out pravougaonici.dat

PRAVOUGAONICI.DAT
2 4 p1
3 3 p2
1 6 p3

IZLAZ:
p2 8
```

### Primer 2

```
POKRETANJE: ./a.out dva.dat

DVA.DAT
5 2 pm
4 7 pv

IZLAZ:
28
```

### Primer 3

```
POKRETANJE: ./a.out tri.dat

TRI.DAT
5 5 m
3 3 s
8 8 xl

IZLAZ:
m s xl
```

### Primer 4

```
POKRETANJE: ./a.out empty.dat

EMPTY.DAT

IZLAZ:
```

[Rešenje 4.3.20]

**Zadatak 4.3.21** U prvom redu datoteke *studenti.txt* se nalazi broj studenata, a zatim u posebnim linijama za svakog studenta korisničko ime na Alasu

i poslednjih pet ocena koje je dobio. Napisati program koji pronalazi studenta koji je ostvario najbolji uspeh i ispisuje njegove podatke. Ukoliko više studenata ima maksimalni prosek, ispisati sve. Pretpostaviti da broj studenata neće biti veći od 100. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
STUDENTI.TXT
5
mr15239 10 9 9 8 10
mi14005 8 8 9 8 10
ml15112 9 8 8 7 10
mr15007 10 10 10 10 10
mn13208 7 7 9 6 10

IZLAZ:
korisnicko ime: mr15007, prosek ocena: 10.00
```

*Primer 2*

```
STUDENTI.TXT
3
mr16156 10 9 9 10 10
mi17234 9 9 10 10 10
ml17084 9 8 8 8 8

IZLAZ:
korisnicko ime: mr16156, prosek ocena: 9.6
korisnicko ime: mi17234, prosek ocena: 9.6
```

[Rešenje 4.3.21]

**Zadatak 4.3.22** Kreirati strukturu *Student* koja sadrži:

- *ime\_i\_prezime* (u polju se čuva ime i prezime studenta, napr. "Marko Markovic", maksimalna dužina polja je 100 karaktera),
- *ocene* (sadrži najviše 10 ocena studenta)
- *br\_ocena* (ukupan broj ocena za studenata)
- *prosek* (prosečna ocena)

U datoteci čije se ime zadaje kao argument komandne linije se nalaze podaci o studentima. Za svakog studenta dato je ime, prezime i niz ocena koji se završava nulom. Svi podaci su razdvojeni razmacima. Napisati program koji učitava podatke o studentima i na standardni izlaz ispisuje podatke za studenta sa najvećim prosekom (prosek ispisati na 2 decimale). Ukoliko ima više takvih studenata, ispisati prvog. Maksimalan broj studenta je 100. UPUTSTVO: *Ime i prezime studenta se mogu pročitati pomoću specifikatora %s a potom se za kreiranje niske ime\_i\_prezime u traženom formatu može iskoristiti funkcija strcat.* U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
POKRETANJE: ./a.out studenti.txt

STUDENTI.TXT
Marko Markovic 5 6 7 8 9 0
Jelena Jankovic 10 10 10 0
Filip Viskovic 10 9 8 7 6 0
Jana Peric 10 10 9 9 8 8 7 0

IZLAZ:
Jelena Jankovic 10 10 10 0 10.00
```

### Primer 2

```
POKRETANJE: ./a.out

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.22]

**Zadatak 4.3.23** Imena ulazne i izlazne datoteke se redom navode kao argumenti komandne linije. U ulaznoj datoteci se nalaze podaci o razlomcima: u prvom redu se nalazi broj razlomaka, a u svakom sledećem redu brojilac i imenilac jednog razlomka. Definirati strukturu koja opisuje razlomak i napisati program koji učitava niz razlomaka iz datoteke, a potom:

- (a) ukoliko je navedena opcija  $x$ , upisati u izlaznu datoteku recipročni razlomak za svaki razlomak iz niza (npr. za  $2/3$  treba upisati  $3/2$ )
- (b) ukoliko je navedena opcija  $y$ , upisati u izlaznu datoteku realnu vrednost recipročnog razlomka svakog razlomka iz niza (npr. za  $2/3$  treba upisati 1.5)

Pretpostaviti da se u ulaznoj datoteci nalazi najviše 100 razlomaka. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt -x

ULAZ.TXT
4
1 5
19 3
-2 7
97 90

IZLAZ.TXT
5/1
3/19
-7/2
90/97
```

### Primer 2

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt -y

ULAZ.TXT
4
1 5
19 3
-2 7
97 90

IZLAZ.TXT
5.000000
0.157894
-3.500000
0.927835
```

*Primer 3*

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt -y
ULAZ.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

*Primer 4*

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt
IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.23]

**Zadatak 4.3.24** Definirati strukturu *Automobil* koja sadrži marku, model i cenu. Napisati program koji iz datoteke čije se ime zadaje sa standardnog ulaza učitava broj automobila i podatke za svaki automobil i zatim:

- (a) ispisuje prosečnu cenu po marki kola
- (b) za maksimalnu cenu koju je kupac spreman da plati, a koja se zadaje kao argument komandne linije, ispisuje automobile u tom cenovnom rangu zajedno sa prosečnom cenom odgovarajuće marke

Pretpostaviti da se model i marka sastoje od jedne reči i da svaka od njih sadrži najviše 30 karaktera kao i da se u datoteci nalaze podaci za najviše 100 automobila. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
POKRETANJE: ./a.out 4000

INTERAKCIJA SA PROGRAMOM:
Unesite naziv datoteke:
dat1.txt

DAT1.TXT
7
renault twingo 2900
renault megan 6250
renault clio 3650
dacia logan 5400
dacia sandero 7800
fiat bravo 4900
fiat linea 4290

IZLAZ:
Informacije o prosečnoj
ceni po markama:
renault 4266.67 3
dacia 6600.00 2
fiat 4595.00 2
Kola u vasem cenovnom rangu:
renault twingo 4266.67
renault clio 4266.67
```

### Primer 2

```
POKRETANJE: ./a.out 5000

INTERAKCIJA SA PROGRAMOM:
Unesite naziv datoteke:
dat1.txt

DAT1.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

### Primer 3

```
POKRETANJE: ./a.out

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.24]

**Zadatak 4.3.25** Kao argumenti komandne linije se zadaju ime datoteke i ceo broj  $k$ . Napisati program koji na standardni izlaz ispisuje sve linije zadate datoteke čija je dužina veća od  $k$ . Maksimalna dužina linije je 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
POKRETANJE: ./a.out test.txt 7

TEST.TXT
Teme koje su obradjuvane:
Petlje
Funkcije
Nizovi
Strukture

IZLAZ:
Teme koje su obradjuvane:
Funkcije
Strukture
```

### Primer 2

```
POKRETANJE: ./a.out test.txt

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.25]



**Zadatak 4.3.26** Napisati program koji u datoteci čije se ime navodi kao argument komandne linije određuje liniju maksimalne dužine i ispisuje je na standardni izlaz. Ukoliko ima više takvih linija, ispisati onu koja je leksikografski prva. Pretpostaviti da su linije maksimalne dužine 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
POKRETANJE: ./a.out test.txt

TEST.TXT
Danas je veoma hladno decembarsko
popodne. Ne pada sneg, kazu mozda
ce sutra.

IZLAZ:
Danas je veoma hladno decembarsko
```

*Primer 2*

```
POKRETANJE: ./a.out in.txt

IN.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

[Rešenje 4.3.26]

**Zadatak 4.3.27** U datoteci čije se ime navodi kao prvi argument komandne linije navedena je reč  $r$  i niz linija. Napisati program koji u datoteku čije se ime navodi kao drugi argument komandne linije upisuje sve linije u kojima se reč  $r$  pojavljuje bar  $n$  puta gde je  $n$  pozitivan ceo broj koji se unosi sa standardnog ulaza. Računaju se i samostalna pojavljivanja reči  $r$  i pojavljivanja u okviru neke druge reči. Ispis treba da bude u formatu broj\_pojavljivanja:linija. Maksimalna dužina reči je 100 karaktera, a maksimalna dužina linije je 500 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*Primer 1*

```
POKRETANJE: ./a.out input.txt output.txt

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 2

INPUT.TXT
sto
stolica lampa
postotak Stopiranje stopa
presto Ostoja stotina prostoriya

OUTPUT.TXT
2: postotak Stopiranje stopa
4: presto Ostoja stotina prostoriya
```

*Primer 2*

```
POKRETANJE: ./a.out input.txt output.txt

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3

INPUT.TXT
red
redar za ovu nedelju
redosled ured
odrediti raspored

OUTPUT.TXT
```

### Primer 3

```
POKRETANJE: ./a.out in.txt out.txt

IN.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

### Primer 4

```
POKRETANJE: ./a.out in.txt

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.27]

**Zadatak 4.3.28** Napisati program koji prebrojava koliko se linija datoteke *ulaz.txt* završava niskom *s* koja se učitava sa standardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera, kao i da dužina niske *s* neće biti veća od 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

### Primer 1

```
ULAZ.TXT
abcde abcde
abcde aab
abcde abcde abcde
abcde abcde Aab
abcde abcde ab
abcde abcde abcde abcde

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 3
```

### Primer 2

```
ULAZ.TXT
abcde abcde
abcde
abcde abcde AB

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 0
```

[Rešenje 4.3.28]

**Zadatak 4.3.29** Napisati program koji linije koje se učitavaju sa standardnog ulaza sve do kraja ulaza prepisuje u datoteku *izlaz.txt* i to, ako je prilikom pokretanja zadata opcija *-v* ili *-V* samo one linije koje počinju velikim slovom, ako je zadata opcija *-m* ili *-M* samo one linije koje počinju malim slovom, a ako je opcija izostavljena sve linije. Pretpostaviti da linije neće biti duže od 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*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

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.

```

*Primer 4*

```

POKRETANJE: ./a.out

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.

```

[Rešenje 4.3.29]

**Zadatak 4.3.30** Napisati program koji poredi dve datoteke i ispisuje redni broj linija u kojima se datoteke razlikuju. Imena datoteka se zadaju kao argumenti komandne linije. Pretpostaviti da je maksimalna dužina linije 200 karaktera. Linije brojati počevši od 1. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

*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

PRIMER2.DAT
danas vezbamo
programiranje
ovo je primer kad su
datoteke razlicite

IZLAZ:
2 3 4

```

### Primer 3

```
POKRETANJE: ./a.out prva.dat druga.dat

PRVA.DAT
ovo je primer
kada su
datoteke
razlicite duzine

DRUGA.DAT
kada su
programiranje
datoteke
razlicite
duzine
i kada treba ispisati broj
tih redova

IZLAZ:
1 4 5 6 7
```

### Primer 4

```
POKRETANJE: ./a.out prva.dat

IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 4.3.30]

## 4.4 Rešenja

### Rešenje 4.3.1

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4
5 int main()
6 {
7     /* Deklaracije potrebnih promenljivih. */
8     int c;
9
10    /* Promenljive ulaz i izlaz predstavljaju pokazivace na ugradjenu
11     strukturu FILE. Unutar ove strukture nalaze se polja
12     neophodna za rad sa datotekama. */
13    FILE *ulaz, *izlaz;
14
15    /* Funkcija fopen služi da otvori datoteku. Prvi argument je
16     putanja do datoteke koja se otvara, a drugi argument je niska
17     koja može imati vrednosti "r", "r+", "w", "w+", "a", "a+".
18     Kada ovaj argument ima vrednost "r" datoteka se otvara za
19     citanje.
20     Ukoliko datoteka ne postoji, funkcija fopen kao povratnu
21     vrednost vraća NULL. */
22    ulaz = fopen("ulaz.txt", "r");
```

```

23     if (ulaz == NULL)
24     {
25         /* Funkcija fprintf vrsi ispis u fajl.
26         Funkcionise isto kao i funkcija printf - razlika je sto se
27         kao prvi argument prosledjuje fajl u koji se ispisuje izlaz.
28
29         Ukoliko je izlaz potrebno ispisati na standardni izlaz za
30         greske, kao prvi argument se navodi stderr. */
31         fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
32             "ulaz.txt.\n");
33         exit(EXIT_FAILURE);
34     }
35
36     /* Ukoliko je drugi argument funkcije fopen "w", tada se
37     prosledjena datoteka otvara za pisanje. */
38     izlaz = fopen("izlaz.txt", "w");
39     if (izlaz == NULL)
40     {
41         fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
42             "izlaz.txt. \n");
43         exit(EXIT_FAILURE);
44     }
45
46     /* Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
47     Povratna vrednost ove funkcije je ASCII kod unetog karaktera.
48     Funkcija fputc ispisuje karakter c u datoteku izlaz. */
49     while ((c = fgetc(ulaz)) != EOF)
50         fputc(c, izlaz);
51
52     /* Nakon zavrsetka rada sa datotekama, neophodno ih je zatvoriti
53     pomocu ugradjene funkcije fclose. */
54     fclose(ulaz);
55     fclose(izlaz);
56
57     return 0;
58 }

```

### Rešenje 4.3.2

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     /* Deklaracije potrebnih promenljivih. */
7     FILE * ulaz, *izlaz;
8     int c;
9
10    /* Datoteka ulaz.txt se otvara za citanje i proverava se da li je
11    otvaranje proslo uspesno. */
12    ulaz = fopen("ulaz.txt", "r");

```

## 4 Ulaz i izlaz programa

```
14  if (ulaz == NULL)
16      {
18          fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
20                  "ulaz.txt.\n");
22          exit(EXIT_FAILURE);
24      }

26  /* Datoteka izlaz.txt se otvara za citanje i proverava se da li je
28      otvaranje proslo uspesno. */
30  izlaz = fopen("izlaz.txt", "w");
32  if (izlaz == NULL)
34      {
36          fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
38                  "izlaz.txt. \n");
40          exit(EXIT_FAILURE);
42      }

44  /* Iz ulazne datoteke se citaju karakteri i svaki treci karakter
46      se upisuje u izlaznu datoteku. */
48  while ((c = fgetc(ulaz)) != EOF)
50      {
52          /* Procitani karakter se upisuje u izlaznu datoteku. */
54          fputc(c, izlaz);

56          /* Naredna dva karaktera se preskacu. */
58          fgetc(ulaz);
60          fgetc(ulaz);
62      }

64  /* Zatvaraju se otvorene datoteke. */
66  fclose(izlaz);
68  fclose(ulaz);

70  return 0;
72 }
```

### Rešenje 4.3.3

```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <ctype.h>

5  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
6      izlaz za greske i izlazi iz programa. */
7  void greska(char* poruka)
8  {
9      fprintf(stderr, "%s", poruka);
10     exit(EXIT_FAILURE);
11 }

13 int main()
```

```
{
15  /* Deklaracije potrebnih promenljivih. */
    FILE *ulaz, *izlaz;
17  char c;

19  /* Datoteka podaci.txt se otvara za citanje i proverava se da li
     je otvaranje proslo uspesno. */
21  ulaz = fopen("podaci.txt", "r");
    if (ulaz == NULL)
23      greska("Greska: neuspesno otvaranje datoteke podaci.txt.\n");

25  /* Datoteka sifra.txt se otvara za pisanje i proverava se da li
     je otvaranje proslo uspesno. */
27  izlaz = fopen("sifra.txt", "w");
    if (izlaz == NULL)
29      greska("Greska: neuspesno otvaranje datoteke sifra.txt.\n");

31  /* U petlji se cita karakter po karakter. */
    while ((c = fgetc(ulaz)) != EOF)
33  {
        /* Procitani karakter se sifruje na trazeni nacin. */
35        if (islower(c))
            {
37            c = toupper(c);
                if (c == 'A')
39                c = 'Z';
                    else
41                c = c - 1;
            }
        else if (isupper(c))
43        {
            c = tolower(c);
                if (c == 'a')
45                c = 'z';
                    else
47                c = c - 1;
            }

51        /* Izmenjeni karakter se upisuje u izlaznu datoteku. */
53        fputc(c, izlaz);
    }

55    /* Zatvaraju se datoteke. */
57    fclose(ulaz);
    fclose(izlaz);

59    return 0;
61 }
```

## Rešenje 4.3.4

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_IME 21
6
7 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
8    izlaz za greske i izlazi iz programa. */
9 void greska(char* poruka)
10 {
11     fprintf(stderr, "%s", poruka);
12     exit(EXIT_FAILURE);
13 }
14
15 int main()
16 {
17     /* Deklaracije potrebnih promenljivih. */
18     FILE *ulaz, *izlaz;
19     char c;
20     char ime_datoteke1[MAKS_IME], ime_datoteke2[MAKS_IME];
21
22     /* Ucitavanje imena datoteka. */
23     printf("Unesite ime prve datoteke: ");
24     scanf("%s", ime_datoteke1);
25     printf("Unesite ime druge datoteke: ");
26     scanf("%s", ime_datoteke2);
27
28     /* Prva datoteka se otvara za citanje i proverava se da li je
29        otvaranje proslo uspesno. */
30     ulaz = fopen(ime_datoteke1, "r");
31     if (ulaz == NULL)
32         greska("Greska: neuspesno otvaranje prve datoteke.\n");
33
34     /* Druga datoteka se otvara za pisanje i proverava se da li je
35        otvaranje proslo uspesno. */
36     izlaz = fopen(ime_datoteke2, "w");
37     if (izlaz == NULL)
38         greska("Greska: neuspesno otvaranje druge datoteke.\n");
39
40     /* Iz datoteke se cita karakter po karakter i za svaku procitanu
41        cifru u izlaznu datoteku se upisuje 0, za svako slovo 1, a
42        za ostale karaktere 2. */
43     while ((c = fgetc(ulaz)) != EOF)
44     {
45         if (isdigit(c))
46             fprintf(izlaz, "0");
47         else if (isalpha(c))
48             fprintf(izlaz, "1");
49         else
50             fprintf(izlaz, "2");
51     }
```



```
53  /* Zatvaraju se datoteke. */  
    fclose(ulaz);  
55  fclose(izlaz);  
  
57  return 0;  
}
```

### Rešenje 4.3.5

```
1  #include<stdio.h>  
  #include<stdlib.h>  
3  #include<ctype.h>  
  
5  #define MAX_NAZIV 21  
  
7  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni  
   izlaz za greske i izlazi iz programa. */  
9  void greska(char* poruka)  
11 {  
    fprintf(stderr, "%s", poruka);  
    exit(EXIT_FAILURE);  
13 }  
  
15 int main()  
16 {  
17     /* Deklaracije potrebnih promenljivih. */  
    FILE *ulaz, *izlaz;  
19     char ime1[MAX_NAZIV], ime2[MAX_NAZIV];  
    char c, karakter;  
  
21     /* Ucitavanje imena datoteka i opcije. */  
23     printf("Unesite imena datoteka i opciju:");  
    scanf("%s%s %c", ime1, ime2, &c);  
  
25     /* Provera ispravnosti opcije. */  
27     if (c != 'u' && c != 'l')  
        greska("Greska: neispravan unos.\n");  
  
29     /* Prva datoteka se otvara za citanje i proverava se da li je  
31     otvaranje proslo uspesno. */  
    ulaz = fopen(ime1, "r");  
33     if (ulaz == NULL)  
        greska("Greska: neuspesno otvaranje prve datoteke.\n");  
  
35     /* Druga datoteka se otvara za pisanje i proverava se da li je  
37     otvaranje proslo uspesno. */  
    izlaz = fopen(ime2, "w");  
39     if (izlaz == NULL)  
        greska("Greska: neuspesno otvaranje druge datoteke.\n");  
41 }
```

## 4 Ulaz i izlaz programa

```
43  /* Ako je uneta opcija 'u', svi karakteri se pretvaraju u velika
    slova, a ako je opcija 'l' u mala. Izmenjena slova se upisuju
    u izlaznu datoteku. */
45  if (c == 'u')
    {
47      while ((karakter = fgetc(ulaz)) != EOF)
          fputc(toupper(karakter), izlaz);
49  }
    else
51  {
      while ((karakter = fgetc(ulaz)) != EOF)
53          fputc(tolower(karakter), izlaz);
    }
55
57  /* Zatvaraju se datoteke. */
    fclose(ulaz);
    fclose(izlaz);
59
    return 0;
61 }
```

### Rešenje 4.3.6

```
#include <stdio.h>
2  #include <stdlib.h>
   #include <ctype.h>
4
int main()
6 {
   /* Deklaracije potrebnih promenljivih. */
8   FILE * ulaz;
   int c, broj_malih = 0;
10
   /* Datoteka podaci.txt se otvara za citanje i proverava se da li je
12      otvaranje proslo uspesno. */
   ulaz = fopen("podaci.txt", "r");
14   if (ulaz == NULL)
   {
16       fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
                       "podaci.txt.\n");
18       exit(EXIT_FAILURE);
   }
20
   /* Cita se karakter po karakter i prebrojavaju se mala slova. */
22   while ((c = fgetc(ulaz)) != EOF)
   {
24       if (islower(c))
           broj_malih++;
26   }
28
   /* Ispis rezultata. */
```

```

printf("Broj malih slova je: %d\n", broj_malih);
30
/* Zatvara se datoteka.*/
32 fclose(ulaz);

34 return 0;
}

```

### Rešenje 4.3.7

```

1  #include <stdio.h>
   #include <stdlib.h>
3  #include <ctype.h>

5  #define MAKS_IME 21
   #define MAKS_CIFARA 10
7
int main()
9 {
   /* Deklaracije potrebnih promenljivih. */
11 FILE *ulaz;
   char c;
13 char ime_datoteke[MAKS_IME];
   int brojaci[MAKS_CIFARA];
15 int i, maks_i;

17 /* Ucitavanje imena ulazne datoteke. */
   printf("Unesite ime datoteke: ");
19 scanf("%s", ime_datoteke);

21 /* Ulazna datoteka se otvara za citanje i proverava se da li je
   otvaranje proslo uspesno. */
23 ulaz = fopen(ime_datoteke, "r");
   if (ulaz == NULL)
25 {
       fprintf(stderr, "Greska: neuspesno otvaranje datoteke.\n");
27 exit(EXIT_FAILURE);
   }
29

   /* Brojaci za cifre se inicijalizuju na nule.
   Indeks niza brojaci oznacava cifru (brojaci[0] se koristi za
   prebrojavanje cifre 0, brojaci[1] za 1, ..., brojaci[9] za
   cifru 9).*/
33 for (i = 0; i < MAKS_CIFARA; i++)
35     brojaci[i] = 0;

37 /* Iz datoteke se cita karakter po karakter i za svaku procitanu
   cifru se uvecava odgovarajuci brojac. */
39 while ((c = fgetc(ulaz)) != EOF)
   {
41     if (isdigit(c))

```

## 4 Ulaz i izlaz programa

```
    brojaci[c - '0']++;
43 }

45 /* Pronalazi se cifra koja se najvise puta pojavljuje u
    datoteci. */
47 maks_i = 0;
    for (i = 1; i < MAKS_CIFARA; i++)
49     if (brojaci[maks_i] < brojaci[i])
        maks_i = i;

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

55 /* Zatvara se datoteka. */
    fclose(ulaz);

57 return 0;
59 }
```

### Rešenje 4.3.8

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

4 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
    izlaz za greske i izlazi iz programa. */
6 void greska(char* poruka)
{
8     fprintf(stderr, "%s", poruka);
    exit(EXIT_FAILURE);
10 }

12 int main(int argc, char **argv)
{
14     /* Deklaracije potrebnih promenljivih. */
    FILE *ulaz;
16     char c;
    int broj_zagrada = 0;
18     int nisu_uparene = 0;

20     /* Proverava se broj argumenata komandne linije. */
    if (argc != 2)
22         greska("Greska: neispravan poziv.\n");

24     /* Ulazna datoteka se otvara za citanje i proverava se da li je
        otvaranje proslo uspesno. */
26     ulaz = fopen(argv[1], "r");
    if (ulaz == NULL)
28         greska("Greska: neuspesno otvaranje datoteke.");

30     /* Cita se karakter po karakter i proverava se da li je procitana
```

```

zagrada.
32 Ako se naidje na otvorenu zagradu, brojac se uvecava.
   Ako se naidje na zatvorenu zagradu, brojac se smanjuje.
34 Zagrade su ispravno uparene ukoliko je ovaj brojac na kraju 0.
   Dodatno, ukoliko brojac u bilo kom trenutku postane negativan,
36 to znaci da je zatvorena zagrada procitana pre otvorene,
   tako da ni u tom slucaju zagrade nisu uparene. */
38 while ((c = fgetc(ulaz)) != EOF)
{
40     if (c == '(')
        broj_zagrada++;
42     else if (c == ')')
        broj_zagrada--;
44
46     if (broj_zagrada < 0)
        {
48         nisu_uparene = 1;
            break;
        }
50 }

52 /* Ispis rezultata. */
   if (broj_zagrada != 0 || nisu_uparene)
54     printf("Zagrade nisu uparene.\n");
   else
56     printf("Zagrade jesu uparene.\n");

58 /* Zatvara se datoteka. */
   fclose(ulaz);
60
62 return 0;
}

```

### Rešenje 4.3.9

### Rešenje 4.3.10

```

#include <stdio.h>
2 #include <string.h>
#include <stdlib.h>
4
#define MAKS_NISKA 21
6
/* Funkcija rotira nisku s duzine n za jedno mesto u desno. */
8 void rotiraj_zal(char *s, int n)
{
10     int i;
    char c = s[0];
12
    for (i = 0; i < n - 1; i++)

```

## 4 Ulaz i izlaz programa

```
14     s[i] = s[i + 1];
16     s[n - 1] = c;
17 }
18
19 int main()
20 {
21     /* Deklaracije potrebnih promenljivih. */
22     char s[MAKS_NISKA];
23     int n, i;
24     FILE *izlaz;
25
26     /* Datoteka rotacije.txt se otvara za pisanje i proverava se da
27        li je datoteka uspesno otvorena. */
28     izlaz = fopen("rotacije.txt", "w");
29     if (izlaz == NULL)
30     {
31         fprintf(stderr, "Greska: neuspesno otvaranje datoteke.");
32         exit(EXIT_FAILURE);
33     }
34
35     /* Ucitava se rec koju je potrebno rotirati. */
36     printf("Unesite rec: ");
37     scanf("%s", s);
38
39     /* Izracunava se duzina reci. */
40     n = strlen(s);
41
42     /* U petlji se uneta rec rotira za 1 i ispisuje u datoteku.
43        Postupak se ponavlja n puta. */
44     for (i = 0; i < n; i++)
45     {
46         fprintf(izlaz, "%s\n", s);
47         rotiraj_za_1(s, n);
48     }
49
50     /* Zatvara se datoteka. */
51     fclose(izlaz);
52
53     return 0;
54 }
```

### Rešenje 4.3.11

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_REC 101
6 #define MAKS_IME 21
7
```

```
9  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i izlazi iz programa. */
11 void greska(char* poruka)
   {
13     fprintf(stderr, "%s", poruka);
       exit(EXIT_FAILURE);
   }

15
17 /* Funkcija u nisku rezultat smesta nisku rec rotiranu za k
   mesta u desno. */
19 void rotiraj(char *rec, int k, char *rezultat)
   {
21     int i, n;

23     /* Izracunava se duzina reci. */
       n = strlen(rec);

25     /* Ako je duzina reci npr. 5, a k ima vrednost 13, onda
       je zapravo potrebno izvesti rotaciju za 3 mesta (nema
       potrebe da se vrte dva cela kruga pre toga). */
27     k = k % n;

29     /* Karakteri koji se u pocetnoj reci nalaze na pozicijama 0-k,
       u rezultujucoj reci treba da budu na pozicijama
       od n-k do n-1.*/
31     for (i = 0; i < k; i++)
       rezultat[n - k + i] = rec[i];

33     /* Slicno, karakteri koji se u pocetnoj reci nalaze na pozicijama
       od k do n-1, u rezultujucoj reci treba da budu na pozicijama
       od 0 do n-k-1. */
35     for (i = k; i < n; i++)
       rezultat[i - k] = rec[i];

37     /* Na kraj rezultujuce niske se upisuje terminalna nula. */
41     rezultat[n] = '\0';
43 }

45
47 int main()
   {
49     /* Deklaracije potrebnih promenljivih. */
       FILE *ulaz, *izlaz;
       char ime_datoteke[MAKS_IME];
51     char rec[MAKS_REC];
       char rezultat[MAKS_REC];
53     int k;

55     /* Ucitava se ime ulazne datoteke. */
       printf("Unesite ime datoteke: ");
57     scanf("%s", ime_datoteke);

59     /* Ulazna datoteka se otvara za citanje i proverava se da li je
```

```
        otvaranje proslo uspesno. */
61  ulaz = fopen(ime_datoteke, "r");
    if (ulaz == NULL)
63      greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

65  /* Datoteka rotirano.txt se otvara za citanje i proverava se da
    li je otvaranje proslo uspesno. */
67  izlaz = fopen("rotirano.txt", "w");
    if (izlaz == NULL)
69      greska("Greska: neuspesno otvaranje izlazne datoteke.\n");

71  /* Ucitava se broj k. */
    printf("Unesite broj k: ");
73  scanf("%d", &k);
    if( k < 0)
75      greska("Greska: neispravan unos broja k.\n");

77  /* Iz datoteke se cita rec po rec, sve dok se ne dodje do kraja
    ulaza. */
79  while (fscanf(ulaz, "%s", rec) != EOF)
    {
81      /* Procitana rec se rotira za k mesta i upisuje u izlaznu
        datoteku. */
83      rotiraj(rec, k, rezultat);
        fprintf(izlaz, "%s ", rezultat);
85    }

87  /* Zatvaraju se datoteke. */
    fclose(ulaz);
89    fclose(izlaz);

91  return 0;
}
```

### Rešenje 4.3.12

```
1  #include <stdio.h>
    #include <stdlib.h>
3  #include <string.h>
    #include <ctype.h>

5
    #define MAKS_REC 201
7
    /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
9    izlaz za greske i izlazi iz programa. */
    void greska(char* poruka)
11  {
        fprintf(stderr, "%s", poruka);
13    exit(EXIT_FAILURE);
    }
15
```



```

17  /* Funkcija proverava da li je prosledjena rec palindrom.
    Pri proveru se ignorise razlika izmedju malih i velikih slova. */
18  int palindrom(char rec[])
19  {
20      int n = strlen(rec);
21      int i;
22
23      for (i = 0; i < n / 2; i++)
24          if (tolower(rec[i]) != tolower(rec[n - i - 1]))
25              return 0;
26
27      return 1;
28  }
29
30  int main()
31  {
32      /* Deklaracije potrebnih promenljivih. */
33      FILE *ulaz, *izlaz;
34      char rec[MAKS_REC];
35
36      /* Ulazna datoteka se otvara za citanje i proverava se da li je
37         otvaranje proslo uspesno. */
38      ulaz = fopen("razno.txt", "r");
39      if (ulaz == NULL)
40          greska("Greska: neuspesno otvaranje ulazne datoteke.\n");
41
42      /* Izlazna datoteka se otvara za pisanje i proverava se da li je
43         otvaranje proslo uspesno. */
44      izlaz = fopen("palindromi.txt", "w");
45      if (izlaz == NULL)
46          greska("Greska: neuspesno otvaranje izlazne datoteke.\n");
47
48      /* Iz datoteke se cita rec po rec i u izlaznu datoteku se upisuju
49         reci koje su palindromi. */
50      while (fscanf(ulaz, "%s", rec) != EOF)
51      {
52          if (palindrom(rec))
53              fprintf(izlaz, "%s ", rec);
54      }
55
56      /* Zatvaraju se datoteke. */
57      fclose(ulaz);
58      fclose(izlaz);
59
60      return 0;
61  }

```

### Rešenje 4.3.13

```

1  #include <stdio.h>
2  #include <stdlib.h>

```

```
1  #include <string.h>
2
3
4  #define MAKS_BROJ_RECI 256
5
6  #define MAKS_DUZINA_RECI 51
7
8  #define MAKS_IME 21
9
10 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
11    izlaz za greske i izlazi iz programa. */
12 void greska(char* poruka)
13 {
14     fprintf(stderr, "%s", poruka);
15     exit(EXIT_FAILURE);
16 }
17
18 int main()
19 {
20     /* Deklaracije potrebnih promenljivih. */
21     char ime_datoteke[MAKS_IME];
22     char niz_reci[MAKS_BROJ_RECI][MAKS_DUZINA_RECI];
23     FILE *ulaz, *izlaz;
24     int n, i, k, indikator;
25
26     /* Ucitava se ime ulazne datoteke. */
27     printf("Unesite ime datoteke: ");
28     scanf("%s", ime_datoteke);
29
30     /* Ulazna datoteka se otvara za citanje i proverava se da li je
31        otvaranje proslo uspesno. */
32     ulaz = fopen(ime_datoteke, "r");
33     if (ulaz == NULL)
34         greska("Greska: neuspesno otvaranje ulazne datoteke.\n");
35
36     /* Iz datoteke se ucitava broj reci. */
37     fscanf(ulaz, "%d", &n);
38     if (n < 0 || n > MAKS_BROJ_RECI)
39         greska("Greska: neispravna vrednost broja reci.\n");
40
41     /* Ucitava se rec po rec iz datoteke, smesta se u niz i ispisuje
42        se na standardni izlaz. */
43     for (i = 0; i < n; i++)
44     {
45         fscanf(ulaz, "%s", niz_reci[i]);
46         printf("%s ", niz_reci[i]);
47     }
48
49     /* Izlazna datoteka se otvara za pisanje i proverava se da li je
50        otvaranje proslo uspesno. */
51     izlaz = fopen("rez.txt", "w");
52     if (izlaz == NULL)
53         greska("Greska: neuspesno otvaranje izlazne datoteke.\n");
54
55     /* U izlaznu datoteku se upisuju reci, izostavljajuci duplikate. */
```

```

56   for (i = 0; i < n; i++)
57   {
58       /* Za rec na poziciji i se proverava da li se ona nalazi negde
59          na pozicijama od 0 do i. Ukoliko se nalazi, to znaci da je
60          vec upisana u datoteku i da je treba preskociti.
61          U tom slucaju vrednost promenljive indikator ce biti
62          postavljena na 1. */
63       indikator = 0;
64       for (k = 0; k < i; k++)
65       {
66           if (strcmp(niz_reci[k], niz_reci[i]) == 0)
67           {
68               indikator = 1;
69               break;
70           }
71       }
72       /* Ako indikator ima vrednost 0, to znaci da se doslo do prvog
73          pojavljivanja reci i da je treba upisati u izlaznu
74          datoteku. */
75       if (!indikator)
76           fprintf(izlaz, "%s\n", niz_reci[i]);
77   }
78
79   /* Zatvaraju se datoteke. */
80   fclose(ulaz);
81   fclose(izlaz);
82
83   return 0;
84 }

```

### Rešenje 4.3.14

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
6    izlaz za greske i izlazi iz programa. */
7 void greska(char* poruka)
8 {
9     fprintf(stderr, "%s", poruka);
10    exit(EXIT_FAILURE);
11 }
12
13 /* Funkcija racuna broj cifara broja x. */
14 int broj_cifara(int x)
15 {
16     int brojac = 0;
17
18     do

```

```
{
20     brojac++;
    x /= 10;
22 } while (x);

24 return brojac;
}

26 /* Funkcija broji koliko ima k-tocifrenih brojeva u datoteci f. */
28 int prebrojavanje(FILE * f, int k)
{
30     int n, x, i, brojac;

32     /* Ucitava se broj brojeva u datoteci. */
    fscanf(f, "%d", &n);
34     if (n <= 0)
        greska("Greska: neispravna vrednost broja n.\n");

36     /* Cita se broj po broj i za svaki procitani broj se racuna
38         broj cifara. Ukoliko je on jednak k, uvecava se odgovarajuci
        brojac. */
40     brojac = 0;
    for (i = 0; i < n; i++)
42     {
        fscanf(f, "%d", &x);
44         if (broj_cifara(x) == k)
            brojac++;
46     }

48     /* Povratna vrednost funkcije je broj k-tocifrenih brojeva. */
    return brojac;
50 }

52 int main(int argc, char *argv[])
{
54     /* Deklaracije potrebnih promenljivih. */
    int k;
56     FILE *ulaz;

58     /* Proverava se broj argumenata komandne linije. */
    if (argc != 3)
60         greska("Greska: neispravan poziv.\n");

62     /* Ulazna datoteka se otvara za citanje i proverava se da li je
        otvaranje proslo uspesno. */
64     ulaz = fopen(argv[1], "r");
    if (ulaz == NULL)
66         greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

68     /* Cita se broj k i vrsi se provera ispravnosti. */
    k = atoi(argv[2]);
70     if (k <= 0)
```

```

    greska("Greska: neispravna vrednost broja k.\n");
72
    /* Ispis rezultata. */
74    printf("Broj %d-cifrenih brojeva: %d\n", k,
           prebrojavanje(ulaz, k));
76
    /* Zatvara se datoteka. */
78    fclose(ulaz);
80
    return 0;
}

```

### Rešenje 4.3.15

```

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

4  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i izlazi iz programa. */
6  void greska(char* poruka)
   {
8      fprintf(stderr, "%s", poruka);
      exit(EXIT_FAILURE);
10 }

12 int main()
   {
14     /* Deklaracije potrebnih promenljivih. */
     FILE * ulaz;
16     float broj, najveci_broj;

18     /* Otvara se datoteka brojevi.txt i proverava se da li je
       otvaranje proslo uspesno. */
20     ulaz = fopen("brojevi.txt", "r");
     if (ulaz == NULL)
22         greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

24     /* Promenljiva u koju se smesta najveci broj se inicijalizuje na
       prvi broj iz datoteke. Ukoliko se pri prvom citanju dodje do
       kraja datoteke, ispisuje se odgovarajuca poruka. */
26     if(fscanf(ulaz, "%f", &najveci_broj) == EOF)
28         greska("Greska: datoteka je prazna.\n");

30     /* Iz datoteke se cita broj po broj, sve dok se ne dodje do kraja
       datoteke i trazi se najveci procitani broj. */
32     while (fscanf(ulaz, "%f", &broj) != EOF)
     {
34         if (broj > najveci_broj)
             najveci_broj = broj;
36     }
}

```

## 4 Ulaz i izlaz programa

```
38  /* Ispis rezultata. */
    printf("Najveci broj je: %.2f\n", najveci_broj);
40
    /* Zatvara se datoteka. */
42    fclose(ulaz);
44
    return 0;
}
```

### Rešenje 4.3.16

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

4 #define MAKS_DIM 50

6 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i izlazi iz programa. */
8 void greska(char* poruka)
{
10     fprintf(stderr, "%s", poruka);
    exit(EXIT_FAILURE);
12 }

14 int main()
{
16     /* Deklaracije potrebnih promenljivih. */
    FILE * ulaz;
18     float a[MAKS_DIM][MAKS_DIM];
    int i, j, n, m;

20     /* Ulazna datoteka se otvara za citanje i proverava se da li je
       otvaranje proslo uspesno. */
22     ulaz = fopen("matrica.txt", "r");
    if (ulaz == NULL)
24         greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

26     /* Ucitavaju se dimenzije matrice i vrsi se proverava
       ispravnosti. */
28     fscanf(ulaz, "%d%d", &n, &m);
    if (n <= 0 || n > MAKS_DIM || m <= 0 || m > MAKS_DIM)
30         greska("Greska: neispravne dimenzije matrice.\n");

32     /* Ucitavaju se vrednosti matrice. */
    for (i = 0; i < n; i++)
34         for (j = 0; j < m; j++)
            fscanf(ulaz, "%f", &a[i][j]);

36     /* Za svaku poziciju (i,j) vrsi se proverava traženog uslova. */
    int k, l;
40     float suma = 0;
```

```

42  for (i = 0; i < n; i++)
    {
44      for (j = 0; j < m; j++)
        {
            /* Za poziciju (i,j) racuna se suma suseda. Ona se moze
            46      dobiti kao suma podmatrice 3*3 ciji je gornji levi
            ugao (i-1, j-1), a donji desni (i+1, j+1).
            48      Pri racunanju ove sume treba voditi racuna da se ne izadje
            iz granica originalne matrice. */
            suma = 0;
            for (k = i - 1; k <= i + 1; k++)
            52      {
                for (l = j - 1; l <= j + 1; l++)
                    {
                        /* Ako se nije izaslo iz granica originalne matrice,
                        56      vrednost a[k][l] se dodaje na sumu. */
                        if (k >= 0 && k < n && l >= 0 && l < m)
                            suma += a[k][l];
                        58      }
                    }
                }
            60      }

            /* Kako suma ukljucuje i centralni element (i,j), njega je
            62      potrebno oduzeti jer je potrebno sumirati samo njegove
            susede. */
            64      suma -= a[i][j];

            /* Ako je suma suseda elementa a[i][j] jednaka njegovoj
            66      vrednosti, odgovarajuce pozicije i vrednost elementa
            se ispisuju na standardni izlaz. */
            68      if (a[i][j] == suma)
                printf("(%d, %d, %g)\n", i, j, a[i][j]);
            70      }
        }
    72  }

    /* Zatvara se datoteka. */
    74  fclose(ulaz);
    76  return 0;
    78  }

```

## Rešenje 4.3.17

## Rešenje 4.3.18

```

1  #include <stdio.h>
   #include <stdlib.h>
3  #include <math.h>

5  #define MAKS_TACAKA 50

7  typedef struct

```

## 4 Ulaz i izlaz programa

---

```
{
9   int x,y;
} Tacka;

11
/* Funkcija ispisuje prosledjenu poruku o gresci na standardni
13   izlaz za greske i izlazi iz programa. */
void greska(char* poruka)
15 {
    fprintf(stderr, "%s", poruka);
17   exit(EXIT_FAILURE);
}

19
/* Funkcija racuna rastojanje tacke od koordinatnog pocetka. */
21 double rastojanje_od_koordinatnog_pocetka(const Tacka* a)
{
23   return sqrt(pow(a->x, 2) + pow(a->y, 2));
}

25
int main()
27 {
    /* Deklaracije potrebnih promenljivih. */
29   FILE* ulaz, *izlaz;
    int n, i;
31   Tacka t, maks_t;
    double r, maks_r = -1;

33
    /* Ulazna datoteka se otvara za citanje i proverava se da li je
35     otvaranje proslo uspesno. */
    ulaz = fopen("tacke.txt", "r");
37   if(ulaz == NULL)
        greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

39
    /* Izlazna datoteka se otvara za pisanje i proverava se da li je
41     otvaranje proslo uspesno. */
    izlaz = fopen("rastojanja.txt", "w");
43   if(izlaz == NULL)
        greska("Greska: neuspesno otvaranje izlazne datoteke.\n");

45
    /* Ucitava se broj tacaka i vrsi se provera ispravnosti. */
47   fscanf(ulaz, "%d", &n);
    if(n < 0 || n > MAKS_TACAKA)
49     greska("Greska: neispravan broj tacaka.\n");

51
    /* Iz datoteke se cita tacka po tacka. */
    for(i=0; i<n; i++)
53     {
        fscanf(ulaz, "%d%d", &t.x, &t.y);

55
        /* Racuna se rastojanje tacke t od koordinatnog pocetka. */
57     r = rastojanje_od_koordinatnog_pocetka(&t);

59
        /* Rastojanje se ispisuje u datoteku. */
```



```

        fprintf(izlaz, "%.21f\n", r);
61
        /* Ukoliko je vece od trenutno najveceg rastojanja, azuriraju
63         se vrednosti maksimalnog rastojanja i odgovarajuće tačke. */
        if(r > maks_r)
65         {
            maks_r = r;
67             maks_t = t;
        }
69     }

71     /* Ispis rezultata. */
    printf("Najudaljenija tačka: (%d, %d)\n", maks_t.x, maks_t.y);
73
    /* Zatvara se datoteka. */
75     fclose(ulaz);

77     return 0;
}

```

### Rešenje 4.3.19

Rešenje je analogno rešenju zadatka 4.3.18.

### Rešenje 4.3.20

```

#include <stdio.h>
2  #include <stdlib.h>
#include <string.h>
4
#define MAKS_IME 5
6
typedef struct
8 {
    unsigned int a, b;
10     char ime[MAKS_IME];
} Pravougaonik;
12

/* Funkcija ispisuje prosledjenu poruku o gresci na standardni
14     izlaz za greske i izlazi iz programa. */
void greska(char* poruka)
16 {
    fprintf(stderr, "%s", poruka);
18     exit(EXIT_FAILURE);
}
20

int main(int argc, char *argv[])
22 {
    /* Deklaracije potrebnih promenljivih. */
24     unsigned int max_pov = 0;

```

## 4 Ulaz i izlaz programa

```
Pravougaonik p;

26
/* Proverava se broj argumenata komandne linije. */
28 if (argc != 2)
    greska("Greska: neispravan poziv.\n");

30
/* Ulazna datoteka se otvara za citanje i proverava se da li je
32 otvaranje proslo uspesno. */
FILE *ulaz = fopen(argv[1], "r");
34 if (ulaz == NULL)
    greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

36
/* Iz datoteke se citaju podaci o pravougaonicima. */
38 while (fscanf(ulaz, "%u%u%s", &p.a, &p.b, p.ime) == 3)
{
40     /* Vrsi se provera ispravnosti duzina stranica. */
    if (p.a == 0 || p.b == 0)
42         greska("Greska: duzina stranice ne moze biti 0.\n");

44     if (p.a == p.b)
    {
46         /* Ako je u pitanju kvadrat, njegovo ime se ispisuje na
            standardni izlaz. */
48         printf("%s ", p.ime);
    }
    else
    {
50         /* Ako je u pitanju pravougaonik, njegova površina se poredi
            sa maksimalnom. */
52         if (p.a * p.b > max_pov)
            max_pov = p.a * p.b;
54     }
56 }

58
/* Ukoliko je bilo ucitanih pravougaonika, ispisuje se površina
60 najveceg. */
if (max_pov != 0)
62     printf("Maksimalna površina: %u\n", max_pov);
else
64     printf("\n");

66
/* Zatvara se datoteka. */
fclose(ulaz);
68 return 0;
}
```

### Rešenje 4.3.21

```
1 #include <stdio.h>
  #include <stdlib.h>
3
```

```
5  #define MAKS_STUDENATA 100
6
7  typedef struct
8  {
9      char korisnicko_ime[8];
10     float prosek;
11 } Student;
12
13 int main()
14 {
15     /* Deklaracije potrebnih promenljivih. */
16     FILE * ulaz;
17     Student studenti[MAKS_STUDENATA];
18     int ocena1, ocena2, ocena3, ocena4, ocena5, zbir_ocena;
19     int i = 0, n;
20     float maks_prosek = 0;
21
22     /* Ulazna datoteka se otvara za citanje i proverava se da li je
23        otvaranje proslo uspesno. */
24     ulaz = fopen("studenti.txt", "r");
25     if (ulaz == NULL)
26     {
27         fprintf(stderr, "Greska: neuspesno otvaranje "
28                     "ulazne datoteke.\n");
29         exit(EXIT_FAILURE);
30     }
31
32     /* Citaju se podaci o studentima sve dok se ne dodje do kraja
33        ulaza. */
34     while (fscanf(ulaz, "%s%d%d%d%d", studenti[i].korisnicko_ime,
35                  &ocena1, &ocena2, &ocena3, &ocena4, &ocena5) != EOF)
36     {
37         /* Racuna se prosek trenutnog studenta. */
38         zbir_ocena = ocena1 + ocena2 + ocena3 + ocena4 + ocena5;
39         studenti[i].prosek = zbir_ocena / 5.0;
40
41         /* Azurira se maksimalni prosek. */
42         if (studenti[i].prosek > maks_prosek)
43             maks_prosek = studenti[i].prosek;
44
45         /* Prelazi se na sledeceg studenta. */
46         i++;
47     }
48
49     /* Promenljiva n cuva ukupan broj studenata. */
50     n=i;
51
52     /* Ispis svih studenata sa maksimalnim prosekom. */
53     for(i=0; i<n; i++)
54     {
55         if(studenti[i].prosek == maks_prosek)
56             printf("korisnicko ime: %s, prosek ocena: %.2f\n",
```

## 4 Ulaz i izlaz programa

---

```
        studenti[i].korisnicko_ime, studenti[i].prosek);
57     }

59     /* Zatvara se datoteka. */
    fclose(ulaz);

61     return 0;

63 }
```

### Rešenje 4.3.22

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

4
#define MAKS_PUNO_IME 101
6 #define MAKS_OCENA 10
#define MAKS_STUDENATA 100

8
typedef struct {
10     char puno_ime[MAKS_PUNO_IME];
    int ocene[MAKS_OCENA];
12     int br_ocena;
    float prosek;
14 } Student;

16 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
    izlaz za greske i izlazi iz programa. */
18 void greska(char* poruka)
{
20     fprintf(stderr, "%s", poruka);
    exit(EXIT_FAILURE);
22 }

24 int main(int argc, char **argv)
{
26     /* Deklaracije potrebnih promenljivih. */
    FILE *ulaz;
28     char ime[MAKS_PUNO_IME], prezime[MAKS_PUNO_IME];
    int i = 0, j, ocena, suma_ocena;
30     Student s, maks_s;
    float maks_prosek;

32
    /* Proverava se broj argumenata. */
34     if (argc != 2)
        greska("Greska: neispravan poziv.\n");

36
    /* Ulazna datoteka se otvara za citanje i proverava se da li je
    otvaranje proslo uspesno. */
38     ulaz = fopen(argv[1], "r");
    if (ulaz == NULL)
40
```

```
42     greska("Greska: neuspesno otvaranje ulazne datoteke.\n");
43
44     /* Iz datoteke se ucitavaju podaci o studentima sve dok se ne
45        dodje do kraja ulaza. */
46     while (fscanf(ulaz, "%s%s", ime, prezime) != EOF)
47     {
48         /* Od imena i prezimena se formira puno ime. */
49         strcpy(s.puno_ime, ime);
50         strcat(s.puno_ime, " ");
51         strcat(s.puno_ime, prezime);
52
53         /* Ucitavaju se ocene sve dok se ne ucita broj 0. */
54         j = 0;
55         suma_ocena = 0;
56         while(1)
57         {
58             fscanf(ulaz, "%d", &ocena);
59             if(ocena == 0)
60                 break;
61
62             s.ocene[j] = ocena;
63             suma_ocena += ocena;
64             j++;
65         }
66
67         /* Racuna se prosek ocena. */
68         s.br_ocena = j;
69         s.prosek = (float) suma_ocena / j;
70
71         /* Ukoliko je u pitanju student ciji je prosek veci od trenutno
72            najveceg proseka, pamte se njegovi podaci i azurira se
73            vrednost najveceg proseka. */
74         if(s.prosek > maks_prosek)
75         {
76             maks_prosek = s.prosek;
77             maks_s = s;
78         }
79     }
80
81     /* Ispis podataka o studentu sa najvećim prosekom. */
82     printf("%s ", maks_s.puno_ime);
83     for(i=0; i<maks_s.br_ocena; i++)
84         printf("%d ", maks_s.ocene[i]);
85     printf("%.2f\n", maks_s.prosek);
86
87     /* Zatvara se datoteka. */
88     fclose(ulaz);
89
90     return 0;
91 }
```

### Rešenje 4.3.23

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4 #include <string.h>
5
6 #define MAKS_RAZLOMAKA 100
7
8 typedef struct
9 {
10     int br;
11     int im;
12 } Razlomak;
13
14 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
15    izlaz za greske i izlazi iz programa. */
16 void greska(char* poruka)
17 {
18     fprintf(stderr, "%s", poruka);
19     exit(EXIT_FAILURE);
20 }
21
22 /* Funkcija ucitava razlomke u niz razlomaka i kao povratnu
23    vrednost vraca broj ucitanih razlomaka. */
24 int ucitaj_razlomke(Razlomak niz[], FILE * f)
25 {
26     int i, n;
27     fscanf(f, "%d", &n);
28
29     for (i = 0; i < n; i++)
30     {
31         fscanf(f, "%d %d", &niz[i].br, &niz[i].im);
32         if (niz[i].im == 0)
33             greska("Greska: Imenilac ne moze biti 0.\n");
34     }
35     return n;
36 }
37
38 /* Funkcija racuna razlomak reciprocan razlomku r. */
39 Razlomak reciprocn(const Razlomak* r)
40 {
41     if(r->br == 0)
42         greska("Greska: nije moguće izracunati recipročni razlomak.\n");
43
44     Razlomak r2;
45     r2.im = r->br;
46     r2.br = r->im;
47     return r2;
48 }
49
50 /* Funkcija racuna brojevnu vrednost razlomka r. */
```

```
float vrednost(const Razlomak* r)
52 {
53     return 1.0 * r->br / r->im;
54 }

55 int main(int argc, char *argv[])
56 {
57     /* Deklaracije potrebnih promenljivih. */
58     FILE *ulaz, *izlaz;
59     int i, n, opcija_x = 0, opcija_y = 0;
60     Razlomak razlomci[MAKS_RAZLOMAKA];
61     Razlomak r;

62     /* Proverava se broj argumenata. */
63     if (argc != 4)
64         greska("Greska: neispravan poziv.\n");

65     /* Ulazna datoteka se otvara za citanje i proverava se da li je
66        otvaranje proslo uspesno. */
67     ulaz = fopen(argv[1], "r");
68     if (ulaz == NULL)
69         greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

70     /* Izlazna datoteka se otvara za pisanje i proverava se da li je
71        otvaranje proslo uspesno. */
72     izlaz = fopen(argv[2], "w");
73     if (izlaz == NULL)
74         greska("Greska: neuspesno otvaranje izlazne datoteke.\n");

75     /* Ucitava se zadata opcija i postavlja se vrednost odgovarajuceg
76        indikatora. */
77     if(strcmp(argv[3], "-x") == 0)
78         opcija_x = 1;
79     else if(strcmp(argv[3], "-y") == 0)
80         opcija_y = 1;
81     else if(strcmp(argv[3], "-xy") == 0 || strcmp(argv[3], "-yx") == 0)
82         opcija_x = opcija_y = 1;
83     else
84         greska("Greska: neispravna opcija.\n");

85     /* Ucitavaju se podaci o razlomcima. */
86     n = ucitaj_razlomke(razlomci, ulaz);

87     /* Prolazi se kroz niz razlomaka. */
88     for (i = 0; i < n; i++)
89     {
90         /* Racuna se recipročni razlomak. */
91         r = recipročni(&razlomci[i]);

92         /* U zavisnosti od unetih opcija, vrši se odgovarajući ispis. */
93         if (opcija_x)
94             fprintf(izlaz, "%d/%d ", r.br, r.im);
95     }
```

## 4 Ulaz i izlaz programa

---

```
104     if (opcija_y)
        fprintf(izlaz, "%f ", vrednost(&r));
        fprintf(izlaz, "\n");
106 }

108 /* Zatvaraju se datoteke. */
        fclose(ulaz);
110        fclose(izlaz);

112    return 0;
}
```

### Rešenje 4.3.24

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

4
    #define MAKS_IME 31
6    #define MAKS_AUTOMOBILA 100

8    typedef struct
    {
10        char marka[MAKS_IME];
        char model[MAKS_IME];
12        float cena;
    } Automobil;

14
    /*
16        Struktura Info sadrzi naziv marke automobila, prosek cena za tu
        marku i broj automobila te marke */
18    typedef struct {
        char marka[MAKS_IME];
20        float prosecna_cena;
        int n;
22    } Info;

24    /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
        izlaz za greske i izlazi iz programa. */
26    void greska(char* poruka)
    {
28        fprintf(stderr, "%s", poruka);
        exit(EXIT_FAILURE);
30    }

32    /* Funkcija ucitava informacije o automobilima iz fajla i smesta ih
        u niz struktura. Kao povratnu vrednost vraca broj ucitanih
        automobila. */
34    int ucitaj(FILE * f, Automobil a[])
36    {
        int i, n;
```



```
38     fscanf(f, "%d", &n);
40     if (n <= 0 || n > MAKS_AUTOMOBILA)
41         greska("Greska: neispravan broj automobila.\n");
42
43     for (i = 0; i < n; i++)
44         fscanf(f, "%s %s %f", a[i].marka, a[i].model, &a[i].cena);
45
46     return n;
47 }
48
49 /* Funkcija proverava da li se u nizu sa informacijama o markama
50    nalazi prosledjena marka. Ukoliko se nalazi, vraca odgovarajucu
51    poziciju, a u suprotnom vraca -1. */
52 int sadrzi(Info info[], int n, char marka[])
53 {
54     int i;
55     for (i = 0; i < n; i++)
56         if (strcmp(info[i].marka, marka) == 0)
57             return i;
58
59     return -1;
60 }
61
62 /* Funkcija popunjava niz sa informacijama o markama na osnovu
63    podataka datih u nizu automobila. */
64 void izracunaj_proseke(Automobil a[], int an, Info info[], int *n)
65 {
66     int i, pozicija, j = 0;
67     for(i=0; i < an; i++)
68     {
69         pozicija = sadrzi(info, j, a[i].marka);
70         if(pozicija == -1)
71         {
72             strcpy(info[j].marka, a[i].marka);
73             info[j].prosecna_cena = a[i].cena;
74             info[j].n = 1;
75             j++;
76         }
77         else
78         {
79             info[pozicija].prosecna_cena += a[i].cena;
80             info[pozicija].n += 1;
81         }
82     }
83
84     for(i=0; i<j; i++)
85         info[i].prosecna_cena /= info[i].n;
86
87     *n = j;
88 }
```

## 4 Ulaz i izlaz programa

---

```
90  /* Funkcija ispisuje informacije o prosecnim cenama za svaku
    marku. */
92  void ispisi_informacije(Info info[], int n)
    {
94      int i;
        printf("Informacije o prosečnoj ceni po markama:\n");
96      for (i = 0; i < n; i++)
            printf("%s %.2f\n", info[i].marka, info[i].prosecna_cena);
98  }

100 /* Funkcija ispisuje podatke o automobilima cija je cena manja ili
    jednaka budzetu kojim korisnik raspolaže. */
102 void ispisi_kandidate(Automobil a[], int an, float budzet, Info info
    [],
                                int n)
104 {
    int i, pozicija;
106    printf("Kola u vasem cenovnom rangui:\n");
    for (i = 0; i < an; i++)
108    {
        if (a[i].cena < budzet)
110        {
            pozicija = sadrzi(info, n, a[i].marka);
112            printf("%s %s %.2f\n", a[i].marka, a[i].model,
                                info[pozicija].prosecna_cena);
114        }
    }
116 }

118 int main(int argc, char *argv[])
    {
120    /* Deklaracije potrebnih promenljivih. */
        Automobil automobili[MAKS_AUTOMOBILA];
122    FILE *ulaz;
        char ime_datoteke[MAKS_IME];
124    float granica;
        Info info[MAKS_AUTOMOBILA];
126    int n, info_n;

128    /* Proverava se broj argumenata komandne linije. */
        if (argc != 2)
130        greska("Greska: neispravan poziv.\n");

132    /* Ucitava se budzet. */
        granica = atof(argv[1]);
134

136    /* Ucitava se naziv datoteke. */
        printf("Unesite naziv datoteke: ");
        scanf("%s", ime_datoteke);
138

140    /* Ulazna datoteka se otvara za citanje i proverava se da li je
        otvaranje proslo uspesno. */
```

```

142     ulaz = fopen(ime_datoteke, "r");
143     if (ulaz == NULL)
144         greska("Greska: neuspesno otvaranje ulazne datoteke.\n");
145
146     /* Ucitavaju se podaci o automobilima. */
147     n = ucitaj(ulaz, automobili);
148
149     /* Izracunavaju se proseci za svaku marku. */
150     izracunaj_proseke(automobili, n, info, &info_n);
151
152     /* Ispisuju se podaci za sve marke automobila. */
153     ispisi_informacije(info, info_n);
154
155     /* Ispisuju se podaci o automobilima cija je cena manja ili
156        jednaka granici koju je korisnik uneo. */
157     ispisi_kandidate(automobili, n, granica, info, info_n);
158
159     /* Zatvara se datoteka. */
160     fclose(ulaz);
161
162     return 0;
163 }

```

### Rešenje 4.3.25

```

1  #include <stdio.h>
2  #include <string.h>
3  #include <stdlib.h>
4
5  #define MAKS_LINIJA 81
6
7  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
8     izlaz za greske i izlazi iz programa. */
9  void greska(char* poruka)
10 {
11     fprintf(stderr, "%s", poruka);
12     exit(EXIT_FAILURE);
13 }
14
15 int main(int argc, char *argv[])
16 {
17     /* Deklaracije potrebnih promenljivih. */
18     FILE * ulaz;
19     char linija[MAKS_LINIJA];
20     int k;
21
22     /* Proverava se da li je program ispravno pozvan. */
23     if (argc != 3)
24         greska("Greska: neispravan poziv.");
25
26     /* Otvara se datoteka cije se ime zadaje kao prvi argument

```

## 4 Ulaz i izlaz programa

```
28     komandne linije i proverava se da li je otvaranje proslo
        uspesno. */
30     ulaz = fopen(argv[1], "r");
        if (ulaz == NULL)
            greska("Greska: neuspesno otvaranje ulazne datoteke.\n");
32
34     /* Cita se brojevena vrednost drugog argumenta komandne linije. */
        k = atoi(argv[2]);
36
38     /* Funkcija fgets cita jednu liniju iz datoteke.
        Prima tri argumenta:
        1. Niska u koju ce biti smestena procitana linija
        2. Maksimalna duzina linije
        3. Datoteku iz koje se cita.
        Kada dodje do kraja datoteke, kao povratnu vrednost
        vraca NULL. */
42     while (fgets(linija, MAKS_LINIJA, ulaz) != NULL)
44     {
        /* Na standardni izlaz se ispisuju sve linije iz datoteke
        cija je duzina veca od k. */
46         if (strlen(linija) > k)
48             printf("%s", linija);
        }
50     printf("\n");
52     /* Zatvara se datoteka. */
        fclose(ulaz);
54
56     return 0;
}
```

### Rešenje 4.3.26

```
1  #include <stdio.h>
        #include <stdlib.h>
3  #include <string.h>
5
6  #define MAKS_LINIJA 81
7
8  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
        izlaz za greske i izlazi iz programa. */
9  void greska(char* poruka)
10 {
11     fprintf(stderr, "%s", poruka);
        exit(EXIT_FAILURE);
13 }
15
16 int main(int argc, char *argv[])
17 {
        /* Deklaracije potrebnih promenljivih. */
        char linija[MAKS_LINIJA];
}
```

```

19 char najduza_linija[MAKS_LINIJA];
   int duzina;
21 int maks_duzina;
   FILE *ulaz;

23
   /* Provera broja argumenata komandne linije. */
25 if (argc != 2)
    greska("Greska: neispravan poziv.\n");
27
   /* Ulazna datoteka se otvara za citanje i proverava se da li je
29    otvaranje proslo uspesno. */
   ulaz = fopen(argv[1], "r");
31 if (ulaz == NULL)
    greska("Greska: neuspesno otvaranje ulazne datoteke.\n");
33
   /* Pronalazi se najduza linija u fajlu. */
35 maks_duzina = 0;
   while (fgets(linija, MAKS_LINIJA, ulaz) != NULL)
37 {
    duzina = strlen(linija);
39
    if (duzina > maks_duzina ||
41        (duzina == maks_duzina && strcmp(linija, najduza_linija) < 0))
    {
43        strcpy(najduza_linija, linija);
        maks_duzina = duzina;
45    }
    }
47
   /* Ispis najduze linije na standardni izlaz. */
49 printf("%s", najduza_linija);

51 /* Zatvara se datoteka. */
   fclose(ulaz);
53
   return 0;
55 }

```

### Rešenje 4.3.27

```

#include <stdio.h>
2 #include <stdlib.h>
#include <string.h>
4
#define MAKS_LINIJA 81
6 #define MAKS_REC 31

8 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i izlazi iz programa. */
10 void greska(char* poruka)
{

```

```
12     fprintf(stderr, "%s", poruka);
13     exit(EXIT_FAILURE);
14 }

16 /* Funkcija broji koliko puta se niska t javlja u okviru niske s. */
17 int broj_pojavljivanja(char s[], char t[])
18 {
19     int br = 0, i;
20     int tn = strlen(t);
21     int sn = strlen(s);

22     /* Funkcija strncmp(s,t,n) poredi prvih n karaktera niski s i t.
23      U petlji se vrši poredjenje niske t sa svim podniskama niske
24      s čija je dužina tn.
25      Na primer, ako je s=abcbab, a t=ab, tada je sn=5, a tn=2.
26      za i=0, zove se strncmp("abcbab", "ab", 2) i na taj način se
27      porede "ab" i "ab".
28      za i=1, zove se strncmp("bcbab", "ab", 2) i na taj način se
29      porede "bc" i "ab".
30      ...
31      za i=sn-st=5-2=3, zove se strncmp("ab", "ab", 2) i na taj
32      se porede "ab" i "ab". */
33     for (i = 0; i <= sn-tn; i++)
34         if(strncmp(s + i, t, tn) == 0)
35             br++;

36     return br;
37 }

38
39 int main(int argc, char *argv[])
40 {
41     /* Deklaracije potrebnih promenljivih. */
42     char rec[MAKS_REC];
43     char linija[MAKS_LINIJA];
44     FILE *ulaz, *izlaz;
45     int n, br;

46     /* Provera se broj argumenata komandne linije. */
47     if (argc != 3)
48         greska("Greska: neispravan poziv.");

49     /* Ulazna datoteka se otvara za citanje i proverava se da li je
50      otvaranje proslo uspesno. */
51     ulaz = fopen(argv[1], "r");
52     if (ulaz == NULL)
53         greska("Greska: neuspesno otvaranje ulazne datoteke.\n");

54     /* Izlazna datoteka se otvara za pisanje i proverava se da li je
55      otvaranje proslo uspesno. */
56     izlaz = fopen(argv[2], "w");
57     if (izlaz == NULL)
58         greska("Greska: neuspesno otvaranje izlazne datoteke.\n");
```

```

64  /* Ucitava se broj n i vrsi se provera ispravnosti. */
66  printf("Unesite broj n: ");
68  scanf("%d", &n);
    if (n <= 0)
        greska("Greska: neispravan unos.\n");

70  /* Ucitava se trazena rec. */
72  fscanf(ulaz, "%s", rec);

74  /* Iz ulazne datoteke se cita linija po linija i u izlaznu
    datoteku se upisuju sve linije koje trazenu rec sadrze bar
    n puta. */
76  while (fgets(linija, MAKS_LINIJA, ulaz) != NULL)
78  {
        br = broj_pojavljivanja(linija, rec);
80        if (br >= n)
            fprintf(izlaz, "%d: %s\n", br, linija);
82    }

84  /* Zatvaraju se datoteke. */
86  fclose(ulaz);
    fclose(izlaz);

88  return 0;
}

```

### Rešenje 4.3.28

```

#include <stdio.h>
2  #include <string.h>
    #include <stdlib.h>

4
    #define MAKS_LINIJA 81
6  #define MAKS_NISKA 21

8  /* Funkcija prebrojava koliko linija datoteke ulaz se završava
    niskom s. */
10 int brojLinija(FILE * ulaz, char *s)
    {
12     char linija[MAKS_LINIJA];
        int broj_linija = 0;
14     int duzina_s = strlen(s);
        int duzina_linije;

16
        /* Iz datoteke se cita linija po linija, sve dok se ne dodje
        do kraja datoteke. */
18     while (fgets(linija, MAKS_LINIJA, ulaz) != NULL)
20     {
        /* Racuna se duzina procitane linije. */
22         duzina_linije = strlen(linija);
    }
}

```

```
24      /* Sklanja se znak za novi red sa kraja linije. */
25      if (linija[duzina_linije - 1] == '\n')
26      {
27          linija[duzina_linije - 1] = '\0';
28          duzina_linije--;
29      }
30
31      /* Poredi se kraj linije sa niskom s.
32       * Kraj linije se moze dobiti tako sto se izvrši 'pomeranje' u
33       * desno do kraja linije, a zatim 'pomeranje' u levo onoliko
34       * mesta koliko je dugacka niska s.
35       * Na primer, ako je linija "abcdefghijk", a niska s "ab",
36       * onda sa linija + duzina_linije se vrši pomeranje na karakter
37       * iza karaktera 'k', a sa:
38       * linija + duzina_linije - duzina_s na karakter 'j'.
39       * Ukoliko se funkcija strcmp pozove sa:
40       * strcmp(linija + duzina_linije - duzina_s, s),
41       * vrsice se poredjenje niske "jk" i "ab", sto je i bio cilj. */
42      if (strcmp(linija + duzina_linije - duzina_s, s) == 0)
43          broj_linija++;
44  }
45
46      /* Kao povratna vrednost se vraca broj linija koje se završavaju
47       * niskom s. */
48      return broj_linija;
49  }
50
51  int main()
52  {
53      FILE *ulaz;
54      char s[MAKS_NISKA];
55
56      /* Otvara se datoteka ulaz.txt i proverava se da li je otvaranje
57       * proslo uspesno. */
58      ulaz = fopen("ulaz.txt", "r");
59      if (ulaz == NULL)
60      {
61          fprintf(stderr, "Greska: neuspesno otvaranje ulazne "
62                      "datoteke.\n");
63          exit(EXIT_FAILURE);
64      }
65
66      /* Ucitava se niska s. */
67      printf("Unesite nisku s: ");
68      scanf("%s", s);
69
70      /* Ispis rezultata. */
71      printf("Broj linija: %d\n", brojLinija(ulaz, s));
72
73      /* Zatvara se datoteka. */
74      fclose(ulaz);
```



```
76     return 0;
77 }
```

### Rešenje 4.3.29

```
1  #include <stdio.h>
2  #include <string.h>
3  #include <ctype.h>
4  #include <stdlib.h>
5
6  #define MAKS_LINIJA 81
7
8  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
9     izlaz za greske i izlazi iz programa. */
10 void greska(char* poruka)
11 {
12     fprintf(stderr, "%s", poruka);
13     exit(EXIT_FAILURE);
14 }
15
16 int main(int argc, char *argv[])
17 {
18     char linija[MAKS_LINIJA];
19     FILE * izlaz;
20     int ispisi_v = 0, ispisi_m = 0;
21
22     if (argc > 2)
23         greska("Greska: neispravan poziv.\n");
24
25     if (argc == 1)
26         ispisi_v = ispisi_m = 1;
27     else
28     {
29         /* Funkcija strcasecmp poredi niske ignorisuci razliku izmedju
30            malih i velikih slova. */
31         if (strcasecmp(argv[1], "-v") == 0)
32             ispisi_v = 1;
33         else if (strcasecmp(argv[1], "-m") == 0)
34             ispisi_m = 1;
35         else
36             greska("Greska: neispravna opcija.\n");
37     }
38
39     izlaz = fopen("izlaz.txt", "w");
40     if (izlaz == NULL)
41         greska("Greska: neuspesno otvaranje izlazne datoteke.\n");
42
43     while (fgets(linija, MAKS_LINIJA, stdin) != NULL)
44     {
45         if ((ispisi_m && islower(linija[0])) ||
```

## 4 Ulaz i izlaz programa

---

```
46         (ispisi_v && isupper(linija[0])) ||
          (ispisi_m && ispisi_v))
48         fputs(linija, izlaz);
49     }
50
51     /* Zatvara se datoteka. */
52     fclose(izlaz);
53
54     return 0;
55 }
```

### Rešenje 4.3.30

```
#include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_LINIJA 201
6
7 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
8    izlaz za greske i izlazi iz programa. */
9 void greska(char* poruka)
10 {
11     fprintf(stderr, "%s", poruka);
12     exit(EXIT_FAILURE);
13 }
14
15 int main(int argc, char **argv)
16 {
17     /* Deklaracije potrebnih promenljivih. */
18     int i = 1;
19     char *d1, *d2;
20     FILE *dat1, *dat2;
21     char linija1[MAKS_LINIJA];
22     char linija2[MAKS_LINIJA];
23
24     /* Proverava se broj argumenata komandne linije. */
25     if (argc != 3)
26         greska("Greska: neispravan poziv.\n");
27
28     /* Datoteke ciji su nazivi dati kao argumenti komandne linije
29        se otvaraju za citanje. */
30     dat1 = fopen(argv[1], "r");
31     dat2 = fopen(argv[2], "r");
32
33     /* Proverava se da li su obe datoteke uspesno otvorene. */
34     if (dat1 == NULL || dat2 == NULL)
35         greska("Greska: neuspesno otvaranje datoteke.\n");
36
37     /* Iz obe datoteke se cita prva linija. */
38     d1 = fgets(linija1, MAKS_LINIJA, dat1);
```

```
40     d2 = fgets(linija2, MAKS_LINIJA, dat2);
42     /* Dok god se ne dodje do kraja jedne od datoteka citaju se
43        preostale linije. */
44     while (d1 != NULL && d2 != NULL)
45     {
46         /* Vrsi se poredjenje ucitanih linija. */
47         if (strcmp(linija1, linija2) != 0)
48             printf("%d ", i);
49
50         /* Prelazi se na sledece linije. */
51         d1 = fgets(linija1, MAKS_LINIJA, dat1);
52         d2 = fgets(linija2, MAKS_LINIJA, dat2);
53
54         i++;
55     }
56
57     /* Iz prethodne petlje je moglo da se izadje u 3 slucaja:
58        1. Doslo se do kraja prve datoteke.
59        2. Doslo se do kraja druge datoteke.
60        3. Doslo se do kraja obe datoteke.
61        U slucaju da se desio treci slucaj, nijedna od naredne dve
62        petlje se nece izvrstiti.
63        U prvom slucaju ce se izvrstiti samo prva petlja, a u
64        drugom slucaju druga.
65    */
66
67     /* Ispisuju se preostali redni brojevi linija prve datoteke. */
68     while (d1 != NULL)
69     {
70         printf("%d ", i);
71         d1 = fgets(linija1, MAKS_LINIJA, dat1);
72         i++;
73     }
74
75     /* Ispisuju se preostali redni brojevi linija druge datoteke. */
76     while (d2 != NULL)
77     {
78         printf("%d ", i);
79         d2 = fgets(linija2, MAKS_LINIJA, dat2);
80         i++;
81     }
82
83     /* Zatvaraju se datoteke. */
84     fclose(dat1);
85     fclose(dat2);
86
87     return 0;
88 }
```