

## PROGRAMIRANJE 1



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

# **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

PROGRAMIRANJE 1

Zbirka zadataka

# Sadržaj

<b>1</b>	<b>Uvodni zadaci</b>	<b>1</b>
1.1	Naredba izraza . . . . .	1
1.2	Rešenja . . . . .	11
<b>2</b>	<b>Kontrola toka</b>	<b>29</b>
2.1	Naredbe grananja . . . . .	29
2.2	Rešenja . . . . .	41
2.3	Petlje . . . . .	67
2.4	Rešenja . . . . .	94
2.5	Funkcije . . . . .	153
2.6	Rešenja . . . . .	168
<b>3</b>	<b>Predstavljanje podataka</b>	<b>203</b>
3.1	Nizovi . . . . .	203
3.2	Rešenja . . . . .	220
3.3	Pokazivači . . . . .	261
3.4	Rešenja . . . . .	268
3.5	Niske . . . . .	288
3.6	Rešenja . . . . .	299
3.7	Višedimenzioni nizovi . . . . .	320
3.8	Rešenja . . . . .	334
3.9	Strukture . . . . .	363
3.10	Rešenja . . . . .	375
<b>4</b>	<b>Ulaz i izlaz programa</b>	<b>409</b>
4.1	Datoteke . . . . .	409
4.2	Rešenja . . . . .	424



# 1

## Uvodni zadaci

### 1.1 Naredba izraza

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

*Primer 1*

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

[Rešenje 1.1.1]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.2]

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

## 1 Uvodni zadaci

---

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.3]

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.4]

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

### Primer 1

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

### Primer 1

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

[Rešenje 1.1.5]

**Zadatak 1.1.6** Napisati program koji pomaže kasirki da obračuna kusur koji treba da vrati kupcu. Za unetu cenu artikla, količinu artikla i iznos koji je kupac dao, program treba da ispiše vrednost kusura. NAPOMENA: *Pretpostaviti*



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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.6]

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.7]

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesi dve celobrojne vrednosti: 5 7
pre zamene: x=5, y=7
posle zamene: x=7, y=5
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesi dve celobrojne vrednosti: 237 -592
pre zamene: x=237, y=-592
posle zamene: x=-592, y=237
```

[Rešenje 1.1.8]

**Zadatak 1.1.9** Date su dve 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:
Unesi dve celobrojne vrednosti: 5 7
Nove vrednosti su: a=12, b=-2
```

### Primer 2

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

## 1 Uvodni zadaci

---

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.10]

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.11]

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.12]

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

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

NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.13]

**Zadatak 1.1.14** Napisati program koji ispisuje broj koji se dobija izbacivanjem cifre desetica u unetom prirodnom broju.

*Primer 1*

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

*Primer 2*

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

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

## 1 Uvodni zadaci

---

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.15]

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

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesi 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 decimale. UPUTSTVO: *Jedna milja ima 1.609344 kilometara.*

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.17]

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.18]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.19]

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

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

Pri ispisu vrednost zaokružiti na 4 decimale.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.21]

## 1 Uvodni zadaci

---

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

*Primer 1*

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

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite duzinu poluprecnika kruga: 14.932  
|| Obim: 93.82, površina: 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 decimalne. NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

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

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite duzinu stranice trougla: 2  
|| Obim: 6.00  
|| Površina: 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 decimalne. NAPOMENA: *Pretpostaviti da je unos ispravan.*

*Primer 1*

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

*Primer 2*

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

[Rešenje 1.1.24]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 1.1.26]

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

*Primer 1*

```

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

```

[Rešenje 1.1.27]

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

## 1 Uvodni zadaci

---

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

### Primer 1

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

### Primer 1

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

[Rešenje 1.1.28]

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

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

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.29]

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

### Primer 1

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

### Primer 2

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

[Rešenje 1.1.30]



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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.1.36]

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

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

zaokruženu na dve decimale.

*Primer 1*

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

*Primer 1*

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

[Rešenje 1.1.32]

## 1.2 Rešenja

### Rešenje 1.1.1

```
2  /* Navedeni program definise funkciju koja se zove main. Program
   * moze da definise vise funkcija, ali obavezno mora da definise
   * funkciju koja se zove main i izvršavanje programa uvek pocinje
4  * od te funkcije. Pored naziva, zapis svake funkcije cine i
   * povratna vrednost funkcije (u ovom slucaju int), lista
6  * argumenata koje funkcija koristi (u ovom slucaju funkcija nema
   * argumenata pa se navode samo prazne zagrade, ()) i telo
8  * funkcije koje je ograniceno viticastim zagradama ({ i }). 0
   * ovim pojmovima bice vise reci u narednim poglavljima.

10
   *
12  Unutar tela funkcije navode se naredbe. Unutar navedenog
   * programa postoji jedna naredba koja predstavlja poziv funkcije
   * printf. Funkcija printf služi za ispis teksta na standardni
```

## 1 Uvodni zadaci

---

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

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

28      #include<stdio.h>

30      int main()
32      {
34      /* printf: funkcija pomocu koje se vrsi ispis
Specijalni karakter \n : prelazak u novi red
Svaka naredba zavrшава se karakterom ; */
36      printf("Zdravo svima!\n");

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

### Rešenje 1.1.2

```
#include <stdio.h>

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

10     int n;

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

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

```

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

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

32     return 0;
}

```

### Rešenje 1.1.3

```

#include<stdio.h>

2
int main()
4
{
    /* Promenljive istog tipa mogu se deklarirati jedna za drugom. */
6
    int x, y, rezultat;

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

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

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

18
    /* Mozemo ispisivati direktno vrednost izraza x-y i bez njegovog
        dodeljivanja posebnoj promenljivoj */
20
    printf("%d - %d = %d\n", x, y, x - y);
    printf("%d * %d = %d\n", x, y, x * y);

22
    /* Kada bilo koju aritmeticku operaciju primenimo na dve
        promenljive istog tipa (u ovom slucaju dva celobrojne
        promenljive), rezultat ce biti tog istog tipa. Specijalno, za
        operaciju deljenja: kada operator / primenimo na dva
        celobrojna argumenta x i y, kao rezultat dobijemo 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. Na primer, 7%2 ima vrednost 1 (jer je
        7=3*2+1). Da bismo odstampali karakter %, u naredbi printf
        pisemo %% */
34
    printf("%d %% %d = %d\n", x, y, x % y);
36
}

```

```
38     return 0;
}
```

### Rešenje 1.1.4

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

### Rešenje 1.1.5

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

### Rešenje 1.1.6

```
1  #include <stdio.h>
3  int main()
4  {
5      /* S obzirom da su sve promenljive pozitivini celi brojevi,
6         koristimo tip unsigned int (skraceno unsigned) */
7      unsigned cena;
8      unsigned kolicina;
9      unsigned iznos;
10     unsigned kusur;
11
12     /* Ucitavamo potrebne podatke. Unutar jednog scanf-a mozemo
13        ucitati vise podataka odjednom. Za svaki treba navesti
14        odgovarajuci format za tip podataka koji se unosi (%u za
15        unsigned). */
16     printf("Unesite cenu, kolicinu i iznos: ");
17     scanf("%u%u%u", &cena, &kolicina, &iznos);
18
19     /* Izracunavamo kusur: */
20     kusur = iznos - kolicina * cena;
21
22     /* I ispisujemo trazenu vrednost: */
23     printf("Kusur je %u dinara.\n", kusur);
24
25     return 0;
26 }
```

### Rešenje 1.1.7

```
1  #include <stdio.h>
2
3  int main()
4  {
5
6      unsigned poletanje, poletanje_sat, poletanje_minut;
7      unsigned sletanje, sletanje_sat, sletanje_minut;
8      unsigned duzina, duzina_sat, duzina_minut;
9
10     printf("Unesite vreme poletanja: ");
11     scanf("%u%u", &poletanje_sat, &poletanje_minut);
12
13
14     printf("Unesite vreme sletanja: ");
15     scanf("%u%u", &sletanje_sat, &sletanje_minut);
16
17     /* Pretvoricemo i vreme poletanja i vreme sletanja u sekunde */
18     poletanje = poletanje_sat * 3600 + poletanje_minut * 60;
19     sletanje = sletanje_sat * 3600 + sletanje_minut * 60;
20
21     /* I izracunati razliku u sekundama */
22     duzina = sletanje - poletanje;
23
24     /* Izdvajamo broj sati i broj minuta. */
25     duzina_sat = duzina / 3600;
26     duzina_minut = (duzina % 3600) / 60;
27
28     /* I ispisujemo rezultat */
29     printf("Duzina trajanja leta je %u h i %u min\n", duzina_sat,
30           duzina_minut);
31
32     return 0;
33 }
```

### Rešenje 1.1.8

```
1  #include<stdio.h>
2
3  int main()
4  {
5      int x, y;
6      int p;
7
8      printf("Unesi dve celobrojne vrednosti:");
9      scanf("%d%d", &x, &y);
10
11     printf("pre zamene: x=%d, y=%d\n", x, y);
12
13     /* Pomocna promenljiva p je potrebna da sacuva vrednost
14        promenljive x pre nego sto se ona izmeni i dobije vrednost
```

## 1 Uvodni zadaci

---

```
    promenljive y. */
16  p = x;
    x = y;
18  y = p;
    printf("posle zamene: x=%d, y=%d\n", x, y);
20
    return 0;
22 }
```

### Rešenje 1.1.10

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* S obzirom da broj treba da bude pozitivan, koristimo tip
        unsigned. */
6      unsigned x;
7
8      /* Promenljive koje cuvaju cifre treba da budu najmanjeg
        celobrojnog tipa jer nece sadrzati druge vrednosti osim
11     jednocifrenih celih brojeva. Zbog toga za njih biramo tip
        char. */
12     char cifra_jedinice;
13     char cifra_desetice;
14     char cifra_stotine;
15
16     printf("Unesi trocifreni broj:");
17     scanf("%u", &x);
18
19     /* Na primer, neka je uneti broj 374. Potrebno je da koriscenjem
        racunskih operacija za rad sa celim brojevima pristupimo
21     njegovoj cifri jedinice, cifri desetice i cifri stotine.
22
23     Prisetimo najpre sledece:  $374/10 = 37$   $374\%10 = 4$  Dakle,
        operacijama celobrojnog deljenja i ostatka pri deljenju mozemo
25     iz svakog broja izdvojiti njegovu poslednju cifru (u ovom
        slucaju 4) i broj sastavljen od svih cifara osim poslednje (u
27     ovom slucaju 37).
28
29     Cifri jedinice sada lako pristupamo koriscenjem ostatka pri
        deljenju sa 10. Ona iznosi upravo 4.
31
32     Pri trazenju cifre desetice mozemo ponovo primeniti princip
        izdvajanja poslednje cifre kao ostatka pri deljenju sa 10.
33     Razlika je sto ne mozemo desetice izdvojiti ako primenimo %10
        na 374 (time dobijamo 4), vec %10 primenjujemo na 37, pri cemu
35     37 dobijamo kao ceo deo pri deljenju broja 374 brojem 10.
        Dakle, cifru desetice dobijamo kao  $(374/10)\%10$ .
37
38     S obzirom da znamo da je u pitanju trocifreni broj, cifru
39
```

```

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

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

57     /* 2. nacin, bez uvodjenja dodatnih promenljivih cifra_jedinice,
        cifra_desetice i cifra_stotine:

59         printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10,
61         x/100); */
        return 0;
63 }

```

### Rešenje 1.1.11

```

1  #include <stdio.h>

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

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

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

```

## 1 Uvodni zadaci

---

```

    x = x % 1000;
27  printf("%u*500 +", x / 500);
    x = x % 500;
29  printf("%u*200 +", x / 200);
    x = x % 200;
31  printf("%u*100 +", x / 100);
    x = x % 100;
33  printf("%u*50 +", x / 50);
    x = x % 50;
35  printf("%u*20 +", x / 20);
    x = x % 20;
37  printf("%u*10 +", x / 10);
    x = x % 10;
39  printf("%u*1\n", x);

41  return 0;
}
```

### Rešenje 1.1.12

```

1  #include <stdio.h>

3  int main()
{
5      unsigned x;
      unsigned obrnuto_x;
7      char cifra_jedinice;
      char cifra_desetice;
9      char cifra_stotine;

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

13

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

17

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

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

23     return 0;
}
```

### Rešenje 1.1.13

```

1  #include <stdio.h>
```



```

3 int main()
4 {
5
6     unsigned n, broj_obrnuto, broj_zamena;
7     char j, d, s, h;
8     int proizvod_cifara, razlika_cifara, suma_kvadrata;
9
10    /* Ucitavamo vrednost sa ulaza */
11    printf("Unesite cetvorocifreni broj: ");
12    scanf("%u", &n);
13
14    /* Izdvajamo cifre broja i to redom: j -jedinice, d - desetice,
15       s - stotine i h - hiljade */
16    j = n % 10;
17    d = (n / 10) % 10;
18    s = (n / 100) % 10;
19    h = n / 1000;
20
21    /* Izracunavamo proizvod cifara */
22    proizvod_cifara = j * d * s * h;
23    printf("Proizvod cifara: %d\n", proizvod_cifara);
24
25    /* Izracunavamo razliku sume krajnjih i srednjih cifara */
26    razlika_cifara = (h + j) - (s + d);
27    printf("Razlika sume krajnjih i srednjih: %d\n", razlika_cifara);
28
29    /* Izracunavamo sumu kvadrata cifara */
30    suma_kvadrata = j * j + d * d + s * s + h * h;
31    printf("Suma kvadrata cifara: %d\n", suma_kvadrata);
32
33    /* Odredjujemo broj zapisan istim ciframa ali u obrnutom
34       redosledu */
35    broj_obrnuto = j * 1000 + d * 100 + s * 10 + h;
36    printf("Broj u obrnutom poretku: %u\n", broj_obrnuto);
37
38    /* Odredjujemo broj u kojem su cifra jedinica i cifra stotina
39       zamenile mesta */
40    broj_zamena = h * 1000 + j * 100 + d * 10 + s;
41    printf("Broj sa zamenjenom cifrom jedinica i stotina: %u\n",
42           broj_zamena);
43
44    return 0;
45 }

```

### Rešenje 1.1.15

```

#include <stdio.h>

2 int main()
3 {
4

```

## 1 Uvodni zadaci

---

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

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

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

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

20
      /* Kada levi deo pomnozimo sa 100 000, dobijamo 1 200 000 Kada m
22       pomnozimo sa 1000, dobijamo 67 000 Dobijene vrednosti saberemo
       sa desnim delom 345 ----- Konacan rezultat: 1 267 345 */
24     novibroj = levi * 100000 + m * 1000 + desni;

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

28     return 0;
  }
```

### Rešenje 1.1.16

```
1  #include <stdio.h>

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

9      printf("Unesi broj inca: ");

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

13
      /* 1 inch = 2.54 cm */
15     cm = in * 2.54;

17     /* "%.2f" - ispis realne promenljive na 4 decimale */
      printf("%.2f in = %.2f cm\n", in, cm);

19     return 0;

21  }
```

Rešenje 1.1.17      Zadatak se rešava analogno zadatku 1.1.16.

Rešenje 1.1.18      Zadatak se rešava analogno zadatku 1.1.16.

Rešenje 1.1.19      Zadatak se rešava analogno zadatku 1.1.16.

Rešenje 1.1.21

```
#include <stdio.h>

2
int main()
4 {
    float a, b;
6     float obim, površina;

    /* Učitavamo potrebne podatke */
8     printf("Unesite dužine stranica pravougaonika: ");
10    scanf("%f%f", &a, &b);

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

    /* Površina */
14    površina = a * b;

    /* Ispisujemo tražene vrednosti */
18    printf("Obim: %.2f\n", obim);
20    printf("Površina: %.2f\n", površina);

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

Rešenje 1.1.22

```
1 #include <stdio.h>
2 #include <math.h>
3
4 /* Zaglavlje math.h sadrži deklaracije velikog broja matematičkih
5    funkcija i konstanti. U ovom zadatku se koristi zbog konstante
6    pi (M_PI)
7
8    Ukoliko se koristi i neka funkcija matematičke biblioteke, za
9    prevodjenje je neophodno uključiti opciju -lm npr. gcc primer.c
10   -lm */
11 int main()
12 {
13     float r;
```

## 1 Uvodni zadaci

---

```
float O;  
15 float P;  
printf("Unesite duzinu poluprecnika kruga:");  
17 scanf("%f", &r);  
  
19 O = 2 * r * M_PI;  
P = r * r * M_PI;  
  
21 printf("Obim: %.2f, površina: %.2f\n", O, P);  
23 return 0;  
25 }
```

### Rešenje 1.1.23

```
#include <stdio.h>  
2 #include <math.h>  
  
4 int main()  
{  
6 float a;  
float P, O;  
  
8 printf("Unesi duzinu stranice trougla:");  
10 scanf("%f", &a);  
  
12 O = 3 * a;  
P = (a * a * sqrt(3)) / 4;  
  
14 printf("Obim: %.2f\n", O);  
16 printf("Površina: %.2f\n", P);  
  
18 return 0;  
}
```

### Rešenje 1.1.24

```
#include <stdio.h>  
2 #include <math.h>  
  
4 int main()  
{  
6 float a, b, c;  
float obim, s, površina;  
  
8 /* Ucitavamo potrebne podatke */  
10 printf("Unesite duzine stranica trougla: ");  
scanf("%f%f%f", &a, &b, &c);  
12 }
```

```

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

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

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

24  return 0;
}

```

### Rešenje 1.1.26

```

1  #include<stdio.h>

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

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

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

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

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

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

33     printf("Aritmeticka sredina unetih brojeva je %.2f\n", as);
        return 0;
35 }

```

### Rešenje 1.1.27

```
1 #include <stdio.h>
3 int main()
4 {
5     unsigned duzina, sirina, visina;
6     unsigned cena;
7     float površina_za_krecenje;
8     float ukupna_cena;
9
10    /* Ucitavamo duzinu, sirinu i visinu sobe */
11    printf("Unesite dimenzije sobe: ");
12    scanf("%u%u%u", &duzina, &sirina, &visina);
13
14    /* Ucitavamo cenu krecenja */
15    printf("Unesite cenu po m2: ");
16    scanf("%u", &cena);
17
18    /* Povrsina za krecenje odgovara površini kvadra - bez poda jer
19       se on ne kreci */
20    površina_za_krecenje = 0.8 * (duzina * sirina +
21                                  2 * duzina * visina +
22                                  2 * sirina * visina);
23    ukupna_cena = površina_za_krecenje * cena;
24
25    /* Ispisujemo trazene podatke */
26    printf("Moler treba da okreći %.2f m2\n", površina_za_krecenje);
27
28    printf("Cena krecenja je %.2f\n", ukupna_cena);
29
30    /* Završavamo sa programom */
31    return 0;
32 }
```

### Rešenje 1.1.28

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     unsigned x, p;
7     char c;
8     unsigned levo, desno;
9     unsigned novo_x;
```

```

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

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

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

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

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

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

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

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

    /* Završavamo sa programom */
    return 0;
}

```

### Rešenje 1.1.29

```

1  #include <stdio.h>

```

## 1 Uvodni zadaci

---

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

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

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

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

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

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

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

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

31     return 0;
}
```

### Rešenje 1.1.30

```
1 #include <stdio.h>

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

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

11        izraz1 ? izraz2 : izraz3;

13        Izraz izraz1 se izracunava prvi. Ako je njegova vrednost
14        razlicita od nule (tj. ako ima istinitosnu vrednost tacno),
15        onda se izracunava vrednost izraza izraz2 i to je vrednost
16        citavog uslovnog izraza. U suprotnom, izracunava se vrednost
17        izraz3 i to je vrednost citavog uslovnog izraza

18        */
}
```



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

Rešenje [2.1.36](#)      Zadatak se rešava analogno zadatku [2.1.36](#)

Rešenje [1.1.32](#)

```
1  #include <stdio.h>
3  int main()
4  {
5      float a, b, rez;
6      float min, max;
7      printf("Unesite dva realna broja:");
8      scanf("%f%f", &a, &b);
9
10     /* Odredjujemo minimalnu i maksimalnu vrednost unetih brojeva */
11     min = (a < b) ? a : b;
12     max = (a > b) ? a : b;
13
14     /* Racunamo vrednost promenljive rez */
15     rez = (min + 0.5) / (1 + max * max);
16
17     printf("Rezultat je %.2f\n", rez);
18
19     return 0;
20 }
```



## 2

# Kontrola toka

## 2.1 Naredbe grananja

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva cela broja: 5 18  
| Minimum je 5.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva cela broja: 43 -16  
| Minimum je -16.
```

[Rešenje 2.1.1]

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva cela broja: 141 67  
| Maksimum je 141.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite dva cela broja: -893 -54  
| Maksimum je -54.
```

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

## 2 Kontrola toka

---

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan broj: 7.42  
|| Njegova apsolutna vrednost je: 7.42
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan broj: -562.428  
|| Njegova apsolutna vrednost je: 562.43
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan broj: 0  
|| Njegova apsolutna vrednost je: 0.00
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan broj: 52  
|| Njegova apsolutna vrednost je: 52.00
```

[Rešenje 2.1.3]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan ceo broj: 0  
|| Nedozvoljeno deljenje nulom.
```

### Primer 4

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

[Rešenje 2.1.4]

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

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cela broja: -15 81 0  
|| Suma unetih pozitivnih brojeva: 81
```

### Primer 3

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

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cela broja: 16 2 576  
|| Suma unetih pozitivnih brojeva: 594
```

[Rešenje 2.1.5]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 2.1.6]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 2.1.7]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite vreme (broj sati u intervalu [0,24),
broj minuta u intervalu [0,60)): 24 20
Neispravan unos.
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite vreme (broj sati u intervalu [0,24),
broj minuta u intervalu [0,60)): 14 0
Do ponoci je ostalo 10 sati i 0 minuta.
```

[Rešenje 2.1.8]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: A
Uneti karakter: A, njegov ASCII kod: 65
odgovarajuće malo slovo: a, njegov ASCII kod: 97
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: v
Uneti karakter: v, njegov ASCII kod: 118
odgovarajuće veliko slovo: V, njegov ASCII kod: 86
```

[Rešenje 2.1.9]

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

### Primer 1

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

### Primer 2

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

[Rešenje 2.1.10]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.1.11]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: B6(vV  
|| 1
```

[Rešenje 2.1.12]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.1.13]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.1.14]

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

*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: -1234  
|| Proizvod parnih cifara: 8
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 288  
|| Broj nije četvorocifren!
```

[Rešenje 2.1.15]

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

*Primer 1*

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

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 247  
|| Broj nije četvorocifren!
```



### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 1192
9112
```

### Primer 4

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

[Rešenje 2.1.16]

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

[Rešenje 2.1.17]

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

**Zadatak 2.1.19** Napisati program za rad sa intervalima. Za dva intervala realne prave  $[a_1, b_1]$  i  $[a_2, b_2]$ , program treba da odredi:

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

### Primer 1

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

### Primer 2

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

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite jedan ceo broj: 17
Broj je veci od nule.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite jedan ceo broj: 0
Broj je jednak nuli.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan ceo broj: -586  
|| Broj je manji od nule.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan ceo broj: 62  
|| Broj je veci od nule.
```

[Rešenje 2.1.20]

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

### Primer 1

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

### Primer 2

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

[Rešenje 2.1.21]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite cetvorocifreni broj: 88  
|| Uneti broj nije cetvorocifren.
```

[Rešenje 2.1.22]

**Zadatak 2.1.23** Napisati program koji učitava karakter i:

- a) ako je  $c$  malo slovo, ispisuje odgovarajuće veliko
- b) ako je  $c$  veliko slovo, ispisuje odgovarajuće malo

- c) ako je  $c$  cifra, ispisuje poruku *cifra*  
d) u ostalim slučajevima, ispisuje karakter  $c$  između dve zvezdice.

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karakter: o
|| 0
```

[Rešenje 2.1.23]

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

### Primer 1

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

### Primer 2

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

[Rešenje 2.1.24]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 2.1.25]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.1.26]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.1.27]

**Zadatak 2.1.28** 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 pogrešnog unosa ispisati odgovarajuću poruku.

*Primer 1*

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

*Primer 2*

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

### Primer 3

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

### Primer 4

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

[Rešenje [2.1.28](#)]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje [2.1.29](#)]

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

[Rešenje [2.1.30](#)]

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

### Primer 1

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

### Primer 2

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

[Rešenje [2.1.31](#)]

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

### Primer 1

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

### Primer 2

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

[Rešenje 2.1.32]

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

### Primer 1

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

### Primer 2

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

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

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

Program treba da sračuna vrednost izraza  $P \text{ op1 } Q \text{ op2 } R$  i da ga ispiše.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite tri cela broja: 0 1 2
|| Unesite dva karaktera cela broja: k m
|| 1
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite tri cela broja: -3 -1 0
|| Unesite dva karaktera cela broja: d k
|| 0
```

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

- ukoliko je uneti karakter  $k$  - proverava da li su date tačke temena pravougona čije su stranice paralelne koordinatnim osama i u slučaju da jesu, ispisuje vrednost obima datog pravougona. Možemo podrazumevati da će

korisnik koordinate tačaka unosi redom  $A, B, C, D$ , pri čemu  $ABCD$  opisuje pravougaonik čije su stranice  $AB, BC, CD, DA$ , a dijagonale  $AC$  i  $BD$ . Na primer, tačke  $(1, 1), (2, 1), (2, 2), (1, 2)$  čine pravougaonik čije su stranice paralelne koordinatnim osama i čiji je obim 4 a tačke  $(1, 1), (2, 2), (3, 3), (4, 4)$  ne čine pravougaonik.

- ukoliko je uneti karakter  $h$  - proverava da li su unete tačke kolinearne i ukoliko jesu, ispisuje jednačinu prave kojoj pripadaju. Na primer, tačke  $(1, 2), (2, 3), (3, 4), (4, 5)$  su kolinearne i pripadaju pravoj  $y = x + 1$ , tačke  $(1, 1), (1, 2), (1, 3), (1, 4)$  su kolinearne i pripadaju pravoj  $x = 1$ , a tačke  $(1, 1), (2, 1), (2, 2), (1, 2)$  nisu kolinearne.
- ukoliko je uneti karakter  $j$  - Kramerovim pravilom proverava da li je sistem jednačina  $x_1 * p + x_2 * q = x_4 - x_3, y_1 * p + y_2 * q = y_4 - y_3$  određen, neodređen ili nema rešenja, i u slučaju da je određen ispisuje rešenja.

[Rešenje 2.1.35]

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

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

## 2.2 Rešenja

### Rešenje 2.1.1

```

1 #include <stdio.h>
2
3 int main()
4 {
5     int a, b, min;
6     printf("Unesite dva cela broja: ");
7     scanf("%d%d", &a, &b);
8
9     /* Promenljiva min dobija vrednost promenljive a. */
10    min = a;

```

## 2 Kontrola toka

---

```
11  /* Ako je b<a, promenljiva min ce promeniti vrednost tj. bice joj
13     dodeljena vrednost promenljive b. U suprotnom, vrednost ostaje
        ista. */
15
16     if (b < a)
17         min = b;
18
19     printf("Minimum je %d\n", min);
20
21     return 0;
22 }
```

Rešenje 2.1.2      Rešenje je analogno rešenju broj 2.1.1.

### Rešenje 2.1.3

```
1  #include<stdio.h>
2
3  int main()
4  {
5      float x;
6      float apsolutno_x;
7
8      printf("Unesite jedan realan broj:");
9      scanf("%f", &x);
10
11     apsolutno_x = x;
12     if (x < 0)
13         apsolutno_x = -x;
14
15     printf("Njegova apsolutna vrednost je %.2f\n", apsolutno_x);
16
17     /* 2. nacin, pomocu funkcije fabs za koju je neophodno ukljuciti
        zaglavlje math.h: apsolutno_x=fabs(x); */
18     return 0;
19 }
```

### Rešenje 2.1.4

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int x;
6      float rx;
7
8      printf("Unesite jedan ceo broj:");
```



```

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

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

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

23

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

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

```

### Rešenje 2.1.5

```

1  #include<stdio.h>

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

9

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

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

17

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

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

```

## 2 Kontrola toka

---

```
23     if (b > 0)
24         s += b;
25
26     if (c > 0)
27         s += c;
28
29     printf("Suma unetih pozitivnih brojeva: %d\n", s);
30     return 0;
31 }
```

### Rešenje 2.1.6

```
#include <stdio.h>
2
3 int main()
4 {
5     unsigned a, b, c;
6     unsigned min;
7     unsigned cena_bez_popusta, cena_sa_popustom;
8
9     printf("Unesite cene tri artikla:");
10    scanf("%u%u%u", &a, &b, &c);
11
12    /*
13     * Racunamo minimum tri broja. Dodeljujemo promenljivoj min
14     * vrednost prvog broja. */
15    min = a;
16
17    /*
18     * Ako je drugi broj manji od minimuma, to znaci da promenljiva
19     * min ne sadrzi najmanji broj. Dodeljujemo joj vrednost drugog
20     * broja. */
21    if (min > b)
22        min = b;
23
24    /*
25     * Ako je treci broj manji od minimuma, to znaci da promenljiva
26     * min ne sadrzi najmanji broj. Dodeljujemo joj vrednost treceg
27     * broja. */
28    if (min > c)
29        min = c;
30
31    cena_bez_popusta = a + b + c;
32    cena_sa_popustom = cena_bez_popusta - min + 1;
33
34    printf("Cena sa popustom: %u\nUsteda: %u\n",
35           cena_sa_popustom, cena_bez_popusta - cena_sa_popustom);
36
37    return 0;
38 }
```

## Rešenje 2.1.7

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int n, j, d, s, h, max;
7
8      /* Ucitavamo broj */
9      printf("Unesite cetvorocifreni broj: ");
10     scanf("%d", &n);
11
12     /* Za slucaj da je broj negativan, uzimamo apsolutnu vrednost
13        unetog broja */
14     n = abs(n);
15
16     /* Ako uneti broj nije cetvorocifren, ispisujemo poruku o gresci
17        i prekidamo izvršavanje programa. */
18     if (n < 1000 || n > 9999) {
19         printf("Greska: Niste uneli cetvorocifren broj!\n");
20         return -1;
21     }
22
23     /* Ako je broj cetvorocifren, izdvajamo cifre broja: j - jedinice,
24        d - desetice, s - stotine i h - hiljade */
25     j = n % 10;
26     d = (n / 10) % 10;
27     s = (n / 100) % 10;
28     h = n / 1000;
29
30     /* Odredjujemo maksimalnu cifru */
31     max = j;
32
33     if (d > max)
34         max = d;
35
36     if (s > max)
37         max = s;
38
39     if (h > max)
40         max = h;
41
42     /* II nacin: if(j>d && j>s && j>h) max=j; if(d>j && d>s && d>h)
43        max=d; if(s>j && s>d && s>h) max=s; if(h>j && h>d && h>s)
44        max=h; */
45
46     /* Ispisujemo rezultat */
47     printf("Najveca cifra je: %d\n", max);
```

```
49     return 0;
    }
```

### Rešenje 2.1.8

```
#include<stdio.h>

2
int main()
4 {
    int sati;
    int minuti;
    int preostali_sati;
    int preostali_minuti;

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

14
    printf("Unesite vreme (broj sati u intervalu [0,24],\n"
16         "broj minuta u intervalu [0,60]):");
    scanf("%d%d", &sati, &minuti);

18
    /* U slucaju da je unos neispravan, ispisujemo poruku o gresci i
    prekidamo dalje izvorsavanje programa.

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

28
30     if (sati > 24 || sati < 0 || minuti > 59 || minuti < 0) {
32         printf("Neispravan unos.\n");
        return -1;
34     }

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

38
    if (preostali_minuti == 60) {
40         preostali_sati++;
        preostali_minuti = 0;
42     }

44
    /* Ukoliko naredbu printf zelimo da napisemo u dva reda i pritom
    ne prekidamo deo pod navodnicima, to mozemo uraditi bez
```

```

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

50     return 0;
}

```

### Rešenje 2.1.9

```

#include <stdio.h>

2
int main()
4 {
    char c;
6     printf("Unesite jedan karakter:");
    scanf("%c", &c);

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

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

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

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

22    return 0;
24 }

```

### Rešenje 2.1.10

```

1  #include <stdio.h>

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

7

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

11

        /* Proveravamo da li je prvi karakter malo slovo */
13        if (c1 >= 'a' && c1 <= 'z') {

```

## 2 Kontrola toka

```
15     /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
17 }
19 /* Proveravamo da li je drugi karakter malo slovo */
    if (c2 >= 'a' && c2 <= 'z') {
21         /* I ako jeste, uvecavamo broj malih slova */
        broj_malih_slova++;
23     }
25 /* Proveravamo da li je treci karakter malo slovo */
    if (c3 >= 'a' && c3 <= 'z') {
27         /* I ako jeste, uvecavamo broj malih slova */
        broj_malih_slova++;
29     }
31 /* Proveravamo da li je cetvrti karakter malo slovo */
    if (c4 >= 'a' && c4 <= 'z') {
33         /* I ako jeste, uvecavamo broj malih slova */
        broj_malih_slova++;
35     }
37 /* Proveravamo da li je peti karakter malo slovo */
    if (c5 >= 'a' && c5 <= 'z') {
39         /* I ako jeste, uvecavamo broj malih slova */
        broj_malih_slova++;
41     }
43 /* Ispisujemo rezultat */
    printf("Broj malih slova: %d\n", broj_malih_slova);
45     return 0;
}
```

### Rešenje 2.1.11

```
1 #include <stdio.h>
   #include <ctype.h>
3
4 int main()
5 {
6     /* Broj pojavljivanja slova a i A se inicijalizuje na 0 */
7     int br_a = 0;
8
9     /* Funkcija getchar ucitava jedan karakter. Njena povratna
       vrednost je ASCII kod ucitanog karaktera.
10
11     Funkcija tolower za dati karakter vraca: - odgovarajuće malo
12     slovo, ako je dati karakter veliko slovo - taj isti karakter,
13     u suprotnom 0va funkcija je definisana u biblioteci ctype.h
14
15 }
```

```

17     U slucaju da je uslov ispunjen, uvecavamo brojac br_a za jedan
18     pomocu operatora inkrementacije ++ */
19     if (tolower(getchar()) == 'a')
20         br_a++;
21     if (tolower(getchar()) == 'a')
22         br_a++;
23     if (tolower(getchar()) == 'a')
24         br_a++;
25     if (tolower(getchar()) == 'a')
26         br_a++;
27     if (tolower(getchar()) == 'a')
28         br_a++;
29     printf("%d\n", br_a);
30
31     return 0;
32 }

```

### Rešenje 2.1.12

```

1  #include <stdio.h>
2  #include <ctype.h>
3
4  int main()
5  {
6      int br_cif = 0;
7
8      /* Funkcija isdigit vraca 1 ako je dati karakter cifra i 0 u
9       suprotnom. Nalazi se u biblioteci ctype.h. */
10     if (isdigit(getchar()))
11         br_cif++;
12     if (isdigit(getchar()))
13         br_cif++;
14     if (isdigit(getchar()))
15         br_cif++;
16     if (isdigit(getchar()))
17         br_cif++;
18     if (isdigit(getchar()))
19         br_cif++;
20
21     printf("%d\n", br_cif);
22
23     return 0;
24 }

```

### Rešenje 2.1.13

```

1  #include <stdio.h>

```

## 2 Kontrola toka

---

```
3 int main()
4 {
5     int x;
6     printf("Unesite godinu:");
7     scanf("%d", &x);
8
9     if ((x % 4 == 0 && x % 100 != 0) || x % 400 == 0)
10         printf("Godina je prestupna\n");
11     else
12         printf("Godina nije prestupna\n");
13
14     return 0;
15 }
```

### Rešenje 2.1.14

```
1 #include <stdio.h>
2 #include <stdlib.h>          /* abs */
3
4 int main()
5 {
6     int n, j, d, s;
7
8     /* Ucitavamo broj */
9     printf("Unesite broj: ");
10    scanf("%d", &n);
11
12    /* Uzimamo apsolutnu vrednost broja za slucaj da je uneti broj
13       negativan */
14    n = abs(n);
15
16    /* Ako broj nije trocifren, izdajemo poruku o gresci i prekidamo
17       dalje izvršavanje programa */
18    if (n < 100 || n > 999) {
19        printf("Greska: Niste uneli trocifren broj!\n");
20        return -1;
21    }
22
23    /*
24       Izdvajamo cifre broja: j -jedinice, d - desetice, s - stotine */
25    j = n % 10;
26    d = (n / 10) % 10;
27    s = n / 100;
28
29    /* Proveravamo da li je broj Armstrongov */
30    if (n == j * j * j + d * d * d + s * s * s)
31        printf("Broj je Armstrongov.\n");
32    else
33        printf("Broj nije Armstrongov.\n");
34
35    return 0;
36 }
```



}

## Rešenje 2.1.15

```
1  #include <stdio.h>
3  int main()
4  {
5      int n, j, d, s, h;
6      int broj_parnih, proizvod_parnih;
7
8      printf("Unesite cetvorocifreni broj: ");
9      scanf("%d", &n);
10
11     n = abs(n);
12
13     if (n < 1000 || n > 9999) {
14         printf("Broj nije cetvorocifren.\n");
15         return -1;
16     }
17
18     /* Izdvajamo cifre broja: j -jedinice, d - desetice, s - stotine
19        i h - hiljade */
20     j = n % 10;
21     d = (n / 10) % 10;
22     s = (n / 100) % 10;
23     h = n / 1000;
24
25     /* Inicijalizujemo broj parnih cifara na 0 */
26     broj_parnih = 0;
27     /* Postavljamo proizvod parnih cifara na 1 (neutral za mnozenje) */
28     proizvod_parnih = 1;
29
30     /* Proveravamo da li je cifra jedinica parna */
31     if (j % 2 == 0) {
32         proizvod_parnih = proizvod_parnih * j;
33         broj_parnih++;
34     }
35
36     /* Proveravamo da li je cifra desetica parna */
37     if (d % 2 == 0) {
38         proizvod_parnih = proizvod_parnih * d;
39         broj_parnih++;
40     }
41
42     /* Proveravamo da li je cifra stotina parna */
43     if (s % 2 == 0) {
44         proizvod_parnih = proizvod_parnih * s;
45         broj_parnih++;
46     }
47 }
```

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

53
/* Proveravamo da li u zapisu broja ima parnih cifara i
55     ispisujemo rezultat */
if (broj_parnih == 0) {
57     printf("Nema parnih cifara.\n");
} else {
59     printf("Proizvod parnih cifara: %d\n", proizvod_parnih);
}

61     return 0;
63 }
```

### Rešenje 2.1.16

```
1 #include <stdio.h>

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

7
    // Da bismo lakse odredili da li je cetvorocifren
9     int absBroj = broj < 0 ? -broj : broj;
    if (absBroj <= 999 || absBroj >= 10000) {
11         printf("Broj nije cetvorocifren.");
        return -1;
13     }

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

19
    int max = a, min = a;
21     // cuvamo i stepen da bismo lakse zamenili cifre
    /* Ideja: 4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1, ^ ^
23         odnosno oduzemo 100 i dodamo 900 */
    int stepenMax = 1, stepenMin = 1;

25
    if (b > max) {
27         max = b;
        stepenMax = 10;
29     }
    if (b < min) {
31         min = b;
        stepenMin = 10;
    }
```

```

33     }

35     if (c > max) {
36         max = c;
37         stepenMax = 100;
38     }
39     if (c < min) {
40         min = c;
41         stepenMin = 100;
42     }
43
44     if (d > max) {
45         max = d;
46         stepenMax = 1000;
47     }
48     if (d < min) {
49         min = d;
50         stepenMin = 1000;
51     }
52
53     int rez;
54     /* Ideja: 4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1, ^ ^
55        odnosno oduzemo 100 i dodamo 900 */
56     if (broj > 0)
57         rez = broj - max * stepenMax + min * stepenMax
58             - min * stepenMin + max * stepenMin;
59     else
60         rez = broj + max * stepenMax - min * stepenMax
61             + min * stepenMin - max * stepenMin;
62
63     printf("%d\n", rez);
64
65     return 0;
66 }

```

### Rešenje 2.1.20

```

1  #include <stdio.h>

3  int main()
4  {
5      int x;
6      printf("Unesite jedan ceo broj:");
7      scanf("%d", &x);

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

```

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

### Rešenje 2.1.21

```
1  #include <stdio.h>
   #include <math.h>
3
4  int main()
5  {
6      float a, b, c;
7      float D;
8      float x1, x2;
9      printf("Unesite koeficijente A, B i C:");
10     scanf("%f%f%f", &a, &b, &c);
11
12     /* Proveravamo da li je kvadratna jednačina korektno zadata. */
13     if (a == 0)
14         if (b == 0)
15             /* slucaj a==0 && b==0 && c==0 */
16             if (c == 0)
17                 printf("Jednacina ima beskonacno mnogo resenja\n");
18     /* slucaj a==0 && b==0 && c!=0 */
19     else
20         printf("Jednacina nema resenja\n");
21     /* slucaj a!=0 && b!=0 */
22     else {
23         x1 = -c / b;
24         printf("Jednacina ima jedinstveno realno resenje %.2f\n", x1);
25     }
26     /* slucaj a!=0 */
27     else {
28         D = b * b - 4 * a * c;
29         if (D < 0)
30             printf("Jednacina nema realnih resenja\n");
31         else if (D > 0) {
32             /* funkcija sqrt nalazi se u biblioteci math.h (prevodjenje
33              sa -lm opcijom) */
34             x1 = (-b + sqrt(D)) / (2 * a);
35             x2 = (-b - sqrt(D)) / (2 * a);
36             printf("Jednacina ima dva razlicita realna resenja %.2f ",
37                  "i %.2f\n", x1, x2);
38         } else {
39             x1 = (-b) / (2 * a);
40             printf("Jednacina ima jedinstveno realno resenje %.2f\n", x1);
41         }
42     }
43
44     return 0;
45 }
```

## Rešenje 2.1.22

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     int x;
7     char c1;
8     char c10;
9     char c100;
10    char c1000;
11
12    printf("Unesi jedan cetvorocifreni broj:");
13    scanf("%d", &x);
14
15    /* Uzimamo apsolutnu vrednost unetog broja kako u slucaju da je
16       negativan ne bismo za cifre dobili negativne brojeve. Funkcija
17       abs nalazi se u zaglavlju stdlib.h */
18    x = abs(x);
19
20    if (x < 1000 || x > 9999) {
21        printf("Uneti broj nije cetvorocifren\n");
22        return -1;
23    }
24
25    /* Izdvajamo cifre broja. */
26    c1 = x % 10;
27    c10 = (x / 10) % 10;
28    c100 = (x / 100) % 10;
29    c1000 = (x / 1000) % 10;
30
31    if (c1000 <= c100 && c100 <= c10 && c10 <= c1)
32        printf("Cifre su uredjene neopadajuce \n");
33    else if (c1000 >= c100 && c100 >= c10 && c10 >= c1)
34        printf("Cifre su uredjene nerastuce \n");
35    else
36        printf("Cifre nisu uredjene\n");
37
38    return 0;
39 }
```

## Rešenje 2.1.23

```
1 #include <stdio.h>
2
3 int main()
```

```
{
5   char c;

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

9

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

21  return 0;
}
```

### Rešenje 2.1.24

```
1  #include <stdio.h>

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

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

9

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

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

19

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

23

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

27

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

```

31     /* Ujedno, za neparno k se trazi cifra jedinica izdvojenog
32        broja */
33     printf("Na %d-toj poziciji je broj %d.\n", k, broj % 10);
34
35 } else {
36     /* Ako je k paran broj, zapisan je ceo broj dvocifrenih
37        brojeva i zapoceto je sa zapisom sledeceg
38
39        9 oduzimamo jer je 9 broj cifara potrebnih za zapis
40        jednocifrenih brojeva */
41     n = (k - 9) / 2 + 1;
42
43     /* Broj o kojem se radi je */
44     broj = 9 + n;
45
46     /* Ujedno, za parno k se trazi cifra desetica izdvojenog
47        broja */
48     printf("Na %d-toj poziciji je broj %d.\n", k, broj / 10);
49
50 }
51 } else {
52     printf("Greska: Nedoizvoljena vrednost broja k!\n");
53 }
54
55 return 0;
56 }

```

### Rešenje 2.1.25

```

1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      float x;
7      int k;
8      float F;
9
10     printf("Unesite redom x i k: ");
11     scanf("%f %d", &x, &k);
12
13     /* Proveravaju se vrednosti za k */
14     if (k < 1 || k > 3) {
15         printf("Greska: nedozvoljena vrednost za k!\n");
16         return 0;
17     }
18     printf("F(%f,%d)=", x, k);
19
20     /* Analiziraju se moguci slucajevi */
21     if (k == 1) {

```

```

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

### Rešenje 2.1.26

```

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

7     printf("Unesite broj: ");
    scanf("%d", &dan);
9
11    switch (dan) {
        case 1:
13        printf("ponedeljak\n");
            break;
        case 2:
15        printf("utorak\n");
            break;
        case 3:
17        printf("sreda\n");
            break;
        case 4:
19        printf("cetvrtak\n");
            break;
        case 5:
21        printf("petak\n");
            break;
        case 6:
23        printf("subota\n");
            break;
        case 7:
25        printf("nedelja\n");
            break;
31    }
```



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

### Rešenje 2.1.27

```
1  #include <stdio.h>
3  int main()
4  {
5      char c;
6      printf("Unesite jedan karakter:");
7      scanf("%c", &c);
8
9      /* Da bi se utvrdilo da li je karakter samoglasnik, neophodno je
10       proveriti da li odgovara nekom od sledecih karaktera:
11       A,E,I,O,U,a,e,i,o,u */
12     switch (c) {
13     case 'A':
14     case 'E':
15     case 'I':
16     case 'O':
17     case 'U':
18     case 'a':
19     case 'e':
20     case 'i':
21     case 'o':
22     case 'u':
23         printf("Uneti karakter je samoglasnik\n");
24         break;
25     default:
26         printf("Uneti karakter nije samoglasnik\n");
27         break;
28     }
29
30     return 0;
31 }
```

### Rešenje 2.1.28

```
1  #include <stdio.h>
3  int main()
4  {
5      char op;
6      int x, y;
```

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

11  switch (op) {
12      case '+':
13          printf("Rezultat je: %d\n", x + y);
14          break;
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      case '/':
22          if (y == 0)
23              printf("Greska: deljenje nulom nije dozvoljeno!\n");
24          else
25              printf("Rezultat je: %f\n", x * 1.0 / y);
26          break;
27      case '%':
28          printf("Rezultat je: %d\n", x % y);
29          break;
30      default:
31          printf("Greska: nepoznat operator!\n");
32      }
33
34      return 0;
35 }
```

### Rešenje 2.1.29

```
1  #include <stdio.h>

3  int main()
4  {
5      int d, m;
6      printf("Unesite dan i mesec");
7      scanf("%d%d", &d, &m);

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

```
21     else
22         printf("prolece\n");
23         break;
24 case 4:
25 case 5:
26     printf("prolece\n");
27     break;
28 case 6:
29     if (d < 21)
30         printf("prolece\n");
31     else
32         printf("leto\n");
33     break;
34 case 7:
35 case 8:
36     printf("leto\n");
37     break;
38 case 9:
39     if (d < 23)
40         printf("leto\n");
41     else
42         printf("jesen\n");
43     break;
44 case 10:
45 case 11:
46     printf("jesen\n");
47     break;
48 case 12:
49     if (d < 22)
50         printf("jesen\n");
51     else
52         printf("zima\n");
53 }
54
55 return 0;
56 }
```

### Rešenje 2.1.30

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int godina;
6      int mesec;
7      int prestupna;
8
9      printf("Unesite godinu: ");
10     scanf("%d", &godina);
11
12     if (godina < 0) {
```

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

17 /* Provera da li je godina prestupna, zbog februara */
18 if ((godina % 4 == 0 && godina % 100 != 0)
19     || godina % 400 == 0)
20     prestupna = 1;
21 else
22     prestupna = 0;

23
24 printf("Unesite redni broj meseca: ");
25 scanf("%d", &mesec);

26
27 switch (mesec) {
28     case 1:
29         printf("Januar, 31 dan\n");
30         break;
31     case 2:
32         if (prestupna)
33             printf("Februar, 29 dana\n");
34         else
35             printf("Februar, 28 dana\n");
36         break;
37     case 3:
38         printf("Mart, 31 dan\n");
39         break;
40     case 4:
41         printf("April, 30 dana\n");
42         break;
43     case 5:
44         printf("Maj, 31 dan\n");
45         break;
46     case 6:
47         printf("Jun, 30 dana\n");
48         break;
49     case 7:
50         printf("Jul, 31 dan\n");
51         break;
52     case 8:
53         printf("Avgust, 31 dan\n");
54         break;
55     case 9:
56         printf("Septembar, 30 dana\n");
57         break;
58     case 10:
59         printf("Oktobar, 31 dan\n");
60         break;
61     case 11:
62         printf("Novembar, 30 dana\n");
63         break;
64     case 12:
```

```

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

71     return 0;
}

```

### Rešenje 2.1.31

```

1  #include <stdio.h>

3  int main()
4  {
5      int dan, mesec, godina, dozvoljen_broj_dana;

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

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

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

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

```

## 2 Kontrola toka

```
41     else
42         /* Ako nije, dozvoljeni broj dana za februar je 28 */
43         dozvoljen_broj_dana = 28;
44         break;
45     case 4:
46     case 6:
47     case 9:
48     case 11:
49         /* Dozvoljeni broj dana za april, jun, septembar i novembar je
50            30 */
51         dozvoljen_broj_dana = 30;
52         break;
53 }
54 /* Proveravamo dan */
55 if (dan < 0 || dan > dozvoljen_broj_dana) {
56     printf("Datum nije korektan (neispravan dan)!\n");
57     return 0;
58 }
59
60 /* Sve provere su ispunjene pa zakljucujemo da je datum korektan */
61 printf("Ispravan datum!\n");
62
63 return 0;
64 }
```

### Rešenje 2.1.32

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int dan, mesec, godina;
6     int prethodni_dan, prethodni_mesec, prethodni_godina;
7
8     /* Citamo datum */
9     printf("Unesite datum: ");
10    scanf("%d.%d.%d", &dan, &mesec, &godina);
11
12    /* Racunamo dan, mesec i godinu prethodnog dana */
13    prethodni_dan = dan - 1;
14    prethodni_mesec = mesec;
15    prethodni_godina = godina;
16
17    /* I po potrebi vrsimo korekcije */
18
19    /* Ako je u pitanju prvi u mesecu */
20    if (prethodni_dan == 0) {
21        /* Treba korigovati mesec */
22        prethodni_mesec = mesec - 1;
23        /* Ako je u pitanju januar */
24        if (prethodni_mesec == 0) {
```

```

25     /* Treba korigovati i godinu */
    prethodni_mesec = 12;
27     prethodni_godina = godina - 1;
    }

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

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

59     return 0;
61 }

```

Rešenje 2.1.33      Rešenje je analogno rešenju zadatka 2.1.32.

### Rešenje 2.1.35

```

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

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

```

```
float dab, dad;
10 float delta, deltap, deltaq;
float 0;
12 float k, n;

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

16

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

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



```

62     deltaq = x1 * (y4 - y3) - y1 * (x4 - x3);
    if (delta != 0)
        printf("Sistem ima jedinstveno resenje %.2f, %.2f\n",
64             deltap / delta, deltaq / delta);
    else if (deltap == 0 && deltaq == 0)
66         printf("Sistem je neodredjen ili nema resenja.\n");
    else
68         printf("Sistem nema resenja\n");
        break;
70 default:
    printf("Nekorektan unos\n");
72 }

74 return 0;
}

```

## 2.3 Petlje

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

*Primer 1*

```

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

```

[Rešenje 2.3.1]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 2.3.2]

## 2 Kontrola toka

---

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

*Primer 1*

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

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo pozitivan broj: -10
|| Neispravan unos. Promenljiva mora biti
|| pozitivna!
```

[Rešenje 2.3.3]

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

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva cela broja: -2 4
|| -2 -1 0 1 2 3 4
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva cela broja: 10 6
|| Neispravan unos. Nisu dobro zadate granice
|| intervala!
```

[Rešenje 2.3.4]

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite pozitivan broj: 18
|| Faktorijel = 640237370572800
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite pozitivan broj: 8
|| Faktorijel = 40320
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite pozitivan broj: 40
|| Broj je veliki, dolazi do
|| prekoracenja.
```

[Rešenje 2.3.5]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.6]

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

### Primer 1

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

### Primer 2

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

[Rešenje 2.3.7]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM: ||
|| Unesite ceo broj veci od 0: 100 ||
|| 2 4 5 10 20 25 50              ||
```

### Primer 2

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

[Rešenje 2.3.8]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.9]

## 2 Kontrola toka

---

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.10]

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 12  
|| Deljiv je sumom svojih  
||   cifara.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2564  
|| Nije deljiv sumom svojih  
||   cifara.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -4  
|| Neispravan ulaz.
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 0  
|| Neispravan ulaz.
```

[Rešenje 2.3.11]

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 7  
|| Unesite brojeve:  
|| 8 -50 45 2007 -67 -123 14  
|| Suma pozitivnih: 2074  
|| Suma negativnih: -240
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 5  
|| Unesite brojeve:  
|| -5 -20 -4 -200 -8  
|| Suma pozitivnih: 0  
|| Suma negativnih: -237
```

*Primer 3*

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

[Rešenje 2.3.12]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -4
Neispravan unos.

```

[Rešenje 2.3.13]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

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

```

*Primer 4*

```

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

```

[Rešenje 2.3.14]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj m: 2
|| Unesite broj n: -4
|| Broj artikala ne može biti negativan.
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj m: 30
|| Unesite broj n: 4
|| Unesite n brojeva: 67 -100 23 98
|| Cena ne može biti negativna.
```

[Rešenje 2.3.15]

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

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve:
|| -5 -200 -43 0
|| Nisu uneseni pozitivni
|| brojevi.
```

### Primer 2

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

[Rešenje 2.3.16]

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

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -235515
|| Cifra 5 se nalazi u zapisu!
```

[Rešenje 2.3.17]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.18]

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

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cene: 6.32 -9
|| Cena ne može biti negativana.
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cene: 0
|| Nisu unesene cene.
```

[Rešenje 2.3.19]

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

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 10
|| Unesite brojeve:
|| 7.82 4.3 -1.2 56.8 -3.4 -72.1 8.9 11.2 -11.2
|| -102.4
|| Broj promena je 5.
```

### Primer 2

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

### Primer 3

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

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 0
|| Broj promena je 0.
```

[Rešenje 2.3.20]

## 2 Kontrola toka

---

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

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj artikla: 6
|| Unesite artikle:
|| 12 3.4 90 100.53 53.2 12.8
|| Minimalna cena je: 3.400000
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj artikla: 3
|| Unesite artikle: 4 -8 92
|| Cena ne može biti negativna.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj artikla: -9
|| Neispravan unos.
```

[Rešenje 2.3.21]

**Zadatak 2.3.22** Program učitava ceo pozitivan broj  $n$ , a zatim i  $n$  celih brojeva. Napisati program koji ispisuje broj sa najvećom cifrom desetica. Ukoliko ima više takvih, ispisati prvi.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -12
|| Neispravan unos.
```

[Rešenje 2.3.22]

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

*Primer 1*

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

[Rešenje 2.3.23]



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

*Primer 1*

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

*Primer 1*

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

[Rešenje 2.3.24]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.25]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.26]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.27]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 2.3.28]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje 2.3.29]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

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

```

[Rešenje 2.3.30]

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

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj: 10
|| Trazeni broj je: 55

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj: -100
|| Neispravan unos. Pozicija u Fibonacijevom
|| nizu mora biti pozitivan broj koji nije 0!

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj: 78
|| Trazeni broj je: 375819880

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj: 20
|| Trazeni broj je: 6765

```

[Rešenje 2.3.31]

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

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

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

*Primer 1*

```

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

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo broj: -48
|| Nekorektan unos. Broj mora biti pozitivan.

```

*Primer 3*

```

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

```

*Primer 4*

```

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

```

[Rešenje 2.3.32]

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

*Primer 1*

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

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite format papira: 3  
|| 297 420
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite format papira: 7  
|| 74 105
```

*Primer 4*

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

[Rešenje 2.3.33]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.34]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.35]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.36]

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

*Primer 1*

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

[Rešenje 2.3.37]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.38]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.39]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.40]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.41]

\* **Zadatak 2.3.42** Napisati program koji učitava realane brojeve  $x$  i  $eps$  i sa zadatom tačnošću  $eps$  izračunava i ispisuje sumu  $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$ . 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 faktoriјel odvojeno zbog neefikasnosti takvog rešenja i zbog mogućnosti prekoračenja.

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.42]

\* **Zadatak 2.3.43** Napisati program koji učitava realane brojeve  $x$  i  $eps$  i sa zadatom tačnošću  $eps$  izračunava i ispisuje sumu  $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} \dots$ . 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 2.3.43]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.44]

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

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje [2.3.45](#)]

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

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

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

[Rešenje [2.3.46](#)]

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

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

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

### Primer 1

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

### Primer 2

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



### Primer 3

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

### Primer 4

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

[Rešenje 2.3.47]

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

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

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
|| Neispravan unos
```

### Primer 4

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

[Rešenje 2.3.48]

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

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

U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

### Primer 1

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

### Primer 2

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

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva relana broja x i a:: 2.6 0.3
Unesite prirodan broj: 3
Izraz = 5800.970129
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva relana broja x i a:: 5.4 7
Unesite prirodan broj: -2
Neispravan unos.
```

[Rešenje 2.3.49]

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

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

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

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
1 2 3
4 5 6
7 8 9
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
1 2 3 4
5 6 7 8
9 10 11 12
13 14 15 16
```

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

- (d) Napisati program koji za uneto  $n$  iscrtava pravougli „trougao” sačinjen od „koordinata” svojih tačaka. „Koordinata” tačke je oblika  $(i, j)$  pri čemu  $i, j = 0, \dots, n$ . 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 2.3.50]

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

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

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

*Primer 1*

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

*Primer 2*

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

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

*Primer 1*

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

*Primer 1*

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

[Rešenje 2.3.51]

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 2.3.52]

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Pogresan unos.

```

[Rešenje 2.3.53]

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

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

*Primer 1*

```

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

```

*Primer 1*

```

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

```

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

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

*Primer 1*

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

*Primer 1*

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

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

*Primer 1*

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

*Primer 2*

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

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

*Primer 1*

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

*Primer 2*

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

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 2.3.54]

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

- (a) Slika sadrži jednakokranični trougao stranice  $n$  koji je sastavljen od zvezdica.

*Primer 1*

```

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

```

*Primer 2*

```

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

```

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

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

### Primer 1

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

## Primer 2

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

*
***
*****
*****
*****
*****
*****
***
*
```

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

### Primer 1

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

### Primer 1

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

  *
 * *
*   *
 *   *
* * * *

```

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

### Primer 1

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

  *
 * *
* * *
 * *
  *
```

## Primer 2

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

  *
 * *
*   *
* * * *
* * * * *
 *   *
  * *
   *
```

\* **Zadatak 2.3.56** Napisati program koji za uneti prirodan broj  $n$  iscrta va strelice dimenzije  $n$ . Pretpostaviti da je unos korektan.



*Primer 1*

```

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

```

*Primer 2*

```

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

```

[Rešenje 2.3.56]

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

*Primer 1*

```

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

```

[Rešenje 2.3.57]

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

Primer 1

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

Primer 2

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

[Rešenje 2.3.58]

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

Primer 1

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

Primer 2

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

[Rešenje 2.3.59]

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

Primer 1

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

Primer 1

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

[Rešenje 2.3.60]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.3.61]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -23
|| Neispravan unos.
```

[Rešenje 2.3.62]

### 2.4 Rešenja

#### Rešenje 2.3.1

```
1  #include <stdio.h>
2
3  int main()
4  {
5      /* Promenljiva i kontrolise koliko puta ce se petlja izvesti i
6       naziva se brojac petlje. Njenu pocetnu vrednost postavljamo na
7       0 jer se u pocetku petlja nije ni jednom izvela. */
8      int i = 0;
9
10     /* Pre ulaska u telo petlje proverava se da li je ispunjen uslov
11     ulaska u petlju. */
12     while (i < 5) {
13         /* Ukoliko uslov ulaska u petlju jeste ispunjen, izvela se
14         telo petlje. */
15         /* Ispisujemo trazeni tekst. */
16         printf("Mi volimo da programiramo.\n");
17
18         /* Uvecava se brojac za jedan jer je jednom izvelo telo
19         petlje. Ako bi ova vrednost ostala nepromenjena, petlja bi
20         se izvela beskonacno. */
21         i++;
22
23         /* Nakon poslednje naredbe tela petlje ponovo se ispituje uslov
24         petlje. */
25     }
26
27     return 0;
28 }
```

#### Rešenje 2.3.2

```
1  #include<stdio.h>
2
3  int main()
4  {
5      /* Brojac u petlji. */
6      int i = 0;
7      /* Promenljiva koja oznacava koliko puta cemo ispisati trazeni
8      tekst. */
9      int n;
10
11     printf("Unesite ceo broj: ");
12     scanf("%d", &n);
13
14     /* Pre ulaska u telo petlje proverava se da li je ispunjen uslov
```

```
        ulaska u petlju. */
16 while (i < n) {
    printf("Mi volimo da programiramo.\n");
18     i++;
    }
20
    return 0;
22 }
```

### Rešenje 2.3.3

```
1  #include <stdio.h>
3  int main()
4  {
5      /* Promenljivu x koristimo u dve svrhe. Prvo, ova promenljiva
        kontrolise koliko puta se petlja izvršila. Drugo, ovu
7      promenljivu koristimo za ispis potrebnih vrednosti. */
    int x;
9    /* Promenljiva n se unosi i odredjuje koliko brojeva ispisujemo. */
    int n;
11
    printf("Unesi pozitivan ceo broj: ");
13    scanf("%d", &n);
15
    /* U slucaju neispravnih podataka ispisujemo odgovarajucu poruku
       i izlazimo iz programa. */
17    if (n < 0) {
        printf("Neispravan unos. Promenljiva mora biti pozitivna!\n");
19        return -1;
    }
21
    /* Ispis pocinjemo od 0, zato promenljivu x postavljamo na 0. */
23    x = 0;
    while (x <= n) {
25        /* Ispisujemo broj. */
        printf("%d\n", x);
27        /* Uvecavamo promenljivu za jedan jer smo broj ispisali i sada
           zelimo da ispisemo sledeci broj. */
29        x++;
    }
31
    return 0;
33 }
```

### Rešenje 2.3.4

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

## 2 Kontrola toka

---

```
3 #include <stdio.h>

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

11
    printf("Unesi dva cela broja: ");
13     scanf("%d%d", &n, &m);

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

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

29     printf("\n");

31     return 0;
33 }
```

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

3 #include <stdio.h>

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

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

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

```

21
23  /* naredba i=n se izvršava jednom, pre prve iteracije */
24  for (i = n; i <= m; i++)      /* uslov petlje i<=m se proverava
25                                pre svake iteracije */
26      printf("%d ", i);        /* naredba i++ se izvršava nakon
27                                svake iteracije */
28
29  printf("\n");
30
31  return 0;
32 }
33
1  /* Resenje pod c). */
2  #include <stdio.h>
3
4  int main()
5  {
6      /* Promenljive koje oznacavaju granice intervala. */
7      int n, m;
8      /* Promenljiva koja oznacava trenutno ispisani broj intervala. */
9      int i;
10
11     printf("Unesi dva cela broja: ");
12     scanf("%d%d", &n, &m);
13
14     if (m < n) {
15         printf
16             ("Neispravan unos. Nisu dobro zadate granice intervala!\n");
17         return -1;
18     }
19
20     /* Uslov petlje se proverava na kraju svake iteracije. */
21     /* Zbog toga se do while petlja izvršava bar jednom, cak i u
22        slucaju da uslov petlje nikada nije ispunjen. */
23     i = n;
24     do {
25                                     /* Petlja se zapocinje bez provere
26                                     uslova. */
27         printf("%d ", i);          /* Stampa se vrednost promenljive
28                                     i. */
29         i++;                      /* Uvecava se vrednost promenljive
30                                     i. */
31     }
32     while (i <= m);                /* Proverava se uslov i ukoliko je
33                                     ispunjen, nastavlja se sa
34                                     sledecom iteracijom. */
35     /* U suprotnom, petlja se završava i program se nastavlja od prve
36        naredbe koja sledi za petljom. */
37     printf("\n");
38
39     return 0;

```

```
39 }
```

### Rešenje 2.3.5

```
1  #include<stdio.h>
3  int main()
4  {
5      int x;
6      /* U promenljivoj f se pamti izracunati faktorijel. Kako
7         faktorijel je jako veliki broj, za tip podataka se uzima
8         unsigned long, da bi mogla da se upise sto veka vrednost. */
9      unsigned long f;
10     int i;
11     int original;
12
13     printf("Unesite pozitivan broj: ");
14     scanf("%d", &x);
15
16     if (x < 0) {
17         printf("Nekorektan unos\n");
18         return -1;
19     }
20
21     if (x >= 22) {
22         printf("Broj je veliki, dolazi do prekoracenja.\n");
23         return -1;
24     }
25
26     original = x;
27     f = 1;
28
29     while (x > 1) {
30         f = f * x;
31         x--;
32     }
33
34     printf("Faktorijel = %lu\n", f);
35     return 0;
36 }
37 }
```

### Rešenje 2.3.6

```
1  #include <stdio.h>
3  int main()
4  {
5      int n;
```



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

9

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

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

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

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

23

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

27 return 0;
}
```

### Rešenje 2.3.7

```
1  #include <stdio.h>

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

7

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

11

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

15

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

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

## 2 Kontrola toka

---

```
27     }
29     return 0;
}
```

### Rešenje 2.3.8

```
1  #include<stdio.h>
   #include<math.h>
3
   int main()
5  {
       int x;
7      /* Brojac u petlji. */
       int i;
9
       /* Ucitavamo broj. */
11     printf("Unesi ceo broj veci od 0: ");
       scanf("%d", &x);
13
       if (x <= 0) {
15         printf("Neispravan unos.\n");
           return -1;
17     }

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

27
       /* 2. nacin (brzi) -- Ne proveravaju se svi brojevi od 2 do x,
29         vec se petlja izvrsava dok ne stignemo do korena broja. */
       printf("----- 2. nacin ----- \n");
31     for (i = 2; i <= sqrt(x); i++) {
           /* Proveravamo da li i deli broj x. */
33         if (x % i == 0)
           /* U slucaju kada je delilac koren broja, npr. 4 za 16,
35             ispisujemo ga jednom. */
           if (i == x / i)
37             printf("%d \n", i);
           /* U suprotnom, npr. 2 za 16, ispisujemo i 2 i 8. */
39         else
           printf("%d %d \n", i, x / i);
41     }

43     return 0;
}
```

## Rešenje 2.3.9

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

## Rešenje 2.3.10

```
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  int main()
5  {
6      int x;
7      char cifra;
8      printf("Unesi ceo broj:");
9      scanf("%d", &x);
11
12     /* Pretvaranje u apsolutnu vrednost se vrši za slučaj kada je
13        unet negativan broj kako bismo osigurali da ce nam izdvojene
14        cifre biti pozitivne. */
15     x = abs(x);
16
17     /* Kako uklanjamo cifre broja (pogledati telo petlje) u nekom
18        trenutku broj ce postati 0 jer smo uklonili sve njegove cifre.
```

```
    Tada prekidamo rad petlje. */
19 while (x > 0) {
    /* Izdvajamo poslednju cifru broja x. */
21     cifra = x % 10;
    printf("%d\\n", cifra);
23     /* Uklanjammo poslednju cifru broja x. */
    x /= 10;
25 }

27 return 0;
}
```

### Rešenje 2.3.11

```
1  #include <stdio.h>

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

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

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

17     if (n <= 0) {
19         printf("Neispravan unos.\\n");
        return -1;
21     }

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

27     while (pom_n != 0) {
29         /* Na sumu dodajemo poslednju cifru. */
        suma += pom_n % 10;
31         /* Sa broja skidamo poslednju cifru. */
        pom_n /= 10;
33     }

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

```
39     else
        printf("Nije deljiv sumom svojih cifara.\n");
41     return 0;
}
```

## Rešenje 2.3.12

```
1  #include<stdio.h>
3  int main()
4  {
5      int n;
6      /* Oznaka broja koji unosimo u jednoj iteraciji petlje. */
7      int x;
8      int suma_poz;
9      int suma_neg;
10     /* Brojac. */
11     int i;

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

14     if (n < 0) {
15         printf("Neispravan unos.\n");
16         return -1;
17     }

18     /* Promenljivama koje ce sadržati sume se pre ulaska u petlju
19        dodeljuje 0 (neutral za sabiranje). */
20     suma_poz = 0;
21     suma_neg = 0;
22     i = 0;

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

26         if (x < 0)
27             suma_neg += x;
28         else
29             suma_poz += x;

30         i++;
31     }

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

### Rešenje 2.3.13

```
1 #include <stdio.h>
3 int main()
{
5     /* Promenljiva x oznacava tekuci uneti broj. */
    int n, x;
7     /* Brojac. */
    int i;
9     int zbir = 0;

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

13
15    if (n < 0) {
        printf("Neispravan unos.\n");
        return -1;
17    }

19    printf("Unesite n brojeva: ");

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

27
29        /* Proverava se da li broj negativan i neparan. */
        if (x < 0 && x % 2 != 0) {
            /* Ako jeste, dodajemo ga na zbir. */
            zbir = zbir + x;
31        }

33
35        /* Uvecava se brojac iteracija. */
        i++;
37    }

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

41    return 0;
}
```

### Rešenje 2.3.14

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

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

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

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

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

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

25  printf("Suma je %d.\n", suma);
27  return 0;
29 }
```

### Rešenje 2.3.15

```
1  #include <stdio.h>

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

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

13    if (m < 0) {
15        printf("Nikola ne moze imati negativno novaca.\n");
        return -1;
17    }

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

21    if (n < 0) {
```

## 2 Kontrola toka

---

```
23     printf("Broj artikala ne moze biti negativan.\n");
      return -1;
25 }

27 printf("Unesite cene artikala: ");

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

33         if (cena <= 0) {
35             printf("Cena ne moze biti negativna.\n");
              return -1;
37         }

39         /* Provera da li je cena manji od zadatog broja m. */
          if (cena < m) {
41             /* Ako jeste, uvecava se broj brojeva za 1. */
              broj_brojeva++;
43         }

45         i++;
      }

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

49 return 0;
51 }
```

### Rešenje 2.3.16

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

5      /* U promenljivoj p se cuva proizvod. */
      int p;

7      /* Promenljiva u služi za proveru da li su brojevi uopšte
          uneseni. Na početku se pretpostavlja da nisu i postavlja se na
11      0. */
      int u = 0;

13      /* Promenljiva unesen_pozitivan služi za proveru da li su
          pozitivni brojevi uopšte uneseni. Na početku se pretpostavlja
15      da nisu i postavlja se na 0. */
      int unesen_pozitivan = 0;

17      p = 1;

19 }
```



```

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

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

27

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

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

33

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

37

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

43

    /* Ako je makar jedan broj pozitivan, promenljiva
    unesen_pozitivan se postavlja na 1. */
45    unesen_pozitivan = 1;
    p = p * x;

47    }

51

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

59    return 0;
61 }

```

### Rešenje 2.3.17

```

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

```

## 2 Kontrola toka

---

```
7   int indikator = 0;

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

13  if (n < 0)
    n = abs(n);

15

17  /* Sve dok imamo cifara u zapisu broja. */
while (n > 0) {

19      /* Izdvajamo poslednju cifru broja. */
    cifra = n % 10;

21

23      /* Proveravamo da li je bas ona jednaka broju 5 */
    if (cifra == 5) {
        /* Ako jeste postavljamo indikator na vrednost 1 tako da
25         znamo da smo pronasli peticu i prekidamo sa izvršavanjem
        petlje. */
        indikator = 1;
27         break;
29     }
    /* Ako izvdvojena cifra nije jednaka broju 5, broj delimo sa 10
31     * kako bi mogli da izdvojimo i preostale cifre broja na isti
    * nacin.
33     */
    n = n / 10;
35 }

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

43
45 return 0;
}
```

### Rešenje 2.3.18

```
1  #include <stdio.h>

3  int main()
{
5      int x;
    int broj_brojeva;
7      int suma;

9      broj_brojeva = 0;
```

```

11     suma = 0;
13     printf("Unesite brojeve: ");
15     while (1) {
16         /* Ucitavanje broja. */
17         scanf("%d", &x);
19         /* Ako je unesena 0, prekida se petlja. */
20         if (x == 0)
21             break;
23         /* Procitani broj dodaje se na sumu. */
24         suma += x;
26         /* I uvecava se broj ucitanih brojeva. */
27         broj_brojeva++;
29     }
31     if (broj_brojeva == 0)
32         printf("Nisu uneseni brojevi.\n");
33     else
34         /* Prilikom deljenja celih brojeva kao rezultat se dobija ceo
35          broj. Kako je aritmeticka sredina realan broj, potrebno je
36          izvršiti konverziju prilikom deljenja da bi se dobio
37          ispravan rezultat. */
38         printf("Aritmeticka sredina: %.4f\n",
39             (double) suma / broj_brojeva);
41     return 0;
42 }

```

### Rešenje 2.3.19

```

1  #include <stdio.h>
3  int main()
4  {
5      float cena;
6      int broj_artikla;
7      float suma;
9      broj_artikla = 0;
10     suma = 0;
12     printf("Unesite cene: ");
14     while (1) {
15         scanf("%f", &cena);
17         if (cena == 0)

```

## 2 Kontrola toka

---

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

25    suma += cena;

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

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

### Rešenje 2.3.20

```
1  #include <stdio.h>

3  int main()
{
5      int n;
    /* Ucitavaju se dva broja, broj i sledbenik, i proverava se da li
7      su razlicitog znaka. */
    double broj, sledbenik;
    /* Brojac. */
9      int i;
    int broj_promena = 0;

11

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

15

17    if (n < 0) {
        printf("Neispravan unos.\n");
        return -1;
19    }

21    /* Prvo se proveara da li uopste ima unosa, i ako unosa nema,
        ispisuje se odgovarajuca poruka i izlazi iz programa. */
23    if (n == 0) {
        printf("Broj promena je 0.\n");
        return 0;
25    }

27

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

```

29  /* Pre petlje učitava se jedan broj, a u petlji se učitava njegov
    sledbenik i proverava se da li su razlicitog znaka. */
31  scanf("%lf", &broj);

33  /* Kako je vec jedan broj unesen, brojac se postavlja na 1, a ne
    na 0. */
35  for (i = 1; i < n; i++) {
    /* Učitava se sledbenik. */
37    scanf("%lf", &sledbenik);

39    /* Ako su razlicitog znaka proizvod je manji od 0. */
    if (sledbenik * broj < 0)
41      broj_promena++;
    /* Problem je ako je proizvod jednak 0. Tada mora proveriti da li
43     je jedan od brojeva negativan jer tada postoji promena
    znaka. */
45    else if (sledbenik * broj == 0 && (sledbenik < 0 || broj < 0))
      broj_promena++;
47
    /* Tekuci sledbenik postaje tekuci broj, a u sledecoj iteraciji
49     petlje se učitava sledeci sledbenik. */
    broj = sledbenik;
51  }

53  printf("Broj promena je %d.\n", broj_promena);

55  return 0;
}

```

### Rešenje 2.3.21

```

1  #include <stdio.h>

3  int main()
{
5    /* Broj artikala. */
    int n;
7    /* Brojac. */
    int i;
9    /* Cena trenutno unetnog artikla. */
    float cena;
11   /* Minimalna cena. */
    float min_cena;
13
    printf("Unesite broj artikala:");
15    scanf("%d", &n);

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

```

## 2 Kontrola toka

---

```
21  /* Prva cena se unosi iznad petlje kako bi bio njegova vrednost
23     bila dodeljena promenljivoj min_cena. Neophodno je da
25     promenljiva min bude inicijalizovana pre ulaska u petlju da bi
        uslov x<min mogao da bude ispitan u prvoj iteraciji. */
27  printf("Unesite cenu artikala:");
29  scanf("%f", &cena);
31  /* Proveravamo da li je cena isprano uneta vrednost. */
33  if (cena <= 0) {
35      printf("Cena ne moze biti negativna.\n");
37      return -1;
39  }

41  min_cena = cena;
43  i = 0;
45  while (i < n - 1) {
47      scanf("%f", &cena);

49      if (cena <= 0) {
51          printf("Cena ne moze biti negativna.\n");
53          return -1;
55      }

57      /* Provera da li je uneta cena manja od tekuće minimalne cene. */
59      if (cena < min_cena)
61          min_cena = cena;
63      i++;
65  }

67  printf("Minimalna cena je: %f\n", min_cena);
69
71  return 0;
73 }
```

### Rešenje 2.3.22

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int n;
7      int x, x_desetica;
8      int max_desetica, broj;
9      int i;
10
11     printf("Unesite broj n: ");
12     scanf("%d", &n);
13
14     if (n < 0) {
15         printf("Neispravan unos.\n");
16     }
17 }
```

```

17     return -1;
18 }
19 if (n == 0) {
20     printf("Nisu uneseni brojevi.\n");
21     return 0;
22 }
23
24 /* Maksimalna cifra desetice se postavlja na 0 jer 0 je
25    svakako najmanja cifra pa je pocetna vrednost neutralna tj. ne
26    moze da utice na izracunavanje maksimuma. Ipak, treba biti
27    pazljiv jer nije uvek zgodno pretpostaviti da je maksimalna
28    vrednost 0. Na primer, ako je zadatak naci maksimum celih
29    brojeva, a korisnik unese -32 -7 i -22, maksimalni je broj -7,
30    sto je manje od 0. */
31 max_desetica = 0;
32
33 printf("Unesite brojeve: ");
34 for (i = 0; i < n; i++) {
35     scanf("%d", &x);
36
37     /* Izdvajanje cifre desetice procitanog broja. */
38     x_desetica = (abs(x) / 10) % 10;
39
40     /* Proverava da li je izdvojena cifra veca od trenutne
41        maksimalne cifre desetice. */
42     if (x_desetica > max_desetica) {
43         /* Ako jeste vece, pamti se nova najveca cifra, kao i broj u
44            kom se pojavila. */
45         max_desetica = x_desetica;
46         broj = x;
47     }
48 }
49
50 printf("Broj sa najvecom cifrom desetice je %d\n", broj);
51
52 return 0;
53 }

```

### Rešenje 2.3.23

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6
7     int n;
8     int x, x_kopija, br_cifara;
9     int max_br_cifara, broj;
10    int i;

```

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

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

19 if (n == 0) {
21     printf("Nisu uneseni brojevi.\n");
    return 0;
23 }

25 /* Maksimalan broj cifara se postavlja na 0, svaki broj ima vise
    od 0 cifara pa je ova vrednost neutralna. */
27 max_br_cifara = 0;

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

33         /* Odredjivanje broja cifara unetog broja x. */
        x_kopija = abs(x);
35         br_cifara = 0;
        while (x_kopija != 0) {
37             x_kopija = x_kopija / 10;
            br_cifara++;
39         }

41         /* Ako je broj cifara unetog broja veci od maksimalnog */
        if (br_cifara > max_br_cifara) {
43             /* Cuvamo ga */
            max_br_cifara = br_cifara;
45             /* I zbog ispisa rezultata, cuvamo i originalni broj */
            /* Zbog ovoga smo morali i da racunamo broj cifara nad
47                kopijom broja x kako ne bismo promenili njegovu vrednost */
            broj = x;
49         }
    }

51 printf("Najvise cifara ima broj %d\n", broj);

53 return 0;
55 }
```

### Rešenje 2.3.24

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



```

4  int main()
   {
6
       int n;
8       int x, x_kopija;
       int broj;
10      int vodeca_cifra, max_vodeca_cifra;
       int i;
12
       /* Citamo vrednost sa ulaza */
14      printf("Unesite broj n: ");
       scanf("%d", &n);
16
       /* Postavljamo maksimalnu vodecu cifru na 0 - cifre broja su vece
18          ili jednake od 0 pa je ova vrednost neutralna */
       max_vodeca_cifra = 0;
20
       /* Ucitavamo broj po broj */
22      printf("Unesite n brojeva: ");
       for (i = 0; i < n; i++) {
24          scanf("%d", &x);

26          /* Odredjujemo vodecu cifru broja */
           x_kopija = abs(x);
28           while (x_kopija > 10) {
               x_kopija = x_kopija / 10;
30           }
           vodeca_cifra = x_kopija;
32
           /* Ako je izdvojena cifra veca od maksimalne vodece cifre */
34           if (vodeca_cifra > max_vodeca_cifra) {
               /* Cuvamo je */
36               max_vodeca_cifra = vodeca_cifra;
               /* I zbog ispisa, cuvamo i broj u kojem se ona pojavljuje */
38               /* Zbog ovoga smo morali i da racunamo vodecu cifru nad
                   kopijom broja x kako ne bismo promenili njegovu vrednost */
40               broj = x;
           }
42     }

44     /* Ispisujemo rezultat */
       printf("%d\n", broj);
46
       return 0;
48   }

```

### Rešenje 2.3.25

```

1  #include <stdio.h>

```

## 2 Kontrola toka

---

```
3 int main()
4 {
5     int x;
6     int min, max;
7
8     printf("Unesite brojeve: ");
9
10    /* Prvi broj se učitava izvan petlje zbog inicijalizacije
11       maksimuma i minimuma. */
12    scanf("%d", &x);
13    max = x;
14    min = x;
15
16    /* Sve dok se ne unese 0, učitavaju se brojevi u petlji. */
17    while (x != 0) {
18
19        /* Provera da li je procitani broj veci od aktuelnog maksimuma.
20           */
21        if (x > max)
22            max = x;
23        /* Provera da li je procitani broj manji od aktuelnog minimuma.
24           */
25        if (x < min)
26            min = x;
27
28        /* Učitavanje narednog broja. */
29        scanf("%d", &x);
30    }
31
32    printf("Razlika: %d\n", max - min);
33
34    return 0;
35 }
```

### Rešenje 2.3.26

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main()
5 {
6     int n;
7     int d;
8     /* Uzastopni brojevi za koje se racuna rastojanje. */
9     int x, y;
10    int broj_parova;
11    int i;
12
13
14    printf("Unesite brojeve n i d: ");
15    scanf("%d %d", &n, &d);
```

```

17  if (n < 0 || d < 0) {
18      printf("Neispravan unos.\n");
19      return -1;
20  }
21
22  broj_parova = 0;
23
24  printf("Unesite n brojeva: ");
25
26  /* Prvi broj se učitava pre petlje. */
27  scanf("%d", &x);
28
29  for (i = 1; i < n; i++) {
30      scanf("%d", &y);
31
32      /* Provera da li su brojevi na rastojanju d. */
33      if (abs(y - x) == d)
34          broj_parova++;
35
36      /* Broj iz tekuće iteracije se čuva kako bi mogao da se
37         upotrebljava u narednoj iteraciji. */
38      x = y;
39  }
40
41  printf("Broj parova: %d\n", broj_parova);
42
43  return 0;
44  }

```

### Rešenje 2.3.27

```

1  #include <stdio.h>
2
3  int main()
4  {
5      int x;
6      /* Tezina trenutne pozicije u broju. Može biti 1, 10, 100, 1000
7         itd. */
8      int pozicija;
9      /* Trenutna izdvojena cifra iz broja x. */
10     int cifra;
11     /* Broj dobijen nakon transformacije. */
12     unsigned int y;
13
14     printf("Unesite broj: ");
15     scanf("%d", &x);
16
17
18     if (x <= 0) {
19         printf("Nekorektan unos.\n");

```

```
    return -1;
21 }

23 /* Posto pocinjemo sa izdvajanjem cifara od cifre jedinica,
    postavlja se tezinu (stepen) pozicije na 1. */
25 pozicija = 1;
    y = 0;

27 /* Provera da li ima cifara u zapisu broja. */
29 while (x > 0) {

31     /* Izdvaja se poslednja cifra iz zapisa. */
        cifra = x % 10;

33     /* Provera da li je cifra parna. */
35     if (cifra % 2 == 0) {
        /* I ako jeste, uvecava se. */
37         cifra++;

39     }

41     /* Novi broj se formira tako sto se izdvojena cifra pomnozi
        odgovarajucom tezinom (stepenom) pozicije. */
43     y += cifra * pozicija;

45     /* Priprema se broj za izdvajanje naredne cifre, uklanja se
        poslednja cifra broja. */
47     x /= 10;

49     /* Uvecava se tezinu (stepen) pozicije. */
        pozicija *= 10;

51 }

53 /* Ispisuje se izracunatu vrednost. */
    printf("%d\n", y);

55     return 0;
57 }
```

### Rešenje 2.3.28

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

5 int main()
  {
7     int x;
    /* Tezina trenutne pozicije u broju. Moze biti 1, 10, 100, 1000
9     itd. */
    int stepen_deset;
```

```
11  /* Trenutna izdvojena cifra iz broja x. */
    int cifra;
13  /* Redni broj cifre koja se trenutno obradjuje, gledano s desna
    na levo. */
15  int rbr;
    /* Broj dobijen nakon transformacije. */
17  int y;
    /* Promenljiva znak cuva znak unesenog broja. Moze biti -1 za
19  negativnu vrednost ili 1 za pozitiivnu vrednost. */
    int znak = 1;

21  /* Ucitavanje broja. */
23  printf("Unesite broj: ");
    scanf("%d", &x);

25
    if (x <= 0) {
27         x = abs(x);
        znak = -1;
29     }
    /* Postavlja se vrednost stepena na 0 - to znaci da se prvo mnozi
31     sa 10^0=1. */
    stepen_deset = 0;

33
    /* Postavlja se vrednost broja koji se formira na 0. */
35    y = 0;
    /* Postavlja se redni broj pozicije na 0. */
37    rbr = 0;

39    /* Provera da li ima cifara u zapisu broja. */
    while (x > 0) {
41
        /* Izdvajanje cifre. */
43        cifra = x % 10;

45        /* Proverava se da li je pozicija izdvojene cifre parna - cifre
        na parnim pozicijama se zadrzavaju. */
47        if (rbr % 2 == 0) {
            /* Ako jeste parna izdvojena cifra se dodaje novom broju.
49            Neophodno je izvršiti promenu tipova, jer je double
            povratni tip funkcije pow. */
51            y += cifra * ((int) pow(10, stepen_deset));

53            /* Uvecava se stepen zbog naredne cifre. */
            stepen_deset++;
55        }

57        /* Azurira se redni broj cifre. */
        rbr++;
59        /* I priprema se broj za naredno izdvajanje. */
        x /= 10;
61    }
```

## 2 Kontrola toka

---

```
63  y = znak * y;

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

67  return 0;

69 }
```

### Rešenje 2.3.29

```
1  #include <stdio.h>

3  int main()
{
5      int n, novo_n;
      int stepen;
7      int cifra_levo, cifra_sredina, cifra_desno;

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

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

17     /* Stepen broja 10 sa kojim se mnoze cifre izdvojenog broja. */
19     stepen = 1;

21     /* Nova vrednost broja. */
     novo_n = 0;

23     /* Provera da li u zapisu broja postoje barem tri cifre. */
25     while (n > 99) {
         /* Izdvaja se srednja cifra, cifra desno od nje i cifra levo od
27         nje: npr. za trojku 583 8 je srednja cifra, 3 je cifra
            desno, a 5 cifra levo. */
29         cifra_desno = n % 10;
            cifra_sredina = (n / 10) % 10;
31         cifra_levo = (n / 100) % 10;

33         /* U novi broj se smesta desna cifra. */
            novo_n += cifra_desno * stepen;

35         /* Azurira se vrednost stepena. */
37         stepen = stepen * 10;

39         /* Provera da li je srednja cifra jednaka zbiru leve i desne
            cifre. */
41         if (cifra_levo + cifra_desno == cifra_sredina) {
```

```

43     /* Treba izbaciti srednju cifru, pa broj n se azurira tako
        sto se podeli sa 100. */
45     n = n / 100;
    } else {
47
        /* Inace, zadrzava se srednja cifra i odbacuje se samo
49         poslednja. */
        n = n / 10;
51     }
    }
53
    /* Na novi broj se dodaje preostali dvocifreni ili jednocifreni
55     broj. */
    novo_n = n * stepen + novo_n;
57
    /* Ispisivanje rezultata. */
59     printf("%d\n", novo_n);
61
63     return 0;
}

```

### Rešenje 2.3.30

```

1  #include <stdio.h>
   #include <math.h>
3
   int main()
5  {
       int x;
       int broj_cifara;
       int min_stepen, max_stepen;
       int pom;
       int leva_cifra, desna_cifra;
11      int indikator;

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

       /* Ako je korisnik uneo negativan broj, analizira se njegova
17        apsolutna vrednost. */
       if (x < 0)
19         x = -x;

21      /* Odredjuje se broj cifara u zapisu broja x da bi moglo da se
        izdvajaju istovremeno cifre i sa leve i sa desne strane. */
23      broj_cifara = 0;
       pom = x;
25      while (pom > 0) {

```

## 2 Kontrola toka

```
    pom /= 10;
27     broj_cifara++;
    }

29
    /* Odredjuje se stepen koji stoji uz krajnju levu cifru broja. */
31     max_stepen = (int) pow(10, broj_cifara - 1);

33     /* Indikator je promenljiva koja ukazuje da li je broj palindrom
        ili ne. */
35     indikator = 1;
    while (x != 0 && indikator == 1) {
37         /* Izdvaja se leva cifra. */
        leva_cifra = x / max_stepen;
39         /* Izdvaja se desna cifra. */
        desna_cifra = x % 10;
41         /* Ako su cifre razlicite, odmah se moze zakljuciti da broj
            nije palindrom i prekida se izvršavanje petlje. */
43         if (leva_cifra != desna_cifra) {
            indikator = 0;
45             break;
        }
47         /* Formira se nova vrednost broja x tako sto se odbacuje
            krajnja leva i krajnja desna cifra. */
49         x = (x % max_stepen - x % 10) / 10;
        /* Koriguje se maksimalan stepen tako dobijenog broja - deli se
51         sa 100 jer su odbacene 2 cifre. */
        max_stepen = max_stepen / 100;
53     }

55     /* Ispisuje se rezultat. */
    if (indikator == 1)
57         printf("Broj je palindrom!\n");
    else
59         printf("Broj nije palindrom!\n");

61     return 0;
}
```

### Rešenje 2.3.31

```
1  #include <stdio.h>

3  int main()
{
5     /* Pamtimozastopna dva Fibonacijeva broja i na osnovu njih
        racunamo sledeci. */
7     /* Promenljive prvi i drugi su brojevi koje pamtimo i na osnovu
        njih racunamo treci. */
9     /* Na osnovu teksta zadatka, promenljive prvi i drugi postavljamo
        na 1. */
11    int prvi = 1;
```



```

13  int drugi = 1;
14  int treci;
15  /* Promenljiva pozicija je podatak koji ucitavamo i odnosi se na
    poziciju u Fibonacijevom nizu za koju treba izracunati
    vrednost. */
17  int pozicija;
18  /* Promenljiva i oznacava do koje pozicije smo izracunali
    vrednosti. Kako imamo prve dve vrednosti, ovu promenljivo
    postavljamo na 2. */
21  int i = 2;

23  printf("Unesite poziciju u Fibonacijevom nizu: ");
24  scanf("%d", &pozicija);

25
26  /* Pozicija ne moze biti 0 i ne moze biti negativan broj. */
27  if (pozicija < 1) {
28      printf
29      ("Neispravan unos. Pozicija u Fibonacijevom nizu mora biti
    pozitivan broj koji nije 0!\n");
30      return -1;
31  }

33  while (i < pozicija) {
34      /* Na osnovu dva uzastopna racunamo treci. */
35      treci = prvi + drugi;

37      /* Potom razmenjujemo vrednosti. Uzastopna dva koja pamtimo
    postaju sledeca uzastopna dva broja Fibonacijevog niza. */
39      prvi = drugi;
40      drugi = treci;

41      /* Prelazimo na racunanje sledeceg broja na sledecoj poziciji. */
43      i++;
44  }

45  printf("Trazeni broj je: %d\n", drugi);
47  return 0;
49  }

```

### Rešenje 2.3.32

```

#include<stdio.h>

2
int main()
4 {
    int a0;
    int an, an1;
    printf("Unesi pocetni clan niza brojeva:");
    scanf("%d", &a0);
    if (a0 <= 0) {

```

## 2 Kontrola toka

```
10     printf("Nekorektan unos. Broj mora biti pozitivan.\n");
11     return -1;
12 }
13 printf("%d\n", a0);
14 an = a0;
15 while (an != 1) {
16     if (an % 2) {
17         /* Ukoliko je vrednost izraza an%2 razlicita od nule, izraz
18            se tumaci kao tacan i izvrsavaju se naredbe iz if grane. */
19         an1 = (3 * an + 1) / 2;
20     } else {
21         /* U suprotnom, ukoliko je vrednost izraza an%2 jednaka nuli,
22            izraz se tumaci kao netacan i izvrsavaju se naredbe iz
23            else grane. */
24         an1 = an / 2;
25     }
26     printf("%d\n", an1);
27     an = an1;
28 }
29 return 0;
30 }
```

### Rešenje 2.3.33

```
1  #include <stdio.h>
2  #include <math.h>
3
4  int main()
5  {
6      int format;
7      /* Pomocna promenljiva koja služi kao brojac u petlji. */
8      int i;
9      /* Trenutne vrednosti za sirinu i visinu i pomocna promenljiva za
10         promene u petlji. */
11     double sirina, duzina, nova_duzina;
12     unsigned int konacna_sirina, konacna_duzina;
13
14     printf("Uneti format papira: ");
15     scanf("%d", &format);
16
17     if (format <= 0) {
18         printf("Neispravan unos.\n");
19         return -1;
20     }
21
22     /* duzina/sirina = 1 : sqrt(2) duzina*sirina = 1000x1000mm^2 Na
23        osnovu ovih odnosa dobijamo pocetnu vrednost za sirinu i
24        duzinu, odnosno vrednosti za papir A0. */
25     duzina = sqrt(1000 * 1000 / sqrt(2));
26     sirina = sqrt(2) * duzina;
27 }
```

```

29  /* Kako vec imamo odredjenu sirinu i duzinu za papir A0, petlju
    krecemo od izracunavanja za papir A1, pa brojac i postavljamo
    na 1. */
31  for (i = 1; i <= format; i++) {
    nova_duzina = sirina / 2;
33    sirina = duzina;
    duzina = nova_duzina;
35  }

37  /* Duzina i sirina celi brojevi. */
    konacna_sirina = (unsigned int) sirina;
39    konacna_duzina = (unsigned int) duzina;

41    printf("%u %u\n", konacna_duzina, konacna_sirina);

43    return 0;
}

```

### Rešenje 2.3.34

```

1  #include <stdio.h>

3  int main()
{
5    char c;

7    /* Funkcija getchar ucitava jedan karakter. Naredbom dodele
    (c=getchar()) promenljivoj c bice dodeljena vrednost ascii
    koda unetog karaktera. Obratiti posebnu paznju na zagrade. */
9

11   while ((c = getchar()) != '.') {
    if (c >= 'A' && c <= 'Z')
13       /* Razlika izmedju ascii koda svakog malog i odgovarajuceg
        velikog slova je konstanta koja se moze sracunati izrazom
        'a'-'A' (i iznosi 32). */
15       putchar(c + 'a' - 'A');
    else if (c >= 'a' && c <= 'z')
17       putchar(c - 'a' + 'A');
    else
19       putchar(c);
21   }

23   return 0;
}

```

### Rešenje 2.3.35

```

#include <stdio.h>
2

```

```
int main()
4 {
    char c;

6     /* Inicijalizacija brojaca na 0. */
    int br_v = 0;
    int br_m = 0;
10    int br_c = 0;
    int br_b = 0;
12    int br_k = 0;
    int suma = 0;

14    /* Petlja se završava kada korisnik ne unese karakter, već zada
16    konstantu EOF . Ova konstanta se zadaje kombinacijom tastera
    CTRL+D. U tom slučaju, getchar() vraća -1. */
18    while ((c = getchar()) != EOF) {
        if (c >= 'A' && c <= 'Z')
20            br_v++;
        else if (c >= 'a' && c <= 'z')
22            br_m++;
        else if (c >= '0' && c <= '9') {
24            br_c++;
            /* Kada od promenljive tipa char oduzimamo karakter (ili neku
26            drugu promenljivu tipa char), zapravo se vrši oduzimanje
            njihovih ascii vrednosti i dobija se broj. */
28            suma = suma + c - '0';
        } else if (c == '\t' || c == '\n' || c == ' ')
30            br_b++;

32        br_k++;
    }

34    printf("velika: %d, mala: %d, cifre: %d, beline: %d \n", br_v,
36            br_m, br_c, br_b);
    printf("suma cifara: %d\n", suma);

38    return 0;
40 }
```

### Rešenje 2.3.36

```
#include <stdio.h>

2
int main()
4 {
    /* Promenljiva i je brojac. */
    int n, i;
6    /* Brojaci za svaki od samoglasnika. */
    int broj_a = 0, broj_e = 0, broj_i = 0, broj_o = 0, broj_u = 0;
8    /* Promenljiva c je tekuci ucitani karakter. */
    char c, belina;
10 }
```

```
12 printf("Unesite broj n: ");
13 scanf("%d", &n);
14
15 if (n < 0) {
16     printf("Neispravan unos.\n");
17     return -1;
18 }
19
20 for (i = 0; i < n; i++) {
21     /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
22        pa tek posle procitane beline se cita uneseni karakter. */
23     scanf("%c%c", &belina, &c);
24
25     /* Provera da li je ucitani karakter samoglasnik. */
26     switch (c) {
27         case 'a':
28         case 'A':
29             broj_a++;
30             break;
31         case 'e':
32         case 'E':
33             broj_e++;
34             break;
35         case 'i':
36         case 'I':
37             broj_i++;
38             break;
39         case 'o':
40         case 'O':
41             broj_o++;
42             break;
43         case 'u':
44         case 'U':
45             broj_u++;
46             break;
47     }
48 }
49
50 printf("samoglasnik a: %d\n", broj_a);
51 printf("samoglasnik e: %d\n", broj_e);
52 printf("samoglasnik i: %d\n", broj_i);
53 printf("samoglasnik o: %d\n", broj_o);
54 printf("samoglasnik u: %d\n", broj_u);
55
56 return 0;
57 }
```

## Rešenje 2.3.37

```
1  /* Sa standardnog ulaza se unosi ceo broj n, a zatim i n karaktera.
   Napisati program koji proverava da li se od unetih karaktera
3   moze napisati rec Zima. */

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

7
   int main()
9  {
   int n;
11  int broj_Z, broj_i, broj_m, broj_a;
   char novi_red, c;
13  int i;

15  broj_Z = 0;
   broj_i = 0;
17  broj_m = 0;
   broj_a = 0;

19

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

23  /* Ucitavanje karakter po karakter. */
   for (i = 0; i < n; i++) {
25     printf("Unestite %d. karakter: ", i + 1);
     /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
27     pa tek posle procitane beline se cita uneseni karakter. */
     scanf("%c%c", &novi_red, &c);

29

     /* Analiziramo karakter */
31     switch (c) {
       case 'Z':
33         broj_Z++;
         break;
35         case 'i':
           broj_i++;
37           break;
           case 'm':
39             broj_m++;
             break;
41             case 'a':
               broj_a++;
43               break;
           }
45     }

47     /* Ako u unosu ima barem jedno veliko slovo z i barem po jedno
       malo slovo i, m i a, rec se moze napisati. A u suprotnom ne
49     moze. */
     if (broj_Z && broj_i && broj_m && broj_a) {
51         printf("Moze se napisati rec Zima.\n");
     } else {
```

```
53     printf("Ne moze se napisati rec Zima.\n");
54 }
55
56     return 0;
57 }
```

### Rešenje 2.3.38

```
1  #include <stdio.h>
2  int main()
3  {
4      int n;
5      /* Brojac. */
6      int i;
7      /* Promenljiva u kojoj se cuva suma kubova. */
8      int s;
9
10     printf("Unesite pozitivan ceo broj:");
11     scanf("%d", &n);
12     if (n < 0) {
13         printf("Neispravan unos.\n");
14         return -1;
15     }
16
17     for (s = 0, i = 1; i <= n; i++)
18         s += i * i * i;
19
20     printf("Suma kubova od 1 do %d: %d\n", n, s);
21
22     return 0;
23 }
```

### Rešenje 2.3.39

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int n;
6      /* Brojac. */
7      int i;
8      /* Promenljiva u kojoj se cuva suma kubova. */
9      int s;
10
11     printf("Unesite pozitivan ceo broj:");
12     scanf("%d", &n);
13
14     if (n < 0) {
15         printf("Neispravan unos.\n");
16     }
17 }
```

## 2 Kontrola toka

---

```
16     return -1;
17 }
18
19 i = 1;
20 s = 0;
21
22 for (i = 1; i <= n; i++) {
23     s += i * i * i;
24     printf("i=%d, s=%d\n", i, s);
25 }
26
27 return 0;
28 }
```

### Rešenje 2.3.40

```
1  #include <stdio.h>
2
3  int main()
4  {
5      int n, i;
6      float x, S, stepen;
7
8      printf("Unesite redom brojeve x i n: ");
9      scanf("%f %d", &x, &n);
10
11     if (n < 0) {
12         printf("Neispravan unos.\n");
13         return -1;
14     }
15
16     /* Inicijalizacija sume. */
17     S = 0;
18
19     /* Stepen promenljiva ce sadrzati vrednosti stepena x^n. Pocetna
20        vrednost joj je 1 jer je x^0 = 1. */
21     stepen = 1;
22
23     for (i = 1; i <= n; i++) {
24         stepen = stepen * x;
25         S = S + i * stepen;
26     }
27
28     printf("S=%f\n", S);
29
30     return 0;
31 }
```

### Rešenje 2.3.41



```
1 #include <stdio.h>
3 int main()
4 {
5     unsigned n, i;
6     float x, S, stepen;
7
8     printf("Unesite redom brojeve x i n: ");
9     scanf("%f %u", &x, &n);
10
11     S = 1;
12     stepen = 1;
13     for (i = 1; i <= n; i++) {
14         stepen = stepen * x;
15         S = S + 1 / stepen;
16     }
17
18     printf("S=%f\n", S);
19
20     return 0;
21 }
```

### Rešenje 2.3.42

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     /* Promenljiva i je brojac, promenljiva S cuva izracunatu sumu, a
7        promenljiva clan je tekuci clan niza. */
8     int i;
9     float S;
10    float x, eps;
11    float clan;
12
13    printf("Unesite x: ");
14    scanf("%f", &x);
15
16    printf("Unesite tacnost eps: ");
17    scanf("%f", &eps);
18
19    S = 0;
20    /* Prvi clan sume je 1. */
21    clan = 1;
22    i = 1;
23    while (clan > eps) {
24        S = S + clan;
25        clan = clan * x / i;
26        i++;
27    }
```

```
    }
28     printf("S=%f\n", S);
30     return 0;
32 }
```

### Rešenje 2.3.43

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

4 int main()
5 {
6     /* Promenljiva i je brojac, promenljiva S cuva izracunatu sumu,
7        promenljiva znak moze biti 1 ili -1 i odredjuje znak trenutnog
8        clana sume, a promenljiva clan je tekuci clan niza. */
9     int i, znak;
10    float S;
11    float x, eps, clan;
12
13    printf("Unesite x: ");
14    scanf("%f", &x);
15
16    printf("Unesite tacnost eps: ");
17    scanf("%f", &eps);
18
19
20    S = 0;
21    clan = 1;
22    i = 1;
23    znak = -1;
24
25    /* Kako clanovi sume mogu biti negativni, potrebno je posmatrati
26       apsolutnu vrednost clana. */
27    while (fabs(clan) > eps) {
28        S = S + clan;
29
30        /* Promena znaka. */
31        clan = clan * x / i;
32        clan *= znak;
33
34        i++;
35    }
36
37    printf("S=%f\n", S);
38
39    return 0;
40 }
```

## Rešenje 2.3.44

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main()
5 {
6     int n, i;
7     double x;
8     double stepen = 1;
9     double proizvod = 1;
10
11     printf("Unesite redom brojeve x i n: ");
12     scanf("%lf %d", &x, &n);
13
14     if (n <= 0) {
15         printf("Neispravan unos.\n");
16         return -1;
17     }
18
19     for (i = 0; i < n; i++) {
20         stepen *= x;
21         proizvod *= 1 + cos(stepen);
22     }
23
24     printf("Proizvod = %lf\n", proizvod);
25
26     return 0;
27 }
```

## Rešenje 2.3.45

```
1 #include <stdio.h>
2
3 int main()
4 {
5     int n, i;
6     double Razlomak;
7
8     printf("Unesite prirodan broj: ");
9     scanf("%d", &n);
10
11     if (n <= 0) {
12         printf("Neispravan unos.\n");
13         return -1;
14     }
15
16     Razlomak = n;
17
18     /* Razlomak se izracunava "od nazad", odnosno, krece se od
```

## 2 Kontrola toka

```
19     najnizeg razlomka 1/n i od njega se nadalje formira sledeci,  
21     "visi" razlomak itd. Završava se kada se stigne do koraka 0 +  
21     1/R. */  
23     for (i = n - 1; i >= 0; i--)  
23         Razlomak = i + 1 / Razlomak;  
25     printf("Razlomak = %lf\n", Razlomak);  
27     return 0;  
}
```

### Rešenje 2.3.46

```
1  #include <stdio.h>  
  
3  int main () {  
5      /* Promenljiva i je brojac , promenljiva S cuva izracunatu sumu ,  
5      promenljiva znak moze bito 1 ili -1 i odredjuje znak trenutnog  
7      clana sume , a promenljiva clan je tekuci clan niza. */  
7  
9      int i, znak , n;  
9      float S;  
9      float x, clan;  
11     printf(" Unesite x i n: ");  
11     scanf("%f%d", &x, &n);  
13  
15     if (n <= 0) {  
15         printf(" Neispravan unos .\n");  
15         return -1;  
17     }  
19  
19     S = 1;  
19     clan = 1;  
21     i = 1;  
21     znak = -1;  
23  
25     /* Kako clanovi sume mogu biti negativni , potrebno je posmatrati  
25     apsolutnu vrednost clana. */  
25     while (i <= 2 * n - 1) {  
27  
27         /* Promena znaka. */  
29         /* Svaki clan suma se od prethodnog clana razlikuje za  
29         x^2/(i*(i+1)). */  
31         clan = clan * x * x / (i * (i + 1));  
31         clan *= znak;  
33  
33         S = S + clan;  
35  
35         i += 2;  
37     }  
}
```

```
39     printf("S=%f\n", S);
41     return 0;
}
```

## Rešenje 2.3.47

```
1  #include <stdio.h>
3  int main()
4  {
5      int n, i;
6      /* Promenljiva clan je deo proizvoda i predstavlja 1/i!. */
7      double clan;
8      double S = 1;
9
10     printf("Unesite prirodan broj: ");
11     scanf("%d", &n);
12
13     if (n <= 1) {
14         printf("Neispravan unos.\n");
15         return -1;
16     }
17
18     clan = 1;
19     for (i = 2; i <= n; i++) {
20         clan = clan / i;
21         S *= 1 + clan;
22     }
23
24     printf("S = %lf\n", S);
25
26     return 0;
27 }
```

## Rešenje 2.3.48

```
1  #include <stdio.h>
3  int main()
4  {
5      int n, i, znak = -1;
6      /* Promenljiva clan je deo proizvoda i predstavlja 1*3*5*...*i. */
7      long int clan;
8      long int S = 0;
9
10     printf("Unesite prirodan broj: ");
11     scanf("%d", &n);
```

## 2 Kontrola toka

---

```
13  if (n < 5 || n % 2 == 0) {
14      printf("Neispravan unos.\n");
15      return -1;
16  }
17
18  clan = 1 * 3;
19  for (i = 5; i <= n; i += 2) {
20      clan = znak * clan * i;
21      S += clan;
22  }
23
24  printf("S = %ld\n", S);
25
26  return 0;
27 }
```

### Rešenje 2.3.49

```
#include <stdio.h>
2
3 int main()
4 {
5     int n, i;
6     double P;
7     double x, a;
8
9     printf("Unesite dva relana broja x i a: ");
10    scanf("%lf%lf", &x, &a);
11
12    printf("Unesite prirodan broj: ");
13    scanf("%d", &n);
14
15    if (n <= 0) {
16        printf("Neispravan unos.\n");
17        return -1;
18    }
19
20    P = x;
21    for (i = 0; i < n; i++)
22        P = (P + a) * (P + a);
23
24    printf("Izraz = %lf\n", P);
25
26    return 0;
27 }
```

### Rešenje 2.3.50

*Rešenje (a)*

```
1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n, i, j;
6
7      printf("Unesite broj n: ");
8      scanf("%u", &n);
9
10     for (i = 1; i <= n; i++) {
11         for (j = 1; j <= n; j++)
12             /* U tablici mnozenja vrednost svakog polja je proizvod vrste
13              i kolone u kojoj se nalazi. */
14             printf("%3d ", i * j);
15
16         printf("\n");
17     }
18
19     return 0;
20 }
```

Rešenje (b)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n, i, j;
6
7      printf("Unesite broj n: ");
8      scanf("%u", &n);
9
10     /* Brojac koji broji koliko brojeva je ispisano u jednom redu. */
11     j = 0;
12     for (i = 1; i <= n * n; i++) {
13         printf("%3d ", i);
14         /* Uvecavamo brojac */
15         j++;
16
17         /* Kada je ispisano n brojeva u jednom redu, ispisuje se znak
18          za novi red, da bi ispis krenuo u novom redu i vrednost
19          brojaca j se postavlja na 0 jer u novom redu jos ni jedan
20          broj nije ispisan. */
21         if (j == n) {
22             j = 0;
23             printf("\n");
24         }
25     }
26
27     return 0;
28 }
```

## 2 Kontrola toka

---

28 | }

### Rešenje (c)

```
#include <stdio.h>

2
int main()
4 {
    unsigned int n, i, j;

6
    printf("Unesite broj n: ");
    scanf("%u", &n);

8
    for (i = 1; i <= n; i++) {
        for (j = 0; j < n; j++)
10             if ((j + i) % n == 0)
                printf("%3d", n);
12             else
                printf("%3d", (j + i) % n);
14
        printf("\n");
16
    }
18
20 return 0;
}
```

### Rešenje (d)

```
1 #include <stdio.h>

3 int main()
{
5     unsigned int n, i, j;

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

9
    for (i = 0; i < n; i++) {
        for (j = 0; j < n - i; j++)
11             printf("(%d, %d)", i, j);

13
        printf("\n");
15
    }

17 return 0;
}
```



## Rešenje 2.3.51

Rešenje (a)

```
1 #include <stdio.h>
2
3 int main()
4 {
5     unsigned int n, i, j;
6
7     printf("Unesite broj n: ");
8     scanf("%u", &n);
9
10    for (i = 0; i < n; i++) {
11        /* Kvadrat predstavlja tabelu sa n vrsta gde svaka vrsta sadrzi
12         n polja, a svako polje je isto i predstavlja karakter *. */
13        for (j = 0; j < n; j++)
14            printf("*");
15        printf("\n");
16    }
17
18    return 0;
19 }
```

Rešenje (b)

```
1 #include <stdio.h>
2
3 int main()
4 {
5     unsigned int n, i, j;
6
7     printf("Unesite broj n: ");
8     scanf("%u", &n);
9
10    for (i = 0; i < n; i++) {
11        /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter
12         *, a unutrasnjost kvadrata je karakter blanko. */
13        for (j = 0; j < n; j++)
14            /* Provera da li je ivica. */
15            if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
16                printf("*");
17            else
18                printf(" ");
19        printf("\n");
20    }
21
22    return 0;
23 }
```

## 2 Kontrola toka

---

Rešenje (c)

```
1 #include <stdio.h>
3 int main()
{
5     unsigned int n, i, j;

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

9     for (i = 0; i < n; i++) {
11        /* Kvadrat predstavlja tabelu sa n vrsta gde su ivice karakter
            *, a unutrasnjost kvadrata je karakter blanko osim na
13        mestima na kojima je glavna dijagonala. */
        for (j = 0; j < n; j++)
15            /* Provera da li je ivica ili glavna dijagonala. */
            if (j == 0 || j == n - 1 || i == 0 || i == n - 1 || i == j)
17                printf("*");
            else
19                printf(" ");
            printf("\n");
21    }

23    return 0;
}
```

### Rešenje 2.3.52

```
1 #include <stdio.h>
3 int main()
{
5     unsigned int n, i, j;

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

9     for (i = 0; i < n; i++) {
11        /* Veliko slovo X se moze posmatrati kao dijagonale kvadrata
            (glavna i sporedna). Zato, treba ispisivati blanko na
13        mestima gde nije dijagonala, a karakter * na mestima gde je
            neka od dijagonala. */
        for (j = 0; j < n; j++)
15            /* Provera da li je mesto glavne ili sporedne dijagonale. */
            if (i == j || i + j == n - 1)
17                printf("*");
            else
19                printf(" ");
            printf("\n");
21    }
```

```
    }  
23     return 0;  
25 }
```

### Rešenje 2.3.53

```
1  #include <stdio.h>  
  
3  int main()  
{  
5     unsigned int n, i, j;  
  
7     printf("Unesite broj n: ");  
    scanf("%u", &n);  
9  
    if (n % 2 == 0) {  
11        printf("Pogresan unos.\n");  
        return -1;  
13    }  
  
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  
    return 0;  
25 }
```

### Rešenje 2.3.54

*Rešenje (a)*

```
#include <stdio.h>  
2  
int main()  
4 {  
    unsigned int n, i, j;  
6  
    printf("Unesite broj n: ");  
8    scanf("%u", &n);  
  
10    for (i = 0; i < n; i++) {  
        for (j = 0; j < n - i; j++)
```

## 2 Kontrola toka

---

```
12     printf("*");
13     printf("\n");
14 }
15
16 return 0;
17 }
```

### Rešenje (b)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n, i, j;
6
7      printf("Unesite broj n: ");
8      scanf("%u", &n);
9
10     for (i = 0; i < n; i++) {
11         for (j = 0; j <= i; j++)
12             printf("*");
13         printf("\n");
14     }
15
16     return 0;
17 }
```

### Rešenje (c)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n, i, j;
6
7      printf("Unesite broj n: ");
8      scanf("%u", &n);
9
10     for (i = 0; i < n; i++) {
11         /* Prvo se ispisuju beline koje prethode karakterima *. */
12         for (j = 0; j < i; j++)
13             printf(" ");
14         /* Posle belina se ispisuje potreban broj karaktera *. */
15         for (j = 0; j < n - i; j++)
16             printf("*");
17         printf("\n");
18     }
19
20     return 0;
21 }
```

21 | }

*Rešenje (d)*

```

1  #include <stdio.h>

3  int main()
4  {
5      unsigned int n, i, j;

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

9      /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
11     for (i = 0; i < n; i++) {
12         /* Prvo se ispisuju beline koje prethode karakterima *. */
13         for (j = 0; j < n - i - 1; j++)
14             printf(" ");
15         /* Posle belina se ispisuje potreban broj karaktera *. */
16         for (j = 0; j <= i; j++)
17             printf("*");
18         printf("\n");
19     }

21     return 0;
22 }

```

*Rešenje (e)*

```

1  #include <stdio.h>

2  int main()
3  {
4      unsigned int n, i, j;

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

9      /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
11     for (i = 0; i < n; i++) {
12         /* Prvo se ispisuju beline koje prethode karakterima *. */
13         for (j = 0; j < n - i - 1; j++)
14             printf(" ");
15         /* Posle belina se ispisuje potreban broj karaktera *. */
16         for (j = 0; j <= i; j++)
17             printf("*");
18         printf("\n");
19     }

20 }

```

## 2 Kontrola toka

---

```
22  /* Potrebno je iscrtati i donji deo slike, odnosno donji trougao.
23     Brojac i odredjuje koji red donjeg trougla se trenutno iscrtava.
24     * Kako je prvi red donjeg trougla vec iscrtan (to je poslednji
25     red gornjeg trougla), brojac se postavlja na 1. */
26  for (i = 1; i < n; i++) {
27      /* Prvo se ispisuju beline koje prethode karakterima *. */
28      for (j = 0; j < i; j++)
29          printf(" ");
30      /* Posle belina se ispisuje potreban broj karaktera *. */
31      for (j = 0; j < n - i; j++)
32          printf("*");
33      printf("\n");
34  }
35
36  return 0;
37 }
```

Rešenje (f)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n, i, j;
6      char c, blanko;
7
8      printf("Unesite broj n: ");
9      scanf("%u", &n);
10
11     printf("Unesite karakter c: ");
12     /* Zbog pritiskanja tastera ENTER nakon unosa promenljive broj
13        potrebno je učitati karakter za novi red u promenljivu blanko
14        pre ucitavanja karaktera kojim se iscrtava trougao. */
15     scanf("%c%c", &blanko, &c);
16
17     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
18     for (i = 0; i < n; i++) {
19         /* Iscrtavaju se samo ivice trougla, ostalo se popunjava
20            belinama. */
21         for (j = 0; j <= i; j++)
22             if (i == n - 1 || j == 0 || j == i)
23                 printf("%c", c);
24             else
25                 printf(" ");
26         printf("\n");
27     }
28
29     return 0;
30 }
```

## Rešenje 2.3.55

Rešenje (a)

```
1  #include <stdio.h>
3  int main()
4  {
5      unsigned int n, i, j;
7      printf("Unesite broj n: ");
8      scanf("%u", &n);
9
10     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
11     for (i = 0; i < n; i++) {
12         /* Prvo se ispisuju beline koje prethode karakterima *. */
13         for (j = 0; j < n - i - 1; j++)
14             printf(" ");
15         /* Posle belina se ispisuje potreban broj karaktera *. */
16         for (j = 0; j < 2 * i + 1; j++)
17             printf("*");
18         printf("\n");
19     }
21     return 0;
22 }
```

Rešenje (b)

```
1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n, i, j;
6
7      printf("Unesite broj n: ");
8      scanf("%u", &n);
9
10     /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
11        izracunavanja koliko zvezdica i praznina je potrebno ispisati
12        u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
13        iteraciji petlje. */
14     for (i = n - 1; 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 potreban broj karaktera *. */
19         for (j = 0; j < 2 * i + 1; j++)
20             printf("*");
21         printf("\n");
22     }
```

## 2 Kontrola toka

```
22      /* Posebna paznja mora da se obrati na cinjenicu da su brojac  
24      tipa unsigned int. Problem nastaje kada je i==0 i pokusa se  
26      oduzimanje (i--). Posto su brojevi unsigned, nova vrednost  
28      nece biti -1, vec pozitivan ceo broj. Imajuci to na umu,  
      uslov i>=0 ne moze da se stavi u uslov za for petlju. Mnogo  
      sigurnije je brojace deklarirati da budu tipa int i izbeci  
      ovakvu vrstu problema. */  
30      if (i == 0)  
          break;  
32  }  
34  return 0;  
}
```

Rešenje (c)

```
1  #include <stdio.h>  
  
3  int main()  
{  
5      unsigned int n;  
      int i, j;  
  
7  
      printf("Unesite broj n: ");  
9      scanf("%u", &n);  
  
11     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */  
      for (i = 0; i < n; i++) {  
13         /* Prvo se ispisuju beline koje prethode karakterima *. */  
         for (j = 0; j < n - i - 1; j++)  
15             printf(" ");  
         /* Posle belina se ispisuje potreban broj karaktera *. */  
17         for (j = 0; j < 2 * i + 1; j++)  
             printf("*");  
19         printf("\n");  
      }  
  
21  
      /* Sada se ispisuje donji trougao. Kako je prvi red donjeg  
23      trougla vec ispisan (poslednji red gornjeg trougla), potrebno  
      je naciniti jednu iteraciju manje. */  
  
25  
      /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg  
27      izracunavanja koliko zvezdica i praznina je potrebno ispisati  
      u svakom redu, i se postavlja na n-1 i smanjuje u svakoj  
29      iteraciji petlje. */  
      for (i = n - 2; i >= 0; i--) {  
31         /* Prvo se ispisuju beline koje prethode karakterima *. */  
         for (j = 0; j < n - i - 1; j++)  
33             printf(" ");  
         /* Posle belina se ispisuje potreban broj karaktera *. */
```



```

35     for (j = 0; j < 2 * i + 1; j++)
36         printf("*");
37     printf("\n");
38 }
39
40 return 0;
41 }

```

## Rešenje (d)

```

1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n, i, j;
6
7      printf("Unesite broj n: ");
8      scanf("%u", &n);
9
10     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
11     for (i = 0; i < n; i++) {
12         /* Prvo se ispisuju beline koje prethode karakterima *. */
13         for (j = 0; j < n - i - 1; j++)
14             printf(" ");
15         /* Posle belina se ispisuje sam troougao. Ako je brojac na
16            ivici onda se ispisuje karakter *, a inace praznina.
17            Takodje, proverava se da li se ispisuje poslednji red (i==n)
18            i u njemu se ispisuje svaki drugi put *, a inace praznina.
19            Kako se ispisuje svaki drugi put vrsi se provera j%2 == 0. */
20         for (j = 0; j < 2 * i + 1; j++)
21             if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
22                 printf("*");
23             else
24                 printf(" ");
25         printf("\n");
26     }
27
28     return 0;
29 }

```

## Rešenje (c)

```

1  #include <stdio.h>
2
3  int main()
4  {
5      unsigned int n;
6      int i, j;
7

```

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

/* Brojac i odredjuje koji red slike se trenutno ispisuje. */
for (i = 0; i < n; i++) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
        printf(" ");
    /* Posle belina se ispisuje sam trougao. Ako je brojac na
       ivici onda se ispisuje karakter *, a inace praznina.
       Takodje, proverava se da li se ispisuje poslednji red (i==n)
       i u njemu se ispisuje svaki drugi put *, a inace praznina.
       Kako se ispisuje svaki drugi put vrsi se provera j%2 == 0. */
    for (j = 0; j < 2 * i + 1; j++)
        if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
            printf("*");
        else
            printf(" ");
    printf("\n");
}

/* Sada se ispisuje donji trougao. Kako je prvi red donjeg
   trougla vec ispisan (poslednji red gornjeg trougla), potrebno
   je naciniti jednu iteraciju manje. */

for (i = n - 2; i >= 0; i--) {
    /* Prvo se ispisuju beline koje prethode karakterima *. */
    for (j = 0; j < n - i - 1; j++)
        printf(" ");
    /* Posle belina se ispisuje potreban broj karaktera *. */
    for (j = 0; j < 2 * i + 1; j++)
        if (j == 0 || j == 2 * i)
            printf("*");
        else
            printf(" ");
    printf("\n");
}

return 0;
}
```

### Rešenje 2.3.56

```
1 #include <stdio.h>

3 int main()
4 {
5     unsigned int n, i, j;

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

```

9      /* Strelica se moze posmatrati kao spojena dva pravougla trougla
11         kojima se ispisuje hipotenuza i jedna, donja kateta. */

13     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
14     for (i = 0; i < n; i++) {
15         for (j = 0; j <= i; j++)
16             /* Proverava se da li se ispisuje karakter na hipotenuzi (j
17                == i-1) ili da se ispisuje poslednji red (i == n-1). */
18                 if (j == i || i == n - 1)
19                     printf("*");
20                 else
21                     printf(" ");
22             printf("\n");
23     }

25     /* Potrebno je iscrtati i donji deo slike, odnosno donji trougao.
26        Brojac i odredjuje koji red donjeg trougla se trenutno iscrtava.
27        * Kako je prvi red donjeg trougla vec iscrtan (to je poslednji
28           red gornjeg trougla), brojac se postavlja na 1. */
29     for (i = 1; i < n; i++) {
30         for (j = 0; j < n - i; j++)
31             /* Provera da li se ispisuje hipotenuza. */
32                 if (j == n - i - 1)
33                     printf("*");
34                 else
35                     printf(" ");
36             printf("\n");
37     }

39     return 0;
}

```

### Rešenje 2.3.57

```

1  #include <stdio.h>

3  int main()
4  {
5      unsigned int n;
6      int i, j, k;

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

11     /* Brojac j odredjuje koliko ukupno karaktera (praznina i
12        karaktera *) u svakom redu se ispisuje. U svakom drugom redu
13        ovaj broj se povecava za 2. Na pocetku je 1 (jer se ispisuje
14        samo jedna zvezda). */
15     j = 1;

```

```
17  /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
    for (i = 1; i <= n; i++) {
19      /* U svakom drugom redu broj ispisanih karaktera se uvecava za
        2. */
21      if (i % 2 == 0)
          j += 2;
23      for (k = 0; k < j; k++)
          /* U svakom drugom redu se naizmenicno ispisuje * ili
            praznina. */
25          if (i % 2 == 0) {
27              if (k % 2 == 0)
                  printf("*");
29              else
                  printf(" ");
31          } else
              printf("*");
33      printf("\n");
35  }
37  return 0;
}
```

### Rešenje 2.3.58

```
#include <stdio.h>
2
int main()
4 {
    unsigned int n, m;
    int i, j, k;
6
    printf("Unesite brojeve n i m: ");
    scanf("%u%u", &n, &m);
8
    for (i = 1; i <= m; i++) {
12        /* Za svaki kvadrat se racuna duzina bez poslednje ivice.
            Kvadrat je sastavljen od (m-1) zvezdice i (m-1) praznine
            (praznine se nalaze izmedju zvezdica). Znac ukupna duzina
            je 2*(m-1) karakter, a kako ima n kvadrata, duzina je
            n*2*(m-1). */
14        for (j = 0; j <= n * 2 * (m - 1); j++)
            /* Provera da li se ispisuje prvi ili poslednji red. */
16            if (i == 1 || i == m)
                /* Naizmenicno se ispisuje * i praznina. */
20                if (j % 2 == 0)
                    printf("*");
22                else
                    printf(" ");
24            else
26                /* Na kraju svakog kvadrata (nakon svake (m-1) zvezdice i
```

```

    (m-1) praznine se ispisuje ivica kvadrata. */
28     if (j % (2 * (m - 1)) == 0)
        printf("*");
30     else
        printf(" ");
32     printf("\n");
34 }
36 return 0;
}

```

### Rešenje 2.3.59

```

1  #include <stdio.h>
3  int main()
4  {
5      unsigned int n;
6      int i, j;
7
8      printf("Unesite broj n: ");
9      scanf("%u", &n);
10
11     /* Potrebno je spojiti sve slike u jednu, sliku gornjeg dela
12        romba i sliku donjeg dela romba. */
13
14     /* Brojac i odredjuje koji red slike se trenutno ispisuje. */
15     for (i = 0; i < n; i++) {
16         /* Prvo se ispisuju * koje prethode karakterima -. */
17         for (j = 0; j < n - i; j++)
18             printf("*");
19         /* Posle * se ispisuje potreban karakter -. */
20         for (j = 0; j < 2 * i; j++)
21             printf("-");
22         /* Potom se ispisuju * koje su nakon karaktera -. */
23         for (j = 0; j < n - i; j++)
24             printf("*");
25         printf("\n");
26     }
27
28     /* Sada se ispisuje donji trougao. Kako je prvi red donjeg
29        trougla vec ispisan (poslednji red gornjeg trougla), potrebno
30        je naciniti jednu iteraciju manje. */
31
32     /* Brojac i odredjuje koliko redova se ispisuje. Radi lakseg
33        izracunavanja koliko zvezdica i praznina je potrebno ispisati
34        u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
35        iteraciji petlje. */
36     for (i = n - 2; i >= 0; i--) {
37         /* Prvo se ispisuju * koje prethode karakterima -. */

```

```

    for (j = 0; j < n - i; j++)
39     printf("*");
    /* Posle * se ispisuje potreban karakter -. */
41     for (j = 0; j < 2 * i; j++)
        printf("-");
43     /* Potom se ispisuju * koje su nakon karaktera -. */
    for (j = 0; j < n - i; j++)
45         printf("*");
    printf("\n");
47 }

49 return 0;
}
```

### Rešenje 2.3.60

```

1  #include <stdio.h>

3  int main()
{
5     unsigned int n, i, j;

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

9     /* Slika se sastoji iz dva dela, trougla i kvadrata i svaki deo
11     se nezavisno iscrtava. */

13     /* Prvo se iscrtava krov, odnosno trougao. */
    for (i = 0; i < n - 1; i++) {
15         /* Prvo se ispisuju beline koje prethode karakterima *. */
        for (j = 0; j < n - i - 1; j++)
17             printf(" ");
        for (j = 0; j < 2 * i + 1; j++)
19             if (j == 0 || j == 2 * i)
                printf("*");
21             else
                printf(" ");
23         printf("\n");
    }

25     /* Potom se iscrtava kvadrat. Da bi iscrtavanje bilo lakse
27     istovremeno se ispisuju dva karaktera. */
    for (i = 0; i < n; i++) {
29         for (j = 0; j < n; j++)
            /* Provera da li je ivica. */
31             if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
                printf("* ");
33             else
                printf(" ");
35         printf("\n");
    }
```

```
    }  
37     return 0;  
39 }
```

## Rešenje 2.3.61

```
1  #include <stdio.h>  
3  int main()  
{  
5     unsigned int n, i, j;  
  
7     printf("Unesite broj n: ");  
    scanf("%u", &n);  
9  
    for (i = 1; i <= (n + 1) / 2; i++) {  
11        for (j = i; j <= n + 1 - i; j++)  
            printf("%d ", j);  
13    }  
  
15    return 0;  
}
```

## Rešenje 2.3.62

```
1  #include <stdio.h>  
3  int main()  
{  
5     unsigned int n, i, j;  
  
7     printf("Unesite broj n: ");  
    scanf("%u", &n);  
9  
    for (i = 1; i <= n; i++) {  
11        for (j = 1; j <= n; j++)  
            if (j % i == 1 || i == 1)  
13                printf("%d ", j);  
  
15        printf("\n");  
    }  
17  
    return 0;  
19 }
```

## 2.5 Funkcije

**Zadatak 2.5.1** Napisati funkciju `int kvadrat(int x)` koja računa kvadrat datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 15  
|| Kvadrat broja 15 je 225
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -89  
|| kvadrat broja -89 je 7921
```

[Rešenje 2.5.1]

**Zadatak 2.5.2** Napisati funkciju `int kub(int x)` koja računa kub datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 15  
|| Kub broja 15 je 3375
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -89  
|| Kub broja -89 je -704969
```

[Rešenje 2.5.2]

**Zadatak 2.5.3** Napisati funkciju `unsigned int apsolutna_vrednost(int x)` koja izračunava apsolutnu vrednost broja  $x$ . Napisati program koji učitava jedan ceo broj i ispisuje rezultat poziva funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -34  
|| Apsolutna vrednost: 34
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 5  
|| Apsolutna vrednost: 5
```

[Rešenje 2.5.3]

**Zadatak 2.5.4** Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji učitava tri cela broja i ispisuje rezultat poziva funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 19 8 14  
|| Minimum je: 8
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: -6 11 -12  
|| Minimum je: -12
```



[Rešenje 2.5.4]

**Zadatak 2.5.5** Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja  $x$ . Napisati program koji učitava jedan realan broj  $i$  ispisuje rezultat poziva funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 8.235
|| Razlomljeni deo: 0.235000
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -5.11
|| Razlomljeni deo: 0.110000
```

[Rešenje 2.5.5]

**Zadatak 2.5.6** Napisati funkciju `float stepen(float x, int n)` koja računa vrednost  $n$ -tog stepena realnog broja  $x$ . Napisati program koji učitava realan broj  $x$  i ceo broj  $n$  i ispisuje rezultat rada funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite jedan realan i jedan ceo broj:
|| 4.5 3
|| 4.5000003=91.125000
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite jedan realan i jedan ceo broj:
|| -33.2 5
|| -33.2000015=-40335776.000000
```

[Rešenje 2.5.6]

**Zadatak 2.5.7** Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva broja  $x$  i  $n$  utvrđuje da li je  $x$  neki stepen broja  $n$ . Ukoliko jeste, funkcija vraća izložilac stepena, a u suprotnom vraća  $-1$ . Napisati program koji učitava dva cela pozitivna broja i ispisuje rezultat poziva funkcije. NAPOMENA: *Pretpostaviti da je unos korektan.*

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: 81 3
|| Jeste: 81 = 34
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: 162 11
|| Broj 162 nije stepen broja 11.
```

[Rešenje 2.5.7]

**Zadatak 2.5.8** Napisati funkciju `int faktorijel(int n)` koja računa faktorijel broja  $n$ . Napisati i program koji učitava dva cela broja  $x$  i  $y$  iz intervala  $[0, 12]$  i ispisuje vrednost zbira  $x! + y!$ .

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 4 5  
|| 144
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 18 -5  
|| Greska: pogresan unos!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 6 0  
|| 721
```

[Rešenje 2.5.8]

**Zadatak 2.5.9** Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najveći zajednički delilac primenom Euklidovog algoritma. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva cela broja: 1024 832  
|| Najveci zajednicki delilac je 64
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva cela broja: -900 112  
|| Najveci zajednicki delilac je -4
```

[Rešenje 2.5.9]

**Zadatak 2.5.10** Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato  $n$  vraća zbir recipročnih vrednosti brojeva od 1 do  $n$ . Napisati program koji učitava ceo broj i ispisuje rezultat rada funkcije zaokružen na dve decimale.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesi jedan pozitivan ceo broj: 10  
|| Zbir reciprocnih je 2.93
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesi jedan pozitivan ceo broj: 100  
|| Zbir reciprocnih je 5.19
```

[Rešenje 2.5.10]

**Zadatak 2.5.11** Napisati funkciju `void ispis(float x, float y, unsigned n)` koja za dva realna broja  $x$  i  $y$  i jedan pozitivan ceo broj  $n$  ispisuje vrednosti sinusne funkcije u  $n$  ravnomerno raspoređenih tačaka intervala  $[x, y]$ . Napisati program koji učitava odgovarajuće vrednosti i testira rad ove funkcije.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva realna broja: 7 32
|| Unesite jedan prirodan broj: 10
|| 0.6570 -0.3457 -0.0108 0.3659 -0.6731
|| 0.8922 -0.9945 0.9666 -0.8122

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva realna broja: 20.5 -8.32
|| Unesite jedan prirodan broj: 5
|| -0.8934 -0.8979 -0.1920 0.6658 0.9968

```

[Rešenje 2.5.11]

**Zadatak 2.5.12** Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji učitava ceo broj i ispisuje rezultat na tri decimale.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 461
|| 3.667

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1001
|| 0.500

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -84723
|| 4.800

```

[Rešenje 2.5.12]

**Zadatak 2.5.13** Napisati funkciju `int sadrzi(int x, int c)` koja ispituje da li se cifra `c` nalazi u zapisu celog broja `x`. Funkcija treba da vrati 1 ako se cifra nalazi u broju, a 0 inače. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: 17890 7
|| Cifra se nalazi u broju.

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: 1982 6
|| Cifra se ne nalazi u broju.

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: 17890 26
|| Neispravan unos.

```

*Primer 4*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj i cifru: -1982 9
|| Cifra se nalazi u broju.

```

[Rešenje 2.5.13]

**Zadatak 2.5.14** Napisati funkciju `int broj_neparnih_cifara(int x)` koja određuje broj neparnih cifre u zapisu datog celog broja. Testirati rad ove funkcije u programu koji učitava cele brojeve dok se ne unese nula i ispisuje broj neparnih cifara svakog unetog broja.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite cele brojeve:
2341
Broj neparnih cifara je 2
78
Broj neparnih cifara je 1
800
Broj neparnih cifara je 0
-99761
Broj neparnih cifara je 4
0
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite cele brojeve:
987611
Broj neparnih cifara je 4
135
Broj neparnih cifara je 3
-701
Broj neparnih cifara je 2
602
Broj neparnih cifara je 0
-884
Broj neparnih cifara je 0
79901
Broj neparnih cifara je 4
0
```

[Rešenje 2.5.14]

**Zadatak 2.5.15** Napisati program za ispitivanje svojstava cifara datog celog broja.

- (a) Napisati funkciju `sve_parne_cifre` koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.
- (b) Napisati funkciju `sve_cifre_jednake` koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

Napisati program koji učitava ceo broj i ispisuje da li su sve cifre parne i da li su sve cifre jednake.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 86422
Sve cifre broja su parne.
Cifre broja nisu jednake.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 55555
Sve cifre broja nisu parne.
Cifre broja su jednake.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: -88
Sve cifre broja su parne.
Cifre broja su jednake.
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj i cifru: -342
Sve cifre broja nisu parne.
Cifre broja nisu jednake.
```

[Rešenje 2.5.15]

**Zadatak 2.5.16** Napisati funkciju `int zapis(int x, int y)` koja proverava da li se brojevi  $x$  i  $y$  zapisuju pomoću istih cifara. Funkcija treba da vrati vrednost 1 ako je uslov ispunjen, a 0 ako nije. Napisati i program koji učitava dva cela broja i ispisuje rezultat primene funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 251 125  
|| Uslov je ispunjen!
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 8898 9988  
|| Uslov nije ispunjen!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: -7391 1397  
|| Uslov je ispunjen!
```

[Rešenje 2.5.16]

**Zadatak 2.5.17** Napisati funkciju `int rastuce(int n)` koja ispituje da li su cifre datog celog broja u rastućem poretku. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i ispisuje poruku da li su cifre unetog broja u rastućem poretku.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2689  
|| Cifre su u rastucem poretku!
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 559  
|| Cifre su u rastucem poretku!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 628  
|| Cifre nisu u rastucem poretku!
```

[Rešenje 2.5.17]

**Zadatak 2.5.18** Napisati funkciju `int par_nepar(int n)` koja ispituje da li su cifre datog celog broja naizmenično parne i neparne. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i testira rad funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2749  
|| Broj ispunjava uslov!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -963  
|| Broj ispunjava uslov!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 27449  
|| Broj ne ispunjava uslov!
```

[Rešenje 2.5.18]

**Zadatak 2.5.19** Napisati funkciju `int ukloni_stotine(int n)` koja modifikuje zadati broj tako što iz njegovog zapisa uklanja cifru stotine (ako postoji). Napisati program koji za brojeve koji se unose sve do pojave broja 0 ispisuje odgovarajuće brojeve kojima je uklonjena cifra stotine.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 1210  
|| 110  
|| Unesite broj: 18  
|| 18  
|| Unesite broj: 3856  
|| 356  
|| Unesite broj: 0
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -9632  
|| -932  
|| Unesite broj: 246  
|| 46  
|| Unesite broj: -52  
|| -52  
|| Unesite broj: 0
```

[Rešenje 2.5.19]

**Zadatak 2.5.20** Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sve do pojave broja 0 ispisuje odgovarajuće rotirane brojeve.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 146  
|| 461  
|| Unesite broj: 18  
|| 81  
|| Unesite broj: 3856  
|| 8563  
|| Unesite broj: 7  
|| 7  
|| Unesite broj: 0
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 89  
|| 98  
|| Unesite broj: -369  
|| -693  
|| Unesite broj: -55281  
|| -52815  
|| Unesite broj: 0
```

[Rešenje 2.5.20]

**Zadatak 2.5.21** Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja  $n$ . Napisati program koji učitava ceo broj  $k$  i ispisuje zbir delilaca svakog broja od 1 do  $k$ .

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj k: 6
|| 1 3 4 7 6 12
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj k: -2
|| Greska: pogresan unos!
```

[Rešenje 2.5.21]

**Zadatak 2.5.22** Broj je prost ako je deljiv samo sa 1 i sa samim sobom. Napisati funkciju `int prost (int x)` koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati 1 ako je broj prost i 0 u suprotnom. Napisati program koji za uneti ceo broj  $n$  ispisuje prvih  $n$  prostih brojeva.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 17
|| Broj je prost!
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 24
|| Broj nije prost!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -11
|| Broj nije prost!
```

[Rešenje 2.5.22]

**Zadatak 2.5.23** Napisati funkciju `void prosti_brojevi(int m)` koja ispisuje sve proste brojeve manje od broja  $m$ . Napisati program koji učitava ceo broj veći od 1 i ispisati rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 15
|| 2 3 5 7 11 13
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 9
|| 2 3 5 7
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1
|| Greska: pogresan unos!
```

**Zadatak 2.5.24** Napisati funkciju `double e_na_x(double x, double eps)` koja računa vrednost  $e^x$  kao parcijalnu sumu reda  $\sum_{n=0}^{\infty} \frac{x^n}{n!}$ , pri čemu se sumiranje vrši dok je razlika sabiraka u redu po apsolutnoj vrednosti manja od  $\varepsilon$ . 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 2.5.24]

**Zadatak 2.5.25** Za dati broj može se formirati niz tako da je svaki sledeći član niza dobijen kao suma cifara prethodnog člana niza. Broj je *srećan* ako se dati niz završava jedinicom. Napisati funkciju `int srecan(int x)` koja vraća 1 ako je broj srećan, a 0 u suprotnom. Napisati program koji za uneti prirodan broj  $n$  ispisuje sve srećne brojeve od 1 do  $n$ . NAPOMENA: *Pretpostaviti da je unos korektan*.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 100
|| Srećni brojevi:
|| 1 10 19 28 37 46 55 64 73 82 91 100
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 0
|| Nema srećnih brojeva.
```

[Rešenje 2.5.25]

**Zadatak 2.5.26** Broj  $a$  je Armstrongov ako je jednak sumi  $n$ -tih stepena svojih cifara, pri čemu je  $n$  broj cifara broja  $a$ . Napisati funkciju `int armstrong(int x)` koja vraća 1 ako je broj Armstrongov, odnosno 0 ako nije. Napisati program koji za učitani ceo broj proverava da li je Armstrongov.



*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 153
|| Broj je Armstrongov!

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 1634
|| Broj je Armstrongov!

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 118
|| Broj nije Armstrongov!

```

[Rešenje 2.5.26]

**Zadatak 2.5.27** Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj  $x$  pojavljuje u nizu brojeva koji se unose sve do pojave broja 0. Napisati program koji učitava vrednost broja  $x$  i testira rad napisane funkcije.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj x: 2.84
|| Unesite brojeve: 8.13 2.84 5 21.6 2.84 11.5 0
|| Broj pojavljivanja broja 2.84 je: 2

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj x: -1.17
|| Unesite brojeve: -128.35 8.965 8.968 89.36 0
|| Broj pojavljivanja broja -1.17 je: 0

```

[Rešenje 2.5.27]

**Zadatak 2.5.28** Napisati funkciju `long unsigned fibonacci(int n)` koja računa  $n$ -ti element Fibonačijevog niza. Napisati i program koji učitava ceo broj  $n$  ( $0 \leq n \leq 50$ ) i ispisuje traženi Fibonačijev broj.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| 21

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 65
|| Greska: nedozvoljena vrednost!

```

[Rešenje 2.5.28]

**Zadatak 2.5.29** Napisati funkciju `int konverzija(int c)` koja prebacuje veliko slovo u ekvivalentno malo i obrnuto. Napisati program koji testira ovu funkciju na karakterima koji se unose do pojave znaka EOF.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: ZDRAVO
|| zdravo
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: Dobro jutro, R2D2!
|| dOBRO JUTRO, r2d2!
```

[Rešenje [2.5.29](#)]

**Zadatak 2.5.30** Napisati funkciju `char sifra(char c, int k)` koja za dati karakter  $c$  određuje šifru na sledeći način: ukoliko je  $c$  slovo, šifra je karakter koji se nalazi  $k$  pozicija ispred njega u abecedi. Karakteri koji nisu slova se ne šifruju. Šifrovanje treba da bude kružno, što znači da je, na primer, šifra za karakter  $b$  i pomeraj 2 karakter  $z$ . Napisati program koji učitava karakter po karakter do kraja ulaza i ispisuje šifrovani tekst.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj k: 2
|| Unesite tekst (CTRL+D za prekid):
|| c
|| a
|| 8
|| 8
|| +
|| +
|| Z
|| X
```

[Rešenje [2.5.30](#)]

**Zadatak 2.5.31** Napisati funkciju `int prestupna(int godina)` koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati 1 ako je godina prestupna ili 0 ako nije. Napisati program koji učitava dva cela broja  $g1$  i  $g2$  i ispisuje sve godine iz intervala  $[g1, g2]$  koje su prestupne.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dve godine: 2001 2010
|| Prestupne godine su: 2004 2008
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dve godine: 2005 2015
|| Prestupne godine su: 2008 2012
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dve godine: 2010 2001
|| Greska: pogresan unos!
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dve godine: 2001 2002
|| Nema prestupnih godina u ovom intervalu!
```

[Rešenje 2.5.31]

**Zadatak 2.5.32** Napisati funkciju `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu. Napisati program koji učitava dva cela broja (mesec i godinu) i ispisuje broj dana u datom mesecu. U slučaju nekorektnog unosa ispisati odgovarajuću poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite mesec i godinu: 8
||      1998
|| Broj dana je: 31
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite mesec i godinu: 2
||      2004
|| Broj dana je: 29
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite mesec i godinu: 24
||      2004
|| Neispravan unos.
```

**Zadatak 2.5.33** Napisati funkciju `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan. Napisati program koji učitava tri cela broja (dan, mesec, godinu) i ispisuje da li je datum ispravan ili ne.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 24.8.1998.
|| Datum je ispravan.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 31.4.1789.
|| Datum nije ispravan.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 29.2.2004.
|| Datum je ispravan.
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 29.14.2004.
|| Datum nije ispravan.
```

**Zadatak 2.5.34** Napisati funkciju `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum određuje datum sledećeg dana. Napisati program koji učitava tri cela broja i ispisuje datum sledećeg dana.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 24.8.1998.
|| Datum sledeceg dana je: 25.8.1998.
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 31.12.1789.
|| Datum sledeceg dana je: 1.1.1790.
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 28.2.2003.
|| Datum sledeceg dana je: 1.3.2004.
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite datum: 31.4.2004.
|| Datum nije ispravan.
```

**Zadatak 2.5.35** Napisati funkciju `int od_nove_godine(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do datog datuma. Napisati program koji učitava tri cela broja i ispisuje koliko dana je proteklo od Nove godine.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 24.8.1998.  
|| Broj dana od Nove godine je: 235
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.12.1680.  
|| Broj dana od Nove godine je: 366
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2003.  
|| Broj dana od Nove godine je: 58
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Datum nije ispravan.
```

**Zadatak 2.5.36** Napisati funkciju `int do_kraja_godine(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati program koji učitava tri cela broja i ispisuje broj dana do kraja godine.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 24.8.1998.  
|| Broj dana do Nove godine je: 129
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.12.1680.  
|| Broj dana od Nove godine je: 0
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2004.  
|| Broj dana od Nove godine je: 307
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Datum nije ispravan.
```

[Rešenje 2.5.37]

**Zadatak 2.5.37** Napisati funkciju `int broj_dana_između(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati program koji učitava šest celih brojeva, koji označavaju dva datuma, i na standardni izlaz ispisuje broj dana između ta dva datuma. U slučaju pogrešnog unosa ispisati poruku o grešci.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite prvi datum: 12.3.2008.  
|| Unesite drugi datum: 5.12.2008.  
|| Broj dana između dva datuma je: 268
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite prvi datum: 26.9.1986.  
|| Unesite drugi datum: 2.2.1701.  
|| Broj dana između dva datuma je: 104319
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 24.8.1998.
Unesite drugi datum: 12.10.2010.
Broj dana izmedju dva datuma je: 4432
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 24.8.1998.
Unesite drugi datum: 31.4.2004.
Ulaz nije ispravan.
```

[Rešenje 2.5.37]

**Zadatak 2.5.38** Napisati funkciju `void romb(int n)` koja iscrtava romb čija je stranica dužine  $n$ . Napisati program koji učitava ceo pozitivan broj i ispisuje rezultat poziva funkcije.

### Primer 1

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

### Primer 2

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

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -5
Greska: pogresna dimenzija!
```

[Rešenje 2.5.38]

**Zadatak 2.5.39** Napisati funkciju `void grafikon_h(int a, int b, int c, int d)` koja iscrtava horizontalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i prikazuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 4 1 7 5
****
*
*****
*****
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 8 -2 5 4
Greska: pogresan unos!
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 5 2 2 10
*****
**
**
*****
```

[Rešenje [2.5.39](#)]

**Zadatak 2.5.40** Napisati funkciju `void grafikon_v(int a, int b, int c, int d)` koja iscrtava vertikalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i ispisuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 4 1 7 5
*
*
**
* **
* **
* **
****
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 8 -2 5 4
Greska: pogresan unos!
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 5 2 2 4
*
* *
* *
****
****
```

[Rešenje [2.5.40](#)]

## 2.6 Rešenja

### Rešenje [2.5.1](#)

```
1 #include <stdio.h>
3 int kvadrat(int x)
```

```

5 {
6     /* promenljive u listi argumenata funkcije, kao i one
7        deklarisanе u samoj funkciji, lokalne su za tu funkciju
8        sto znaci da se promenljive x i y neće "videti" nigde izvan
9        funkcije kvadrat (ni u funkciji main ni u funkciji kub)
10    */
11
12    int y;
13    y = x*x;
14    return y;
15 }
16
17 int main()
18 {
19     int a, kv, kb;
20     printf("Unesi ceo broj:");
21     scanf("%d", &a);
22
23     kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
24        funkcije kvadrat */
25
26     printf("Kvadrat broja %d je %d\n", a, kv);
27     return 0;
28 }

```

### Rešenje 2.5.2

```

1 #include <stdio.h>
2
3 int kub(int a)
4 {
5     /*
6        u listi argumenata funkcije mozemo, a ne moramo, imati
7        promenljivu
8        istog naziva kao promenljiva koja je deklarisanа u main
9        funkciji
10       (u ovom slucaju promenljiva a); ova promenljiva se razlikuje
11       od promenljive a deklarisanе u main funkciji i vidljiva je
12       samo unutar funkcije kub
13    */
14    return a*a*a;
15 }
16
17 int main()
18 {
19     int a, kb;
20     printf("Unesi ceo broj:");
21     scanf("%d", &a);
22
23     kb = kub(a); /* promenljivoj kb dodeljujemo povratnu vrednost
24        funkcije kub */
25 }

```

## 2 Kontrola toka

---

```
23     printf("Kub broja %d je %d\n", a, kb);
    return 0;
25 }
```

### Rešenje 2.5.3

```
1  #include <stdio.h>

3  /* Funkcija koja racuna apsolutnu vrednost */
   unsigned int apsolutna_vrednost(int x){
5     /* Kako funkcija vraca unsigned, a x je tipa int, vrsimo kastovanje
       rezultata u tip unsigned */
       return (unsigned)(x<0?-x:x);
7  }

9  int main(){
       int n;

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

15     /* Ispisujemo njegovu apsolutnu vrednost */
17     printf("Apsolutna vrednost: %u\n", apsolutna_vrednost(n));

19     return 0;
   }
```

### Rešenje 2.5.4

```
1  #include <stdio.h>

3  /*
   Funkcija koja racuna minimum tri cela broja
5  */
   int min(int x, int y, int z){
7     int min;

9     min=x;

11    if(min>y)
        min=y;

13    if(min>z)
        min=z;

15    return min;
17 }
}
```



```

19 int main(){
21     int x,y,z;

23     /* Ucitavamo brojeve */
    printf("Unesite brojeve: ");
25     scanf("%d%d%d", &x, &y, &z);

27     /* Pozivamo funkciju i ispisujemo rezultat */
    printf("Minimum je: %d\n", min(x,y,z));
29     return 0;
31 }

```

### Rešenje 2.5.5

```

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

3

/* Funkcija koja vraca razlomljeni deo prosledjenog broja */
5 float razlomljeni_deo(float x){

7     /* Funkcija fabs vraca apsolutnu vrednost realnog broja
   * NAPOMENA: funkcija fabs se nalazi u zaglavlju math.h
   * NAPOMENA2: funkcija abs se nalazi u zaglavlju stdlib.h, ali se
   * koristi samo za cele brojeve!
   */
11    x = fabs(x);

13    /* Razlomljeni deo broja dobijamo tako sto od samog broja oduzmemo
   njegov ceo deo*/
    return x - (int)x;
15 }

17 int main(){
    float n;

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

23     /* Ispisujemo rezultat */
    printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));
25     return 0;
27 }

```

### Rešenje 2.5.6

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija za realan broj x i ceo broj n odredjuje vrednost stepena
   x^n. */
5 float stepen(float a, int b)
6 {
7     float s=1;
8     int i, abs_b = abs(b);
9
10    for(i=0;i<abs_b;i++)
11        s=s*a;
12
13    /* ukoliko je izlozilac b negativan, izracunamo a^|b| i vracamo
       reciprocnu vrednost izracunatog stepena */
14    return b>0 ? s : 1/s;
15 }
16
17 int main()
18 {
19     int n;
20     float x;
21     float s;
22
23     printf("Unesi jedan realan i jedan ceo broj:");
24     scanf("%f%d",&x,&n);
25
26     s = stepen(x,n);
27
28     printf("%f^%d=%f\n",x,n,s);
29
30     return 0;
31 }
```

### Rešenje 2.5.7

```
1 #include <stdio.h>
2
3 /* Funkcija za dva uneta neoznacena broja x i n utvrdjuje da li je x
   neki stepen
   broja n. Ukoliko jeste, funkcija vraca izlozilac stepena, a u
   suprotnom vraca -1. */
4 int je_stepen(unsigned x, unsigned n)
5 {
6     int i=1;
7     int s=n;
8
9     while(s<x)
10    {
11        s=s*n;
12    }
```

```
13     i++;
14 }
15
16     if (s==x)
17         return i;
18
19     return -1;
20 }
21
22 int main()
23 {
24     unsigned x;
25     unsigned n;
26     int st;
27
28     scanf("%u%u",&x,&n);
29
30     st = je_stepen(x,n);
31
32     if (st!=-1)
33         printf("u=%u^%d\n",x,n,st);
34     else
35         printf("u nije stepen broja %u\n",x,n);
36
37     return 0;
38 }
```

### Rešenje 2.5.8

```
1  #include <stdio.h>
2
3  /* Funkcija racuna faktoriyel broja.
4   * Faktoriyel formiramo mnozenjem sa trenutnom vrednoscu broja x,
5   * a zatim smanjujuci tu vrednost za 1.
6   * Ukoliko je x = 5, f = 5 * 4 * 3 * 2 * 1
7   */
8  int faktoriyel(int x) {
9
10     int f = 1;
11     while(x) {
12         f *= x;
13         x--;
14     }
15     return f;
16 }
17
18 int main() {
19
20     int x, y;
21
22     printf("Unesite dva broja: ");
```

## 2 Kontrola toka

---

```
23 scanf("%d%d", &x, &y);
25 /* Provera uslova.
   *
27  * Faktorijel je veoma brza funkcija, tj.
   * s povecanjem broja x, drasticno brzo uvecava se i vrednost x!.
29  * Tip podatka int ima ogranicenje u velicini broja koji moze da
   * sadrzi.
   * Za 13! i vece, int ne bi mogao da sacuva sve cifre potrebne za
   * zapis tako velikog broja,
31  * te bi doslo do prekoracenja.
   *
33  * Slicno, faktorijel nije definisan nad skupom negativnih celih
   * brojeva.
   */
35 if(x < 0 || y < 0 || x > 12 || y > 12) {
   printf("Greska: pogresan unos!\n");
37 }
   else{
39 printf("%d\n", faktorijel(x) + faktorijel(y));
   }
41 return 0;
}
```

### Rešenje 2.5.9

```
1 #include <stdio.h>
3 int euklid(int x, int y)
{
5     int r;
   /* Euklidov algoritam */
7     while(y) /* algoritam se zaustavlja kada vrednost */
   { /* promenljive y postane nula */
9         r=x%y;
         x=y;
11        y=r;
   }
13
   return x; /* nzd je sacuvan u promenljivoj x */
15 }
17 int main()
{
19     int a,b;
     int nzd;
21
   printf("Unesite dva cela broja:");
23     scanf("%d%d", &a,&b);
```

```

25     nzd = euklid(a,b); /* promenljivoj nzd dodeljujemo povratnu
        vrednost funkcije euklid */
27     printf("Najveci zajednicki delilac je %d\n", nzd);
29     return 0;
}

```

### Rešenje 2.5.10

```

1  #include <stdio.h>
3  float zbir_reciprocnih(int n)
4  {
5      float z=0;
6      int i;
7      for(i=1;i<=n;i++)
8          z+=1.0/i; /* da bismo dobili reciprocnu vrednost broja, vazno je
        da izbegnemo celobrojno deljenje dva cela broja */
9      return z; /* tako sto ce npr deljenik biti 1.0 umesto 1 */
10 }
11
12 int main()
13 {
14     int n;
15     printf("Unesi jedan pozitivan ceo broj:\n");
16     scanf("%d", &n);
17     printf("Zbir je %.2f\n", zbir_reciprocnih(n));
18     /* povratna vrednost funkcije zbir_reciprocnih je float; funkciju
        mozemo pozvati u okviru
19     naredbe printf i umesto specifikatora %.2f bice ispisana
        povratna vrednost funkcije
        zbir_reciprocnih zaokruzena na dve decimale */
21     return 0;
22 }

```

### Rešenje 2.5.11

```

1  #include <stdio.h>
2  #include <math.h>
4  void ispis(float x, float y, int n) /* funkcija nema povratnu
        vrednost; zbog toga je povratni tip void */
5  {
6      float i;
7      float korak=(y-x)/(n-1);
8
9      for(i=x;i<=y;i+=korak)
10         printf("%.4f ", sin(i));

```

```
12     printf("\n");
14 }
16 int main()
17 {
18     float a,b;
19     int n;
20     float t;
21     printf("Unesi dva realna broja:");
22     scanf("%f%f",&a,&b);
23     printf("Unesi jedan prirodan broj:");
24     scanf("%u",&n);
26     if (n<=1 || a==b)
27     {
28         printf("Nekorektan unos\n");
29         return -1;
30     }
31     if (b<a) /* u slucaju da je desni kraj intervala manji od levog */
32     {       /* zamenimo im mesta */
33         t=a;
34         a=b;
35         b=t;
36     }
38
40     ispis(a,b,n);
42
43     return 0;
44 }
```

### Rešenje 2.5.12

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija racuna aritmeticku sredinu cifara datog celog broja */
5 float aritmeticka_sredina(int x) {
6
7     /* Proveravamo slucaj broja 0 */
8     if(x==0)
9         return 0;
10
11     /* Brojaci sa svojim pocetnim vrednostima */
12     int zbir_cifara = 0;
13     int broj_cifara = 0;
```

```

15  /* U slucaju da je broj negativan,
16     * ostaci pri deljenju sa 10 bi takodje bili negativni
17     * sto bi se moralo dodatno proveravati.
18     * Stoga je zgodnije posmatrati samo apsolutnu vrednost broja.
19     */
20  x = abs(x);
21
22  /* Izdvajamo cifru po cifru s kraja zapisa broja x.
23     * Uslov while(x > 0) ekvivalentan je uslovu while(x)
24     * Broj x je pozitivan (apsolutna vrednost), pa cifre izdvajati
25     * sve dok ima cifara izdvajanje, tj. broj x nije deljenjem sa 10
26     * postao 0
27     */
28  while(x) {
29
30      zbir_cifara += x % 10;
31      broj_cifara++;
32
33      x /= 10;
34  }
35
36  /* Da nije izvršena implicitna konverzija (kastovanje),
37     * operator / obavljao bi celobrojno deljenje.
38     * Zato je potrebno da bar jedan operand bude ceo broj,
39     * sto je u ovom slucaju deljenik.
40     * Operator kastovanja je unaran, pa ima veci prioritet od /
41     */
42  return (float) zbir_cifara / broj_cifara;
43 }
44
45 int main() {
46
47     int x;
48
49     printf("Unesite ceo broj: ");
50     scanf("%d", &x);
51
52     printf("Aritmeticka sredina cifara broja %d je %.3f\n", x,
53         aritmeticka_sredina(x));
54
55     /* Iako smo u funkciji aritmeticka_sredina menjali broj x,
56        * prosledjivanjem argumenata funkciji pravi se lokalna kopija
57        * promenljive x za tu funkciju,
58        * tako da je ona menjana unutar funkcije aritmeticka_sredina,
59        * a promenljiva x iz funkcije main() ostaje netaknuta.
60        */
61
62     return 0;
63 }

```

## Rešenje 2.5.13

## 2 Kontrola toka

---

```
1 #include<stdio.h>
2 #include<stdlib.h>
3
4 /* Funkcija ispituje da li se cifra c nalazi u zapisu celog broja x.
5  */
6 int sadrzi(int x, int c)
7 {
8     char cifra;
9     x=abs(x);
10    while(x)
11    {
12        cifra = x%10;
13        if (cifra==c)
14            return 1;
15        x/=10;
16    }
17    return 0;
18 }
19 int main()
20 {
21     int x;
22     int c;
23     printf("Unesi jedan ceo broj i jednu cifru:");
24     scanf("%d%d",&x,&c);
25     if (sadrzi(x,c))
26         printf("Cifra %d se nalazi u zapisu broja %d\n",c,x);
27     else
28         printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
29     return 0;
30 }
```

### Rešenje 2.5.14

```
1 #include<stdio.h>
2 #include<stdlib.h>
3
4 int broj_ncifara(int x)
5 {
6     int s=0;
7     char cifra;
8     x = abs(x);
9
10    while(x)
11    {
12        cifra = x%10;
13        s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
14                       neparna,
15                       a 0 kada je cifra parna */
16        x/=10;
17    }
18 }
```



```

17     return s;
19 }

21 int main()
22 {
23     int x;
24     do
25     {
26         scanf("%d",&x);
27         printf("%d\n", broj_ncifara(x));
28     } while(x!=0);
29
30     return 0;
31 }

```

### Rešenje 2.5.15

```

/*
2
a) Napisati funkciju sve_parne_cifre koja ispituje da li se dati ceo
    broj sastoji isključivo iz parnih cifara. Funkcija treba
4 da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.

6 b) Napisati funkciju sve_cifre_jednake koja ispituje da li su sve
    cifre datog celog broja jednake. Funkcija treba
    da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

8 c) Napisati potom glavni program koji na uneti ceo broj primenjuje
    napisane funkcije i ispisuje odgovarajuće poruke.

10 Na primer, za uneti broj 222, program treba da ispise:
12 Sve cifre broja su parne.
    Sve cifre broja su jednake.

14 A za uneti broj -284:
16 Sve cifre broja su parne.
    Broj sadrži različite cifre

18 */
20 #include <stdio.h>
    #include <stdlib.h>

22 int sve_parne_cifre(int x) /* funkcija vraća 1 ako su sve cifre broja
    parne i 0 u suprotnom */
24 {
26     char d;
    x=abs(x);          /* uzimamo apsolutnu vrednost broja za slučaj da je
        broj negativan */
    while (x>0)
28     {

```

```

    d=x%10;          /* izdvajamo cifru broja */
30
    if (d%2==1)      /* u slucaju da je neparna, to znaci da nisu sve
    cifre broja parne */
32        return 0;   /* vracamo 0 */

    x/=10;           /* "uklanjamo" poslednju cifru broja celobrojnim
    deljenjem sa 10 */
34
}

36
return 1;           /* ukoliko se while petlja zavrсила, to znaci da
    uslov d%2==1 nije
38        nijednom bio ispunjen i da su sve cifre broja
    parne; zbog toga
        vracamo 1
40        */
42
}

44 int sve_cifre_jednake(int x) /* funkcija vraca 1 ako su sve cifre
    broja jednake i 0 u suprotnom */
{
46     char d;
    char prva_cifra;
48     x=abs(x);
    prva_cifra = x%10; /* izdvajamo prvu cifru broja */
50     x/=10;          /* broj delimo sa 10 jer smo prvu cifru vec
        izdvojili */

52     while(x)
    {
54         d = x%10;

56         if (d!=prva_cifra)
            return 0;

58         x/=10;

60     }

62     return 1;
}

64 main()
{
66     int x;
    int d;

68     printf("unesi ceo broj:");
70     scanf("%d", &x);

72     if (sve_parne_cifre(x))
        printf("Sve cifre broja su parne\n");
74     else

```

```

    printf("Broj sadrzi bar jednu neparnu cifru\n");
76
    if (sve_cifre_jednake(x))
78        printf("Sve cifre broja su jednake\n");
    else
80        printf("Broj sadrzi razlicite cifre \n");
82 }

```

### Rešenje 2.5.16

```

1  #include <stdio.h>
3  /* Funkcija int zapis(int x, int y) proverava da li su dva cela broja
   napisana
   * pomocu istih cifara, kao i da li se te cifre pojavljuju
5  * isti broj puta.
   * Ideja je sledeca:
7  * iz broja x izdvajaju se redom cifra po cifra s kraja,
   * a zatim se svaka takva cifra trazi i u broju y.
9  * Ukoliko postoji, eliminise se prvi put kada se pojavi (dakle,
   samo jednom).
   * Ukoliko su sve cifre iste (**redosled nije bitan**),
11 * na kraju ce i iz x i iz y biti sve cifre eliminisane",
   * te ostaju nule u oba broja.
13 *
   * Broj novo_y formira se, zbog jednostavnosti, pomocu Heronovog
   obrasca.
15 * Ovaj postupak obradjen je u okviru funkcije int izbaci_cifru(int
   y, int cifra).
   */
17 int izbaci_cifru(int y, int cifra) {
19     int novo_y = 0;
21     int indikator = 0;
     int izdvojena_cifra;
23
     while(y) {
25         izdvojena_cifra = y % 10;
27         /* U slucaju da se cifra razlikuje od one koju treba eliminisati,
           * ili ukoliko je jedna cifra vec eliminisana =>
29         * tekucu cifru ukljuciti prilikom formiranja novog y
           * */
31         if(izdvojena_cifra != cifra || indikator)
33             /* Heronov obrazac.
               * Menja poredak cifara, ali on u ovom slucaju i nije bitan.
35             */
             novo_y = novo_y*10 + izdvojena_cifra;

```

```
37     else
38
39         /* U slucaju da je cifra vec eliminisana,
40          * ne treba je opet eliminisati.
41          * Za svaku pojavu cifre iz x,
42          * eliminiše se jedna odgovarajuća pojava
43          * te cifre iz y.
44          */
45         indikator = 1;
46
47     y /= 10;
48 }
49
50 return novo_y;
51 }
52
53 int zapis(int x, int y) {
54
55     /* Cifra koja se izdvaja iz x, a onda eliminiše iz y */
56     int cifra;
57
58     /* U slucaju da su prosledjeni brojevi negativni */
59     x = abs(x);
60     y = abs(y);
61
62     while(x) {
63
64         cifra = x % 10;
65         x /= 10;
66
67         y = izbacij_cifru(y, cifra);
68
69         /* otkomentarisati donju liniju radi lakšeg praćenja rada
70          programa: */
71         // printf("Iz x izdvojeno: %d\n\tx = %d, y = %d\n\n", cifra, x, y
72         );
73     }
74
75     return (x == 0 && y == 0);
76 }
77
78 int main() {
79
80     int x, y;
81     printf("Unesite dva cela broja: ");
82     scanf("%d%d", &x, &y);
83
84     if(zapis(x, y))
85         printf("Uslov je ispunjen!\n");
86     else
87         printf("Uslov nije ispunjen!\n");
88 }
```

```
87     return 0;
    }
```

### Rešenje 2.5.17

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   /* Funkcija proverava da li se
5   * cifre u zapisu broja nalaze u rastucem poretku.
   *
7   * Situacija od interesa je kada za dve uzastopne cifre to nije
   * slucaj.
   * Tada ne treba proveravati i za ostale cifre,
9   * vec odmah prekinuti izvršavanje funkcije.
   *
11  * Ukoliko funkcija nije ranije prekinuta,
   * to znaci da cifre jesu u rastucem poretku
13  * (odnosno, kako izdvajamo cifre od nazad, u stvari proveravamo
   * opadajuci poredak),
   * te treba vratiti 1.
15  */

17 int rastuce(int n) {
   int tekuca_cifra;
   int prethodna_cifra;
21
   n = abs(n);
23
   /* Prvu cifru (odnosno, poslednju u zapisu broja)
25  * izdvajamo izvan petlje
   * kako bismo mogli da je poredimo sa narednom
27  */
   tekuca_cifra = n % 10;
29   n /= 10;

31   while(n) {

33       /* Cifra koja je bila tekuca u prethodnoj iteraciji petlje,
   * u novoj iteraciji postaje prethodna.
35       *
   * Novoizdvojena cifra je tekuca.
37       */
       prethodna_cifra = tekuca_cifra;
39       tekuca_cifra = n % 10;

41       /* Ukoliko smo naisli na cifre koje kvare rastuci poredak,
   * prekidamo izvršavanje funkcije sa odgovarajucom povratnom
   * vrednoscu 0.
43       */
```

## 2 Kontrola toka

```

    if(prethodna_cifra < tekuca_cifra)
45 return 0;

    /* Inace, nastavljamo sa izdvajanjem cifara */
    n /= 10;
49 }

51 return 1;
}

53
int main() {
55
    int x;
57 printf("Unesite broj: ");
    scanf("%d", &x);
59
    if(rastuce(x))
61     printf("Cifre su u rastucem poretku!\n");
    else
63     printf("Cifre nisu u rastucem poretku!\n");
65
    return 0;
}
```

### Rešenje 2.5.18

```

1 #include <stdio.h>
  #include <stdlib.h>
3
  /* Funkcija proverava da li su dve uzastopne cifre
5   * razlicite parnosti.
   *
7   * Interesantna situacija je ukoliko su dve uzastopne cifre
   * obe parne, odnosno obe neparne.
9   * Ovaj uslov svodimo na poredjenje njihovih ostataka pri deljenju sa
   * 2:
11  * ukoliko su ostaci isti, cifre su iste parnosti,
   * te ne treba dalje proverati da li je uslov zadovoljen,
   * vec odmah prekinuti sa izvršavanjem funkcije.
13  *
   * Ukoliko dve uzastopne cifre ni u jednom slucaju nisu bile iste
   * parnosti,
15  * a izdvojene su sve cifre iz broja x,
   * uslov je ispunjen, pa funkcija vraca 1.
17  */
  int par_nepar(int x) {
19
    int prethodna_cifra;
21    int tekuca_cifra;

23    /* u slucaju da je uneti broj negativan */
```

```

25     x = abs(x);
26
27     /* jednu cifru izdvajamo van petlje
28      * kako bismo mogli da je odmah u petlji poredimo sa narednom
29      */
30     prethodna_cifra = x % 10;
31     x /= 10;
32
33     while(x) {
34
35         tekuca_cifra = x % 10;
36
37         if(tekuca_cifra % 2 == prethodna_cifra % 2)
38             return 0;
39
40         /* tekuca cifra postaje prethodna cifra za narednu iteraciju */
41         prethodna_cifra = tekuca_cifra;
42         x /= 10;
43     }
44
45     return 1;
46 }
47
48 int main() {
49
50     int x;
51     printf("Unesite broj: ");
52     scanf("%d", &x);
53
54     if(par_nepar(x))
55         printf("Broj ispunjava uslov!\n");
56     else
57         printf("Broj ne ispunjava uslov!\n");
58
59     return 0;
60 }

```

### Rešenje 2.5.19

```

1  #include <stdio.h>
2
3  /* Funkcija koja uklanja broj stotina iz broja n */
4  int ukloni_stotine(int n){
5
6      /* Ako je broj izmedju -100 i 100 nema cifru desetica pa onda
7       vracamo isti taj broj */
8      if(n>-100 && n<100)
9          return n;
10     else
11     {
12         /* U suprotnom vracamo broj sa uklonjenom cifrom stotina */
13     }
14 }

```

```
12      /* Odredjujemo znak broja */
14      int znak=(n<0)? -1 : 1;

16      /* I nadalje radimo sa apsolutnom vrednoscu broja */
18      n=abs(n);

20      return znak*((n/1000)*100 + n%100);
21  }
22  }

23  /* Funkcija koja vraca znak broja */
24  int znak(int broj){
25      return broj<0?-1:1;
26  }

27
28  int main(){
29
30      int broj;
31
32      while(1){
33
34          /* Ucitavamo broj sa standardnog ulaza */
35          printf("Unesite broj: ");
36          scanf("%d", &broj);
37
38          /* Broj 0 oznacava kraj rada */
39          if(broj == 0)
40              break;
41
42          /* Ispisujemo rezultat, vodeci racuna da program treba da radi
43             ispravno i za negativne brojeve */
44          printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
45      }
46
47      return 0;
48  }
```

### Rešenje 2.5.20

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

3
4  int rotacija(int n){
5
6      /* U promenljivoj broj pamtimo originalnu vrednost n */
7      int broj, br = 0, znak;
8
9      /* Odredjujemo znak broja */
10     znak=(n<0) ? -1: 1;
```



```
12  /* I nadalje radimo sa apsolutnom vrednoscu broja */
    n=abs(n);
14
16  /* U promenljivoj broj cuvamo kopiju broja n */
    broj=n;
18
20  /* Ako je broj jednocifren, nema potrebe da ga rotiramo. */
    if(n>-10 && n < 10)
        return n;
22
24  /* Petljom izdvajamo cifru po cifru, kako bismo dosli do krajnje
    leve cifre broja
    (one koja treba da postane krajnje desna), npr za n = 1234, treba
    da dobijemo 1,
    zatim da "pomerimo" 234 u levo i da na kraj nalepimo 1 = 2341 */
26
28  /* Na kraju ove petlje, u n se nalazi najlevlja cifra broja (koja
    treba da postane krajnje desna),
    dok se u br nalazi broj cifara unetog broja */
    while(n >=10){
        n/=10;
        br++;
    }
32
34  /*
    Levi deo (234) dobijamo kao n%(10^broj_cifara)
    Zatim levi deo pomnozimo sa 10, da bi dobili 2340
    Zatim na levi deo dodamo desni deo (1) koja se nalazi u
    promenljivoj n
    */
38
40  return znak* ((broj%(int)pow(10, br))*10 + n);
}

42 int main(){
44     int n;
    while(1){
46
48         /* Ucitavamo broj */
        printf("Unesite broj: ");
        scanf("%d", &n);
50
52         /* Ako je uneta 0, izlazimo iz petlje */
        if(n == 0)
            break;
54
56         /* Stampamo broj rotiran za jedno mesto u levo */
        printf("%d\n", rotacija(n));
    }
58
```

```
60     return 0;
    }
```

### Rešenje 2.5.21

```
1  #include <stdio.h>

3  /* Funkcija koja racuna zbir delilaca broja x */
   int zbir_delilaca(int x){
5     int i=0;

7     /* Na pocetku zbir inicijalizujemo na 0 */
     int zbir = 0;

9     /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
11    for(i=1; i<=x; i++){
        if(x % i == 0)
13        zbir += i;
    }

15    /* Vracamo dobijeni zbir */
17    return zbir;
   }

19  int main(){

21     int k, i;

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

27     /* Proveravamo korektnost ulaza */
29     if(k <= 0)
         printf("Greska: pogresan unos!\n");
31     else{

33         /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
         for(i=1; i<=k; i++)
35             printf("%d ", zbir_delilaca(i));

37         printf("\n");
     }

39

41     return 0;
   }
```

## Rešenje 2.5.22

```
1  #include <stdio.h>
2  #include <math.h>
3
4  /* Funkcija vraca 1 ako je broj prost i 0 u suprotnom. */
5  int prost (int x) /* 1-broj je prost, 0-broj nije prost */
6  {
7      int i;
8
9      if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
10         return 1;
11
12     if (x%2==0)          /* parni brojevi nisu prosti */
13         return 0;
14
15     for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
16         if (x%i==0) /* ako je pronadjen, to znaci da broj nije prost */
17             return 0; /* završavamo funkciju */
18
19     /* ukoliko izvršavanje funkcije dodje do poslednje naredbe return,
20        to znaci da broj nije ispunio nijedan od prethodnih uslova
21        (nije ni 2, ni 3, ni paran, niti ima ijednog delioca), odakle
22        sledi da je prost i zbog toga vracamo 1
23     */
24     return 1;
25 }
26
27 int main()
28 {
29     int n;
30     scanf("%d",&n);
31     int i,j;
32
33     i=1; /* kandidat za prost broj */
34     j=0; /* brojac prostih brojeva */
35     while(j<n)
36     {
37         if (prost(i))          /* ako je broj prost */
38         {
39             printf("%d\n", i); /* stampamo ga i */
40             j++;               /* uvecavamo brojac prostih brojeva */
41         }
42         i++; /* bilo da je i prost ili ne, uvecavamo ga za 1 i
43            nastavljamo sa sledecom iteracijom */
44     }
45
46     return 0;
47 }
```

### Rešenje 2.5.24

```
1  #include<stdio.h>
2  #include<math.h>
3
4  /* Funkcija racuna vrednost e^x kao parcijalnu sumu reda
5     suma(x^n/n!), gde indeks n ide od
6     od 0 do beskonacno, pri cemu se sumiranje vrsi dok
7     je razlika sabiraka u redu po apsolutnoj vrednosti
8     manja od eps. */
9  double e_na_x(double x, double eps)
10 {
11     double s=1;
12     double clan=1;
13     int n=1;
14
15     /*
16     parcijalnu sumu formiramo tako sto u svakoj iteraciji petlje
17     promenljivoj s dodamo jedan sabirak sume oblika (x^n)/n! koji
18     cuvamo u promenljivoj clan
19
20     svaki sabirak mozemo da dobijemo na osnovu prethodnog tako sto
21     ga pomnozimo sa x i podelimo sa n, koje predstavlja redni broj
22     sabirka u sumi
23
24     prvi sabirak (kome odgovara n=0) iznosi 1; zbog toga promenljive
25     s i clan inicijalizujemo na vrednost 1
26
27     sumiranje se sprovodi dogod je sabirak po apsolutnoj vrednosti
28     veci od trazene tacnosti eps
29     */
30
31     do
32     {
33         clan = (clan*x)/n;
34         s += clan;
35         n++;
36     } while(fabs(clan)>eps);
37
38     return s;
39 }
40
41 int main()
42 {
43     double x,eps;
44     printf("x=");
45     scanf("%lf", &x);
46     printf("eps=");
47     scanf("%lf", &eps);
48
49     printf("e^x=%f\n", x, e_na_x(x,eps));
50     return 0;
```

```
}

```

### Rešenje 2.5.25

```

1  /*
   Za dati broj moze se formirati niz tako da je svaki sledeci clan niza
   dobijen
3  kao suma cifara prethodnog clana niza. Broj je srecan ako se dati niz
   zavrшава sa
   jedinicom. Napisati program koji za uneti broj odredjuje da li je
   srecan.
5  Na primer:
   - broj 1234 je srecan jer je zbir njegovih cifara 10, dalje zbir
     cifara broja 10 je 1.
7  - broj 999 nije srecan jer je njegov zbir cifara 27, zbir cifara
     broja 27 je 9.
   - broj 991 je srecan, zbir njegovih cifara je 19, zbir cifara broja
     19 je 10, zbir cifara
9  broja 10 je 1.
   - broj 372 nije srecan, zbir njegovih cifara je 12, zbir cifara broja
     12 je 3

11  Napisati funkciju koja vraca 1 ako je broj srecan, a 0 u suprotnom.
13
   Napisati program koji omogucava korisniku da unese prirodan broj,
   poziva funkciju
15  i ispisuje da li je dati broj srecan. Potom traziti od korisnika da
   unese prirodan
   broj n i ispisati sve srecne brojeve od 1 do n.
17  */

19  #include<stdio.h>

21  int zbir_cifara(int x)
   {
23     int s=0;
     char cifra;
25     while(x)
     {
27         cifra = x%10;
         s+=cifra;
29         x/=10;
     }
31     return s;
   }

33  int srecan(int x)
35  {
     int s; /* promenljiva s sadrzi sumu cifara */
37
     do

```

## 2 Kontrola toka

```
39     {
40         s=zbir_cifara(x);
41         x=s; /* kada izracunamo sumu cifara, dodeljujemo je promenljivoj
42             x jer iz te promenljive izdvajamo cifre u funkciji zbir_cifara
43             */
44     } while(x>=10);
45
46     return (x==1);
47 }
48
49 int main()
50 {
51     unsigned n;
52     int i;
53     printf("Unesi jedan neoznaceni broj:");
54     scanf("%u",&n);
55
56     for(i=1;i<=n;i++)
57         if (srecan(i))
58             printf("%d je srecan\n", i);
59
60     return 0;
61 }
```

### Rešenje 2.5.26

```
1 #include <stdio.h>
2
3 /* Funkcija racuna broj x na n-ti stepen */
4 int stepen(int x, int n) {
5
6     int i;
7     /* Promenljiva u kojoj se cuva proizvod broja x sa samim sobom, n
8     puta */
9     int st = 1;
10
11     for(i = 1; i <= n; i++)
12         st *= x;
13
14     return st;
15 }
16
17 /* Funkcija proverava da li je broj Armstrongov. */
18 int armstrong(int x) {
19
20     /* u y se cuva zbir i-tih stepena cifara */
21     int y;
22     /* stepen za koji se proverava */
23     int i = 1;
24     /* prilikom izdvajanja cifara, broj x se menja,
```

```

25  * te treba imati promenlju koja cuva pravu vrednost x
    */
    int original = x;
27
    do {
29
        y = 0;
31        /* Racunamo i-te stepene za svaku cifru,
        * i istovremeno te stepen sabiramo.
33        * Rezultat pamtimo u promenljivoj y.
        */
35        while(x) {

37            y += stepen(x % 10, i);
            x /= 10;
39        }

41        /* x je sada promenjen, pa ga treba vratiti na pravu vrednost. */
        x = original;
43        i++;

45    } while(y < x); /* Petlju vrtimo sve dok je zbir stepena cifara
        manji od datog broja. */

47    /* Ukoliko smo nasli i, takvo da je zbir i-tih stepena cifara
    * jednak upravo broju x, takav broj je Armstrongov,
49    * te izraz x == y vraca 1.
    *
51    * Inace, vraca 0, tj. broj nije Armstrongov.
    */
53    return x == y;
}

55 int main() {

57     int x;
59     printf("Unesite broj: ");
    scanf("%d", &x);

61
    if(armstrong(x))
63         printf("Broj je Armstrongov!\n");
    else
65         printf("Broj nije Armstrongov!\n");

67     return 0;
}

```

### Rešenje 2.5.27

```

#include <stdio.h>
2

```

```
4  /* Funkcija broji koliko puta se realan broj x
   * javlja u nizu unetih brojeva sa tastature.
   *
6  * Brojevi se unose sve do pojave 0,
   * pa treba koristiti do..while petlju,
8  * kako bi bar jedan broj bio unet (makar bio i 0).
   */
10 int prebrojavanje(float x) {
12     /* y prihvata uneti broj sa tastature */
    float y;
14     /* br_pojavljivanja je brojac koji broji koliko puta se broj x
       javlja u unetom nizu brojeva */
    int br_pojavljivanja = 0;
16
18     printf("Unesite brojeve: ");
    do {
20         /* Unosimo broj. */
        scanf("%f", &y);
22
24         /* Poredimo uneti broj sa datim brojem.
           * Ukoliko je unet bas trazeni broj,
           * uvecavamo brojac.
           * */
        if(x == y)
28             br_pojavljivanja++;
30     } while(y); /* Sve dok nije uneta 0 */
32     return br_pojavljivanja;
34 }
36
38 int main() {
    float x;
    int br_pojavljivanja;
40
42     printf("Unesite broj x: ");
    scanf("%f", &x);
44
    br_pojavljivanja = prebrojavanje(x);
    printf("Broj pojavljivanja broja %.2f je: %d\n", x,
46         br_pojavljivanja);
    return 0;
}
```

### Rešenje 2.5.28

---



```
1 #include <stdio.h>

3 /* Funkcija racuna n-ti clan Fibonaccijevog niza.
   * Clanovi ovog niza zadaju se rekurzivno tj. u zavisnosti od
   * prethodnih clanova.
5  * Fibonaccijevi brojevi od 0. do 47. se mogu smestiti u tip int, a
   * kako n moze uzimati vrednosti
   * od 1 do 50, povratni tip funkcije je long int.
7  */
long int fibonaci(int n) {
9
11     int i;

13     /* f0 i f1 su prva dva clana niza */
    int f0 = 1;
    int f1 = 1;
15     /* promenljiva u kojoj se cuvaju opsti clanovi: n+2, n+1. i n-ti
       clan */
    long int fn2, fn1, fn;

17     /* ukoliko treba vratiti multi ili prvi clan,
19     * njih ne treba racunati
       * jer su vec dati.
21     */
    if(n == 0 || n == 1)
23         return 1;

25     /* postavljamo prethodne clanove niza */
    fn = f0;
    fn1 = f1;
27     /* racunamo od drugog clana, pa dok ne dodjemo do n-tog */
    for(i = 2; i <= n; i++) {
29
31         /* izracunamo n+2-i clan niza sabiranjem prethodna dva clana */
        fn2 = fn1 + fn;
33         /* promenimo prethodne clanove niza, zbog naredne iteracije */
        fn = fn1;
35         fn1 = fn2;
    }

37     return fn2;
39 }

41 int main() {

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

47     /* Provera vrednosti za broj n */
    if(n < 0 || n > 50) {
49         printf("Greska: nedozvoljena vrednost!\n");
```

```
    }
51  else{
    printf("%ld\n", fibonaci(n));
53  }

55  return 0;
}
```

### Rešenje 2.5.29

```
1  /*
   . a) Napisati funkciju
3
   int konverzija (int c)
5
   koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.
7
   b) Napisati program koji omogućava korisniku da unese niz karaktera
9   sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
   Na primer, za uneti tekst "Kolokvijum iz Progi je 1.12." program
11  treba da ispise "kOLOVKIJUM IZ pROG1 JE 1.12."
13  */
   #include <stdio.h>
15
   int konverzija(int c)
17  {
    /* kljucna rec return vraca povratnu vrednost funkcije (ako je ima)
       */
19    /* i završava izvršavanje funkcije */

21    if (c>='A' && c<='Z')
        return c+'a'-'A';

23
        if (c>='a' && c<='z')
25            return c-'a'+'A';

27    return c;
   }
29
   int main()
31  {
    int c;
33
    while((c=getchar())!=EOF) /* korisnik unosi karakter po karakter
        do konstante EOF */
35        putchar(konverzija(c)); /* funkcija putchar ispisuje jedan
                                   karakter na standardni izlaz */
37
    return 0;
39  }
```

## Rešenje 2.5.30

```
1  #include <stdio.h>
3  /* Funkcija vraca karakter koji se u abecedi
   * nalazi k mesta pre datog karaktera c
   */
5  char sifra(char c, int k) {
7
   /* Ukoliko je uneto malo slovo ... */
9  if(c >= 'a' && c <= 'z')
   /* Pri tome karakter koji je k pozicija pre datog karaktera
   ispada iz opsega malih slova ... */
11  if(c-k < 'a')
   /* Treba krenuti s drugog kraja abecede, racunajuci i
   preskocena slova.
13     *
   * Na primer, ukoliko je c = 'b' i k = 2
15     * Jedan karakter pre 'b' je 'a'.
   * Dva karaktera pre 'b' je 'z' (kruzno).
17     *
   * Karakter iz prvog dela abecede, koji je preskocen, je 'a'.
19     * Broj preskocenih karaktera iz prvog dela abecede
   * racunamo tako sto izracunamo c - 'a' (rastojanje od datog
   karaktera do malog slova a)
21     * sto je u ovom slucaju 'b' - 'a' = 1.
   *
23     * Ostatak karaktera do k ispisujemo, ali gledavsi unazad od z.
   * Zato racunamo k - (c - 'a') - 1.
25     *
   * Od k oduzimamo rastojanje izmedju c i 'a',
27     * kako bismo dobili preostali broj karaktera koji treba
   preskociti.
   */
29     return 'z' - (k - (c - 'a') - 1);
   else
31     /* U suprotnom, karakter ne ispada iz opsega malih slova, te je
   dovoljno bas njega i vratiti */
   return c-k;
33
   /* Ukoliko je uneto veliko slovo ... */
35  else if(c >= 'A' && c <= 'Z')
   if(c-k < 'A')
37     return 'Z' - (k - (c - 'A') - 1);
   else
39     return c-k;
41
   return c;
```

```
43 }
45 int main() {
47     int k;
48     char c;
49
50     printf("Unesite broj k: ");
51     scanf("%d", &k);
52
53     printf("Unesite tekst (CTRL + D za prekid): ");
54     while((c = getchar()) != EOF)
55         putchar(sifra(c, k));
56
57     return 0;
58 }
```

### Rešenje 2.5.31

```
1  #include<stdio.h>
2
3  /* Funkcija koja proverava da li je godina prestupna */
4  int prestupna(int godina){
5      if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
6          return 1;
7      else
8          return 0;
9  }
10
11 /* Funkcija koja proverava da li postoji prestupna godina u datom
12     intervalu */
13 int postoji_prestupna(int g1, int g2){
14     for(; g1<=g2; g1++){
15         if(prestupna(g1))
16             return 1;
17     }
18     return 0;
19 }
20
21 int main(){
22
23     int g1, g2;
24
25     /* Ucitavamo godine */
26     printf("Unesite dve godine: ");
27     scanf("%d%d", &g1, &g2);
28
29     /* Proveravamo korektnost ulaza */
30     if(g1 < 0 || g2 < 0 || g1>g2){
31         printf("Greska: pogresan unos!\n");
32     }
```

```

else{
33
/* Proveravamo da li uopste postoji prestupna godina u datom
   intervalu */
35 if(postoji_prestupna(g1,g2)){
   /* Ako postoje, ispisujemo ih */
37   printf("Prestupne godine su: ");
   for(; g1<=g2; g1++){
39     if(prestupna(g1))
       printf("%d ", g1);
41   }
   printf("\n");
43 }else{
   /* U suprotnom, stampamo odgovarajucu poruku */
45   printf("Nema prestupnih godina u ovom intervalu!\n");
   }
47 }
   return 0;
49 }

```

Rešenje 2.5.37

Rešenje 2.5.37

Rešenje 2.5.38

```

1  #include<stdio.h>
3  /* Funkcija koja iscrtava romb */
   void romb(int n){
5     int i, j;

7     /* U svakoj liniji */
       for(i=0; i<n; i++){
9
10    /* Prvo ispisujemo n-i-1 razmaka */
11    for(j=0; j<n-i-1; j++)
       printf(" ");
13
14    /* Zatim ispisujemo n zvezdica */
15    for(j=0; j<n; j++)
       printf("*");
17
18    /* Na kraju svake linije stoji oznaka za novi red */
19    printf("\n");
   }
21 }
23

```

```
int main(){
25     int n;

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

31     /* Proveravamo korektnost ulaza i ispisujemo rezultat */
    if(n<=0)
33     printf("Greska: pogresna dimenzija!\n");
    else
35     romb(n);

37     return 0;
}
```

### Rešenje 2.5.39

```
#include<stdio.h>
2
/* Funkcija koja stampa n zvezdica za kojima sledi znak za novi red
   */
4 void stampaj_zvezdice(int n){
    int i;
6     for(i=0; i<n; i++)
        printf("*");
8
    printf("\n");
10 }

12 /* Funkcija koja crta grafikon */
void grafikon_h(int a, int b, int c, int d)
14 {
    int i;
16
    /* Prvo ispisujemo a zvezdica */
18     stampaj_zvezdice(a);

20     /* Zatim u sledecem redu b zvezdica */
    stampaj_zvezdice(b);

22     /* Zatim u sledecem redu c zvezdica */
    stampaj_zvezdice(c);

24     /* Zatim u poslednjem redu d zvezdica */
    stampaj_zvezdice(d);
26
28 }

30 int main(){
32     int a,b,c,d;
```

```

34  /* Ucitavamo vrednosti a,b,c,d */
    printf("Unesite vrednosti: ");
36  scanf("%d%d%d%d", &a, &b, &c, &d);

38  /* Proveravamo korektnost ulaza i ispisujemo rezultat */
    if(a <0 || b<0 || c<0 || d<0){
40  printf("Greska: pogresan unos!\n");
    }else{
42  grafikon_h(a,b,c,d);
    }

44  return 0;
46 }

```

### Rešenje 2.5.40

```

1  #include<stdio.h>

3  int maksimum(int a, int b, int c, int d) {
    int max;

5     max=a;
7     if(b>max)
        max=b;
9     if(c>max)
        max=c;
11    if(d>max)
        max=d;

13    return max;
15 }

17 void stampaj_znak(int polje, int granica) {
    if(polje<granica)
19     printf(" ");
    else
21     printf("*");
}

23 /* Funkcija koja iscrtava vertikalni grafikon */
25 void grafikon_v(int a, int b, int c, int d){
    int i, max;

27    /* Na pocetku je potrebno pronaci najveću od ove četiri vrednosti
        */
29    max=maksimum(a, b, c, d);

31    /* Grafikon ukupno ima max horizontalnih linija */
    for(i=0; i<max; i++){
33

```

```
35     /* U svakoj od horizontalnih linija se nalazi po 4 polja:
    polje za a,b,c i d uspravnu liniju.
    U svako od polja treba da se upise ili * ili belina,
37     u zavisnosti od vrednosti i toga u kojoj liniji se trenutno
    nalazimo
    */
39
    /* Stampamo znak za polje a */
41     stampaj_znak(i, max-a);

43     /* Proveravamo uslov za polje b */
    stampaj_znak(i, max-b);
45
47     /* Proveravamo uslov za polje c */
    stampaj_znak(i, max-c);

49     /* Proveravamo uslov za polje d */
    stampaj_znak(i, max-d);
51

53     /* Na kraju svake horizontalne linije stampamo novi red */
    printf("\n");
    }
55 }

57 int main(){
    int a,b,c,d;
59

    /* Ucitavamo vrednosti a,b,c,d */
61     printf("Unesite vrednosti: ");
    scanf("%d%d%d%d", &a, &b, &c, &d);
63

65     /* Proveravamo korektnost ulaza i stampamo grafikon */
    if(a <0 || b<0 || c<0 || d<0)
        printf("Greska: pogresan unos!\n");
67     else
        grafikon_v(a,b,c,d);
69

    return 0;
71 }
```



# 3

## Predstavljanje podataka

### 3.1 Nizovi

**Zadatak 3.1.1** Ako su  $a = (a_1, \dots, a_n)$  i  $b = (b_1, \dots, b_n)$  vektori dimenzije  $n$ , njihov skalarni proizvod je  $a \cdot b = a_1 \cdot b_1 + \dots + a_n \cdot b_n$ . 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.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju vektora:
5
Unesite koordinate vektora a:
8 -2 0 2 4
Unesite koordinate vektora b:
35 12 5 -6 -1
Skalarni proizvod vektora a i b:
240
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju vektora:
3
Unesite koordinate vektora a:
-1 0 1
Unesite koordinate vektora b:
5 5 5
Skalarni proizvod vektora a i b:
0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju vektora:
120
Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.1]

**Zadatak 3.1.2** Napisati program koji za učitani niz ispisuje:

### 3 Predstavljanje podataka

---

(a) elemente niza koji se nalaze na parnim pozicijama.

(b) parne elemente niza.

Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
6
Unesite elemente niza:
1 8 2 -5 -13 75
Elementi niza na parnim pozicijama:
1 2 -13
Parni elementi niza:
8 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
3
Unesite elemente niza:
11 81 -63
Elementi niza na parnim pozicijama:
11 -63
Parni elementi niza:
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
-4
Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.2]

**Zadatak 3.1.3** Napisati program koji za učitani ceo broj, ispisuje broj pojavljivanja svake od cifara u zapisu tog broja. UPUTSTVO: *Za evidenciju broja pojavljivanja svake cifre pojedinačno, koristiti niz.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj:
2355623
U zapisu broja 2355623, cifra 2 se 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 3.1.3]

**Zadatak 3.1.4** Napisati program koji za dva cela broja  $x$  i  $y$  koja se učitavaju sa standardnog ulaza proverava da li se zapisuju pomoću istih cifara.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 251 125
Brojevi se zapisuju istim ciframa!
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 8898 9988
Brojevi se ne zapisuju istim ciframa!
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: -7391 1397
Brojevi se zapisuju istim ciframa!
```

[Rešenje 3.1.4]

**Zadatak 3.1.5** Napisati program koji učitava karaktere sa standardnog ulaza sve do kraja ulaza i izračunava koliko se puta u unetom tekstu pojavila svaka od cifara, svako malo slovo i svako veliko slovo. Ispisati broj pojavljivanja samo za karaktere koji su se u unetom tekstu pojavili barem jednom. UPUTSTVO: Za evidenciju broja pojavljivanja cifara, malih i velikih 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
```

[Rešenje 3.1.5]

**Zadatak 3.1.6** Sa standardnog ulaza se unosi jedna linija teksta. Napisati program koji izračunava i ispisuje koliko puta se pojavilo svako od slova engleskog alfabeta u unetom tekstu. Ne praviti razliku između malih i velikih slova.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
haHIjkl
a:1 b:0 c:0 d:0 e:0 f:0 g:0 h:2 i:0 j:2 k:1 l:1 m:0
n:0 o:0 p:0 q:0 r:0 s:0 t:0 u:0 v:0 w:0 x:0 y:0 z:0
```

### 3 Predstavljanje podataka

---

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
DanaS j3 _j_utRo laBU78d
a:1 b:0 c:0 d:0 e:0 f:0 g:0 h:2 i:0 j:2 k:1 l:1 m:0
n:0 o:0 p:0 q:0 r:0 s:0 t:0 u:0 v:0 w:0 x:0 y:0 z:0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Sao PaoLo 1998 _JuZna Amerika90
a:5 b:0 c:0 d:2 e:1 f:0 g:0 h:0 i:1 j:1 k:1 l:1 m:1
n:1 o:3 p:1 q:0 r:1 s:1 t:0 u:1 v:0 w:0 x:0 y:0 z:0
```

**Zadatak 3.1.7** Napisati program koji za dva učitana niza  $a$  i  $b$  dimenzije  $n$  formira i na izlaz ispisuje niz  $c$  koji se dobija naizmeničnim raspoređivanjem elemenata nizova  $a$  i  $b$ , tj.  $c = [a_0, b_0, a_1, b_1, \dots, a_{n-1}, b_{n-1}]$ . Pretpostaviti da dimenzija učitanih nizova nije veća od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova:
5
Unesite elemente niza a:
2 -5 11 4 8
Unesite elemente niza b:
3 3 9 -1 17
Rezultujući niz:
2 3 -5 3 11 9 4 -1 8 17
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova:
105
Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.7]

**Zadatak 3.1.8** Sa standardnog ulaza se učitava ceo broj  $n$  (manji od 100) i elementi dvaju nizova  $a$  i  $b$  dimenzije  $n$ . Napisati program koji formira i ispisuje niz  $c$  čiju prvu polovinu čine elementi niza  $b$ , a drugu polovinu elementi niza  $a$ .

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
Unesite elemente niza a: 4 -8 32
Unesite elemente niza b: 5 2 11
5 2 11 4 -8 32
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Unesite elemente niza a: 1 0 -1 0
Unesite elemente niza b: 5 5 5 3
5 5 5 3 1 0 -1 0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 145
Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.8]

**Zadatak 3.1.9** Napisati program koji sa standardnog ulaza učitava 10 celih brojeva i razdvaja ih na parne i neparne tako što parne brojeve upisuje na početak niza, a neparne brojeve na kraj niza. Ispisati niz dobijen na ovaj način. NAPOMENA: *Nije dozvoljeno koristiti pomoćne nizove.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite 10 brojeva:
-2 8 11 53 59 20 17 -8 3 14
Rezultujući niz:
-2 8 20 -8 14 3 17 59 53 11
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite 10 brojeva:
9 142 -9 -278 -69 33 34 28 -6 14
Rezultujući niz:
142 -278 34 28 -6 14 33 -69 -9 9
```

**Zadatak 3.1.10** Napisati program koji učitava dimenziju  $n$  celobrojnog niza  $a$  i njegove elemente, i iz niza  $a$  izbacuje sve elemente koji nisu deljivi svojom poslednjom cifrom. Izuzetak su elementi čija je poslednja cifra 0 koje treba zadržati. Program treba da ispiše izmenjeni niz na standardni izlaz. Niz  $a$  sadrži najviše 100 elemenata.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
9
Unesite elemente niza a:
173 -25 23 7 17 25 34 61 -4612
Niz a nakon izmene:
-25 7 25 61 -4612
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
0
Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.10]

**Zadatak 3.1.11** Napisati program koji u nizu dužine  $n$  (broj manji od 100) čiji se elementi učitavaju sa ulaza eliminiše sve brojeve koji nisu deljivi svojim indeksom. Niz reorganizovati tako da nema *rupa* koje su nastale eliminacijom elemenata i ispisati na standardni izlaz. NAPOMENA: *Nulti element niza treba zadržati jer nije dozvoljeno deljenje nulom.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 10
Unesite elemente niza:
4 2 1 6 7 8 10 2 16 3
4 2 6 16
```

#### Zadatak 3.1.12

Napisati funkcije za rad sa nizovima celih brojeva.

- (a) Napisati funkciju `void ucitaj(int a[], int n)` koja učitava elemente niza `a` dimenzije `n`.
- (b) Napisati funkciju `void stampaj(int a[], int n)` koja štampa elemente niza `a` dimenzije `n`.
- (c) Napisati funkciju `int suma(int a[], int n)` koja računa i vraća sumu elemenata niza `a` dimenzije `n`.
- (d) Napisati funkciju `int prosek(int a[], int n)` koja računa i vraća prosečnu vrednost (aritmetičku sredinu) elemenata niza `a` dimenzije `n`.
- (e) Napisati funkciju `int minimum(int a[], int n)` koja izračunava i vraća minimum elemenata niza `a` dimenzije `n`.
- (f) Napisati funkciju `int pozicija_maksimuma(int a[], int n)` koja izračunava i vraća poziciju maksimalnog elementa u nizu `a` dimenzije `n`. U slučaju više pojavljivanja maksimalnog elementa, vratiti najmanju poziciju.

Napisati program koji testira rad zadatih funkcija. Sa standardnog ulaza učitati dimenziju niza (broj ne veći od 100).

##### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza:
|| 5
|| 2 5 -2 8 11
|| Ucitani niz: 2 5 -2 8 11
|| Suma elemenata niza: 24
|| Prosečna vrednost elemenata niza: 4.80
|| Minimumalni element niza: -2
|| Indeks maksimalnog elementa niza: 4
```

##### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza:
|| -5
|| Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.12]

#### Zadatak 3.1.13

Napisati funkcije za rad sa nizovima celih brojeva.

- (a) Napisati funkciju koja proverava da li niz sadrži zadatu vrednost `m`. Povratna vrednost funkcije je 1 ako je vrednost sadržana u nizu ili 0 ako nije.
- (b) Napisati funkciju koja vraća vrednost prve pozicije na kojoj se nalazi element koji ima vrednost `m` ili `-1` ukoliko element nije u nizu.

- (c) Napisati funkciju koja vraća vrednost poslednje pozicije na kojoj se nalazi element koji ima vrednost  $m$  ili  $-1$  ukoliko element nije u nizu.
- (d) Napisati funkciju koja proverava da li elementi niza čine palindrom.
- (e) Napisati funkciju koja proverava da li su elementi niza uređeni neopadajuće.

Napisati i program koji testira rad napisanih funkcija za uneti broj  $m$  i niz čija dimenzija nije veća od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
7
8 11 -2 14 -2 11 8
Ucitani niz: 8 11 -2 14 -2 11 8
Unesite jedan ceo broj:
11
Niz sadrzi element cija je vrednost 11.
Niz sadrzi element cija je vrednost 11.
Indeks njegovog prvog pojavljivanja u nizu je 1.
Niz sadrzi element cija je vrednost 11.
Indeks njegovog poslednjeg pojavljivanja u nizu je 5.
Elementi niza cine palindrom.
Niz nije sortiran neopadajuće.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
-5
Greska: Nedozvoljena vrednost!
```

[Rešenje ??]

**Zadatak 3.1.14** Napisati funkcije za rad sa nizovima celih brojeva.

- (a) Napisati funkciju koja sve vrednosti niza uvećava za zadatu vrednost  $m$ .
- (b) Napisati funkciju koja obrće elemente niza.
- (c) Napisati funkciju koja rotira niz ciklično za jedno mesto u levo.
- (d) Napisati funkciju koja rotira niz ciklično za  $k$  mesta u levo.

Napisati i program koji testira rad napisanih funkcija za uneti broj  $m$  i niz čija dimenzija nije veća od 100.

### 3 Predstavljanje podataka

---

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
6
7 -3 11 783 26 -19
Unesite jedan ceo broj:
2
Elementi niza nakon uvecanja za 2:
9 -1 13 785 28 -17
Elementi niza nakon obrtanja:
-17 28 785 13 -1 9
Elementi niza nakon rotiranja za 1 mesto ulevo:
28 785 13 -1 9 -17
Unesite jedan pozitivan ceo broj:
3
Elementi niza nakon rotiranja za 3 mesto ulevo:
-1 9 -17 28 785 13
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
252
Greska: Nedozvoljena vrednost!
```

[Rešenje 3.1.14]

**Zadatak 3.1.15** Napisati program koji transformiše uneti niz tako što kvadrira sve negativne elemente niza. Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 6
Unesite elemente niza:
12.34 -6 1 8 32.4 -16
12.34 36 1 8 32.4 256
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 9
Unesite elemente niza:
-8.25 6 17 2 -1.5 1 -7 2.65 -125.2
68.0625 6 17 2 2.25 1 49 2.65 15675.04
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza:
9.53 5 1 4.89
9.53 5 1 4.89
```

[Rešenje 3.1.15]

**Zadatak 3.1.16** Sa standardnog ulaza se učitava dimenzija niza, elementi niza i jedan ceo broj  $k$ . Napisati program koji štampa indekse elemenata koji su deljivi sa  $k$ . Pretpostaviti da je dimenzija niza broj koji nije veći od 100.



*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 10 14 86 20
Unesite broj k: 5
0 3

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 6 14 8 9
Unesite broj k: 5
U nizu nema elemenata koji su deljivi brojem 5!

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza: 8 9 11 -4 8 11
Unesite broj k: 2
0 3 4

```

[Rešenje 3.1.16]

**Zadatak 3.1.17** Napisati program koji učitava dimenziju i elemente niza i štampa niz u kojem su najveći i najmanji element niza razmenili mesta. Ukoliko se najmanji ili najveći element više puta pojavljuju u nizu, uzeti u obzir njihova prva pojavljivanja. Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 5
Unesite elemente niza: 8 -2 11 19 4
8 19 11 -2 4

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 10
Unesite elemente niza:
46 -2 51 8 -5 66 2 8 3 14
46 -2 51 8 66 -5 2 8 3 14

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 145
Greska: Nedozvoljena vrednost!

```

[Rešenje 3.1.17]

**Zadatak 3.1.18** Napisati funkciju `int min_max(int a[], int n)` koja pronalazi indekse najmanjeg i najvećeg elementa u nizu  $a$  dimenzije  $n$  koristeći samo jedan prolaz kroz niz. Funkcija kao povratnu vrednost vraća manji od ta dva indeksa. Napisati program koji testira ovu funkciju za učitane nizove celih brojeva maksimalne dužine 100 elemenata.

### 3 Predstavljanje podataka

---

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza:
7
Unesite elemente niza:
5 8 -4 11 17 89 1
2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza:
3
Unesite elemente niza:
9 11 6
1
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza:
-45
Greska: Nedozvoljena vrednost!
```

**Zadatak 3.1.19** Napisati program koji učitane karaktere (najviše njih 100, učitavaju se sve do pojave karaktera \*) ispisuje u redosledu suprotnom od redosleda čitanja.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: a
Unesite karakter: 8
Unesite karakter: 5
Unesite karakter: Y
Unesite karakter: I
Unesite karakter: o
Unesite karakter: ?
Unesite karakter: *
? o I Y 5 8 a
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: g
Unesite karakter: g
Unesite karakter: 2
Unesite karakter: 2
Unesite karakter: )
Unesite karakter: )
Unesite karakter: *
) ) 2 2 g g
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite karakter: U
Unesite karakter: 4
Unesite karakter: a
Unesite karakter: u
Unesite karakter: *
u a 4 U
```

[Rešenje 3.1.19]

#### Zadatak 3.1.20

Sa standardnog ulaza se unosi broj elemenata niza  $a$  i njegovi elementi. Napisati program koji od datog niza formira niz  $b$  u koji ulaze elementi niza  $a$  koji se pojavljuju tačno tri puta. Pretpostaviti da će uneti niz imati najviše 100 elemenata. Pretpostaviti da je dimenzija niza broj koji nije veći od 100.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza a:
4 11 4 6 8 4 6 6
Elementi niza b: 4 6

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 13
Unesite elemente niza a:
-8 26 7 2 1 1 7 2 2 2 7 5 1
Elementi niza b: 7 1

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 2
Unesite elemente niza a:
9 5
Elementi niza b:

```

[Rešenje 3.1.20]

**Zadatak 3.1.21** Napisati funkciju `int sadrzi_bar_dva(int a[], int na, int b[], int nb)` koja proverava da li niz  $a$  dužine  $na$  sadrži barem dva broja koja se pojavljuju u nizu  $b$  dužine  $nb$ . Napisati i program koji učitava redom dimenzije i elemente nizova  $a$  i  $b$  i ispisuje da li uneti nizovi ispunjavaju traženo svojstvo. Pretpostaviti da će uneti nizovi imati najviše 100 elemenata.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a:
5
Unesite elemente niza a:
5 8 7 -2 6
Unesite broj elemenata niza b:
6
Unesite elemente niza b:
11 -11 7 -7 6
Svojstvo je ispunjeno.

```

**Zadatak 3.1.22** Sa standardnog ulaza se, redom, učitavaju dimenzije i elementi dva niza,  $a$  i  $b$ . Napisati program koji određuje i ispisuje njihovu uniju, presek i razliku (redosled prikaza elemenata nije bitan). Pretpostaviti da će uneti nizovi imati najviše 100 elemenata.

### 3 Predstavljanje podataka

---

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 5
Unesite elemente niza a: 2 8 1 5 2
Unesite broj elemenata niza b: 3
Unesite elemente niza b: 5 7 8
Unija: 2 8 1 5 2 5 7 8
Presek: 5
Razlika: 2 1 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 3
Unesite elemente niza a: 11 4 4
Unesite broj elemenata niza b: 2
Unesite elemente niza b: 18 9
Unija: 11 4 4 18 9
Presek:
Razlika: 11 4 4
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 6
Unesite elemente niza a: 12 7 9 12 5 1
Unesite broj elemenata niza b: 4
Unesite elemente niza b: 1 12 22 12
Unija: 12 7 9 12 5 1 1 12 22 12
Presek: 12 12 1
Razlika: 7 9 5
```

[Rešenje 3.1.22]

**Zadatak 3.1.23** Napisati program koji za učitani niz formira i ispisuje niz koji se dobija izbacivanjem svih neparnih elemenata niza. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Pretpostaviti da će uneti niz imati najviše 100 elemenata.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza: 8 9 15 12
8 12
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 6
Unesite elemente niza: 21 5 3 22 19 188
22 188
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza: 133 129 121 101
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza: 15 -22 -23 13 18 46 14 -31
-22 18 46 14
```

[Rešenje 3.1.23]

**Zadatak 3.1.24** Napisati program koji za učitani niz ispisuje niz koji se dobija izbacivanjem svih elemenata koji su prosti brojevi. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Pretpostaviti da će uneti niz imati najviše 100 elemenata. NAPOMENA: *Broj 1 nije prost.*

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: 11 5 6 48 8
6 48 8
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza: 11 5 19 21
21
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: 12 18 9 31 7
12 18 9
```

*Primer 4*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza: -31 11 -19
```

*Primer 5*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: -2 15 -11 8 7
15 8
```

[Rešenje 3.1.24]

**Zadatak 3.1.25** Napisati funkciju `int prebrojavanje(int a[], int n)` koja izračunava broj elemenata celobrojnog niza  $a$  dužine  $n$  koji su manji od poslednjeg elementa niza. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza: 11 2 4 9
2
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza: 7 2 1 14 65 2 8
4
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: 25 18 29 30 14
0
```

[Rešenje 3.1.25]

### 3 Predstavljanje podataka

---

**Zadatak 3.1.26** Napisati funkciju `int prebrojavanje(int a[], int n)` koja izračunava broj parnih elemenata niza celih brojeva  $a$  dužine  $n$  koji pret-hode maksimalnom elementu niza. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza: 11 2 4 9
0
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza: 7 2 1 14 65 2 8
2
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: 25 18 29 30 14
1
```

[Rešenje 3.1.26]

**Zadatak 3.1.27** Napisati funkciju `int cifre(char s[], int n)` koja izraču-nava broj cifara u nizu karaktera  $a$  dužine  $n$ . Napisati program koji za karaktere koji se unose u zasebnim redovima ispisuje broj unetih cifara. Pretpostaviti da dužina niza neće biti veća od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
4
+
A
u
8
Broj cifara je: 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza:
J
M
a
5
5
-
2
Broj cifara je: 3
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza:
e
k
F
Broj cifara je: 0
```

[Rešenje 3.1.27]

**Zadatak 3.1.28** Napisati funkciju `int zbir(int a[], int n, int i, int j)` koja računa zbir elemenata niza celih brojeva  $a$  dužine  $n$  od pozicije  $i$  do pozicije  $j$ . Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: 11 5 6 48 8
Unesite vrednosti za i i j: 0 2
Zbir je: 22
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza: -2 8 1
Unesite vrednosti za i i j: 8 12
Greska: Nekorektne vrednosti granica!
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza: -2 5 9 11 6 -3 -4
Unesite vrednosti za i i j: 2 5
Zbir: 23
```

[Rešenje 3.1.28]

**Zadatak 3.1.29** Napisati funkciju `float zbir_pozitivnih(float a[], int n, int k)` koja izračunava zbir prvih  $k$  pozitivnih elemenata realnog niza  $a$  dužine  $n$ . Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza:
2.34 1 -12.7 5.2 -8 -6.2 7 14.2
Unesite vrednost za k: 3
Zbir je: 8.54
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza:
-6.598 -8.14 -15
Unesite vrednost za k: 4
Zbir je: 0.00
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza:
-35.11 5.29 -1.98 12.1 12.2 -3.33 -4.17
Unesite vrednost za k: 15
Zbir: 29.59
```

[Rešenje 3.1.29]

### 3 Predstavljanje podataka

---

**Zadatak 3.1.30** Napisati funkciju `void kvadriranje(float a[], int n)` koja kvadrira elemente realnog niza  $a$  dužine  $n$  koji se nalaze na parnim pozicijama. Napisati program koji transformiše na ovaj način uneti niz. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza:
2.34 1 -12.7 5.2 -8 -6.2 7 14.2
5.4756 1 161.29 5.2 64 -6.2 49 14.2
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza:
-6 -8.14 -15
36 -8.14 225
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 1
Unesite elemente niza:
-35.11
1232.71
```

[Rešenje 3.1.30]

**Zadatak 3.1.31** Napisati funkciju `int blizu_3(int a[], int n)` koja pronalazi i vraća indeks elementa niza koji je po vrednosti najbliži aritmetičkoj sredini onih elemenata niza koji su deljivi brojem tri. Napisati program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
1 2 3 4 5
2
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
3 6 2 4 7
3
```

**Zadatak 3.1.32** Napisati funkcije za rad sa nizovima celih brojeva.

- Napisati funkciju koja izbacuje poslednji element niza.
- Napisati funkciju koja izbacuje prvi element niza. Zadatak rešiti na dva načina: čuvanjem redosleda elemenata i premeštanjem poslednjeg elementa niza na upražnjenu poziciju.
- Napisati funkciju koja izbacuje element sa date pozicije  $k$ .
- Napisati funkciju koja izbacuje sva pojavljivanja datog elementa  $x$  iz niza.
- Napisati funkciju koja ubacuje dati element  $x$  na kraj niza.



- (f) Napisati funkciju koja ubacuje dati element  $x$  na početak niza.
- (g) Napisati funkciju koja ubacuje dati element  $x$  na datu poziciju  $k$ .

Napisati program koji testira rad zadatih funkcija. Sa standardnog ulaza učitati dimenziju niza (broj ne veći od 100).

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza:
8
2 5 -2 16 33 19 8 11
Niz posle izbacivanja poslednjeg elementa: 2 5 -2 16 33 5 8
Niz nakon izbacivanja prvog elementa: 5 -2 16 33 5 8
Unesite poziciju elementa za izbacivanje:
3
Niz nakon izbacivanja 3. elementa: 5 -2 16 5 8
Unesite element cije pojavljivanje treba izbaciti:
5
Niz nakon izbacivanja elementa 5: -2 16 8
Unesite element koji treba ubaciti u niz:
19
Niz nakon ubacivanja elementa 19 na kraj: -2 16 8 19
Niz nakon ubacivanja elementa 19 na pocetak: 19 -2 16 8 19
Unesite poziciju na koju treba ubaciti element:
2
Niz nakon ubacivanja elementa 19 na poziciju 2: 19 -2 19 16 8 19
```

**Zadatak 3.1.33** Napisati funkcije za rad sa nizovima celih brojeva.

- (a) Napisati funkciju koja određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva.
- (b) Napisati funkciju koja određuje dužinu najvećeg neopadajućeg podniza datog niza celih brojeva.
- (c) Napisati funkciju koja određuje da li se jedan niz javlja kao podniz uzastopnih elemenata drugog niza.
- (d) Napisati funkciju koja određuje da li se jedan niz javlja kao podniz elemenata drugog niza (elementi ne moraju da budu uzastopni, ali je redosled pojavljivanja isti).
- (e) Napisati funkciju koja izbacuje višestruka pojavljivanja elemenata iz datog niza brojeva. Zadatak rešiti na dva načina: zadržavanjem prvog pojavljivanje elementa i zadržavanjem poslednjeg pojavljivanje elementa.

**Zadatak 3.1.34** Napisati funkciju koja iz zadatog niza izbacuje sve elemente koji su deljivi svojim indeksom. Niz reorganizovati tako da nema *rupa* koje su

### 3 Predstavljanje podataka

---

nastale izbacivanjem elemenata. Povratna vrednost funkcije je nova dimenzija niza. Napisati program koji za učitni niz (dimenzije manje od 100) ispisuje niz dobijen nakon poziva funkcije. NAPOMENA: *Element na nultoj poziciji niza zadržati jer nije dozvoljeno deljenje nulom.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 10
Unesite elemente niza:
4 2 1 6 7 10 8 2 16 27
4 1 7 8 2 16
```

**Zadatak 3.1.35** Za celobrojni niz  $a$  dimenzije  $n$  kažemo da je *permutacija* ako sadrži sve brojeve od 1 do  $n$ .

- (a) Napisati funkciju `void brojanje(int a[], int b[], int n)` koja na osnovu celobrojnog niza  $a$  dimenzije  $n$  formira niz  $b$  tako što  $i$ -ti element niza  $b$  odgovara broju pojavljivanja vrednosti  $i$  u nizu  $a$ .
- (b) Napisati funkciju `int permutacija(int a[], int n)` koja proverava da li je zadati niz permutacija. Funkcija vraća vrednost 1 ako je svojstvo ispunjeno, odnosno 0 ako nije. UPUTSTVO: *Koristiti funkciju brojanje iz tačke (a).*

Napisati program koji sa standardnog ulaza učitava dimenziju niza (broj manji od 100) i elemente niza i ispisuje da li je uneti niz permutacija ili ne.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
1 5 4 3 2
Uneti niz je permutacija.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 6
Unesite elemente niza:
2 3 3 1 1 5
Uneti niz nije permutacija.
```

[Rešenje 3.1.35]

## 3.2 Rešenja

### Rešenje 3.1.1

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

```

4  /* Predprocesorska direktiva kojom se definise maksimalni broj
   *   elemenata niza */
6  #define MAX 100
8
6  int main()
   {
8      int a[MAX];
      int b[MAX];
10     int n;
      int i;
12     int skalarni_proizvod;

14
      /* Ucitava se dimenzija vektora i proverava njena ispravnost */
16     printf("Unesite dimenziju vektora: ");
      scanf("%d", &n);
18     if (n<1 || n>100)
       {
20         printf("Nedozvoljena vrednost!\n");
         return -1;
22     }

24     /* Ucitavaju se koordinate vektora */
      printf("Unesite koordinate vektora a: ");
26     for (i=0; i<n; i++)
       {
28         scanf("%d", &a[i]);
       }

30
      printf("Unesite koordinate vektora b: ");
32     for (i=0; i<n; i++)
       {
34         scanf("%d", &b[i]);
       }

36
      /* Izracunava se skalarni proizvod po zadataj formuli */
38     skalarni_proizvod=0;

40     for (i=0; i<n; i++)
         skalarni_proizvod = skalarni_proizvod + a[i]*b[i];

42
      /* I ispisuje se njegova vrednost */
44     printf("Skalarni proizvod vektora a i b: %d\n",skalarni_proizvod);

46     return 0;
   }

```

### Rešenje 3.1.2

```

#include <stdio.h>
2

```

```

4  #define MAX 100
6  int main()
7  {
8      int a[MAX];
9      int n;
10     int i;
11
12     /* Ucitava se dimenzija niza i proverava se njena ispravnost. */
13     printf("Unesi dimenziju niza:\n");
14     scanf("%d", &n);
15     if (n<1 || n>MAX)
16     {
17         printf("Nedozvoljena vrednost!\n");
18         return -1;
19     }
20
21     /* Ucitavaju se elementi niza */
22     printf("Unesi elemente niza:\n");
23     for (i=0; i<n; i++)
24     {
25         scanf("%d", &a[i]);
26     }
27
28     /* Ispisuju se elementi niza na parnim pozicijama */
29     printf("Elementi niza na parnim pozicijama:\n");
30     for (i=0; i<n; i+=2)
31     {
32         printf("%d ", a[i]);
33     }
34     printf("\n");
35
36     /* Ispisuju se parni elementi niza */
37     printf("Parni elementi niza:\n");
38     for (i=0; i<n; i++){
39         if (a[i]%2==0){
40             printf("%d ", a[i]);
41         }
42     }
43     printf("\n");
44
45     return 0;
46 }
```

#### Rešenje 3.1.3

```

1  #include<stdio.h>
2  #include<stdlib.h>
3
4  int main()
```

```
5 {
6     int x;
7     int brojaci[10];
8     char cifra;
9     int original;
10    int i;
11
12    /* Ucitava se ceo broj sa standardnog ulaza */
13    printf("Unesite ceo broj:\n");
14    scanf("%d",&x);
15
16    /* Cuva se njegova originalna vrednost zbog finalnog ispisa */
17    original = x;
18
19    /* I nadalje posmatra apsolutna vrednost */
20    x = abs(x);
21
22    /* Svaki element niza brojaci predstavlja broj za jednu od
23       cifara:
24       brojac[0] predstavlja broj nula u zapisu broja x
25       brojac[1] predstavlja broj jedinica u zapisu broja x
26       ...
27       brojac[9] predstavlja broj devetki u zapisu broja x
28    */
29
30    /* Brojaci se na pocetku inicijalizuju nulama */
31    for(i=0;i<10;i++){
32        brojaci[i]=0;
33    }
34
35    /* Sve dok ima cifara u zapisu broja x */
36    do
37    {
38        /* Izdvaja se krajnja desna cifra */
39        cifra = x%10;
40
41        /* Uvecava se njen broj pojavljivanja */
42        brojaci[cifra]++;
43
44        /* I prelazi se na analiziranje sledece cifre */
45        x/=10;
46    } while(x);
47
48    /* Ispisuju se informacije o ciframa koje se nalaze u zapisu broja
49       x */
50    for(i=0; i<10; i++){
51        if(brojaci[i]){
52            printf("U zapisu broja %d, cifra %d se pojavljuje %d puta\n",
53                original, i, brojaci[i]);
54        }
```

```
    }  
55  
57     return 0;  
}
```

#### Rešenje 3.1.4

```
1  #include <stdio.h>  
3  #define BROJ_CIFARA 10  
5  int main()  
{  
7     char c;  
    int cifrex[BROJ_CIFARA], cifrey[BROJ_CIFARA];  
9     int x, y, i, indikator;  
  
11    /* Ucitavaju se brojevi x i y */  
    printf("Unesite dva broja: ");  
13    scanf("%d%d", &x, &y);  
  
15    /* Za slucaj da su unete vrednosti negativne, posmatra se njihova  
       apsolutna vrednost.  
       Ovo je opravdano iz razloga sto se brojevi x i -x zapisuju istim  
       ciframa. */  
17    x=abs(x);  
    y=abs(y);  
19  
    /* Niz cifrex predstavlja brojace za cifre broja x.  
       Niz cifrey predstavlja brojace za cifre broja y.  
       Na pocetku se ovi nizovi inicijalizuju nulama. */  
23    for(i=0;i<BROJ_CIFARA;i++)  
    {  
25        cifrex[i] = 0;  
        cifrey[i] = 0;  
27    }  
  
29    /* Analiziraju se cifre broja x */  
    while(x)  
31    {  
        c = x%10;  
33        cifrex[c]++;  
        x /= 10;  
35    }  
  
37    /* Analiziraju se cifre broja y */  
    while(y)  
39    {  
        c = y%10;  
41        cifrey[c]++;
```

```

43     y /= 10;
44 }
45
46 /* Promenljiva indikator služi za praćenje da li su oba broja
47    sastavljena od istih cifara. */
48 indikator = 1;
49
50 for(i=0; i<BROJ_CIFARA; i++){
51     /* Ako se broj pojavljivanja cifre i u zapisu broja x razlikuje
52        od broja pojavljivanja cifre i u
53        zapisu broja y, brojevi se ne zapisuju istim ciframa. Zato se
54        vrednost indikatora može postaviti na
55        0 i prekinuti dalje upoređivanje broja pojavljivanja. */
56     if(cifrey[i] != cifrex[i])
57     {
58         indikator = 0;
59         break;
60     }
61 }
62
63 /* Ako je vrednost promenljive indikator ostala 1, to znači da u
64    petlji nije pronađena cifra
65    koja se ne pojavljuje isti broj puta u zapisima brojeva x i y.
66    Zato se može zaključiti da se brojevi
67    zapisuju istim ciframa. */
68 if(indikator)
69     printf("Brojevi se zapisuju istim ciframa!\n");
70 else
71     printf("Brojevi se ne zapisuju istim ciframa!\n");
72
73 return 0;
74 }

```

### Rešenje 3.1.5

```

1 #include <stdio.h>
2
3 int main()
4 {
5     /* Niz u kojem će se čuvati informacije o broju pojavljivanja
6        cifara */
7     int cifre[10];
8
9     /* Niz u kojem će se čuvati informacije o broju pojavljivanja malih
10        slova */
11     int mala_slova[26];
12
13     /* Niz u kojem će se čuvati informacije o broju pojavljivanja
14        velikih slova */
15     int velika_slova[26];

```

```
14  int c, i;

16  /* Brojaci karaktera se na pocetku inicijalizuju nulama */
17  for(i=0;i<10;i++){
18      cifre[i]=0;
19  }

20
21  for(i=0;i<26;i++)
22  {
23      mala_slova[i]=0;
24      velika_slova[i]=0;
25  }

26
27  /* Ucitavaju se karakteri sve do kraja ulaza */
28  while((c = getchar()) != EOF)
29  {
30      /* Ako je procitani karakter veliko slovo ... */
31      if (c>='A' && c<='Z'){
32          velika_slova[c-'A']++;
33      }
34      else{
35          /* Ako je procitani karakter malo slovo ... */
36          if (c>='a' && c<='z'){
37              mala_slova[c-'a']++;
38          }
39          else{
40              /* Ako je procitani karakter cifra ... */
41              if(c >='0' && c <= '9'){
42                  cifre[c-'0']++;
43              }
44          }
45      }
46  }

47
48  /* Ispisuju se trazene informacije */
49  for(i = 0; i < 10; i++){
50      if (cifre[i]!=0)
51          printf("Karakter %c se pojavljuje %d puta\n", '0' + i, cifre[i]);
52  }

53
54  for(i = 0; i < 26; i++){
55      if (mala_slova[i]!=0)
56          printf("Karakter %c se pojavljuje %d puta\n", 'a' + i,
57              mala_slova[i]);
58  }

59
60  for(i = 0; i < 26; i++){
61      if (velika_slova[i]!=0)
62          printf("Karakter %c se pojavljuje %d puta\n", 'A' + i,
63              velika_slova[i]);
```



```
62     }  
64     return 0;  
66 }
```

### Rešenje 3.1.6

### Rešenje 3.1.7

```
#include <stdio.h>  
2  
#define MAX 100  
4  
int main()  
6 {  
    int a[MAX];  
    int b[MAX];  
8  
    /* Rezultujući niz ima najviše 2*MAX elemenata */  
    int c[2*MAX];  
12  
    int n;  
    int i,j;  
14  
16  
    /* Učitava se dimenzija nizova i proverava njena ispravnost */  
    printf("Unesite dimenziju nizova:\n");  
    scanf("%d", &n);  
    if (n<1 || n>MAX)  
    {  
22        printf("Nedozvoljena vrednost!\n");  
        return -1;  
24    }  
26  
    /* Učitavaju se elementi prvog niza */  
    printf("Unesite elemente niza a:\n");  
    for(i=0;i<n;i++)  
    {  
30        scanf("%d", &a[i]);  
    }  
32  
    /* Učitavaju se elementi drugog niza */  
    printf("Unesite elemente niza b:\n");  
    for(i=0;i<n;i++)  
    {  
36        scanf("%d", &b[i]);  
38    }  
40  
    /*
```

### 3 Predstavljanje podataka

```
42     Formira se treci niz.
      Koriste se dva indeksa:

44     - indeks i pomocu kojeg se pristupa elementima nizova a i b i
      koji treba uvecati za 1 nakon svake iteracije

46     - indeks j pomocu kojeg se pristupa elementima rezultujuceg
      niza c;
48     s obzirom da se u svakoj iteraciji u niz c smestaju dva
      elementa, jedan iz niza a i jedan iz niza b,
      indeks j se uvecava za 2 nakon svake iteracije
50 */
for(i=0,j=0;i<n;i++,j+=2)
52 {
    c[j]=a[i];
54     c[j+1]=b[i];
    }
56
/* Ispisuju se elementi rezultujuceg niza */
58 printf("Rezultujuci niz:\n");
for(i=0;i<2*n;i++)
60     printf("%d ",c[i]);
printf("\n");
62
64 return 0;
}
```

#### Rešenje 3.1.8

```
1  #include <stdio.h>

3  #define MAX 100

5  int main()
{
7      int a[MAX], b[MAX], c[2*MAX];
      int i, n;

9      /* Ucitava se broj elemenata nizova i proverava se njegova
      ispravnost */
11     printf("Unesite broj n: ");
     scanf("%d", &n);
13     if(n<1 || n>MAX)
     {
15         printf("Greska: Nedozvoljena vrednost!\n");
         return -1;
17     }

19     /* Ucitavaju se elementi nizova */
     printf("Unesite elemente niza a: ");
21     for(i=0;i<n;i++)
```

```

    scanf("%d", &a[i]);
23
printf("Unesite elemente niza b: ");
25
for(i=0;i<n;i++)
    scanf("%d", &b[i]);
27
/*
29
    Niz c ima 2*n elemenata: prvih n elemenata su elementi niza b, a
    narednih n elemenata elementi niza a.
    Elementi niza b se nalaze na pozicijama 0,1,2,...n-1, a elementi
    niza a na pozicijama
31
    n,n+1,...2*n-1. Jednim prolaskom kroz petlju na poziciju i u nizu
    c se postavlja element b[i] niza b,
    a na poziciju n+i element a[i] niza a.
33
*/
for(i=0;i<n;i++)
35
{
    c[i] = b[i];
37
    c[n+i] = a[i];
}
39
/* Ispisuju se elementi niza c */
41
for(i=0;i<2*n;i++)
    printf("%d ", c[i]);
43
printf("\n");
45
return 0;
}

```

### Rešenje 3.1.9

### Rešenje 3.1.10

```

#include <stdio.h>
2
#define MAX 100
4
int main()
6
{
8
    int a[MAX];
    int n;
10
    int i,j;
    char poslednja_cifra;
12
    int novo_n;

14
    /* Ucitava se dimenzija niza i proverava njena ispravnost */
    printf("Unesite dimenziju niza:\n");
16
    scanf("%d", &n);
    if (n<1 || n>MAX)

```

```
18 {
19     printf("Nedozvoljena vrednost!\n");
20     return -1;
21 }
22
23 /* Ucitavaju se elementi niza a */
24 printf("Unesite elemente niza a:\n");
25 for(i=0; i<n; i++)
26 {
27     scanf("%d", &a[i]);
28 }
29
30
31 /* Obilaze se svi elementi niza a */
32 for(i=0, j=0; i<n; i++)
33 {
34     /* Izdvaja se poslednja cifra tekuceg elementa */
35     poslednja_cifra = a[i]%10;
36
37     /* Ako je poslednja cifra 0 ili je element deljiv svojom
38     poslednjom cifrom,
39     zadržavamo ga i smestamo na poziciju j */
40     if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
41     {
42         a[j]=a[i];
43         j++;
44     }
45 }
46
47 /* Dimenzija novog niza odgovara posledjoj vrednosti brojac j */
48 novo_n=j;
49
50 /* Ispisuje se rezultujuci niz */
51 printf("Niz a nakon izmena:\n");
52 for(i=0; i<novo_n; i++)
53     printf("%d ", a[i]);
54     printf("\n");
55
56     return 0;
57 }
```

Rešenje 3.1.11

Rešenje 3.1.12

```
#include <stdio.h>
2
#define MAX 100
4
```

```
/* a) Napisati funkciju koja ucitava elemente niza. */
6 void ucitaj(int a[], int n)
{
8     int i;
    for(i=0;i<n;i++)
10     {
        scanf("%d",&a[i]);
12     }
}

14 /* b) Napisati funkciju koja stampa elemente niza. */
16 void stampaj(int a[], int n)
{
18     int i;
    for(i=0;i<n;i++)
20     printf("%d ",a[i]);
    printf("\n");
22 }

24 /* c) Napisati funkciju koja racuna sumu elemenata niza. */
int suma(int a[], int n)
26 {
    int i;
28     int s=0;
    for(i=0;i<n;i++)
30     s+=a[i];
    return s;
32 }

34 /* d) Napisati funkciju koja racuna prosechnu vrednost elemenata niza.
    */
36 float prosek(int a[], int n)
{
38     int i;
    int s = suma(a,n);
40     return (float) s/n;
}

42

44 /* e) Napisati funkciju koja izracunava minimum elemenata niza.*/
int minimum (int a[],int n)
46 {
    int m;
48     int i;

50     /* Minimum inicijalizujemo prvim elementom niza (a[0]), a zatim
        prolazimo kroz ostatak niza.
        U svakom koraku poredimo vrednost minimuma sa tekucim elementom
        niza. */
52     m = a[0];
    for(i=1;i<n;i++)
```

### 3 Predstavljanje podataka

---

```
54     if (a[i] < m)
55         m = a[i];
56
57     /* Vraca se izracunata vrednost minimuma */
58     return m;
59 }
60
61 /* f) Napisati funkciju koja izracunava poziciju maksimalnog elementa
62    u nizu. */
63 int pozicija_maksimuma (int a[],int n)
64 {
65     int m;
66     int m_pozicija;
67     int i;
68
69     /* Maksimum inicijalizujemo prvim elementom niza (a[0]), a zatim
70        prolazimo kroz ostatak niza.
71        U svakom koraku poredimo vrednost maksimuma sa tekucim
72        elementom niza. */
73
74     m = a[0];
75     m_pozicija=0;
76     for(i=1;i<n;i++)
77     {
78         if (a[i] > m)
79         {
80             m = a[i];
81             m_pozicija=i;
82         }
83     }
84
85     /* Vraca se izracunata pozicija */
86     return m_pozicija;
87 }
88
89 int main()
90 {
91     int a[MAX];
92     int n;
93
94     /* Ucitava se dimenzija niza i proverava njena ispravnost */
95     printf("Unesite dimenziju niza:");
96     scanf("%d",&n);
97     if (n<1 || n>MAX)
98     {
99         printf("Nedozvoljena vrednost!\n");
100         return -1;
101     }
102
103     /* Testira se funkcija kojom se ucitavaju elementi niza */
```

```

104     ucitaj(a,n);

106     /* Testira se funkcija kojom se ispisuju elementi niza */
    printf("Ucitani niz: ");
    stampaaj(a,n);

108     /* Testira se funkcija kojom se izracunava suma elemenata niza */
110     printf("Suma elemenata niza: %d\n", suma(a,n));

112     /* Testira se funkcija kojom se racuna prosek elemenata niza */
    printf("Prosecna vrednost elemenata niza: %.2f\n", prosek(a,n));

114     /* Testira se funkcija kojom se izracunava minimum niza */
116     printf("Minimumalni element niza: %d\n", minimum(a,n));

118     /* Testira se funkcija kojom se izracunava pozicija maksimalnog
        elementa */
    printf("Indeks maksimalnog elementa niza: %d\n",
        pozicija_maksimuma(a,n));

120     return 0;
122 }

```

### Rešenje 3.1.13

```

1  #include <stdio.h>
2
3  #define MAX 100
4
5  /* Funkcija koja ucitava elemente niza */
6  void ucitaj(int a[], int n)
7  {
8      int i;
9      for(i=0;i<n;i++)
10     {
11         scanf("%d",&a[i]);
12     }
13 }
14
15 /* Funkcija koja ispisuje elemente niza */
16 void stampaaj(int a[], int n)
17 {
18     int i;
19     for(i=0;i<n;i++)
20     {
21         printf("%d ",a[i]);
22     }
23     printf("\n");
24 }
25
26 /* a) Funkcija koja proverava da li niz sadrzi zadatu vrednost m */
27 int sadrzi(int a[], int n, int m)

```

### 3 Predstavljanje podataka

---

```
{
28     int i;

30     /* Poredi se element po element niza a sa zadatim brojem m */

32     for(i=0;i<n;i++){
33         /* Ukoliko je tekuci element niza jednak trazenom broju */
34         if (a[i]==m){
35             /* Funkcija vraca vrednost 1 */
36             return 1;
37         }
38     }

40     /* Ako se stigne do kraja niza i ne naidje na vrednost koja je
       jednaka broju m,
       to znaci da se broj ne nalazi u nizu i da funkcija treba da
       vrati 0. */
42     return 0;
43 }

44 /* b) Funkcija koja vraca vrednost prve pozicije na kojoj se nalazi
       element koji ima vrednost m,
       ili -1 ukoliko element nije u nizu */
46 int prvo_pojavljivanje(int a[], int n, int m)
47 {
48     int i;

50     /* Poredi se element po element niza a sa zadatim brojem m */

52     for(i=0;i<n;i++){
53         /* Ukoliko je tekuci element niza jednak trazenom broju */
54         if (a[i]==m){
55             /* Vraca se njegov indeks */
56             return i;
57         }
58     }

60     /* Ako se stigne do kraja niza i ne naidje na vrednost koja je
       jednaka broju m,
       to znaci da se broj ne nalazi u nizu i da funkcija treba da
       vrati -1. */
62     return -1;
63 }

64 /* c) Funkcija koja vraca vrednost poslednje pozicije na kojoj se
       nalazi element koji ima vrednost m,
       ili -1 ukoliko element nije u nizu */
66 int poslednje_pojavljivanje(int a[], int n, int m)
67 {
68     int i;

70     /* Polazi se od kraja niza i poredi se element po element sa
```



```
        zadanim brojem m*/
74     for(i=n-1;i>=0;i--){
        /* Ukoliko je tekuci element niza jednak trazenom broju */
76         if (a[i]==m){
            /* Vraca se njegov indeks */
78             return i;
        }
80     }

82     /* Ako se stigne do pocetka niza i ne naidje na vrednost koja je
        jednaka broju m,
        to znaci da se broj ne nalazi u nizu i da funkcija treba da
        vrati -1. */
84     return -1;
}

86 /* d) Funkcija koja proverava da li elementi niza cine palindrom */
88 int palindrom(int a[], int n)
89 {
90     int i,j;
92
93     /*
94     Uporedjuje se element na poziciji 0 sa elementom na poziciji n-1
95     Uporedjuje se element na poziciji 1 sa elementom na poziciji n-2
96     Uporedjuje se element na poziciji 2 sa elementom na poziciji n-3
97     .
98     .
99     i tako redom dok je pozicija prvog elementa manja od pozicije
        drugog elementa
100    */

102    for(i=0,j=n-1;i<j;i++,j--){
        /* Ako element na poziciji i nije jednak odgovarajucem elementu
        na poziciji j */
104        if(a[i]!=a[j]){
            /* Moze se odmah zakljuciti da niz nije palindrom */
106            return 0;
        }
108    }

110    /* Ako su svi parovi elemenata jednaki, niz je palindrom i
        funkcija vraca vrednost 1 */
    return 1;
112 }

114 /* e) Funkcija koja proverava da li su elementi niza uredjeni
        neopadajuce */
int neopadajuci(int a[], int n)
116 {
    int i;
```

### 3 Predstavljanje podataka

---

```
118      /* Ako je niz uredjen neopadajuće vase nejednakosti:
120         a[0]<=a[1], a[1]<=a[2], ..., a[i]<=a[i+1], ..., a[n-2]<=a[n-1]
           Zato je dovoljno proveriti da li za parove susednih elemenata
           vazi ovi svojstvo.
122      */
123      for(i=0; i<n-1; i++)
124          if (a[i]>a[i+1])
125              return 0;
126
127      return 1;
128  }
129
130  /* f) Funkcija koja izracunava najduzu uzastopnu seriju jednakih
131     elemenata u nizu. Na primer, za uneti niz 1 2 3 4 4 4 5 6 7 8 9 9
           funkcija
           treba da vrati 3. */
132  int najduza_serija_jednakih(int a[], int n)
133  {
134      int i;
135      int j;
136      int duzina;
137      int max_duzina=0;
138
139
140
141
142      for(i=0, j=0; i<n-1; i++)
143      {
144          if(a[i]==a[i+1])
145          {
146              j++;
147
148              /*
149                 ako se niz završava konstantnim intervalom (nalazimo se u
150                 poslednjoj
151                 iteraciji petlje i tada je i==n-2), ispitujemo da li je
152                 taj konstantni
153                 interval maksimalne duzine
154              */
155              if(i==n-2)
156              {
157                  j++;
158                  if(j>max_duzina)
159                      max_duzina=j;
160              }
161          }
162          else
163          {
164              /*
165                 izašli smo iz konstantnog intervala
166
167                 ukoliko smo imali bar dva elementa u konstantnom
```

```
    intervalu,
166     vrednost promenljive j ce biti 1, a duzina tog intervala
    je 2;
    zbog toga je neophodno takve (pozitivne) j uvecati za 1;
168
    sa druge strane, ako su a[i] i a[i+1] razliciti,
170     duzina tog intervala je 0
    */
172
    if (j>0)
174         j++;

    /* azuriramo maksimalnu duzinu uspona */
    if(j>max_duzina)
176         max_duzina=j;
    /*
180     duzina uspona se postavlja na nulu
    kako bi mogli da je iskoristimo
182     za naredni uspon
    */
184     j=0;

186 }

188 }

190 return max_duzina;
192 }

194 int main()
196 {
    int a[MAX];
198     int n;
    int m;
200     int i;

    /* Ucitava se dimenzija niza i proverava se njena ispravnost */
    printf("Unesite dimenziju niza:");
204     scanf("%d",&n);
    if (n<1 || n>MAX)
206     {
        printf("Nedozvoljena vrednost!\n");
208         return -1;
    }

210
    /* Ucitavaju se i ispisuju elementi niza */
212     ucitaj(a,n);
    printf("Ucitani niz:");
214     stampaj(a,n);
```

### 3 Predstavljanje podataka

```
216      /* Ucitava se vrednost za pretragu */
printf("Unesi jedan ceo broj:");
218      scanf("%d",&m);

220      /* I proverava se rad napisanih funkcija */
if(sadrzi(a,n,m))
222          printf("Niz sadrzi element cija je vrednost %d\n", m);
else
224          printf("Niz ne sadrzi element cija je vrednost %d\n", m);

226      i = prvo_pojavljivanje(a,n,m);
if(i!=-1)
228          printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
prvog pojavljivanja u nizu je %d\n", m,i);
else
230          printf("Niz ne sadrzi element cija je vrednost %d\n", m);

232
234      i = poslednje_pojavljivanje(a,n,m);
if(i!=-1)
236          printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
poslednjeg pojavljivanja u nizu je %d\n", m,i);
else
238          printf("Niz ne sadrzi element cija je vrednost %d\n", m);

240      if(palindrom(a,n))
printf("Elementi niza cine palindrom\n");
else
242          printf("Elementi niza ne cine palindrom\n");

244      if(neopadajuci(a,n))
printf("Niz je sortiraneopadajuće\n");
else
246          printf("Niz nije sortiraneopadajuće\n");

248      printf("Duzina najduzeg konstantnog intervala: %d\n",
najduza_konstanta(a,n));

250
252      return 0;
}
```

#### Rešenje 3.1.14

```
#include<stdio.h>
2
#define MAX 100
4
/* Funkcija kojom se ucitavaju elementi niza a dimenzije n */
6 void ucitaj(int a[], int n)
{
```

```
8     int i;
9     for(i=0;i<n;i++)
10     {
11         scanf("%d",&a[i]);
12     }
13 }
14
15 /* Funkcija kojom se ispisuju elementi niza a dimenzije n */
16 void stampaj(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 /* a) Funkcija koja sve vrednosti niza uvecava za zadatu vrednost m
25    */
26 void uvecaj(int a[], int n, int m)
27 {
28     int i;
29     for(i=0;i<n;i++)
30         a[i]+=m;
31 }
32
33 /* b) Funkcija koja obrce elemente niza */
34 void obrni(int a[], int n)
35 {
36     int t;
37     int i,j;
38
39     /*
40      Za niz a[0], a[1], ..., a[n-2], a[n-1] obrnuti niz je a[n-1], a[
41      n-2], ..., a[1], a[0]
42      Zato je potrebno razmeniti vrednosti elemenata a[0] i a[n-1], a
43      [1] i a[n-2], itd. i zaustaviti se
44      kada je vrednost indeksa prvog elementa veca od vrednosti drugog
45      elementa.
46      */
47     for(i=0,j=n-1;i<j;i++, j--)
48     {
49         t = a[i];
50         a[i] = a[j];
51         a[j] = t;
52     }
53 }
54
55 /* c) Funkcija koja rotira niz ciklicno za jedno mesto u levo */
```

### 3 Predstavljanje podataka

---

```
56 void rotiraj1(int a[], int n)
57 {
58     int i;
59     int tmp;
60
61     /* Izdvaja se prvi element niza */
62     tmp=a[0];
63
64     /* Pomeraju se preostali elementi niza */
65     for(i=0;i<n-1;i++){
66         a[i]=a[i+1];
67     }
68
69     /* Poslednjem elementu se dodeljuje sacuvana vrednost prvog
70     elementa */
71     a[n-1] = tmp;
72 }
73
74 /* d) Funkcija koja rotira niz ciklicno za k mesta u levo */
75 void rotirajk(int a[], int n, int k)
76 {
77     int i;
78
79     /* Odredjuje se vrednost broja k koja je u opsegu od 0 do n-1 kako
80     bi se izbegla suvisna pomeranja */
81     k=k%n;
82
83     /* Niz se rotira za jednu poziciju ulevo k puta */
84     for(i=0;i<k;i++)
85         rotiraj1(a,n);
86 }
87
88 int main()
89 {
90     int a[MAX];
91     int n;
92     int i;
93     int k;
94     int m;
95
96     /* Ucitava se dimenzija niza i proverava se njena ispravnost */
97     printf("Unesite dimenziju niza:");
98     scanf("%d",&n);
99     if (n<1 || n>MAX)
100     {
101         printf("Nedozvoljena vrednost!\n");
102         return -1;
103     }
104
105     /* Ucitavaju se elementi niza */
106     ucitaj(a,n);
```

```

106  /* Testira se rad napisanih funkcija */
108
109  /* a) */
110  printf("Unesite jedan ceo broj:");
111  scanf("%d", &m);
112  printf("Elementi niza nakon uvecanja za %d:\n",m);
113  uvecaj(a,n,m);
114  stampaj(a,n);
115
116  /* b) */
117  printf("Elementi niza nakon obrtanja:\n");
118  obrni(a,n);
119  stampaj(a,n);
120
121  /* c) */
122  printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
123  rotiraj1(a,n);
124  stampaj(a,n);
125
126  /* d) */
127  printf("Unesite jedan pozitivan ceo broj:");
128  scanf("%d",&k);
129  if (k<=0)
130  {
131      printf("Nekorektan unos\n");
132      return -1;
133  }
134  rotirajk(a,n,k);
135  printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n",k);
136  stampaj(a,n);
137
138  return 0;
139
140 }

```

### Rešenje 3.1.15

```

1  #include <stdio.h>
2
3  #define MAX 100
4
5  int main()
6  {
7      float brojevi[MAX];
8      int n, i;
9
10     /* Ucitava se dimenzija niza i proverava se njena ispravnost */
11     printf("Unesite broj elemenata niza: ");
12     scanf("%d", &n);
13     if(n<1 || n>MAX)

```

### 3 Predstavljanje podataka

```
15     {
16         printf("Nedozvoljena vrednost!\n");
17         return -1;
18     }
19
20     /* Ucitavaju se elementi niza */
21     printf("Unesite elemente niza:\n");
22     for(i=0;i<n;i++){
23         scanf("%f", &brojevi[i]);
24     }
25
26     /* Ukoliko je i-ti element niza brojevi[i] negativan broj,
27        kvadriramo ga tako sto ga pomnozimo sa samim sobom. */
28     for(i=0;i<n;i++){
29         if(brojevi[i]<0)
30             brojevi[i] *= brojevi[i];
31     }
32
33     /* Ispisuje se novodobijeni niz */
34     for(i=0;i<n;i++){
35         printf("%g ", brojevi[i]);
36     }
37     printf("\n");
38
39     return 0;
40 }
```

#### Rešenje 3.1.16

```
1  #include <stdio.h>
2
3  #define MAX 100
4
5  int main()
6  {
7      int brojevi[MAX];
8      int n, i, k, indikator;
9
10     /* Ucitava se dimenzija niza i proverava se njena ispravnost */
11     printf("Unesite dimenziju niza: ");
12     scanf("%d", &n);
13     if(n<1 || n>MAX)
14     {
15         printf("Greska: Nedozvoljena vrednost!\n");
16         return -1;
17     }
18
19     /* Ucitavaju se elementi niza */
20     printf("Unesite elemente niza: ");
21     for(i=0;i<n;i++)
22         scanf("%d", &brojevi[i]);
```



```

23
25  /* Ucitava se broj k i proverava se njegova ispravnost */
printf("Unesite broj k: ");
27  scanf("%d", &k);
if(k == 0)
29  {
    printf("Greska: Pogresan unos!\n");
31    return -1;
}

33
/*
35  Promenljiva koja cuva informaciju o tome da li je u nizu
    postojao element koji je deljiv brojem k.
    Inicijalna vrednost je 0.
37  */

39  indikator = 0;

41  /*
    Ukoliko je element niza deljiv brojem k, indikator se postavlja
    na 1
43    i ispisuje se indeks tog elementa.
    */

45  for(i=0;i<n;i++){
47      if(brojevi[i]%k == 0)
          {
49          indikator = 1;
          printf("%d ",i);
51      }
    }

53
/* Ukoliko je indikator jednak nuli to znaci da ne postoji element
   u nizu koji je deljiv brojem k. */

55  if(indikator == 0){
57      printf("U nizu nema elemenata koji su deljivi brojem %d!\n",k);
    }

59

61  return 0;
}

```

### Rešenje 3.1.17

```

1  #include <stdio.h>
3  #define MAX 100
5  int main()

```

```
{
7   int brojevi[MAX];
   int n, i, poz_max, poz_min, max, min, tmp;
9
   /* Ucitava se dimenzija niza i proverava se njena ispravnost */
11  printf("Unesite dimenziju niza: ");
   scanf("%d", &n);
13  if(n<1 || n>MAX)
   {
15      printf("Greska: Nedozvoljena vrednost!\n");
      return -1;
17  }

19  /* Ucitavaju se elementi niza */
   printf("Unesite elemente niza:\n");
21  for(i=0;i<n;i++)
      scanf("%d", &brojevi[i]);
23
   /*
25      Maksimalnim tj. minimalnim elementom niza proglašava se nulti
      element niza.
      Pozicije maksimalnog tj. minimalnog elementa se postavljaju na 0.
27  */
   max = brojevi[0];
29   min = brojevi[0];
   poz_max = 0;
31   poz_min = 0;

33   /* U prolazu kroz niz trazi se maksimalni i minimalni element i
      pamte se njihove pozicije */
   for(i=1;i<n;i++)
35   {
       if(brojevi[i] > max)
37       {
           max = brojevi[i];
           poz_max = i;
39       }

       if(brojevi[i] < min)
41       {
           min = brojevi[i];
           poz_min = i;
43       }
45   }
47   }

49   /* Zamenjuju se elementi na pozicijama poz_min i poz_max */
   tmp = max;
51   brojevi[poz_max] = min;
   brojevi[poz_min] = tmp;
53
   /* Ispisuje se rezultujući niz */
55   for(i=0;i<n;i++)
```

```
        printf("%d ", brojevi[i]);
57     printf("\n");

59     return 0;
61 }
```

### Rešenje 3.1.18

### Rešenje 3.1.19

```
1  #include <stdio.h>

3  #define MAX 100

5  int main()
6  {
7      /* niz karaktera */
8      char karakteri[MAX];
9      char c;
10     int i, n;

11

12     for(i=0; i<MAX; i++)
13     {
14         /* Ucitava se karakter po karakter sa standardnog ulaza sve dok
15         se ne unese * ili
16         se ne prekorači maksimalni broj karaktera */
17         printf("Unesite karakter: ");
18         scanf("%c", &c);

19         /* Cita se znak za novi red nakon unesenog karaktera */
20         getchar();

21         /* Ukoliko je unet karakter * prekida se dalje citanje i izlazi
22         se iz petlje */
23         if(c == '*')
24             break;

25         /* Inace, procitani karakter se smesta u niz */
26         karakteri[i] = c;
27     }

28     /* Broj unetih karaktera je nakon izlaska iz petlje i-1 */
29     n = i-1;

30     /* Ispisuju se karakteri u obrnutom redosledu */
31     for(i=n; i>=0; i--)
32     {
33         printf("%c ", karakteri[i]);
34     }
```

### 3 Predstavljanje podataka

---

```
    }
39    printf("\n");
41
42    return 0;
43 }
```

#### Rešenje 3.1.20

```
1  #include <stdio.h>
3  #define MAX 100
5  /* Funkcija koja vraca broj pojavljivanja broja x u nizu */
6  int broj_pojavljivanja(int niz[], int n, int x)
7  {
8      int i;
9
10     /* Broj pojavljivanja broja x */
11     int brojac = 0;
12
13     /* Obilazi se element po element niza */
14     for(i=0; i<n; i++){
15         /* Ukoliko je tekuci element jednak traženom broju */
16         if(niz[i] == x){
17             /* Uvecava se broj pojavljivanja */
18             brojac++;
19         }
20     }
21
22     /* Vraca se izracunata vrednost */
23     return brojac;
24 }
25
26 int main()
27 {
28     /* Niz elemenata koje zadaje korisnik */
29     int a[MAX];
30
31     /* Niz elemenata koji se pojavljuju tri puta */
32     int b[MAX];
33
34     int i, j, n, n_b;
35
36     /* Ucitava se broj elemenata korisnickog niza i proverava se
37     njegova ispravnost */
38     printf("Unesite broj n: ");
39     scanf("%d", &n);
40     if(n<1 || n>MAX)
41     {
42         printf("Greska: Nedoovoljena vrednost!\n");
43     }
44 }
```

```

43     return -1;
44 }
45
46 /* Ucitavaju se elementi korisnickog niza */
47 printf("Unesite elemente niza a: ");
48 for(i=0;i<n;i++)
49     scanf("%d", &a[i]);
50
51 /* j - brojca elemenata rezultujucega niza b */
52 j = 0;
53
54 /* Obilazi se element po element niza a */
55 for(i=0;i<n;i++)
56 {
57     /* Ukoliko se tekuci element pojavljuje tacno tri puta u nizu a i
58        nije upisan u niz b
59        koji trenutno ima j elemenata, dodaje se u niz b na poziciju j i
60        uvecava se broj elemenata niza b */
61     if(broj_pojavljivanja(a, n, a[i])==3 && broj_pojavljivanja(b, j,
62        a[i])==0)
63     {
64         b[j] = a[i];
65         j++;
66     }
67 }
68
69 /* Ispisuje se rezultujući niz b - broj elemenata u nizu b je j*/
70 n_b = j;
71 for(i=0;i<n_b;i++)
72     printf("%d ", b[i]);
73 printf("\n");
74
75 return 0;
76 }

```

## Rešenje 3.1.21

## Rešenje 3.1.22

```

1 #include <stdio.h>
2
3 #define MAX 100
4
5 /*
6  Funkcija koja vraca 1 ukoliko broj x postoji u nizu, 0 inace.
7  */
8
9 int postoji(int niz[], int n, int x)
10 {
11     int i;

```

```
12     for(i=0;i<n;i++)
13         if(niz[i] == x)
14             return 1;
15
16     return 0;
17 }
18
19
20 int main()
21 {
22     int a[MAX], b[MAX], unija[2*MAX], presek[MAX], razlika[MAX];
23     int i, j, n_a, n_b, n_u, n_p, n_r, indikator;
24
25     printf("Unesite broj elemenata niza a: ");
26     scanf("%d", &n_a);
27
28     if(n_a<1 || n_a>100)
29     {
30         printf("Greska: pogresan unos!\n");
31         return -1;
32     }
33
34     printf("Unesite elemente niza a: ");
35     for(i=0;i<n_a;i++)
36         scanf("%d", &a[i]);
37
38     printf("Unesite broj elemenata niza b: ");
39     scanf("%d", &n_b);
40
41     if(n_b<1 || n_b>100)
42     {
43         printf("Greska: pogresan unos!\n");
44         return -1;
45     }
46
47     printf("Unesite elemente niza b: ");
48     for(i=0;i<n_b;i++)
49         scanf("%d", &b[i]);
50
51     /*
52      Brojaci elemenata u nizovima unija, presek i razlika.
53     */
54     n_u = 0;
55     n_p = 0;
56     n_r = 0;
57
58     for(i=0;i<n_a;i++)
59     {
60         /*
61          Ukoliko se element a[i] ne nalazi u uniji, dodajemo ga u uniju
62          i povecamo brojac elemenata u nizu unija.
63         */
```

```
64     if(postoji(unija,n_u,a[i]) == 0)
66     {
68         unija[n_u] = a[i];
66         n_u++;
68     }

68     /*
70     Ukoliko se element a[i] postoji u nizu b i ne postoji u nizu
70     presek, dodajemo ga u presek i povecavamo brojac elemenata u nizu
70     presek.
70     */
72     if(postoji(b, n_b, a[i])==1 && postoji(presek, n_p, a[i])==0)
74     {
76         presek[n_p] = a[i];
76         n_p++;
78     }

78     /*
78     Ukoliko element a[i] ne postoji u nizu b i ne postoji u nizu
78     razlika, dodajemo ga u razliku i povecavamo brojac elemenata u
78     nizu razlika.
80     */
82     if(postoji(b, n_b, a[i])==0 && postoji(razlika, n_r, a[i])==0)
84     {
86         razlika[n_r] = a[i];
86         n_r++;
88     }

88     /*
88     Elemente niza b koji ne postoje u uniji dodajemo u uniju.
90     */
92     for(i=0;i<n_b;i++)
94     {
96         if(postoji(unija, n_u, b[i]))
98         {
100             unija[n_u] = b[i];
100             n_u++;
102         }

102     printf("Unija: ");
104     for(i=0;i<n_u;i++)
106     printf("%d ", unija[i]);

104     printf("\nPresek: ");
106     for(i=0;i<n_p;i++)
108     printf("%d ", presek[i]);

106     printf("\nRazlika: ");
108     for(i=0;i<n_r;i++)
110     printf("%d ", razlika[i]);

110     return 0;
```

```
}
```

#### Rešenje 3.1.23

```
#include <stdio.h>

2
#define MAX 100
4
int main()
6 {
    int a[MAX];
    int i, j, n_a;

10 /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
    */
    printf("Unesite broj elemenata niza: ");
12 scanf("%d", &n_a);
    if(n_a<1 || n_a>100)
14 {
        printf("Greska: Nedozvoljena vrednost!\n");
16         return -1;
    }

18
    /* Ucitavaju se elementi niza */
20 printf("Unesite elemente niza: ");
    for(i=0; i<n_a; i++)
22         scanf("%d", &a[i]);

24
    /*
26     1. nacin

28     int b[MAX], n_b;

30     // Brojac j predstavlja poziciju u nizu b na koju treba smestiti
        element niza a. Njegova pocetna vrednost je 0.

32     for(i=0, j=0; i<n_a; i++){
        // Ako je tekuci element niza a paran
34         if(a[i]%2 == 0)
        {
36             // Smesta se na poziciju j u nizu b
                b[j] = a[i];
38             // Vrednost brojaca j se priprema za narednu iteraciju
                j++;
40         }

42         // Ako je element niza a neparan, sa njim nista ne treba raditi
    }

44
    // Broj elemenata novodobijenog niza b je j
```



```

46     n_b = j;

48     // Ispisuju se elementi niza b
    for(i=0;i<n_b;i++)
50         printf("%d ", b[i]);

52     */

54     /*
    2. nacin

56     J predstavlja brojac prve slobodne pozicije na koju se moze
        upisati element niza koji treba da ostane u nizu.
58     Kada se naidje na element koji je paran, on se kopira na mesto a[j]
        i poveca se vrednost brojac j.
        Ukoliko se naidje na element koji je neparan, njega treba
        preskociti.
60     */
    for(i=0, j=0;i<n_a;i++)
62     {
        /* Ako je tekuci element niza a paran */
64         if(a[i]%2 == 0)
        {
66             /* Premesta se na poziciju j */
            a[j] = a[i];

68             /* Vrednost brojac j se priprema za narednu iteraciju */
70             j++;
        }

72         /* Ako je tekuci element niza a neparan, sa njim nista ne treba
            raditi */
74     }

76     /* U nizu a se sada na pozicijama od 0,...,j-1 nalaze elementi koji
        su parni, te je njegova nova dimenzija j. */
    n_a=j;

78     /* Ispisuju se elementi modifikovanog niza a */
    for(i=0;i<n_a;i++){
80         printf("%d ", a[i]);
82     }
    printf("\n");

84     return 0;
86 }

```

### Rešenje 3.1.24

```

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

```

```
4 #define MAX 100

6 /*
7  Funkcija koja proverava da li je zadati broj prost broj.
8  Povratna vrednost funkcije je 1 ukoliko broj jeste prost, inace je
9  0.
10 */
11 int prost(int x)
12 {
13     int i;

14     /* Posmatra se apsolutna vrednost broja kako bi se pokrio i slucaj
15        negativnih brojeva */
16     x=abs(x);

17     /* Brojevi 1, 2 i 3 su prosti */
18     if(x == 1 || x == 2 || x == 3)
19         return 1;

20     /* Ako je broj paran nije prost */
21     if(x%2 == 0)
22         return 0;

23     /* Ako broj ima delioce u skupu [3, koren_broja(x)] takodje nije
24        prost */
25     for(i=3;i<=sqrt(x);i+=2){
26         if(x%i == 0)
27             return 0;
28     }

29     /* Ako su svi uslovi ispunjeni, broj je prost */
30     return 1;
31 }

32 int main()
33 {
34     int a[MAX];
35     int i, j, n_a, n_b;

36     /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
37        */
38     printf("Unesite broj elemenata niza: ");
39     scanf("%d", &n_a);
40     if(n_a<1 || n_a>MAX)
41     {
42         printf("Greska: Nedozvoljena vrednost!\n");
43         return -1;
44     }

45     /* Ucitavaju se elementi niza a */
46     printf("Unesite elemente niza: ");
```

```
52     for(i=0;i<n_a;i++)
        scanf("%d", &a[i]);

54
56     /*
57     1. nacin
58
59     int b[MAX];
60
61     for(i=0, j=0;i<n_a;i++){
62         if(prost(a[i]) == 0)
63         {
64             b[j] = a[i];
65             j++;
66         }
67     }
68
69     // Broj elemenata novodobijenog niza b je j
70     n_b = j;
71
72     // Ispisuju se elementi niza b
73     for(i=0;i<n_b;i++)
74         printf("%d ", b[i]);
75     printf("\n");
76
77     */
78
79     /*
80     2. nacin
81     */
82
83     for(i=0, j=0; i<n_a; i++)
84     {
85         if(prost(a[i]) == 0)
86         {
87             a[j] = a[i];
88             j++;
89         }
90     }
91
92     n_a = j;
93
94     /* Ispisuju se elementi modifikovanog niza a */
95     for(i=0;i<n_a;i++)
96         printf("%d ", a[i]);
97     printf("\n");
98
99     return 0;
100 }
```

#### Rešenje 3.1.25

```
1  #include <stdio.h>
2
3  #define MAX 100
4
5  /* Funkcija prebrojavanje vraca broj elemenata niza koji su manji od
6     poslednjeg elementa */
7  int prebrojavanje(int a[], int n)
8  {
9      int i;
10
11     /* Brojac elemenata koji su manji od poslednjeg */
12     int broj_manjih=0;
13
14     /* Obilazi se element po element niza */
15     for(i=0; i<n-1; i++){
16         /* Ako je tekuci element manji od poslednjeg (on se nalazi na
17            poziciji n-1) */
18         if(a[i]<a[n-1]){
19             /* Uvecava se brojac */
20             broj_manjih++;
21         }
22     }
23
24     /* Vraca se izracunata vrednost */
25     return broj_manjih;
26 }
27
28 int main()
29 {
30     int a[MAX];
31     int n;
32     int i;
33
34     /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
35        */
36     printf("Unesite broj elemenata niza:");
37     scanf("%d", &n);
38     if(n<=0 || n>MAX)
39     {
40         printf("Greska: Nedozvoljena vrednost!\n");
41         return 0;
42     }
43
44     /* Ucitavaju se elementi niza */
45     printf("Unesite elemente niza:");
46     for(i=0; i<n; i++)
47         scanf("%d", &a[i]);
48
49     /* Ispisuje se rezultat poziva funkcije */
50     printf("%d\n", prebrojavanje(a,n));
```

```
48     return 0;
50 }
```

### Rešenje 3.1.26

```
#include <stdio.h>
2
#define MAX 100
4
/* Funkcija vraca broj parnih elemenata niza koji prethode
   maksimalnom elementu niza */
6 int prebrojavanje(int a[], int n)
{
8     int i;

10     int maksimum;
    int pozicija_maksimuma;

12     /* Brojac elemenata koji su parni i prethode maksimalnom */
14     int broj_parnih;

16     /* Pronalazi se maksimalni element niza i njegova pozicija */
    maksimum = a[0];
18     pozicija_maksimuma = 0;

20     for(i=1; i<n-1; i++)
        if(a[i] > maksimum)
22     {
            maksimum = a[i];
24         pozicija_maksimuma = i;
        }

26     /* Prebrojavaju se parni elementi koji prethode maksimalnom */
28     broj_parnih = 0;
    for(i=0; i < pozicija_maksimuma; i++){
30         if(a[i]%2==0){
            broj_parnih++;
32         }
    }

34     /* Vraca se izracunata vrednost */
36     return broj_parnih;
}

38 int main()
40 {
    int a[MAX];
42     int n;
    int i;
44
```

### 3 Predstavljanje podataka

---

```
/* Ucitava se broj elemenata niza i proverava se njegova ispravnost */
46 printf("Unesite broj elemenata niza:");
scanf("%d", &n);
48 if(n<=0 || n>MAX)
{
50     printf("Greska: Nedozvoljena vrednost!\n");
    return 0;
52 }

54 /* Ucitavaju se elementi niza */
printf("Unesite elemente niza:");
56 for(i=0;i<n;i++){
    scanf("%d",&a[i]);
58 }

60 /* Ispisuje se rezultat poziva funkcije */
printf("%d\n", prebrojavanje(a,n));
62
    return 0;
64 }
```

#### Rešenje 3.1.27

```
1  #include <stdio.h>
   #include <ctype.h>
3
   #define MAX 100
5
   /* Funkcija prebrojava cifre u datom nizu karaktera */
7  int cifre(char a[], int n)
   {
9      int i;

11     /* Brojac cifara */
    int broj_cifara = 0;

13
    /* Obilazi se element po element niza */
15     for(i=0;i<n;i++){
        /* Ako je tekuci element cifra */
17         if(isdigit(a[i])){
            /* Uvecava se broj cifara */
19             broj_cifara++;
        }
21     }

23     /* Vraca se izracunata vrednost */
    return broj_cifara;
25 }

27 int main()
```

```

{
29  char a[MAX];
    int n;
31  int i;

33  /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
    */
    printf("Unesite broj elemenata niza:");
35  scanf("%d", &n);
    if(n<=0 || n>MAX)
37  {
        printf("Greska: Nedozvoljena vrednost!\n");
39  return 0;
    }

41  /* Ucitavaju se elementi niza */
43  printf("Unesite elemente niza:");
    for(i=0;i<n;i++) {
45      /* Preskace se prethodno uneti znak za novi red */
        getchar();

47      /* A zatim se ucitava sam karakter i smesta u niz */
49      scanf("%c",&a[i]);
    }

51  /* Ispisuje se rezultat poziva funkcije */
53  printf("Broj cifara je: %d\n", cifre(a,n));

55  return 0;
}

```

### Rešenje 3.1.28

```

1  #include<stdio.h>

3  #define MAX 100

5  /* Funkcija racuna zbir elemenata niza od pozicije i do pozicije j */
    int zbir(int a[], int n, int i, int j){
7      int k;

9      /* Zbir elemenata niza iz zadatog opsega */
        int z = 0;

11     /* Obilaze se elementi niza */
13     for(k=i; k<=j; k++){
        z+=a[k];
15     }

17     /* Vraca se izracunata vrednost */
        return z;

```

### 3 Predstavljanje podataka

---

```
19 }
21 int main(){
23     int n, i, j;
24     int a[MAX];
25
26     /* Ucitava se broj elemenata niza i proverava se njegova ispravnost */
27     printf("Unesite broj elemenata niza: ");
28     scanf("%d", &n);
29     if(n <= 0 || n > MAX)
30     {
31         printf("Greska: Nedozvoljena vrednost!\n");
32         return 0;
33     }
34
35     /* Ucitavaju se elementi niza */
36     printf("Unesite elemente niza:");
37     for(i=0; i<n; i++)
38         scanf("%d", &a[i]);
39
40     /* Ucitavaju se vrednosti granica */
41     printf("Unesite vrednosti za i i j: ");
42     scanf("%d%d", &i, &j);
43
44     /* Proverava se korektnost zadatog intervala */
45     if(i < 0 || j < 0 || i > n-1 || j > n-1 || i > j){
46         printf("Greska: Nekorektne vrednosti granica!\n");
47         return 0;
48     }
49
50     /* Ispisuje se rezultat poziva funkcije */
51     printf("Zbir je: %d", zbir(a,n,i,j));
52
53     return 0;
54 }
```

#### Rešenje 3.1.29

```
#include<stdio.h>
2
#define MAX 100
4
/* Funkcija racuna zbir prvih k pozitivnih elemenata niza */
6 float zbir_pozitivnih(float a[], int n, int k){
8     int i;
10
11     /* Zbir pozitivnih elemenata */
12     float zbir=0;
```



```

12      /* Obilazi se element po element niza - postupk se završava ukoliko
13         se dodje do kraja niza
14         ili ukoliko se sabere k pozitivnih elemenata */
15      for(i=0; i<n && k>0; i++){
16          /* Ako je tekuci element pozitivan broj */
17          if(a[i] >= 0){
18              /* Dodaje se zbiru */
19              zbir+=a[i];
20              /* I umanjuje se brojac pozitivnih elemenata */
21              k--;
22          }
23      }
24
25      /* Vraca se izracunata vrednost */
26      return zbir;
27  }
28
29  int main(){
30      int n, i, k;
31      float a[MAX];
32
33      /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
34         */
35      printf("Unesite broj elemenata niza: ");
36      scanf("%d", &n);
37      if(n<=0 || n> MAX){
38          printf("Greska: Nedozvoljena vrednost!\n");
39          return 0;
40      }
41
42      /* Ucitavaju se elementi niza */
43      printf("Unesite elemente niza: ");
44      for(i=0; i<n; i++){
45          scanf("%f", &a[i]);
46      }
47
48      /* Ucitava se broj k i proverava se njegova ispravnost */
49      printf("Unesite vrednost za k: ");
50      scanf("%d", &k);
51      if(k<0 || k>n){
52          printf("Greska: Nedozvoljena vrednost!");
53          return 0;
54      }
55
56      /* Ispisuje se rezultat poziva funkcije */
57      printf("Zbir je: %.2f\n", zbir_pozitivnih(a,n,k));
58
59      return 0;
60  }

```

## Rešenje 3.1.30

### 3 Predstavljanje podataka

---

```
1  #include<stdio.h>
3  #define MAX 100
5  /* Funkcija koja kvadrira elemente niza koji se nalaze na parnim
   pozicijama */
   void kvadriranje(float a[], int n){
7
9     int i;
11
12    /* Obilaze se elementi na parnim pozicijama */
13    for(i=0; i<n; i+=2){
14        /* I kvadriraju se: a[i] = a[i]*a[i] */
15        a[i]*=a[i];
16    }
17
18 }
19
20 int main(){
21
22     int n, i, j;
23     float a[MAX];
24
25     /* Ucitava se broj elemenata niza i proverava se njegova ispravnost
26        */
27     printf("Unesite broj elemenata niza: ");
28     scanf("%d", &n);
29     if(n <=0 || n>MAX)
30     {
31         printf("Greska: Nedozvoljena vrednost!\n");
32         return 0;
33     }
34
35     /* Ucitavaju se elementi niza */
36     printf("Unesite elemente niza:");
37     for(i=0; i<n; i++)
38         scanf("%f", &a[i]);
39
40     /* Poziva se funkcije */
41     kvadriranje(a,n);
42
43     /* Ispisuje se elementi novodobijenog niza */
44     /* Koriscenje specifikatora %g za stampanje realnih brojeva
45        omogucava ispis broja
46        na onoliko decimalnih mesta koliko ima i sam broj */
47     for(i=0; i<n; i++){
48         printf("%g ", a[i]);
49     }
50     printf("\n");
51
52     return 0;
```

Rešenje 3.1.31

Rešenje 3.1.32

Rešenje 3.1.33

Rešenje 3.1.34

Rešenje 3.1.35

### 3.3 Pokazivači

**Zadatak 3.3.1** Napisati funkciju koja uređuje svoja dva celobrojna argumenta tako da se u prvom nalazi manji, a u drugom veći. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite vrednosti promenljivih x i y: 2 5
|| Uredjene promenljive: x=2, y=5
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite vrednosti promenljivih x i y: 11 -4
|| Uredjene promenljive: x=-4, y=11
```

[Rešenje 3.3.20]

**Zadatak 3.3.2** Napisati funkciju koja za boju datu u *rgb* formatu računa *cmv* format po formulama:

$$c = 1 - (r/255)$$

$$m = 1 - (g/255)$$

$$y = 1 - (b/255)$$

Napisati program koji učitava tri cela broja broja (*rgb* format) i ispisuje rezultat poziva funkcije (*cmv* format). NAPOMENA: *Vrednosti boja u rgb formatu su u opsegu [0, 255].*

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite boju u rgb formatu: 56 111 24
|| c=0.78, m=0.56, y=0.91
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite boju u rgb formatu: 156 -90 5
|| Nekorektan unos.
```

### 3 Predstavljanje podataka

---

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite boju u rgb formatu: 9 0 237  
|| c=0.96, m=1.00, y=0.07
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite boju u rgb formatu: 300 11 27  
|| Nekorektan unos.
```

[Rešenje 3.3.2]

**Zadatak 3.3.3** Napisati funkciju koja za dve prave date svojim koeficijentima pravca i slobodnim članovima određuje njihovu tačku preseka. Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju tačku preseka (ako su paralelne). Napisati program koji učitava podatke o pravama, poziva napisanu funkciju i ispisuje odgovarajuću poruku.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite k i n za prvu pravu: 4 5  
|| Unesite k i n za drugu pravu: 11 -4  
|| Prave se seku u tacki (1.29,10.14).
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite k i n za prvu pravu: 0.5 -4.7  
|| Unesite k i n za drugu pravu: 0.5 9.1  
|| Prave su paralelne.
```

[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 vrednost 1 ukoliko je uspešno izračunala vrednosti i 0 ukoliko deljenje nije moguće. Napisati program koji testira rad ove funkcije.

**Zadatak 3.3.5** Napisati funkciju koja za dinarsku vrednost cene artikla izračunava odgovarajuću cenu u evrima i u dolarima. Napisati program koji testira rad ove funkcije.

**Zadatak 3.3.6** Napisati funkciju koja za dužinu trajanja filma kaja je data u sekundama, određuje ukupno trajanje filma u satima, minutama i sekundama. Napisati program koji testira rad ove funkcije.

**Zadatak 3.3.7** Napisati funkciju `void modifikacija(char* s, char* t, int* br_modifikacija)` koja na osnovu niske *s* formira nisku *t* tako što svako malo slovo zamenjuje velikim. Broj izvršenih modifikacija se čuva u okviru argumenta *br\_modifikacija*. Pretpostaviti da niska *s* neće biti duža od 20 karaktera. Napisati program koji testira rad napisane funkcije.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 123abc789XY
Modifikovana niska je: 123ABC789XY
Broj modifikacija je: 3

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: zima
Modifikovana niska je: ZIMA
Broj modifikacija je: 3

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: SNEG
Modifikovana niska je: SNEG
Broj modifikacija je: 0

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 1234
Modifikovana niska je: 1234
Broj modifikacija je: 0

```

[Rešenje 3.3.7]

**Zadatak 3.3.8** Napisati funkciju `void interpunkcija(int* br_tacaka, int* br_zareza)` koja prebrojava tačke i zareze u tekstu koji se unosi sa standardnog ulaza. Napisati program koji testira napisanu funkciju.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
a.b.c.d
a,b,,c,d,e
Broj tacaka: 3
Broj zareza: 5

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
.....789.....
Broj tacaka: 10
Broj zareza: 0

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
sunce
Broj tacaka: 0
Broj zareza: 0

```

[Rešenje 3.3.8]

**Zadatak 3.3.9** Napisati funkciju `void par_nepar(int a[], int n, int parni[], int* pn, int neparni[], int* nn)` koja razbija niz *a* na niz parnih i niz neparnih brojeva. Pokazivači *pn* i *nn* redom treba da sadrže broj elemenata niza parnih tj. niza neparnih elemenata. Pretpostaviti da dužina niza *a* neće biti veća od 50. Napisati program koji testira napisanu funkciju.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 8
Unesite elemente niza:
1 8 9 -7 -16 24 77 4
Niz parnih brojeva: 8 -16 24 4
Niz neparnih brojeva: 1 9 -7 77

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
2 4 6 8 -11
Niz parnih brojeva: 2 4 6 8
Niz neparnih brojeva: -11

```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 2
Unesite elemente niza:
-15 15
Niz parnih brojeva:
Niz neparnih brojeva: -15 15
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 1
Unesite elemente niza:
0
Niz parnih brojeva: 0
Niz neparnih brojeva:
```

[Rešenje 3.3.9]

**Zadatak 3.3.10** Napisati funkciju `void min_max(float a[], int n, float* min, float* max)` koja izračunava minimalni i maksimalni element niza  $a$  dužine  $n$ . Napisati program koji učitava niz realnih brojeva maksimalne dužine 50 i ispisuje vrednosti minimuma i maksimuma zaokruženu na tri decimale.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza:
24.16 -32.11 999.25 14.25 11
Minimum: -32.110
Maksimum: 999.250
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 4
Unesite elemente niza:
-5.126 -18.29 44 29.268
Minimum: -18.290
Maksimum: 44.000
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 1
Unesite elemente niza:
4.16
Minimum: 4.160
Maksimum: 4.160
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza:
7.82 18.989 7.82
Minimum: 7.820
Maksimum: 18.989
```

[Rešenje 3.3.10]

**Zadatak 3.3.11** Napisati program koji ispisuje broj navedenih argumenata komandne linije, a zatim i same argumenate i njihove redne brojeve.

#### Primer 1

```
POKRETANJE: ./a.out abcde 123 -5 3.7
INTERAKCIJA SA PROGRAMOM:
Broj argumenata je 5:
0: ./a.out
1: abcde
2: 123
3: -5
4: 3.7
```

#### Primer 2

```
POKRETANJE: ./a.out
INTERAKCIJA SA PROGRAMOM:
Broj argumenata je 1:
0: ./a.out
```

[Rešenje 3.3.11]

**Zadatak 3.3.12** Napisati program koji ispisuje zbir numeričkih argumenata komandne linije. UPUTSTVO: *Koristiti funkciju atoi.*

*Primer 1*

```
|| POKRETANJE: ./a.out 5 mkp 9 -2 11 a 4 2
|| INTERAKCIJA SA PROGRAMOM:
||   Zbir numerickih argumenata: 29
```

*Primer 2*

```
|| POKRETANJE: ./a.out ab u f hj
|| INTERAKCIJA SA PROGRAMOM:
||   Zbir numerickih argumenata: 0
```

*Primer 3*

```
|| POKRETANJE: ./a.out 33 1 p 44
|| INTERAKCIJA SA PROGRAMOM:
||   Zbir numerickih argumenata: 78
```

*Primer 4*

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
||   Zbir numerickih argumenata: 0
```

[Rešenje 3.3.12]

**Zadatak 3.3.13** Napisati program koji ispisuje argumente komandne linije koji počinju slovom z.

*Primer 1*

```
|| POKRETANJE: ./a.out zima jabuka zvezda Zrak
|| INTERAKCIJA SA PROGRAMOM:
||   zima zvezda
```

*Primer 2*

```
|| POKRETANJE: ./a.out bundeva pomorandza
|| INTERAKCIJA SA PROGRAMOM:
```

*Primer 3*

```
|| POKRETANJE: ./a.out sanke zapad zujanje
|| INTERAKCIJA SA PROGRAMOM:
||   zapad zujanje
```

*Primer 4*

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
```

[Rešenje 3.3.13]

**Zadatak 3.3.14** Napisati program koji ispisuje broj argumenata komandne linije koji sadrže slovo z.

*Primer 1*

```
|| POKRETANJE: ./a.out zvezda grozd jesen kisa
|| INTERAKCIJA SA PROGRAMOM:
||   2
```

*Primer 2*

```
|| POKRETANJE: ./a.out AZBUKA deda mraz
|| INTERAKCIJA SA PROGRAMOM:
||   2
```

### 3 Predstavljanje podataka

---

#### Primer 3

```
|| POKRETANJE: ./a.out japan caj
|| INTERAKCIJA SA PROGRAMOM:
|| 0
```

#### Primer 4

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| 0
```

[Rešenje 3.3.14]

**Zadatak 3.3.15** Napisati program koji na osnovu broja  $n$  koji se zadaje kao argument komandne linije ispisuje cele brojeve iz intervala  $[-n, n]$ .

#### Primer 1

```
|| POKRETANJE: ./a.out 2
|| INTERAKCIJA SA PROGRAMOM:
|| -2 -1 0 1 2
```

#### Primer 2

```
|| POKRETANJE: ./a.out 4
|| INTERAKCIJA SA PROGRAMOM:
|| -4 -3 -2 -1 0 1 2 3 4
```

#### Primer 3

```
|| POKRETANJE: ./a.out 0
|| INTERAKCIJA SA PROGRAMOM:
|| 0
```

#### Primer 4

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| Greska: nedostaje argument komandne linije!
```

[Rešenje 3.3.15]

**Zadatak 3.3.16** Napisati program koji proverava da li se među zadatim argumentima komandne linije nalaze barem dva ista.

#### Primer 1

```
|| POKRETANJE: ./a.out pec zima deda mraz pec
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima ima istih.
```

#### Primer 2

```
|| POKRETANJE: ./a.out xyz abc abc abc efgh
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima ima istih.
```

#### Primer 3

```
|| POKRETANJE: ./a.out 11 15 abc 888
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima nema istih.
```

#### Primer 4

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima nema istih.
```

[Rešenje 3.3.16]

**Zadatak 3.3.17** Napisati funkciju koja za dva data stringa određuje koliko se uzastopnih karaktera prvog stringa nalazi u drugom stringu počev od početka. Napisati program koji testira napisanu funkciju za dva stringa koji se unose kao



argumenti komandne linije.

#### Primer 1

```
|| POKRETANJE: ./a.out aladin bal
|| INTERAKCIJA SA PROGRAMOM:
|| 3
```

#### Primer 2

```
|| POKRETANJE: ./a.out aladin lad
|| INTERAKCIJA SA PROGRAMOM:
|| 4
```

#### Primer 3

```
|| POKRETANJE: ./a.out Aladin ala
|| INTERAKCIJA SA PROGRAMOM:
|| 0
```

#### Primer 4

```
|| POKRETANJE: ./a.out aladin
|| INTERAKCIJA SA PROGRAMOM:
|| Nekorektan poziv
|| Program treba pozvati sa ./a.out arg1 arg2
```

[Rešenje 3.3.17]

**Zadatak 3.3.18** Napisati program koji ispisuje sve opcije koje su navedene u komandnoj liniji.

#### Primer 1

```
|| POKRETANJE: ./a.out -abc input.txt -d -Fg output
|| INTERAKCIJA SA PROGRAMOM:
|| a b c d F g
```

#### Primer 2

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
```

#### Primer 3

```
|| POKRETANJE: ./a.out ulaz.txt
|| INTERAKCIJA SA PROGRAMOM:
```

#### Primer 4

```
|| POKRETANJE: ./a.out file.txt -x -yZ -g output
|| INTERAKCIJA SA PROGRAMOM:
|| x yZ g
```

[Rešenje 3.3.20]

**Zadatak 3.3.19** Napisati funkciju `void sifruj(char s[], char c, int k)` koja šifruje string `s` na sledeći način: svako malo i veliko slovo stringa `s` konvertuje u slovo koje je u abecedi od njega udaljeno `k` pozicija, i to `k` pozicija ulevo, ako je karakter `c` jednak karakteru `'L'` ili udesno ako je karakter `c` jednak karakteru `'D'`. Šifrovanje treba da bude kružno. Ako string `s` sadrži karakter koji nije alfanumerički, ostaviti ga nešifriranog. Napisati program koji testira napisanu funkciju za string i prirodan broj koji se unose kao argumenti komandne linije dok se pravac šifrovanja unosi kao opcija `-p` koja može imati vrednosti `'L'` ili `'D'`. Ukoliko opcija `-p` nije navedena, podrazumevani pravac je udesno. **NAPOMENA:** *Možemo podrazumevati da string sadrži najviše 30 karaktera.*

### Primer 1

```
POKRETANJE: ./a.out abcd 2
INTERAKCIJA SA PROGRAMOM:
cdef
```

### Primer 2

```
POKRETANJE: ./a.out abcd 2 -p D
INTERAKCIJA SA PROGRAMOM:
cdef
```

### Primer 3

```
POKRETANJE: ./a.out abcd 2 -p L
INTERAKCIJA SA PROGRAMOM:
yzab
```

### Primer 4

```
POKRETANJE: ./a.out abcd -3 -p L
INTERAKCIJA SA PROGRAMOM:
Nekorektan unos
```

### Primer 5

```
POKRETANJE: ./a.out abcd 3 -p X
INTERAKCIJA SA PROGRAMOM:
Nekorektan unos
```

### Primer 6

```
POKRETANJE: ./a.out ab12cd 2 -p D
INTERAKCIJA SA PROGRAMOM:
Nekorektan unos
```

[Rešenje 3.3.19]

**Zadatak 3.3.20** Parametri komandne linije su  $n, a$  i  $b$  ( $a < b$ ). Treba popuniti prvih  $n$  elemenata niza  $A$  celim slučajnim brojevima koji su između  $a$  i  $b$ . Istampati niz  $A$  na standardni izlaz. Maksimalan broj elemenata niza  $A$  je 200. Ukoliko nisu zadati svi argumenti komandne linije ili ne zadovoljavaju potrebna svojstva ispisati poruku o grešci.

## 3.4 Rešenja

### Rešenje 3.3.20

```
1  /*
2      Napisati funkciju uredi koja uredjuje svoja dva
3      celobrojna argumenta tako da se u prvom nalazi manji
4      a u drugom veci. Napisati potom glavni program koji
5      ucitava dva cela broja i uredjuje njihove vrednosti
6      primenom napisane funkcije. Na primer, ako su ucitane
7      promenljive x=5 i y=2, njihove vrednosti nakon
8      primene funkcije uredi treba da budu x=2 i y=5.
9  */
11 #include <stdio.h>
13 /*
14      Argumenti funkcije uredi_pogresno, promenljive a i b,
15      predstavljaju lokalne promenljive za ovu funkciju
16      i prestaju da postoje po zavrsetku funkcije. Zbog toga
```

```
17     se efekti razmene vrednosti promenljivih a i b u slucaju
18     da je a>b vide u funkciji, ali se ne vide u glavnom programu.
19 */
20 void uredi_pogresno(int a, int b)
21 {
22     int t;
23
24     if (a>b)
25     {
26         t = a;
27         a = b;
28         b = t;
29     }
30     printf("uredi_pogresno :: a=%d, b=%d\n", a, b);
31     printf("uredi_pogresno :: &a=%p, &b=%p\n", &a, &b);
32 }
33
34 /*
35     Argumenti funkcije uredi_tacno, promenljive pa i pb,
36     takodje su lokalne promenljive za ovu funkciju i
37     prestaju da postoje kada se funkcija završi.
38     Njima prosledjujemo adrese promenljivih a i b koje zelimo
39     da razmenimo u slucaju da je a>b.
40
41     Promenljivoj a pristupamo preko pokazivacke promenljive
42     pa sa *pa i slicno, promenljivoj pb pristupamo sa *pb.
43
44     Vrednosti promenljivih *pa i *pb razmenjujemo kao
45     i vrednosti bilo koje dve celobrojne promenljive.
46
47 */
48 void uredi_tacno(int * pa, int * pb)
49 {
50     int t;
51     if (*pa>*pb)
52     {
53         t = *pa;
54         *pa = *pb;
55         *pb = t;
56     }
57     printf("uredi_tacno :: *pa=%d, *pb=%d\n ", *pa, *pb);
58     printf("uredi_tacno :: pa=%p, pb=%p\n ", pa, pb);
59 }
60
61 int main()
62 {
63     int a,b;
64
65     printf("Unesi dve celobrojne promenljive:");
66     scanf("%d%d",&a,&b);
67
68     printf("main :: a=%d, b=%d\n", a,b);
```

### 3 Predstavljanje podataka

---

```
69 printf("main :: &a=%p, &b=%p\n", &a, &b);
   uredi_pogresno(a,b);
71 printf("main :: nakon uredi_pogresno, a=%d, b=%d\n", a, b);

73 /*
   Funkcija uredi_tacno kao argument ima dve pokazivacke
   promenljive
75 (int*,int*). Zbog toga joj je u pozivu funkcije neophodno
   proslediti
   adrese promenljivih koje zelimo da uredimo rastuce, &a i &b.
77 */

79 uredi_tacno(&a, &b);
   printf("main :: nakon uredi_tacno, a=%d, b=%d\n", a, b);
81
83 return 0;
}
```

#### Rešenje 3.3.2

```
1 /*
   Napisati funkciju koja za boju datu u rgb formatu
3   racuna cmy format po formulama:
   C = 1 - ( R / 255 )
5   M = 1 - ( G / 255 )
   Y = 1 - ( B / 255 )
7
   Napisati program koji ucitava boju u rgb formatu,
9   primenjuje odgovarajucu funkciju i ispisuje boju u cmy formatu.

11 */

13 #include <stdio.h>
   #include <math.h>
15
17 void rgb_to_cmy(float* a, float* b, float* c)
{
   /* Zgrade su neophodne jer aritmeticke operacije
19   imaju veci prioritet od operatora dereferenciranja (*).
   */
21   *a=1-(*a)/255;
   *b=1-(*b)/255;
23   *c=1-(*c)/255;

25   /*
   Pomocu return ne mozemo vratiti vise od jedne vrednosti.
27
   Ceste greske:
29   return a,b,c;           return vraca samo jednu vrednost
   return a; return b; return c; return ce vratiti samo a
31
   */
}
```

```

33     Zato je neophodno da promenljive ciju vrednost
    zelimo da promenimo prenesemo preko pokazivaca.
35     */
37 }
39 int rgb_korektno(float a)
41 {
    if(a<0 || a>255)
43     return 0;
    return 1;
45 }
47 int main()
49 {
    float a,b,c;
51     /*
    Argumenti funkcije rgb_to_cmy su
    pokazivaci na float. Njima prosledjujemo
53     adrese promenljivih a, b i c.
    */
55     printf("Unesi boju u rgb formatu (vrednosti izmedju 0 i 255:");
57     scanf("%f%f%f",&a,&b,&c);
59     if(rgb_korektno(a) && rgb_korektno(b) && rgb_korektno(c))
        rgb_to_cmy(&a,&b,&c);
61     else
    {
63         printf("Nekorektan unos\n");
        return -1;
65     }
67     printf("Nakon konverzije: %.2f,%.2f,%.2f\n", a,b,c);
69     return 0;
    }

```

### Rešenje 3.3.3

```

2     /*
    Napisati funkciju koja za dve prave date svojim koeficijentima
    pravca i slobodnim clanovima odredjuje njihovu tacku preseka.
4     Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju
    tacku preseka (ako su paralelne). Napisati glavni program
6     koji ucitava podatke o pravama, poziva napisanu funkciju i
    ispisuje odgovarajucu poruku.
8     */

```

### 3 Predstavljanje podataka

---

```
10 #include<stdio.h>
12 /*
    Funkcija presek treba da izracuna tri vrednosti:
    1. indikator da li su koeficijenti pravca jednaki ili ne
    2. prvu koordinatu presečne tačke (ukoliko prave nisu paralelne)
    3. drugu koordinatu presečne tačke (ukoliko prave nisu paralelne)

    Indikator funkcija vraća kao povratnu vrednost, preko ključne reci
    return.

    Koordinate presečne tačke (ako postoji) funkcija vraća preko
    liste argumenata, zbog čega promenljive kojima će koordinate
    biti dodeljene prenosimo preko pokazivaca (promenljive px i py)

    Promenljive koje sadrže podatke o pravama (k1,n1,k2,n2) se ne
    menjaju u funkciji i zbog toga ih ne moramo prenositi preko
    pokazivaca.
28 */
30 int presek(float k1, float n1, float k2, float n2, float* px, float*
    py)
{
32     if (k1==k2)
        return 0;
34
    *px = -(n1-n2)/(k1-k2);
36     *py = k1*( *px)+n1;
    return 1;
38 }
40 int main()
{
42     float k1,k2,n1,n2;
    float x,y;
44
    printf("Unesi k i n za prvu pravu:");
46     scanf("%f%f",&k1,&n1);
48
    printf("Unesi k i n za drugu pravu:");
    scanf("%f%f",&k2,&n2);
50
    if(presek(k1,n1,k2,n2,&x,&y))
52         printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
    else
54         printf("Prave su paralelne\n");
56
    return 0;
}
```

#### Rešenje 3.3.7

```

1  #include <stdio.h>
2
3  #define MAX 21
4
5  void modifikacija(char *s, char *t, int *br_modifikacija)
6  {
7      int i;
8      for(i=0; s[i]; i++)
9          if(s[i]>='a' && s[i]<='z')
10             {
11                 t[i] = toupper(s[i]);
12                 (*br_modifikacija)++;
13             }
14         else
15             t[i] = s[i];
16     }
17
18     int main()
19     {
20         char s[MAX], t[MAX];
21         int br_modifikacija = 0;
22
23         printf("Unesite nisku: ");
24         scanf("%s", s);
25
26         modifikacija(s, t, &br_modifikacija);
27
28         printf("Modifikovana niska je: %s\nBroj modifikacija je: %d\n", t,
29               br_modifikacija);
30
31         return 0;
32     }

```

### Rešenje 3.3.8

```

1  /*
2  Napisati funkciju
3  void interpunkcija(int * br_tacaka, int * br_zareza)
4  koja za tekst koji se unosi sa standardnog ulaza sve do kraja ulaza
5  prebrojava
6  broj tacaka i zareza. Napisati zatim program koji testira napisanu
7  funkciju.
8  */
9
10 #include <stdio.h>
11
12 void interpunkcija(int* br_tacaka, int* br_zareza){
13     int tacke=0, zarezi=0;

```

```
15 char c;
17 while((c=getchar())!=EOF){
19     if(c==',')
21         tacke++;
23     if(c=='.')
25         zarezi++;
27 }
29 *br_tacaka=tacke;
31 *br_zareza=zarezi;
33
35 int main(){
37     int br_tacaka, br_zareza;
39     printf("Unesite tekst: \n");
41     interpunkcija(&br_tacaka, &br_zareza);
43     printf("Broj tacaka: %d\n", br_tacaka);
45     printf("Broj zareza: %d\n", br_zareza);
47     return 0;
49 }
```

#### Rešenje 3.3.9

```
2 /*
3  Napisati funkciju
4  void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
5  int* nn)
6  koja razbija niz a na niz parnih i niz neparnih brojeva. Pokazivaci
7  pn i nn
8  redom treba da sadrze broj elemenata niza parnih tj. niza neparnih
9  elemenata.
10 Pretpostaviti da duzina niza a nece biti veca od 50. Napisati program
11 koji
12 testira napisanu funkciju.
13 */
14 #include <stdio.h>
15 #define MAX 50
16
17 void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
18 int* nn){
19     int i, j, k;
```



```

18  /* i - brojac niza a */
19  /* j - brojac niza parnih brojeva */
20  /* k - brojac niza neparnih brojeva */
21
22  for(i=0, j=0, k=0; i<n; i++){
23      /* Ako je element niza paran */
24      if(a[i]%2==0){
25          /* Smestamo ga u niz parnih brojeva i uvecavamo indeks niza
26          j */
27          parni[j]=a[i];
28          j++;
29      }
30      else{
31          /* Inace, smestamo ga u niz neparnih brojeva i uvecavamo
32          indeks niza k */
33          neparni[k]=a[i];
34          k++;
35      }
36  }
37
38  *pn=j;
39  *nn=k;
40
41  }
42
43  int main(){
44      int n, i, j, pn, nn;
45      int a[MAX], parni[MAX], neparni[MAX];
46
47      /* Ucitavamo dimenziju niza */
48      printf("Unesite broj elemenata niza: ");
49      scanf("%d", &n);
50
51      if(n<0 || n>MAX){
52          printf("Greska: pogresna dimenzija niza!\n");
53          return 0;
54      }
55
56      /* Ucitavamo elemente niza */
57      printf("Unesite elemente niza: ");
58      for(i=0; i<n; i++){
59          scanf("%d", &a[i]);
60      }
61
62      /* Pozivamo funkciju koja razbija zadati niz na niz parnih i niz
63      neparnih */
64      par_nepar(a, n, parni, &pn, neparni, &nn);
65
66      /* Ispisujemo dobijene nizove */
67      printf("Niz parnih brojeva: ");
68      for(i=0; i<pn; i++)

```

### 3 Predstavljanje podataka

---

```
66     printf("%d ", parni[i]);
    printf("\n");
68
    printf("Niz neparnih brojeva: ");
70     for(i=0; i<nn; i++)
        printf("%d ", neparni[i]);
72     printf("\n");
74     return 0;
}
```

#### Rešenje 3.3.10

```
/*
2   Napisati funkciju
    void min_max(float a[], int n, float* min, float* max)
4   koja izracunava minimalni i maksimalni element niza a duzine n.
    Napisati zatim i program koji ucitava niz realnih brojeva
        maksimalne
6   duzine 50 i ispisuje vrednosti minimuma i maksimuma na tri decimalne
    .
8 */
10 #include<stdio.h>
    #define MAX 50
12
14 void min_max(float a[], int n, float* min, float* max){
16
18     int i;
20
22     /* Inicijalizujemo vrednosti minimuma i maksimuma */
    *min=a[0];
    *max=a[0];
24
26     /* Obilazimo preostale elemente niza */
    for(i=1; i<n; i++){
28
30         /* Ako je tekuca vrednost veca od maksimalne, azuriramo maksimum
        */
        if(a[i]>*max){
32             *max=a[i];
        }
34
36         /* Ako je tekuca vrednost manja od minimalne, azuriramo minimum
        */
        if(a[i]<*min){
38             *min=a[i];
        }
    }
}
```

```

36 int main(){
    int i, n;
38 float a[MAX], min, max;

40 /* Ucitavamo dimenziju niza */
    printf("Unesite broj elemenata niza: ");
42 scanf("%d", &n);

44 if(n<0 || n>MAX){
    printf("Greska: pogresna dimenzija niza!\n");
46 return 0;
    }

48 /* Ucitavamo elemente niza */
    printf("Unesite elemente niza:\n");
50 for(i=0; i<n; i++){
52     scanf("%f", &a[i]);
    }

54 /* Pozivamo funkciju za racunanje maksimuma i minimuma */
56 min_max(a, n, &min, &max);

58 /* Ispisujemo rezultat */
    printf("Minimum: %.3f\n", min);
60 printf("Maksimum: %.3f\n", max);

62 return 0;
64 }

```

### Rešenje 3.3.11

```

/*
2   Napisati program koji ispisuje broj navedenih argumenata komandne
    linije,
    a zatim i same argumente i njihove redne brojeve.
4 */

6 #include <stdio.h>

8 /*
    Argumenti komandne linije cuvaju se u nizu niski pod nazivom
10 argv. Svaki element tog niza odgovara jednom argumentu komandne
    linije pri cemu prvi element predstavlja naziv programa koji
12 pokrecemo. Celobrojna promenljiva argc predstavlja ukupan
    broj argumenata komandne linije ukljucujuci i argument koji
14 odgovara nazivu programa.
    */
16 int main(int argc, char *argv[])

```

### 3 Predstavljanje podataka

---

```
18 {
    int i;
20
    printf("Broj argumenata je: %d\n",argc);
22
    for(i=0; i<argc; i++)
24         printf("%d: %s\n",i,argv[i]);
26
    return 0;
}
```

#### Rešenje 3.3.12

```
1 #include <stdio.h>
3 int main(int argc, char* argv[]) {
5     int i;
    int s = 0;
7
    /* char *argv[] <--- niz niski koje predstavljaju argumente
       navedene iza poziva programa
9     int argc    <--- ukupan broj niski (sa sve nazivom programa)
       navedenih prilikom pozivanja
11
    Ukoliko je program pozvan sa ./a.out 12 abc 6 5 3ab
13
    argv[0] = "./a.out".
    argv[1] = "12"
15    argv[2] = "abc"
    argv[3] = "6"
17    argv[4] = "5"
    argv[5] = "3ab"
19
    argc iznosi 6
21
23    Kako je argv[] po prirodi niz,
    koristimo tzv. brojacku odnosno
25    for petlju
    i obradjujemo svaki od argumenata.
27    */
29
    /* Funkcija atoi() prihvata nisku,
       i racuna dekadnu vrednost prosledjene niske,
31    dokle god se ona moze racunati.
    Na primer, ukoliko je niska "-123",
33    atoi() vraca broj -123.
    Ako je, pak, niska "123abc",
35    atoi() ce vratiti 123
```

```

37     (prilikom prve pojave karaktera koji nije cifra, funkcija prekida
        izracunavanje).

39     To za posledicu ima da, ukoliko je funkciji
        prosledjeno nesto
41     sto se ne moze pretvoriti u broj,
        na primer niska "abcd",
        funkcija atoi() vraca 0.
43 */

45     for(i = 1; i < argc; i++)
        s += atoi(argv[i]); /* Zbog nacina rada funkcije atoi(), mozemo
        je pozvati nad svim argumentima
47         komandne linije, i sabrati odgovarajuce dekadne
        vrednosti.
        Ukoliko neki argument i nije broj, to ne predstavlja
        problem
49         jer ce u tom slucaju odgovarajuci sabirak biti 0
        */

51

53     printf("Zbir numerickih argumenata: %d\n", s);

55     return 0;
}

```

### Rešenje 3.3.13

```

1  #include <stdio.h>

3  int main(int argc, char* argv[]) {

5      int i;

7      /* Prolazimo for petljom kroz niz argumenata,
        i trazimo one niske ciji je prvi karakter bas 'z'.
9      Ukoliko je trenutni argument koji se ispituje
        argv[i],
11     kako je on sam po sebi niska,
        do prvog karaktera dolazimo kao i pri dosadasnjem
13     radu sa niskama --> argv[i][0]
        ^
15         |
        index prvog karaktera u niski argv[i]

17     */

19     for(i = 1; i < argc; i++)
        if(argv[i][0] == 'z')
21         printf("%s ", argv[i]);

23     putchar('\n');
}

```

### 3 Predstavljanje podataka

---

```
25     return 0;
    }
```

#### Rešenje 3.3.14

```
1  #include <stdio.h>
   #include <string.h>
3
   int main(int argc, char* argv[]) {
5
       int i;
       int br = 0;
7
9      /* Da bismo proverili da li se karakter 'z' (tj. 'Z')
       nalazi u niski argv[i],
       to mozemo uciniti koriscenjem funkcije
       strchr() koja se nalazi u string.h.
11
       Ukoliko je karakter sadržan u okviru niske,
       strchr() vraća pokazivac na taj karakter
       unutar same niske.
       Inace, ukoliko se karakter ne nalazi u niski,
       funkcija vraća NULL.
13
       */
15
17     for(i = 1; i < argc; i++)
19         if(strchr(argv[i], 'z') != NULL || strchr(argv[i], 'Z') != NULL)
21             br++;
23
25     printf("%d\n", br);
27
       return 0;
   }
```

#### Rešenje 3.3.15

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   int main(int argc, char *argv[])
5   {
       int n,i;
7
       /*
9      Ispisujemo gresku ukoliko nema dovoljno argumenata komandne
       linije.
       */
11     if(argc != 2)
```

```

13     {
14         printf("Greska: nedostaje argument komandne linije!\n");
15         return -1;
16     }
17
18     /*
19      Pretvaramo argument komandne linije koji je string u ceo broj
20      koriscenjem funkcije atoi
21     */
22     n = atoi(argv[1]);
23     n = abs(n);
24
25     for(i=(-1)*n; i<=n; i++)
26         printf("%d ", i);
27
28     return 0;
29 }

```

### Rešenje 3.3.16

```

1  #include <stdio.h>
2  #include <string.h>
3
4  int main(int argc, char *argv[])
5  {
6      int indikator = 0;
7      int i, j;
8      /*
9       Ukoliko imamo samo jedan argument komandne linije,
10       ispisujemo da nema istih i završavamo program.
11      */
12      if(argc < 2)
13      {
14          printf("Medju argumentima nema istih.\n");
15          return -1;
16      }
17
18      /*
19       Prolazimo kroz niz argumenata i za svaki posebno proverimo
20       da li medju ostalima postoji neki koji mu je jednak i ako postoji
21       ispisujemo poruku i završavamo program.
22       Ako smo izašli iz prve petlje to znaci da nismo pronašli dva ista
23       elementa
24       i ispisujemo odgovarajucu poruku.
25      */
26      for(i=0; i<argc; i++)
27      {
28          for(j=0; j != i && j<argc; j++)
29              if(strcmp(argv[i], argv[j]) == 0)
30              {
31                  printf("Medju argumentima ima istih.\n");
32                  return -1;
33              }
34      }
35
36      return 0;
37 }

```

### 3 Predstavljanje podataka

---

```
31         return 0;
32     }
33 }
34
35 printf("Medju argumentima nema istih.\n");
36 return 0;
37 }
```

#### Rešenje 3.3.17

```
/*
2   Napisati funkciju koja za dva data stringa str i
   accept odredjuje koliko se uzastopnih karaktera stringa str
4   nalazi u stringu accept pocev od pocetka niza str. Napisati
   potom program koji testira napisanu funkciju za dva stringa
6   koji se unose kao argumenti komandne linije. Primeri upotrebe:

8   1:
   ./a.out aladin bal
10  3

12  2:
   ./a.out aladin lad
14  4

16  3:
   ./a.out Aladin ala
18  0

20 */
22 #include <stdio.h>
   #include <string.h>
24
26 /*
   Funkcija strspn(str,accept) je ugradjena funkcija koja vraca broj
   karaktera
   stringa str koji se nalaze u stringu accept, pocev od pocetka
   stringa str.

28   Funkcija strspn se nalazi u zaglavlju string.h.

30   Funkcija strspn_klon je jedna implementacija funkcije strspn.

32   U zadacima cemo uvek koristiti ugradjenu funkciju strspn osim ako
   u tekstu zadatka
34   nije naglaseno da se ona ne sme koristiti. Funkcija strspn_klon
   sluzi da pokaze na koji
   nacin radi ugradjena funkcija strspn.
36 */
```



```

38     Ugradjena funkcija strspn poziva se na isti nacin kao funkcija
        strspn_klon:
        strspn(s1,s2)

40 */

42 int strspn_klon(char str[], char accept[])
{
44     int br=0;
    int i;

46     for(i=0; str[i];i++)
48         if(strchr(accept, str[i])!=NULL)
            br++;
50     else /* ako pronadjemo karakter u stringu str koji nije */
        break; /* u stringu accept, prekidamo petlju */

52     return br;
54 }

56 int main(int argc, char* argv[])
{
58     int br;

60     if(argc<3)
62     {
        printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
        arg2\n");
64         return -1;
    }

66     br = strspn_klon(argv[1],argv[2]);
68     printf("Broj karaktera stringa %s koji se nalaze u stringu %s,
        pocev od pocetka stringa %s: %d\n", argv[1],argv[2],argv[1],br);
    return 0;
70 }

```

### Rešenje 3.3.20

```

1  #include <stdio.h>

3  void suma(int a, int b, int *s);

5

7  int main()
{
    int a,b,s;

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

11

```

### 3 Predstavljanje podataka

---

```
13     suma(a,b,&s);
15     printf("suma: %d\n",s);
17     return 0;
19 void suma(int a, int b, int *s)
21 {
    *s = a + b;
}
```

#### Rešenje 3.3.19

```
1  /*
2     Napisati funkciju void sifruj(char s[], char c, int k) koja
3     sifruje
4     string s na sledeci nacin: svako malo i veliko slovo stringa s
5     konvertuje u
6     slovo koje je u abecedi od njega udaljeno k pozicija, i to
7     k pozicija ulevo, ako je karakter c jednak karakteru 'L' ili
8     udesno
9     ako je karakter c jednak karakteru 'D'. Sifrovanje treba da bude
10    kruzno. Ako string
11    s sadrzi karakter koji nije alfanumericki, ostaviti ga
12    nesifriranog.
13
14    Napisati potom glavni program koji testira napisanu funkciju za
15    string i prirodan
16    broj koji se unose kao argumenti komandne linije dok se pravac
17    sifrovanja unosi
18    kao opcija -p koja moze imati vrednosti 'L' ili 'D'. Ukoliko
19    opcija -p nije
20    navedena, podrazumevani pravac je udesno.
21
22    Mozemo podrazumevati da string sadrzi najvise 30 karaktera.
23
24    Primeri upotrebe:
25
26    1:
27    ./a.out abcd 2
28    cdef
29
30    2:
31    ./a.out abcd 2 -p D
32    cdef
33
34    3:
35    ./a.out abcd 2 -p L
36    yzab
37
38    */
```

```

4:
31  ./a.out abcd -3 -p L
    Nekorektan unos
33
5:
35  ./a.out abcd 3 -p X
    Nekorektan unos
37
6:
39  ./a.out ab12cd 2 -p D
    cd12ef
41
*/
43
#include <stdio.h>
45 #include <string.h>
#include <stdlib.h>
47 #define MAX 31

49 void sifruj(char s[], char c, int k)
{
51     int i;
    int znak;
53     char t;

55     /*
        S obzirom da ce korektnost unosa podataka
57     biti ispitana pre poziva funkcije, promenljiva
        c ce imati vrednost 'L' ili 'D'.

59
        Promenljiva znak ima vrednost 1 ili -1
61     i sluzi kao pomocna promenljiva u slucaju
        da prilikom sifriranja konvertovani
63     karakter izadje iz opsega malih ili velikih slova.

65     */
    znak=1;
67     if (c=='L')
        znak = -1;
69

71     for(i=0; s[i];i++)
        if(isalpha(s[i]))
73     {
        /*
75         Promenljiva t predstavlja sifrirani karakter s[i].
        Ako je promenljiva t izvan opsega malih ili velikih slova
        ,
77         dodajemo joj ili oduzimamo ukupan broj slova u abecedi
        (26),
        u zavisnosti od pravca sifriranja, kako bismo omogucili
79         kruzno sifriranje.

```

### 3 Predstavljanje podataka

---

```

81     */
    t = s[i]+znak*k;
    if(((islower(s[i]) && (t<'a' || t>'z')) || (isupper(s[i]) &&
83     (t<'A' || t>'Z'))))
        s[i]=t-znak*26;
    else
85         s[i]=t;
    }
87 }

89 int main(int argc, char* argv[])
90 {
91
92     int k;
93     char pravac;
94     char rec[MAX];
95
96     /*
97     Program mozemo pozivati na dva nacina:
98     ./a.out abcd 2
99     ili
100    ./a.out abcd 2 -p D
101
102    Zbog toga, broj argumenata moze biti 3 ili 5.
103    */
104
105    if (argc!=3 && argc!=5)
106    {
107        printf("Nekorektan unos: broj argumenata moze biti 3 ili 5\n");
108        return -1;
109    }
110
111    /*
112    Argumenti komandne linije su stringovi. Ako program pokrecemo
113    na sledeci nacin:
114    ./a.out abcd 2 -p D
115    to znaci da je argument koji odgovara dvojci u stvari
116    string "2". Da bismo string konvertovali u ceo broj,
117    koristimo ugradjenu funkciju atoi iz biblioteke stdlib.h.
118    */
119
120    k = atoi(argv[2]);
121
122    /*
123    Ispitujemo korektnost datih podataka:
124    */
125    if (k<=0)
126    {
127        printf("Nekorektan unos: broj pozicija mora biti pozitivan ceo
128        broj\n");
129        return -1;
130    }
131 }
```

```
131  /* Korektnost unosa je ispitana, sto znaci da
132  argc moze biti 3 ili 5 */
133
134  if (argc==3) /* Ako je argc 3: */
135      pravac='D';
136  else /* Ako argc nije 3, tada je sigurno 5, jer je */
137  { /* korektnost unosa ispitana, a unos je korektan
138      jedino za argc==3 ili argc==5 */
139      /*
140          Ispitujemo korektnost prethodnjeg argumenta koji mora da
141          bude u formatu "-p".
142          Ovaj argument je string argv[3]. Njegovom prvom karakteru (
143          koji treba
144          da bude '-') pristupamo sa argv[3][0] a drugom sa argv
145          [3][1].
146      */
147      if (argv[3][0] != '-')
148      {
149          printf("Nekorektan unos: pri zadavanju opcija prvi karakter
150          mora biti '-' \n");
151          return -1;
152      }
153
154      if (argv[3][1]!='p')
155      {
156          printf("Nekorektan unos: nedozvoljena opcija\n");
157          return -1;
158      }
159
160      /*
161          Nakon argumenta -p sledi argument koji zadaje vrednost ove
162          opcije. To je
163          poslednji argument kome pristupamo sa argv[4]. Ovaj argument
164          treba
165          da sadrzi samo jedan karakter - 'L' ili 'D' i njemu
166          pristupamo sa
167          argv[4][0].
168      */
169      if(argv[4][0]=='L' || argv[4][0]=='D')
170          pravac=argv[4][0];
171      else
172      {
173          printf("Nekorektan unos: pravac moze biti L ili D\n");
174          return -1;
175      }
176  }
177
178  strcpy(rec, argv[1]);
179  sifruj(rec,pravac,k);
180
181  printf("Sifrovana rec: %s\n", rec);
```

```
175     return 0;
    }
```

## 3.5 Niske

**Zadatak 3.5.1** Napisati funkciju `void konvertuj(char s[])` koja menja datu nisku `s` tako što u njoj mala slova zamenjuje odgovarajućim velikim slovima, a velika slova zamenjuje odgovarajućim malim slovima. Napisati program koji testira ovu funkciju za učitane nisku maksimalne dužine 10 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: Beograd
|| bEOgRAD
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: A+B+C
|| a+b+c
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 12345
|| 12345
```

[Rešenje 3.5.1]

**Zadatak 3.5.2** Napisati funkciju `void modifikacija(char s[])` koja modifikuje nisku `s` tako što u njoj svaki drugi karakter zameni zvezdicom. Napisati program koji testira rad napisane funkcije za učitane nisku maksimalne dužine 20 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 123abc789XY
|| Modifikovana niska je: 1*3*b*7*9*Y
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: zima
|| Modifikovana niska je: z*m*
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: SNEG
|| Modifikovana niska je: S*E*
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: *a*b*c*
|| Modifikovana niska je: *****
```

[Rešenje 3.5.2]

**Zadatak 3.5.3** Napisati program koji vrši poređenje niski. Napisati funkcije:

- (a) `int poredjenje(char s1[], char s2[])` — vraća 1 ako su  $s_1$  i  $s_2$  jednake niske, a 0 u suprotnom;
- (b) `void u_velika_slova(char s[])` — pretvara sva slova niske `s` u velika, a ostale znakovne ne menja.

Napisati program koji za učitane dve reči dužine najviše 20 znakova ispituje da li su jednake, pri čemu se zanemaruje razlika između velikih i malih slova.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| isPiT2010
|| IsPiT2010
|| jesu jednake
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Prog1
|| prog2
|| nisu jednake
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| junski
|| septembarski
|| nisu jednake
```

[Rešenje 3.5.30]

**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: abcde
|| Niska se završava samoglasnikom!
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: AaBb+cCdD
|| Niska se ne završava samoglasnikom!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: PrograMiranjE
|| Niska se završava samoglasnikom!
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: pRograMiranjE1
|| Niska se ne završava samoglasnikom!
```

[Rešenje 3.5.4]

**Zadatak 3.5.5** Napisati program koji za učitane 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 3.5.5]

**Zadatak 3.5.6** 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
|| Niska ne sadrzi veliko slovo
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
||   DiopTrija0.75
|| Niska sadrzi veliko slovo
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
||   21.06.2017.
|| Niska ne sadrzi veliko slovo
```

[Rešenje 3.5.30]

**Zadatak 3.5.7** Napisati funkciju `int podniska(char s[], char t[])` koja proverava da li je niska *t* podniska niske *s*. Napisati program koji učitava dve niske maksimalne dužine 10 karaktera i testira rad napisane funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: abcde
|| Unesite nisku t: bcd
|| t je podniska niske s!
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: abcde
|| Unesite nisku t: bCd
|| t nije podniska niske s!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: abcde
|| Unesite nisku t: def
|| t nije podniska niske s!
```

[Rešenje 3.5.7]

**Zadatak 3.5.8** Napisati funkciju `void skрати(char s[])` koja uklanja beline sa kraja date niske. Napisati program koji testira ovu funkciju za učitano liniju maksimalne dužine 100 karaktera. Prikazati učitano i izmenjenu nisku između zvezdica.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: rep belina
|| učitana niska: *rep belina
|| izmenjena niska: *rep belina*
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: tri tabulatora na kraju
|| učitana 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 učitane nisku maksimalne dužine 20 karaktera ispisuje odgovarajuću izmenjenu nisku.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| abcd123ABCD
|| 123
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| 1+2=3
|| 1+2=3
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| malaVELIKA
```

[Rešenje 3.5.30]

**Zadatak 3.5.10** Napisati funkciju `void ukloni(char *s)` koja iz niske uklanja sva slova iza kojih neposredno sledi slovo koje je u abecedi nakon njih, pri čemu se veličina slova zanemaruje. Testirati funkciju u programu za učitane liniju od najviše 100 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| zdRaVo svIma
|| zRVo vma
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 12345AbCd
|| 12345D
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| JeD1aN D52Va.
|| JeD1N D52Va.
```

[Rešenje 3.5.30]

**Zadatak 3.5.11** Napisati program koji učitava nisku `src` i formira nisku `dst` trostrukim nadovezivanjem niske `src`. Možemo pretpostaviti da niska `src` sadrži najviše 30 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: dan
|| dandandan
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 3sesira
|| 3sesira3sesira3sesira
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: a-b=5
|| a-b=5a-b=5a-b=5
```

[Rešenje 3.5.11]

**Zadatak 3.5.12** Napisati program koji za unetu reč maksimalne dužine 20 karaktera formira rezultujuću reč tako što unetu reč kopira 4 puta, pri čemu se između svaka dva kopiranja umeće crtica.

Zadatak uraditi:

### 3 Predstavljanje podataka

---

- (a) pisanjem odgovarajuće funkcije koja vrši nadovezivanje reči,
- (b) koristeći postojeću funkciju `strcat` iz biblioteke `string.h`.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: ana
|| ana-ana-ana-ana
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 123
|| 123-123-123-123
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: x*y
|| x*y-x*y-x*y-x*y
```

[Rešenje 3.5.30]

**Zadatak 3.5.13** Napisati funkciju `void kopiraj_n(char t[], char s[], int n)` koja kopira najviše  $n$  karaktera niske  $s$  u nisku  $t$ . Napisati program koji testira rad napisane funkcije. Pretpostaviti da je maksimalna dužina niske  $s$  20 karaktera.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: petar
|| Unesite broj n: 3
|| Rezultujuca niska: pet
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: gromobran
|| Unesite broj n: 4
|| Rezultujuca niska: grom
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: abc
|| Unesite broj n: 15
|| Rezultujuca niska: abc
```

[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 testira rad napisane funkcije.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: zima
|| zziimmaa
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: A+B+C
|| AA++BB++CC
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: C
|| CC
```

[Rešenje 3.5.14]

**Zadatak 3.5.15** Napisati program koji učitava nisku cifara sa eventualnim vodećim znakom i pretvara je u ceo broj. NAPOMENA: *Zadatak realizovati bez korišćenja ugrađene funkcije `atoi` iz biblioteke `stdlib.h`*

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
-1238
-1238

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
73
73

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
+1
1

```

[Rešenje 3.5.15]

**Zadatak 3.5.16** Napisati program koji učitava ceo broj i pretvara ga u nisku.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj:
-6543
-6543

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj:
84
84

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj:
5
5

```

[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.

### 3 Predstavljanje podataka

---

#### 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. Pretpostviti 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
Trecu linija
Druga linija
12
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite linije:
Danas je lep dan.
Danas je lep dan.
17
```

[Rešenje 3.5.19]

**Zadatak 3.5.20** Napisati funkcije za rad sa rečenicama:

- (a) `int procitaj_recenicu(char s[], int max_len)` koja učitava rečenicu i smešta je u nisku  $s$ . Funkcija vraća dužinu učitane rečenice. Učitavanje se završava nakon učitanoj karaktera `.` ili nakon  $max\_len - 1$  učitanih karaktera.
- (b) `void prebroj(char s[], int *broj_malih, int *broj_velikih)` koja prebrojava mala i velika slova u niski  $s$ .

Napisati program koji učitava rečenice do kraja ulaza i ispisuje onu rečenicu kod koje je razlika broja malih i velikih slova najveća.

[Rešenje 3.5.30]

**Zadatak 3.5.21** Napisati funkciju `char* strchr_klon(char s[], char c)` koja vraća pokazivač na prvo pojavljivanje karaktera `c` u niski `s` ili `NULL` ukoliko se karakter `c` ne pojavljuje u niski `s`. Napisati program koji za učitane nisku maksimalne dužine 20 karaktera i dodatni karakter testira rad napisane funkcije.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: programiranje
|| Unesite karakter c: a
|| Karakter se nalazi u niski!
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: 123456789
|| Unesite karakter c: y
|| Karakter se ne nalazi u
|| niski!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: leto2017
|| Unesite karakter c: 0
|| Karakter se nalazi u niski!
```

[Rešenje 3.5.21]

**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: opqr
|| 3
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku t: aaiioo124
|| Unesite nisku s: aeiou
|| 6
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku t: 5296abc
|| Unesite nisku s: 0123456789
|| 4
```

[Rešenje 3.5.22]

**Zadatak 3.5.23** Napisati funkciju `int duzina(char s[], char t[])` koja izračunava dužinu početnog dela niske `s` sastavljenog isključivo od karaktera sa-  
držanih u niski `t`. Napisati program koji testira ovu funkciju za dve unete niske maksimalne dužine 100 karaktera.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvu nisku:
734a.bf62
Unesite drugu nisku:
0123456789
3
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvu nisku:
abrakadabra
Unesite drugu nisku:
brada
4
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvu nisku:
popokatepetl
Unesite drugu nisku:
opna
2
```

[Rešenje 3.5.30]

**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 3.5.30]

**Zadatak 3.5.25** Napisati funkciju `void rotiraj(char s[], int k)` koja rotira nisku *s* za *k* mesta ulevo. Napisati program koji rotira učitanu nisku maksimalne dužine 20 karaktera i ispisuje rotiranu nisku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
sveska
2
eskasv
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
olovka
6
olovka
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
rezac
8
acrez
```

[Rešenje 3.5.30]

**Zadatak 3.5.26** 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:
|| AbcXyz
|| BcdYza
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| lmnopqr123
|| mnopqrs123
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| 1,2,3,4,5
|| 1,2,3,4,5
```

[Rešenje 3.5.30]

**Zadatak 3.5.27** Napisati funkciju `void sifruj(char rec[], char sifra[])` koja na osnovu date reči formira šifru tako što se svako slovo u reči zameni sa naredna tri slova u abecedi. Napisati program koji testira napisanu funkciju za reč maksimalne dužine 20 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| tamo
|| uvwbcdnopqqr
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| Zec
|| ABCfghdef
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| a+b=c
|| bcd+cde=def
```

[Rešenje 3.5.30]

**Zadatak 3.5.28** Napisati funkciju `void indel(char s1[], char s2[], char c1, char c2)` koja na osnovu niske  $s_1$  formira nisku  $s_2$  udvajanjem svih karaktera  $c_1$  u niski  $s_1$  i izbacivanjem svih karaktera  $c_2$  iz niske  $s_1$ , dok ostali karakteri ostaju nepromenjeni. Napisati program koji testira ovu funkciju za unetu nisku i dva uneta karaktera. Pretpostaviti da uneta niska nije duža od 20 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| flomaster
|| Unesite prvi karakter:
|| m
|| Unesite drugi karakter:
|| s
|| floasster
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| bojica
|| Unesite prvi karakter:
|| b
|| Unesite drugi karakter:
|| a
|| bbojic
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku:
|| patentara
|| Unesite prvi karakter:
|| t
|| Unesite drugi karakter:
|| a
|| pttenttr
```

[Rešenje 3.5.30]

**Zadatak 3.5.29** Napisati funkciju `int prepis(char a[][21], int na, char b[][21])` koja iz niza reči  $a$  dužine  $na$  prepisuje u niz  $b$  one reči koje su sastavljene samo od malih ili samo od velikih slova i vraća dužinu niza  $b$ . Napisati program koji za učitani broj  $n$  ( $0 < n \leq 50$ ) i  $n$  reči razdvojenih blanko znakom ispisuje sve unete reči sastavljene samo od malih ili samo od velikih slova. Pretpostaviti da su unete reči maksimalne dužine 20 karaktera. U slučaju da je  $n$  van dozvoljenog opsega, ispisati odgovarajuću poruku.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 3 abc ABC aBc
|| abc ABC
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 2 mnB RGa
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| -3
|| Nekorektan broj reci!
```

[Rešenje 3.5.30]

**Zadatak 3.5.30** Napisati program za rad sa brojevima zapisanim u različitim brojevnim sistemima.

- Napisati funkciju `unsigned btoi(char s[], unsigned char b)` koja određuje dekadnu vrednost zapisa datog neoznačenog broja  $s$  u datoj osnovi  $b$ .
- Napisati funkciju `void itob(unsigned n, unsigned char b, char s[])` koja datu dekadnu vrednost  $n$  zapisuje u datoj osnovi  $b$  i smešta rezultat u nisku  $s$ . Pretpostaviti da je  $0 < b \leq 16$ .

Napisati program koji za svaku učitanih liniju koja sadrže po jedan dekadni, oktalni ili heksadekadni broj (zapisan kao što se zapisuju konstante u programskom jeziku C) ispisuje odgovarajući binarni zapis. Linije se unose sve do kraja ulaza. Pretpostaviti da će sve linije sadržati ispravne brojeve i da će ti brojevi biti u opsegu tipa `unsigned`.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 0x49 0x1ABC
|| 1001001 11010101111100
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 012 435 0x64FE
|| 1010 110110011 1100100111111110
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 123 0777
|| 1111011 11111111
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 981
|| 1111010101
```



## 3.6 Rešenja

### Rešenje 3.5.1

```
1  /*
2  Napisati funkciju koja konvertuje dati string tako sto
3  mala slova menja u velika a velika u mala. Napisati
4  potom glavni program koji ucitava string, poziva napisanu
5  funkciju i ispisuje konvertovani string. Mozemo pretpostaviti
6  da string ne sadrzi vise od 10 karaktera.
7  */
8
9  #include <stdio.h>
10 #include <ctype.h>
11
12 /*
13 Kada je niz argument funkcije, dodatni argument je obavezno
14 njegova dimenzija. Kod stringova to nije slucaj jer svaki string
15 ima isti poslednji element - terminirajucu nulu - i to je oznaka
16 kraja stringa.
17 */
18 void konvertuj(char s[])
19 {
20     int i;
21
22     for(i=0; s[i]!='\0'; i++)
23         if (s[i]>='a' && s[i]<='z')
24             s[i] = toupper(s[i]); /* toupper - konvertuje malo slovo u
25             odgovarajuce veliko */
26         else if (s[i]>='A' && s[i]<='Z')
27             s[i] = tolower(s[i]); /* tolower - konvertuje veliko slovo
28             u odgovarajuce malo */
29     /*
30      Funkcije toupper i tolower se nalaze u zaglavlju ctype.h.
31
32      Konverzija malog slova u veliko bez upotrebe funkcije toupper:
33      s[i] = s[i]-'a'+'A';
34      Konverzija velikog slova u malo bez upotrebe funkcije tolower:
35      s[i] = s[i]+'a'-'A';
36
37     */
38 }
39
40 int main()
41 {
42     /*
```

### 3 Predstavljanje podataka

---

```
41      Poslednji karakter svakog stringa je terminirajuca
      nula '\0', specijalni karakter ciji je ASCII kod 0.
43
      Ukoliko pretpostavljamo da string sadrzi najvise 30
45      karaktera, neophodno je deklarirati niz od 31 karaktera,
      pri cemu se dodatni izdvaja za terminirajuću nulu.
47
      */
49      char s[31];
      printf("Unesi string:");
51
      /*
53      Za razliku od nizova koji se učitavaju i štampaju
      element po element, stringovi se mogu učitati i
55      odštampati pomoću jedne scanf/printf naredbe korišćenjem
      specifikatora %s.
57
      Funkcija scanf učitava string do prvog pojavljivanja razmaka.
59      */
      scanf("%s", s);
61
      konvertuj(s);
63
      printf("Konvertovani string: %s\n", s);
65
      return 0;
67
}
```

#### Rešenje 3.5.2

#### Rešenje 3.5.30

#### Rešenje 3.5.4

```
1  /*
   a) Napisati funkciju int samoglasnik(char c) koja proverava da li je
      zadati karakter samoglasnik. Funkcija
3  treba da vrati vrednost 1 ako karakter c jeste samoglasnik, odnosno 0
   ako nije.
   b) Napisati funkciju int samoglasnik_na_kraju(char s[]) koja
      proverava da li se niska s završava samoglasnikom
5  (koristiti funkciju iz tačke a)).
   c) Napisati program koji učitava nisku maksimalne dužine 20 karaktera
      i ispisuje da li završava samoglasnikom ili ne.
7  */
9  #include <stdio.h>
   #include <ctype.h>
```

```
11 #include <string.h>
12 #define MAX_DUZINA 20
13
14 /* Funkcija proverava da li je karakter c samoglasnik */
15 int samoglasnik(char c){
16
17     char C;
18
19     /* Konvertujemo karakter u veliko slovo kako bismo smanjili broj
20        proveru */
21     C=toupper(c);
22
23     /* Samoglasnici su slova a, e, i, o i u */
24     if(C=='A' || C=='E' || C=='I' || C=='O' || C=='U')
25         return 1;
26
27     return 0;
28 }
29
30 /* Funkcija koja proverava da li se niska s završava samoglasnikom */
31 int samoglasnik_na_kraju(char s[]){
32     int duzina;
33
34     /* Odredjujemo duzinu niske */
35     duzina=strlen(s);
36
37     /* Proveravamo da li je niska prazna */
38     if(duzina==0)
39         return 0;
40
41     /* Ako niska nije prazna, proveravamo da li se samoglasnik nalazi
42        na kraju */
43     /* Numeracija karaktera u niski počinje nulom pa zato proveravamo
44        poziciju duzina -1 */
45     return samoglasnik(s[duzina-1]);
46 }
47
48 int main(){
49     char s[MAX_DUZINA+1];
50
51     /* Učitavamo nisku */
52     printf("Unesite nisku: ");
53     scanf("%s", s);
54
55     /* Proveravamo da li se završava samoglasnikom i ispisujemo
56        odgovarajuću poruku */
57     if(samoglasnik_na_kraju(s))
58         printf("Niska se završava samoglasnikom!\n");
59     else
60         printf("Niska se ne završava samoglasnikom!\n");
61 }
```

```
59     return 0;
    }
```

#### Rešenje 3.5.5

```
/*
2   Napisati program koji za uneti string s i karakter c utvrđuje
   da li se c pojavljuje u stringu s i ukoliko se pojavljuje,
4   ispisuje indeks prvog pojavljivanja a u suprotnom ispisuje
   odgovarajuću poruku. Možemo pretpostaviti da string ima najviše
6   20 karaktera.
   */
8
   #include <stdio.h>
10  #include <string.h>
12
   int main()
   {
14       char s[21];
       char c;
16       char* p;

18       printf("Unesi karakter:");
       c=getchar();
20       printf("Unesi string:");
       scanf("%s", s);
22
   /*
24       Da smo učitavali obrnutim redom (prvo string pa karakter)
       to bismo realizovali na sledeći način:
26       printf("Unesi string:");
       scanf("%s",s);
28       getchar();
       printf("Unesi karakter:");
30       c=getchar();

32       Dodatni getchar() bi služio da "pokupi" karakter kojim
       razdvajamo unos stringa i karaktera (razmak, novi red ili
34       slično).

36   */
38   /*
       Funkcija strchr(s,c) je ugrađena funkcija koja vraća pokazivač
40       na prvi karakter u stringu s koji je jednak karakteru c, ako
       takav
       postoji, a NULL u suprotnom.

42       Indeks od pokazivaca dobijamo na isti način kao u prethodnom
       zadatku
44       sa strstr.
```

```

46      */
47      p = strchr(s,c);
48      if(p!=NULL)
49          printf("%c se pojavljuje u %s na poziciji %d\n", c, s, p-s);
50      else
51          printf("%c se NE pojavljuje u %s\n",c, s);
52      return 0;
54  }

```

### Rešenje 3.5.30

### Rešenje 3.5.7

```

/*
2  Napisati funkciju int podniska(char s[], char t[]) koja proverava da
   li je niska t podniska niske s.
   Napisati i program koji ucitava dve niske maksimalne duzine 10
   karaktera i testira rad napisane funkcije.
4
   Napomena: u biblioteci string.h postoji funkcija strstr
6  char* strstr(const char* s, const char* t)
   koja vraca adresu pocetka prvog pojavljivanja niske t u niski s ili
   NULL ako se niska t ne javlja
8  u niski s.
   */
10
11 #include<stdio.h>
12 #define MAX 11
13
14 /*Funkcija koja proverava da li je t podniska od s*/
15 int podniska(char s[], char t[]){
16     int i, j, k;
17
18     /*Spoljna petlja ide redom po niski s*/
19     for(i=0; s[i] != '\0'; i++){
20
21         /*Unutrasnja petlja ide redom po niski t*/
22         /*Promenljiva k pamti vrednost i, sluzi za poredjenje s[k] i t[j] i
           *zatim se vrsi pomeranje i po niski s i po niski t (k++, j++) */
23         /*Cim naidjemo na situaciju da se karakteri ne poklapaju, izlazimo
           iz unutrasnje petlje*/
24         for(j=0, k = i; t[j] != '\0'; j++, k++){
25             if(s[k] != t[j])
26                 break;
27         }
28     }
29
30     /*Ako smo prosli celu unutrasnju petlju (do kraja niske t), to
       znaci da su se svi karakteri iz t poklopili

```

### 3 Predstavljanje podataka

---

```

    sa karakterima iz s i onda vracamo 1*/
32 if(t[j] == '\0')
    return 1;
34 }

36 /*Na kraju vracamo 0*/
    return 0;
38 }

40
42 int main(){
    char s[MAX], t[MAX];

44 /*Ucitavamo prvu nisku*/
    printf("Unesite nisku s: ");
46 scanf("%s", s);

48 /*Ucitavamo drugu nisku*/
    printf("Unesite nisku t: ");
50 scanf("%s", t);

52 /*Pozivamo funkciju i ispisujemo odgovarajucu poruku*/
    if(podniska(s,t))
54 printf("t je podniska niske s!\n");
    else
56 printf("t nije podniska niske s!\n");

58 return 0;
}
```

#### Rešenje 3.5.8

```

/*
2   Napisati funkciju skрати koja uklanja beline sa
    kraja datog stringa.

4
    Napisati glavni program koji testira napisanu
6   funkciju na stringu "rep belina"
    ".

8 */

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

12
14 /*
    Funkcija koja racuna duzinu niza
    ne racunajuci '\0'.

16
    U biblioteci string.h definisan je veliki
18 broj funkcija za rad sa stringovima,
    ukljucujuci i funkciju strlen koja racunana
```

```
20     duzinu stringa.

22     Funkcija strlen_klon predstavlja jednu
    implementaciju funkcije strlen.

24
    U zadacima cemo uvek koristiti ugradjenu
26     funkciju strlen osim ako u tekstu zadatka
    nije naglaseno da se ona ne sme koristiti.
28     Funkcija strlen_klon služi da pokaze na koji
    način radi ugradjena funkcija strlen.

30
    Ugradjena funkcija strlen poziva se na
32     isti način kao funkcija strlen_klon:
    strlen(s1)

34
    */
36     int strlen_klon(char s[])
    {
38         int i=0;
        while(s[i]) /* ASCII kod karaktera '\0' je jednak nuli. */
40             i++;

42         return i;
    }

44
    void skрати(char s[])
46     {
        /*
48         Poslednji karakter stringa s(ne racunajuci '\0') ima
        indeks strlen_klon(s)-1; ideja je da pocnemo od poslednjeg
50         karaktera stringa i da smanjujemo indeks dokle god
        je karakter na poziciji i blanko znak.

52
        */
54         int i;
        for(i=strlen_klon(s)-1;i>=0;i--)
56             if (!isspace (s[i])) /* Ako s[i] nije blanko znak, prekidamo
                petlju. */
                    break;

58         s[i+1]='\0'; /* D0dajemo terminirajucu nulu iza indeksa i (prvi
                neblanko karakter gledano sdesna nalevo).*/

60
        /*
62         Ugradjena funkcija isspace nalazi se u biblioteci ctype.h i
        vraća 1 ako
        je dati karakter blanko znak a 0 u suprotnom.

64
        Unarni logicki operator ! oznacava negaciju.

66
        */
68     }
```

```
70 int main()
71 {
72
73     /*
74     Ukoliko string ne zelimo da učitavamo po pokretanju programa
       vec da ga unapred zadamo, to mozemo uraditi na sledeci nacin:
75     */
76     char s[]="rep belina";
77     /* U ovom slucaju nije neophodno navoditi dimenziju stringa vec
       ce ona biti automatski postavljena na broj karaktera u stringu +
       1 za
80     terminirajucu nulu. */
81
82     printf("Pre skracivanja: %s*\n", s);
83     skрати(s);
84     printf("Posle skracivanja: %s*\n", s);
85
86     return 0;
87 }
88
```

Rešenje 3.5.30

Rešenje 3.5.30

Rešenje 3.5.11

```
1  /*
2     Napisati program koji učitava string src i formira string dst
3     trostrukim nadovezivanjem stringa src. Program treba da ispise
4     string dst. Na primer, za uneti string "dan", string dst treba
5     da bude "dandandan". Pretpostaviti da string src nije duzi od
6     30 karaktera.
7     */
8
9     #include <stdio.h>
10    #include <string.h>
11
12    #define MAX 30
13    /*
14     Na stringove ne mozemo primeniti naredbu dodele.
15     Ukoliko zelimo da jedan string "dodelimo" drugom,
16     mozemo koristiti ugradjenu funkciju strcpy(s,t)
17     koja kopira karaktere stringa t
18     u string s zajedno za terminirajucom nulom.
19
20     Funkcija strcpy se nalazi u biblioteci string.h.
21
```



```
23     Funkcija strcpy_klon predstavlja jednu
    implementaciju funkcije strcpy.

25     Karakteri stringa original se, jedan po jedan,
    kopiraju u string kopija. Nakon kopiranja,
27     na kraj stringa kopija dodaje se terminalna
    nula.

29     U zadacima cemo uvek koristiti ugradjenu
    funkciju strcpy osim ako u tekstu zadatka
31     nije naglaseno da se ona ne sme koristiti.
    Funkcija strcpy_klon služi da pokaze na koji
33     način radi ugradjena funkcija strcpy.

35     Ugradjena funkcija strcpy poziva se na
    isti način kao funkcija strcpy_klon:
37     strcpy(dst,src)
    gde karaktere stringa src kopiramo
39     u string dst.

41

43     */
void strcpy_klon(char kopija[], char original[])
45 {
    int i;
47     for(i=0; original[i]; i++)
        kopija[i]=original[i];
49
    kopija[i] = '\0';
51 }

53 int main()
{
55     char src[MAX+1];    /* src, skraceno od source (izvor, odnosno sta
        kopiramo) */
    char dst[3*MAX+1];    /* dst, skraceno od destination (odrediste,
        odnosno gde kopiramo) */
57
    /*
59     Vazno je izdvojiti dovoljno memorijskog prostora
    za string dst: on treba da bude tri puta veci od
61     maksimalne duzine stringa src + jedan karakter za
    terminirajucu nulu.
63     */

65     printf("Unesi jedan string:");
    scanf("%s", src);
67
    strcpy_klon(dst,src);
69

    /*
71     Funkcija strcat(s,t) nadovezuje karaktere stringa
```

```
73     t na kraj stringa s i novi string terminira
      karakterom '\0' .

75     Funkcija strcat se nalazi u biblioteci string.h.
      */
77     strcat(dst,src);
      strcat(dst,src);

79     printf("Kada nadovezemo string %s triput: %s\n",src,dst);

81     return 0;
83 }
```

#### Rešenje 3.5.30

#### Rešenje 3.5.13

```
/*
2  Napisati funkciju void kopiraj_n(char t[], char s[], int n) koja
      kopira najviše n karaktera niske s u nisku t.
      Napisati i program koji učitava nisku maksimalne dužine 20 karaktera
      i jedan ceo broj i testira rad napisane funkcije.
4  */

6  #include <stdio.h>
      #define MAX_DUZINA 20

8
      void kopiraj_n(char t[], char s[], int n){
10     int i;

12     /* Brojac i oznacava tekucu poziciju u niski */
      /* Uslov i<n je neophodan zbog kopiranja najviše n karaktera */
14     /* Uslov s[i]!='\0' (ili skraceno samo s[i]) je neophodan kako bi
      bili sigurni da na poziciji i postoji karakter u niski s */
      for(i=0; i<n && s[i]!='\0'; i++){
16         t[i]=s[i];
      }

18     /* Upisujemo terminirajucu nulu u novodobijenu nisku */
20     t[i]='\0';
      }

22

24 int main(){
      int n;
26     char s[MAX_DUZINA+1], t[MAX_DUZINA+1];

28     /* Ucitavamo nisku */
      printf("Unesite nisku: ");
30     scanf("%s", s);
```

```

32  /* Ucitavamo broj n i proveravamo korektnost unosa */
    printf("Unesite broj n: ");
34  scanf("%d", &n);
    if(n<0 || n>MAX){
36      printf("Greska: pogresan unos!\n");
        return 0;
38  }

    /* Pozivamo funkciju za kopiranje */
    kopiraj_n(t, s, n);

    /* Ispisujemo dobijenu nisku */
44  printf("Rezultujuca niska: %s\n", t);

46  return 0;
}

```

### Rešenje 3.5.14

```

1  /*
    Napisati funkciju void dupliranje(char t[], char s[]) koja na osnovu
        niske s formira nisku t tako sto duplira svaki
3  karakter niske s. Napisati i program koji ucitava nisku maksimalne
        duzine 20 karaktera i testira rad napisane funkcije.
    */
5
    #include <stdio.h>
7  #define MAX_DUZINA 20

9  void dupliranje(char t[], char s[]){
        int i, j;
11
        /* Brojac i oznacava tekucu poziciju u niski s */
13  /* Brojac j oznacava tekucu poziciju u niski t */
        for(i=0, j=0; s[i]!='\0'; i++, j+=2){
15            t[j]=s[i];
            t[j+1]=s[i];
17        }

19  /* Upisujemo terminirajucu nulu u novodobijenu nisku */
        t[j]='\0';
21    }

23
25  int main(){
        int n;
        char s[MAX_DUZINA+1], t[2*MAX_DUZINA+1];

27
        /* Ucitavamo nisku */
29  printf("Unesite nisku: ");

```

### 3 Predstavljanje podataka

---

```
scanf("%s", s);
31
/* Pozivamo funkciju za dupliranje */
33 dupliranje(t, s);

/* Ispisujemo dobijenu nisku */
35 printf("%s\n", t);
37
return 0;
39 }
```

#### Rešenje 3.5.15

```
/*
2 Napisati program koji pretvara nisku u ceo broj.
  Npr. za ulaz "-1238" se generise rezultat -1238
4 Pogladatai funkcije atoi i atof koje postoje u biblioteci stdlib.h
*/
6
#include <stdio.h>
8
#include <ctype.h>
#define MAX 10
10
/*
12 String b se sastoji od karaktera koji
  cine jedan ceo broj, onim redom kojim
  se karakteri pojavljuju u zapisu broja.
14
  Ako je prvi karakter stringa b '-',
16 to znaci da je broj negativan i
  funkcija znak_broja vraca -1
18
  U suprotnom, broj je pozitivan i
20 funkcija znak_broja vraca 1
22 */
24 int znak_broja(char b[])
{
26     if(b[0]=='-')
        return -1;
28     return 1;
30 }

32 /*
34 Funkcija formiraj_broj na osnovu
  karaktera koji cine broj iz stringa
  b vraca ceo broj koji odgovara
36 zapisu datom u stringu b.
38
  Ako su cifre broja a,b,c i d, tada
```

```
40     broj mozemo kreirati kao:
41     a*10^3 + b*10^2 + c*10^1 + d*10^0
42
43     Medjutim, efikasnije je koristiti
44     Hornerovu semu:
45
46     10*(10*(10*(10*0 + a)+b)+c)+d
47
48 */
49
50 int formiraj_broj(char b[])
51 {
52     int i;
53     int n=0;
54     int znak = znak_broja(b);
55
56     /*
57      Ako je broj negativan, cifre u nizu b
58      pocinju od indeksa 1
59     */
60
61     i=0;
62     if(znak==-1)
63         i=1;
64
65     /*
66      Funkcija isdigit proverava da li je broj
67      cifra. Nalazi se u biblioteci ctype.h
68
69      Proveravamo da li je karakter u zapisu
70      broja cifra kako bismo se osigurali
71      od nekorektnog unosa, npr ako korisnik
72      unese -123abc. Ovaj unos je moguc jer
73      se vrsi sa scanf("%s",broj), gde unosimo
74      karaktere do prvog blanko znaka
75
76      Ako naidjemo na karakter koji nije cifra,
77      prekidamo petlju
78
79     */
80     for(; b[i]!='\0'; i++)
81         if(isdigit(b[i]))
82             n = n*10 + b[i] - '0';
83         else
84             break;
85
86     /* Formirani broj mnozimo znakom: */
87
88     n*=znak;
89     return n;
90 }
```

### 3 Predstavljanje podataka

---

```
92 int main()
93 {
94     char broj[MAX];
95     int n;
96
97     /* Ucitavamo broj: */
98     scanf("%s", broj);
99
100    /* Ispisujemo rezultat: */
101    printf("Broj zapisan kao int: %d\n", formiraj_broj(broj));
102
103    return 0;
104 }
```

#### Rešenje 3.5.16

```
/*
2  Napisati program koji pretvara zadatu broj u nisku.
  Npr. za broj -453 treba generisati nisku "-453"
4  */

6  #include <stdio.h>
  #include <string.h>
8  #define MAX 10
  /*

10     Funkcija transformisi_negativan vraca
12     1 ako je broj negativan i 0 u suprotnom, a
  uz to, ako broj jeste negativan, funkcija
14     treba da ga konvertuje u njegovu apsolutnu
  vrednost. S obzirom da funkcija treba da vrati dve
16     vrednosti, to realizujemo na sledeci nacin:
  1. indikator da li je broj negativan
18     ce vratiti kao povratnu vrednost
  2. apsolutnu vrednost broja ce vratiti
20     preko liste argumenata, zbog cega broj
  prenosimo preko pokazivaca

22  */
24  int transformisi_negativan(int* pn)
  {
26      if(*pn<0)
      {
28          *pn = -(*pn);
          return 1;
30      }
      return 0;
32  }

34  int formiraj_niz_cifara(int n, char b[], int neg)
```

```
{
36     int i=0;
    char cifra;
38
    do
40    {
        cifra = n%10;
42
        /* Promenljiva b predstavlja string.
        Da bismo na neku poziciju u stringu
        upisali karakter koji odgovara nekoj
        cifri, npr '2', neophodno je da
        odgovarajucoj poziciji dodelimo vrednost
        ASCII koda te cifre, konkretno za '2'
        ASCII kod je '0'+2.
50
        Greska bi bila navesti b[i]=2
52        jer 2 nije ASCII kod koji odgovara karakteru
        '2'.
54        */
        b[i]=cifra+'0';
56
        n/=10;
58        i++;
    } while(n);
60
    /* Ako je broj negativan, dodajemo znak minus: */
62    if(neg)
    {
64        b[i]='-';
        i++;
66    }

68    /* Svaki string se zavrшава terminirajucom nulom: */
    b[i]='\0';
70 }

72 void obrni(char s[])
{
74
    char t;
76    int i,j;
    /*
78    Karaktere stringa obrcemo tako sto razmenimo karaktere na
    pozicijama 0 i n-1,
    zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
    druge
80    */

82    for(i=0,j=strlen(s)-1;i<j;i++, j--)
    {
84        t = s[i];
```

### 3 Predstavljanje podataka

```

    s[i] = s[j];
86     s[j] = t;
    }
88
89 }
90
91 void broj_u_niz_cifara(int n, char broj[])
92 {
93     int negativan;
94
95     /* Odredjujemo znak broja: */
96     negativan=transformisi_negativan(&n);
97
98     /* Izdvajamo cifre broja i smestamo ih u niz: */
99     formiraj_niz_cifara(n, broj, negativan);
100
101     /* S obzirom da cifre izdvajamo sa kraja broja, u nizu ce biti u
102        obrnutom redosledu.
103        Na primer, za broj 234 niz ce sadrzati cifre 4 3 2. */
104     obrni(broj);
105 }
106
107 int main()
108 {
109     int n;
110     char broj[MAX];
111     int negativan;
112
113     /* Ucitavamo broj: */
114     scanf("%d", &n);
115
116     /* Kreiramo broj na osnovu niza cifara: */
117     broj_u_niz_cifara(n,broj);
118
119     /* Ispisujemo rezultat: */
120     printf("Broj zapisan kao string: %s\n", broj);
121
122     return 0;
123 }
```

#### Rešenje 3.5.17

```

1  /*
   Napisati funkciju int heksa_broj(char s[]) koja proverava da li je
   niskom s zadat korektan heksadekadni broj.
3  Heksadekadni broj je korektno zadat ako pocinje prefiksom 0x ili 0X
   i ako sadrzi samo cifre i mala ili velika slova A, B, C, D, E i F
   .
   Funkcija treba da vrati vrednost 1 ako je niska korektan
   heksadekadni broj, odnosno 0 ako nije.
```



```

5 | Napisati i program koji ucitava nisku maksimalne duzine 7 karaktera
   | i ispisuje rezultat rada funkcije.
   | */
7 |
   | #include<stdio.h>
9 | #define MAX 8
   |
11 |
   | /*
13 | Funkcija koja proverava da li je prosledjeni karakter ispravna
   |   heksadekadna cifra (broj ili slovo a,b,c,d,e,f)
   |   Ukoliko jeste, funkcija vraca 1, u suprotnom 0.
   | */
15 | int heksa_cifra(char c){
17 |     /*Pretvaramo karakter c u veliko slovo*/
   |     c = toupper(c);
19 |
   |     /*Proveravamo da li je u pitanju cifra ili slovo A,B,C,D,E,F i
   |       ukoliko jeste, vracamo 1*/
21 |     if(isdigit(c) || (c >= 'A' && c <= 'F'))
   |         return 1;
23 |
   |     /*Ukoliko nije, vracamo 0*/
25 |     return 0;
   | }
27 |
   | /*Funkcija koja proverava da li prosledjena niska s predstavlja
   |   ispravan heksadekadni broj */
29 | int heksa_broj(char s[]){
   |     int i;
31 |
   |     /*Kako heksadekadni brojevi pocinju sa 0x ili 0X, prvo proveravamo
   |       da li je taj uslov ispunjen,
33 |     tj. da li je s[0] jednak 0 i da li je s[1] jednak X i ako taj uslov
   |       nije ispunjen, onda
   |     niska s ne predstavlja korektan heksadekadni broj */
35 |     if(s[0] != '0' || toupper(s[1]) != 'X')
   |         return 0;
37 |
   |     /*Prolazimo kroz nisku, pocev od pozicije 2 (jer su prve dve
   |       pozicije 0x) i za svaki karakter do kraja
39 |     niske proveravamo da li je ispravna heksadekadna cifra.
   |     Ako naidjemo na bilo koji koji ne ispunjava taj uslov, onda niska
   |     s nije korektan heksadekadni broj
   |     i vracamo 0. */
41 |     for(i=2; s[i] != '\0'; i++)
43 |         if(!heksa_cifra(s[i]))
   |             return 0;
45 |
   |     /*Ako smo stigli do kraja, znaci da su svi karakteri date niske
   |     ispravne heksadekadne cifre
47 |     i zato vracamo 1 */

```

### 3 Predstavljanje podataka

```
    return 1;
49 }

51 int main(){
    char s[MAX];

53     /*Ucitavamo nisku*/
55     printf("Unesite nisku:");
    scanf("%s", s);

57     /*Pozivamo funkciju i stampamo odgovarajucu poruku*/
59     if(heksa_broj(s))
        printf("Korektan heksadekadni broj!\n");
61     else
        printf("Nekorektan heksadekadni broj!\n");
63     return 0;
65 }
```

#### Rešenje 3.5.18

```
1  /* Napisati funkciju int heksa_broj(char s[]) koja izracunava dekadnu
    vrednost heksadekadnog broja zadatog niskom s.
    * Napisati i program koji ucitava nisku maksimalne duzine 7
    karaktera i ispisuje rezultat rada funkcije.
3  * Pretpostaviti da je uneta niska korektan heksadekadni broj. */

5  #include<stdio.h>
    #include<string.h>

7
    #define MAX 8

9
    /*Funkcija koja racuna dekadnu vrednost heksadekadnog broja*/
11 int heksa_broj(char s[]){
    int i,k;
13     char c;

15     /*Racunamo duzinu niske koja predstavlja heksadekadni broj*/
    int n = strlen(s);

17     /*Inicijalizujemo vrednost v na 0*/
19     int v = 0;

21     /*Prolazimo petljom kroz nisku, krenuvsi sa desne strane
    npr: 1a8e = e*1 + 8*16 + a*256 + 1*4096
23     Promenljiva k ce nam biti mnozilac i ona uzima vrednosti 1, 16,
    256, 4096, ...
    Promenljiva c ce nam cuvati trenutnu heksadekadnu cifru (u nasem
    primeru e, 8, a, 1)
25     U svakom koraku treba na ispravan nacin da pomnozimo c i k
    */
    }
```

```

27  for(i=n-1, k=1; i>=2; i--, k*=16)
    {
29  /*U c smestamo trenutnu heksadekadnu cifru.
    Pozivamo funkciju toupper da bi obezbedili da radimo samo sa
        velikim slovima.
31  Ako je s[i] cifra, funkcija toupper je nece promeniti.
        */
33  c = toupper(s[i]);

35  if(isdigit(c)){
        /*Ako je c cifra, onda samu vrednost te cifre dobijamo sa c-'0'
37  NAPOMENA: Nije ispravno napisati c*k jer je c karakter!
        */
39  v += (c-'0')*k;
    }else{
41  /*Ako je c slovo izmedju A i F, mi treba da dobijemo odgovarajucu
        vrednost izmedju 10 i 15.
        Ova vrednost se dobija sa 10 + c - 'A'. npr. za A ce biti 10 + '
        A' - 'A' = 10, za B: 10 + 'B' - 'A' = 11, ...*/
43  v += (c - 'A' + 10)*k;
    }
45  }

47  /*Na kraju vracamo izracunatu vrednost */
    return v;
49  }

51  int main(){
    char s[MAX];

53  /*Ucitavamo nisku*/
55  printf("Unesite nisku:");
    scanf("%s", s);

57  /*Ispisujemo rezultat*/
59  printf("%d\n", hekza_broj(s));

61  return 0;
    }

```

### Rešenje 3.5.19

```

1  /*
    Napisati funkciju int ucitaj_liniju(char s[], int n)
3  koja ucitava liniju maksimalne duzine n u string s
    i vraca duzinu učitane linije. Linija moze da sadrzi
5  blanko znakove ali ne moze da sadrzi \n ili EOF.

7  Napisati potom glavni program koji ucitava linije
    do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko
9  ima vise linija maksimalne duzine, ispisati prvu. Mozemo

```

```
11     pretpostviti da svaka linija sadrzi najvise 80 karaktera,  
12     zajedno sa \n.  
13 */  
14  
15 #include<stdio.h>  
16 #include<string.h>  
17 #define MAX 81  
18  
19 /*  
20     Ukoliko zelimo da učitamo string koji sadrzi beline  
21     (npr liniju teksta), ne mozemo koristiti funkciju  
22     scanf jer ona ucitava string do prvog blanko znaka.  
23  
24     Zbog toga je neophodno napisati funkciju koja ucitava  
25     string karakter po karakter.  
26  
27     Ova funkcija ne dopusta unosenje vise karaktera od  
28     unapred odredjene granice (argument n).  
29  
30     U standardnoj biblioteci stdio.h postoji definisana  
31     funkcija char *gets(char *s) koja ucitava karaktere  
32     dok se ne pojavi novi red ili EOF. Ova funkcija  
33     dopusta unosenje vise karaktera nego sto string  
34     s sadrzi, sto moze dovesti do neocekivanog ponasanja  
35     programa.  
36  
37     Pored funkcije gets, koja vrsi ucitavanje sa standardnog  
38     ulaza, u standardnoj biblioteci stdio.h postoji  
39     i ugradjena funkcija fgets koja vrsi ucitavanje iz  
40     datoteke. Nju cemo koristiti za nekoliko casova  
41     kada budemo radili datoteke. Prototim funkcije fgets je  
42     ovakav:  
43  
44     char *fgets(char *s, int size, FILE *stream);  
45  
46     Argumenti funkcije fgets su:  
47     s - string u koji vrsimo ucitavanje  
48     size - maksimalna duzina unetog stringa  
49     stream - datoteka iz koje vrsimo ucitavanje  
50  
51     Funkcija fgets, za razliku od funkcije gets, ne dopusta  
52     unos vise karaktera od date vrednosti size. Zbog toga  
53     je ona sigurnija nego funkcija gets. Funkciju fgets  
54     mozemo koristiti i za unos sa standardnog ulaza  
55     ukoliko kao treci argument navedemo stdin.  
56  
57 */  
58 int ucitaj_liniju(char s[], int n)  
59 {  
60     int i=0;  
61     int c;
```

```

63  while((c=getchar())!='\n' && i<n-2 && c!=EOF)
64  {
65      s[i] = c;
66      i++;
67  }

69  /* Ucitavamo najvise n-2 karaktera jer na kraju dodajemo jos
70     dva: '\n' i '\0' */

71  s[i]='\n';
72  s[i+1]='\0';

73  return i;

74  }

75  int main()
76  {
77      char linija[MAX];
78      char najduza_linija[MAX];
79      int max_duzina=0;
80      int duzina;

81      /*
82       Petlja se zavrsava ukoliko je promenljiva duzina
83       jednaka nuli, sto cemo postici zadavanjem linije koja ne sadrzi
84       nijedan karakter osim EOF.
85      */

86      while ((duzina=ucitaj_liniju(linija, MAX))>0)
87      {
88          /*
89           Proveravamo da li je uneta linija duza od trenutnog
90           maksimuma i azuriramo promenljive max_duzina i najduza_linija
91           .
92          */
93          if (max_duzina<duzina)
94          {
95              max_duzina = duzina;
96              strcpy(najduza_linija,linija);
97          }
98      }

99      printf("Najduza linija: %s duzine: %d\n", najduza_linija,
100            max_duzina);

101      return 0;
102  }

```

## Rešenje 3.5.30

Rešenje 3.5.21

Rešenje 3.5.22

Rešenje 3.5.30

Rešenje 3.5.30

Rešenje 3.5.30

Rešenje 3.5.30

Rešenje 3.5.30

Rešenje 3.5.30

Rešenje 3.5.30

Rešenje 3.5.30

## 3.7 Višedimenzioni nizovi

**Zadatak 3.7.1** Napisati program koji učitava i zatim ispisuje vrednosti učitane matrice. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ , 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 greške ispisati odgovarajuću poruku.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti matricu celih brojeva:
1 2 3 4
5 6 7 8
9 10 11 12
Matrica je:
1 2 3 4
5 6 7 8
9 10 11 12
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 3
Uneti matricu celih brojeva:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Matrica je:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
```

## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 500 3
Neispravna dimenzija matrice.

```

[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 greške ispisati odovarajuću poruku. UPUTSTVO: *Euklidska norma matrice je kvadratni koren sume kvadrata svih elemenata matrice.*

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti matricu celih brojeva:
1 2 3 4
5 6 7 8
9 10 11 12
Euklidska norma je: 25.495

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 3
Uneti matricu celih brojeva:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Euklidska norma je: 15.875

```

## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 500 3
Neispravna dimenzija matrice.

```

[Rešenje 3.7.2]

**Zadatak 3.7.3** Napisati funkcije za rad sa celobrojnim matricama:

- `void ucitavanje(int mat[][MAX], int* n, int* m)` kojom se prvo učitava dimezija matrice (brojevi  $m$  i  $n$ ), a potom i vrednosti matrice celih brojeva  $mat$ ,
- `void ispis(int mat[][MAX], int n, int m)` kojom se ispisuju vrednosti matrice  $mat$  dimenzije  $m \times n$ .

Napisati program koji najpre učitava, a zatim i ispisuje vrednosti učitane matrice. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju greške ispisati odovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti matricu celih brojeva:
1 2 3 4
5 6 7 8
9 10 11 12
Matrica je:
1 2 3 4
5 6 7 8
9 10 11 12
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 3
Uneti matricu celih brojeva:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Matrica je:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 500 3
Neispravna dimenzija matrice.
```

[Rešenje 3.7.3]

**Zadatak 3.7.4** Napisati funkciju `void transponovana(int a[][max], int m, int n, int b[][max])` koja određuje matricu  $b$  koja je dobijena transponovanjem matrice  $a$ . Napisati program koji za učitane matricu celih brojeva<sup>1</sup> ispisuje odgovarajuću transponovanu matricu. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti matricu celih brojeva:
1 2 3 4
5 6 7 8
9 10 11 12
Transponovana matrica je:
1 5 9
2 6 10
3 7 11
4 8 12
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 3
Uneti matricu celih brojeva:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Transponovana matrica je:
1 5 7 1 0
1 0 8 2 1
2 2 9 4 1
```

---

<sup>1</sup>Pod pojmom *učitati matricu* ili *za datu matricu* uvek se podrazumeva da se prvo unose dimenzije matrice, a potom i sama matrica.



## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 500 3
Neispravna dimenzija matrice.

```

[Rešenje 3.7.4]

**Zadatak 3.7.5** Napisati funkciju `void razmeni(int mat[][max], int m, int n, int k, int t)` u kojoj se razmenjuju elementi  $k$ -te i  $t$ -te vrste matrice  $mat$  dimezije  $m \times n$ . Napisati program koji za učitano 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 greške ispisati odovarajuću poruku.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti matricu celih brojeva:
1 2 3 4
5 6 7 8
9 10 11 12
Uneti indekse vrsta: 0 2
9 10 11 12
5 6 7 8
1 2 3 4

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 3
Uneti matricu celih brojeva:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Uneti indekse vrsta: 1 3
1 1 2
1 2 4
7 8 9
5 0 2
0 1 1

```

## Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 3
Uneti matricu celih brojeva:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Uneti indekse vrsta: -1 50
Neispravni indeksi vrsta.

```

[Rešenje 3.7.5]

**Zadatak 3.7.6** Napisati program koji za učitano 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 greške ispisati odovarajuću poruku. UPUTSTVO: *Broj susednih elemenata matrice zavisi od položaja elementa u matrici. Na slici 3.1 su slovom s obeleženi susedni elementi matrice za elemente a, b i c.*

#### Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 4 5
Uneti elemente matrice:
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
Indeksi elemenata koji su jednaki
zbiru suseda su:
1 1
3 1
3 4

```

#### Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti elemente matrice:
7 10 12 20
-1 -3 1 7
0 -47 2 0
Indeksi elemenata koji su jednaki
zbiru suseda su:
0 3
1 2

```

[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 greške ispisati odovarajuću poruku.

#### Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 4 5
Uneti elemente matrice:
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
Dobijeni niz je:
1.6 3.6 0.6 1.4

```

#### Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti elemente matrice:
7 10 12 20
-1 -3 1 7
0 -47 2 0
Dobijeni niz je:
12.25 1 -11.25

```

[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 broj 1, a nije u relaciji ukoliko se tu nalazi broj 0. Napisati funkcije:

- (a) `int reflektivna(int a[][MAX], int n)` kojom se za relaciju zadatom matricom  $a$  dimenzije  $n \times n$  ispituje da li je reflektivna;
- (b) `int simetricna(int a[][MAX], int n)` kojom se za relaciju zadatom matricom  $a$  dimenzije  $n \times n$  ispituje da li je simetrična;
- (c) `int tranzitivna(int a[][MAX], int n)` kojom se za relaciju zadatom matricom  $a$  ispituje dimenzije  $n \times n$  da li je tranzitivna;
- (d) `int ekvivalencija(int a[][MAX], int n)` kojom se za relaciju koja je zadata 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 može imati za vrednosti elemenata samo brojeve 0 i 1. U slučaju greške ispisati odgovarajuću poruku.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
1 0 0 0
0 1 1 0
0 0 1 0
0 0 0 0
Relacija nije reflektivna.
Relacija nije simetricna.
Relacija jeste tranzitivna.
Relacija nije ekvivalencija.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
1 1 0 0
1 1 1 0
0 0 1 0
0 0 0 1
Relacija jeste reflektivna.
Relacija jeste simetricna.
Relacija nije tranzitivna.
Relacija nije ekvivalencija.
```

[Rešenje 3.7.8]

**Zadatak 3.7.9** Data je kvadratna matrica dimenzije  $n \times n$ .

- (a) Napisati funkciju `float trag(float a[][MAX], int n)` koja računa trag matrice, odnosno zbir elemenata na glavnoj dijagonali matrice.
- (b) Napisati funkciju `float suma_sporodna(float a[][MAX], int n)` koja računa zbir elemenata na sporednoj dijagonali matrice.

### 3 Predstavljanje podataka

---

- (c) Napisati funkciju `float suma_iznad(float a[][MAX], int n)` koja određuje sumu elemenata iznad glavne dijagonale.
- (d) Napisati funkciju `float suma_ispod(float a[][MAX], int n)` koja određuje sumu elemenata ispod sporedne dijagonale matrice.

Napisati program koji za učitano 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 greške ispisati odovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
6 12.08 -1 20.5
8 90 -33.4 19.02
7.02 5 -20 14.5
8.8 -1 3 -22.8
Trag je 53.200.
Suma na sporednoj dijagonali je 0.900.
Suma iznad glavne dijagonale je 31.700.
Suma ispod sporedne dijagonale je -1.820.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 5
Uneti elemente matrice:
1 2 3 5 5
7 8 9 0 1
6 4 3 2 2
8 9 1 3 4
0 3 1 8 6
Trag je 21.000.
Suma na sporednoj dijagonali je 17.000.
Suma iznad glavne dijagonale je 33.000.
Suma ispod sporedne dijagonale je 24.000.
```

[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 učitano kvadratnu matricu proverava da li je ona donje trougaona i ispisuje odgovarajuću poruku. Pretpostaviti da je maksimalna dimenzija matrice  $100 \times 100$ . U slučaju greške ispisati odovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 5
Uneti elemente matrice:
-1 0 0 0 0
2 10 0 0 0
0 1 5 0 0
7 8 20 14 0
-23 8 5 1 11
Matrica jeste donje trougaona.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
2 -2 1
1 2 2
2 1 -2
Matrica nije donje trougaona.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 200
Neispravna dimenzija matrice.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
2 0 0 0
7 80 0 0
-9 4 4 0
14 23 -8 1
Matrica jeste donje trougaona.
```

[Rešenje 3.7.10]

**Zadatak 3.7.11** Napisati program koji za učitane kvadratnu matricu ispisuje redni broj kolone koja ima najveći zbir elemenata. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . U slučaju greške ispisati odovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
1 2 3
7 3 4
5 3 1
Indeks kolone je: 0
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
7 8 9 10
7 6 11 4
3 1 2 -2
8 3 9 9
Indeks kolone je: 2
```

[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 greške ispisati odovarajuću poruku.

### 3 Predstavljanje podataka

---

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
2 3.2 4
7 8.8 1
2.3 1 1
Razlika je: 2.20
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 5
Uneti elemente matrice:
2.3 1 12 8 -20
4 -8.2 7 14.5 19
1 -2.5 9 11 33
3 4.3 -5.7 2 8
9 56 1.08 7 5.5 19.01
Razlika je:-30.38
```

[Rešenje 3.7.12]

**Zadatak 3.7.13** Napisati program koji za učitane 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 greške ispisati odovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti matricu celih brojeva:
1 2 3 4
5 6 7 8
9 10 11 12
Uneti dva cela broja: 3 3
Sume podmatrica su: 54 63
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti matricu celih brojeva:
1 2 3 4
5 6 7 8
9 10 11 12
Uneti dva cela broja: 2 3
Sume podmatrica su: 24 30 48 54
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: -3 200
Neispravna dimenzija matrice.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 3
Uneti matricu celih brojeva:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Uneti dva cela broja: 2 2
Sume podmatrica su: 7 5 20 19 18 23 4 8
```

[Rešenje 3.7.13]

**Zadatak 3.7.14** Napisati program koji za učitane 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 greške ispisati odovarajuću poruku.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 3
Unesi elemente matrice:
1 2 3
4 5 6
7 8 9
Elementi su sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi su sortirani po dijagonalama.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 2
Unesi elemente matrice:
6 9
4 10
Elementi nisu sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi nisu sortirani po dijagonalama.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 4
Unesi elemente matrice:
5 5 7 9
6 10 11 13
8 12 14 15
13 15 16 20
Elementi su sortirani po kolonama.
Elementi nisu sortirani po vrstama.
Elementi su sortirani po dijagonalama.
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesi dimenziju matrice: 1
Unesi elemente matrice:
5
Elementi su sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi su sortirani po dijagonalama.
```

[Rešenje 3.7.14]

**Zadatak 3.7.15** Napisati program koji za učitane kvadratnu matricu ispisuje da li su zbrovi elemenata njenih kolona uređeni u strogo rastućem poretku. Pretpostaviti da je maksimalna dimenzija matrice koja se unosi  $10 \times 10$ . U slučaju greške ispisati odovarajuću poruku.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
Sume nisu uredjenje strogo rastuce.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
1 2 3
4 5 6
7 8 9
Sume jesu uredjenje strogo rastuce.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
2 -2 1
1 2 2
2 1 -2
Sume nisu uredjenje strogo rastuce.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 5
Uneti elemente matrice:
-1 0 3 0 20
0 0 0 10 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
Sume jesu uredjenje strogo rastuce.
```

[Rešenje 3.7.15]

**Zadatak 3.7.16** Matrica je *ortonormirana* ako je skalarni proizvod svakog para različitih vrsta jednak 0, a skalarni proizvod vrste sa samom sobom 1. Napisati program koji za unetu kvadratnu matricu proverava da li je ortonormirana. U slučaju greške ispisati odgovarajuću poruku. NAPOMENA: *Skalarni proizvod vektora*  $a = (a_1, a_2, \dots, a_n)$  *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:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
Matrica jeste ortonormirana.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
1 2 3
4 5 6
7 8 9
Matrica nije ortonormirana.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
2 -2 1
1 2 2
2 1 -2
Matrica nije ortonormirana.
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 5
Uneti elemente matrice:
-1 0 0 0 0
0 0 0 1 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
Matrica jeste ortonormirana.
```

[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 kvadratna matrica magični kvadrat i ispisuje odgovarajuću poruku na



standardni izlaz. Pretpostaviti da je maksimalna dimenzija matrice je  $50 \times 50$ . U slučaju greške ispisati odgovarajuću poruku.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 4
Uneti elemente matrice:
1 5 3 1
2 1 2 5
3 2 2 3
4 2 3 1
Matrica jeste magicni kvadrat.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
1 2 3
4 5 6
-1 3 3
Matrica nije magicni kvadrat.
```

[Rešenje 3.7.17]

**\* Zadatak 3.7.18** Ispisati elemente učitane kvadratne matrice celih brojeva u grupama koje su paralelne sa sporednom dijagonalom matrice počevši od gornjeg levog ugla matrice. Pretpostaviti da je maksimalna dimenzija matrice  $100 \times 100$ . U slučaju greške ispisati odgovarajuću poruku.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3
Uneti elemente matrice:
1 2 3
4 5 6
7 8 9
Ispis je:
1
2 4
3 5 7
6 8
9
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 5
Uneti elemente matrice:
7 -8 1 2 3
90 11 0 5 4
12 -9 14 23 8
80 6 88 17 62
-22 10 44 57 -200
Ispis je:
7
-8 90
1 11 12
2 0 -9 80
3 5 14 6 -22
4 23 88 10
8 17 44
62 57
-200
```

[Rešenje 3.7.18]

**\* Zadatak 3.7.19** Napisati funkciju `void mnozenje(int a[][max], int m, int n, int b[][max], int k, int t, int c[][max])` koja računa matricu  $c$  kao proizvod matrica  $a$  i  $b$ . Dimenzija matrice  $a$  je  $n \times m$ , a dimenzija matrice  $b$  je  $k \times t$ . Napisati program koji ispisuje proizvod učitanih matrica. Ukoliko množenje matrica nije moguće ili je došlo do greške prilikom unosa podataka ispisati odgovarajuću poruku. Maksimalna dimenzija obe matrice je  $50 \times 50$ .

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Uneti dimenzije matrice: 4 2
Uneti elemente matrice:
11 5
6 7
8 9
0 -3
Rezultat mnozenja je:
87 64
2 24
145 83
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 2
Uneti elemente matrice:
1 7
9 0
-10 2
92 3
14 -8
Uneti dimenzije matrice: 2 4
Uneti elemente matrice:
7 8 9 10
-11 2 34 78
Rezultat mnozenja je:
-70 22 247 556
63 72 81 90
-92 -76 -22 56
611 742 930 1154
186 96 -146 -484
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Uneti dimenzije matrice: 5 2
Uneti elemente matrice:
11 5
6 7
8 9
0 -3
4 4
Mnozenje matrica nije moguće.
```

[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 greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 2 3
1 2 3
0 5 6
Sedlo: 0 0 1
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3 3
10 3 20
15 5 100
30 -1 200
Sedlo: 1 1 5
```

[Rešenje 3.7.20]

\* **Zadatak 3.7.21** Napisati program koji ispisuje elemente matrice celih brojeva u spiralnomo 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 greške ispisati odgovarajuću poruku.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 3 3
Uneti elemente matrice:
1 2 3
4 5 6
7 8 9
Ispis je:
1 2 3 6 9 8 7
4 5
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenziju matrice: 5 7
Uneti elemente matrice:
7 -8 1 2 3 -54 87
90 11 0 5 4 9 18
12 -9 14 23 8 -22 74
80 6 88 17 62 38 41
-22 10 44 57 -200 39 55
Ispis je:
7 -8 1 2 3 -54 87 18 74 41 55
39 -200 57 44 10 -22 80 12 90
11 0 5 4 9 -22 38 62 17 88 6
-9 14 23 8
```

[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  $50 \times 50$ .

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Uneti dimenzije matrice: 2 2
Uneti elemente matrice:
2 3
4 10
Druga matrica je sadržana u prvoj matrici.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 3 4
Uneti elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Uneti dimenzije matrice: 2 2
Uneti elemente matrice:
2 8
6 4
Druga matrica nije sadržana u prvoj matrici.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti dimenzije matrice: 5 5
Uneti elemente matrice:
7 -8 1 2 3
90 11 0 5 4
12 -9 14 23 8
80 6 88 17 62
-22 10 44 57 -200
Uneti dimenzije matrice: 3 4
Uneti elemente matrice:
90 11 0 5
12 -9 14 23
80 6 88 17
Druga matrica je sadrzana u prvoj matrici.
```

[Rešenje 3.7.22]

## 3.8 Rešenja

### Rešenje 3.7.1

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX 50
5
6 int main()
7 {
8     int mat[MAX][MAX];
9     int m, n;
10
11     int i, j;
12
13     printf("Uneti dimenzije matrice: ");
14     scanf("%d%d", &m, &n);
15
16     if (n <= 0 || n > MAX || m <= 0 || m > MAX)
17     {
18         printf("Neispravna dimenzija matrice\n");
19         exit(EXIT_FAILURE);
20     }
21
22     printf("Uneti matricu celih brojeva\n");
23
24     for(i=0; i<m; i++)
25         for(j=0; j<n; j++)
```

```

scanf("%d", &mat[i][j]);
27
/* Ispis elemenata matrice. */
29 for(i=0; i<m; i++)
{
31     for(j=0; j<n; j++)
        printf("%d ", mat[i][j]);
33     printf("\n");
}
35
37 return 0;
}

```

### Rešenje 3.7.2

```

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

5  #define MAX 50

7  int main()
{
9     int mat[MAX][MAX];
    int m, n;
11    int suma = 0;

13    int i, j;

15    printf("Uneti dimenzije matrice: ");
    scanf("%d%d", &m, &n);

17    if (n <= 0 || n > MAX || m <= 0 || m > MAX)
19    {
        printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
    }

23    printf("Uneti matricu celih brojeva\n");

25    for(i=0; i<m; i++)
27        for(j=0; j<n; j++)
            scanf("%d", &mat[i][j]);

29

31    for(i=0; i<m; i++)
33        for(j=0; j<n; j++)
            suma += mat[i][j] * mat[i][j];

35    printf("Euklidska norma je %.31f.\n", sqrt(suma));

```

```
37     return 0;
39 }
```

#### Rešenje 3.7.3

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAX 50
5
   void ucitavanje(int mat[][MAX], int* m, int* n)
7 {
   int i, j;
9
   printf("Uneti dimenzije matrice: ");
11  scanf("%d%d", m, n);
13
   if (*m <= 0 || *n > MAX || *m <= 0 || *m > MAX)
   {
15     printf("Neispravna dimenzija matrice\n");
       exit(EXIT_FAILURE);
17 }
19
   printf("Uneti matricu celih brojeva\n");
21
   for(i=0; i<*m; i++)
       for(j=0; j<*n; j++)
23         scanf("%d", &mat[i][j]);
   }
25
   void ispis(int mat[][MAX], int m, int n)
27 {
   int i, j;
29
   for(i=0; i<m; i++)
   {
31     for(j=0; j<n; j++)
33         printf("%d ", mat[i][j]);
       printf("\n");
35     }
   }
37
   int main()
39 {
   int mat[MAX][MAX];
41     int m, n;
43
   ucitavanje(mat, &m, &n);
   ispis(mat, m, n);
45 }
```

```
    return 0;
47 }
```

### Rešenje 3.7.4

```
1  #include <stdio.h>
   #include <stdlib.h>
3
   #define MAX 50
5
   void učitavanje(int mat[][MAX], int* m, int* n)
7 {
   int i, j;
9
   printf("Uneti dimenzije matrice: ");
11  scanf("%d%d", m, n);
13
   if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
   {
15     printf("Neispravna dimenzija matrice\n");
       exit(EXIT_FAILURE);
17 }
19
   printf("Uneti matricu celih brojeva\n");
21
   for(i=0; i<*m; i++)
       for(j=0; j<*n; j++)
23         scanf("%d", &mat[i][j]);
   }
25
   void ispis(int mat[][MAX], int m, int n)
27 {
   int i, j;
29
   for(i=0; i<m; i++)
   {
31     for(j=0; j<n; j++)
33       printf("%d ", mat[i][j]);
       printf("\n");
35   }
   }
37
   void transponovana(int a[][MAX], int m, int n, int b[][MAX])
39 {
   int i, j;
41
   for(i=0; i<m; i++)
43     for(j=0; j<n; j++)
       b[j][i] = a[i][j];
45 }
```

### 3 Predstavljanje podataka

---

```
47 int main()
48 {
49     int mat[MAX][MAX], t[MAX][MAX];
50     int m, n;
51
52     učitavanje(mat, &m, &n);
53     transponovana(mat, m, n, t);
54     ispis(t, n, m);
55
56     return 0;
57 }
```

#### Rešenje 3.7.5

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

#define MAX 50

void učitavanje(int mat[][MAX], int* m, int* n)
{
    int i, j;

    printf("Uneti dimenzije matrice: ");
    scanf("%d%d", m, n);

    if (*m <= 0 || *n > MAX || *m <= 0 || *m > MAX)
    {
        printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
    }

    printf("Uneti matricu celih brojeva\n");

    for(i=0; i<*m; i++)
        for(j=0; j<*n; j++)
            scanf("%d", &mat[i][j]);
}

void ispis(int mat[][MAX], int m, int n)
{
    int i, j;

    for(i=0; i<m; i++)
    {
        for(j=0; j<n; j++)
            printf("%d ", mat[i][j]);
        printf("\n");
    }
}
```



```

38 void razmeni(int mat[][MAX], int m, int n, int k, int t)
{
40     int j, pom;

42     for(j = 0; j < n; j++)
    {
44         pom = mat[k][j];
         mat[k][j] = mat[t][j];
46         mat[t][j] = pom;
    }
48 }

50 int main()
{
52     int mat[MAX][MAX];
    int m, n;
54     int k, t;

56     učitavanje(mat, &m, &n);

58     printf("Uneti indekse vrsta: ");
    scanf("%d%d", &k, &t);

60

62     if (k < 0 || k >= m || t < 0 || t >= m)
    {
        printf("Neispravni indeksi vrsta.\n");
64         return -1;
    }

66

68     razmeni(mat, m, n, k, t);

70     ispis(mat, m, n);

72     return 0;
}

```

### Rešenje 3.7.6

```

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

4 #define MAX 50

6 void učitavanje(int mat[][MAX], int* m, int* n)
{
8     int i, j;

10     printf("Uneti dimenzije matrice: ");
    scanf("%d%d", m, n);

12

    if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)

```

### 3 Predstavljanje podataka

---

```
14 {
15     printf("Neispravna dimenzija matrice\n");
16     exit(EXIT_FAILURE);
17 }
18
19 printf("Uneti matricu celih brojeva\n");
20
21 for(i=0; i<*m; i++)
22     for(j=0; j<*n; j++)
23         scanf("%d", &mat[i][j]);
24 }
25
26 int main()
27 {
28     int mat[MAX][MAX];
29     int m, n, i, j, suma;
30     int k, t;
31
32     učitavanje(mat, &m, &n);
33
34     printf("Indeksi elemenata koji su jednaki zbiru suseda su:\n");
35     for(i=0; i<m; i++)
36         for(j=0; j<n; j++)
37         {
38             suma = 0;
39
40             for(k=-1; k<=1; k++)
41                 for(t=-1; t<=1; t++)
42                     if (i+k >= 0 && i+k < n && j+t >= 0 && j+t < n)
43                         suma += mat[i+k][j+t];
44
45             if (suma - mat[i][j] == mat[i][j])
46                 printf("%d %d\n", i, j);
47         }
48
49     return 0;
50 }
```

#### Rešenje 3.7.7

```
#include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX 50
5
6 void učitavanje(int mat[][MAX], int* m, int* n)
7 {
8     int i, j;
9
10    printf("Uneti dimenzije matrice: ");
```

```

12     scanf("%d%d", m, n);
14     if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
15     {
16         printf("Neispravna dimenzija matrice\n");
17         exit(EXIT_FAILURE);
18     }
19
20     printf("Uneti matricu celih brojeva\n");
21
22     for(i=0; i<*m; i++)
23         for(j=0; j<*n; j++)
24             scanf("%d", &mat[i][j]);
25
26 void kreiraj_niz(int mat[][MAX], int m, int n, double b[])
27 {
28     int i, j, suma;
29
30     for(i=0; i<m; i++)
31     {
32         suma = 0;
33         for(j=0; j<n; j++)
34             suma += mat[i][j];
35
36         b[i] = (double)suma/n;
37     }
38 }
39
40 int main()
41 {
42     int mat[MAX][MAX];
43     double b[MAX];
44     int m, n, i;
45
46     učitavanje(mat, &m, &n);
47
48     kreiraj_niz(mat, m, n, b);
49
50     printf("Dobijeni niz je:\n");
51     for(i=0; i<m; i++)
52         printf("%g ", b[i]);
53     printf("\n");
54
55     return 0;
56 }

```

### Rešenje 3.7.8

```

1 #include <stdio.h>
2 #include <stdlib.h>

```

```
3  #define MAX 50
5
6  void ucitavanje(int mat[][MAX], int* n)
7  {
8      int i, j;
9
10     printf("Uneti dimenzije matrice: ");
11     scanf("%d", n);
12
13     if (*n <= 0 || *n > MAX)
14     {
15         printf("Neispravna dimenzija matrice\n");
16         exit(EXIT_FAILURE);
17     }
18
19     printf("Uneti matricu celih brojeva\n");
20
21     for(i=0; i<*n; i++)
22         for(j=0; j<*n; j++)
23             scanf("%d", &mat[i][j]);
24 }
25
26 int reflektivna(int a[][MAX], int n)
27 {
28     int i;
29
30     for(i=0; i<n; i++)
31         if (a[i][i] != 1)
32             return 0;
33
34     return 1;
35 }
36
37 int simetricna(int a[][MAX], int n)
38 {
39     int i, j;
40
41     for(i=0; i<n; i++)
42         for(j=0; j<n; j++)
43             if (a[i][j] != a[j][i])
44                 return 0;
45
46     return 1;
47 }
48
49 int tranzitivna(int a[][MAX], int n)
50 {
51     int i, j, k;
52
53     for(i=0; i<n; i++)
54         for(j=0; j<n; j++)
```

```

55     for(k=0; k<n; k++)
56         if (a[i][j] == 1 && a[j][k] == 1 && a[i][k] == 0)
57             return 0;
58
59     return 1;
60 }
61
62 int ekvivalencija(int a[][MAX], int n)
63 {
64     if (refleksivna(a, n) && simetricna(a, n) && tranzitivna(a, n))
65         return 1;
66
67     return 0;
68 }
69
70 int main()
71 {
72     int a[MAX][MAX];
73     int n;
74
75     učitavanje(a, &n);
76
77     if (refleksivna(a, n))
78         printf("Relacija jeste refleksivna.\n");
79     else
80         printf("Relacija nije refleksivna.\n");
81
82     if (simetricna(a, n))
83         printf("Relacija jeste simetricna.\n");
84     else
85         printf("Relacija nije simetricna.\n");
86
87     if (tranzitivna(a, n))
88         printf("Relacija jeste tranzitivna.\n");
89     else
90         printf("Relacija nije tranzitivna.\n");
91
92     if (ekvivalencija(a, n))
93         printf("Relacija jeste ekvivalencija.\n");
94     else
95         printf("Relacija nije ekvivalencija.\n");
96
97     return 0;
98 }

```

### Rešenje 3.7.9

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX 50

```

```
6 void učitavanje(float mat[][MAX], int* n)
{
8     int i, j;

10     printf("Uneti dimenzije matrice: ");
    scanf("%d", n);

12     if (*n <= 0 || *n > MAX)
    {
14         printf("Neispravna dimenzija matrice\n");
16         exit(EXIT_FAILURE);
    }

18     printf("Uneti matricu celih brojeva\n");

20     for(i=0; i<*n; i++)
22         for(j=0; j<*n; j++)
            scanf("%f", &mat[i][j]);
24 }

26 float trag(float a[][MAX], int n)
{
28     float suma = 0;
    int i;

30     for(i=0; i<n; i++)
32         suma += a[i][i];

34     return suma;
}

36 float suma_sporodna(float a[][MAX], int n)
38 {
    float suma = 0;
    int i;

40     for(i=0; i<n; i++)
42         suma += a[i][n-i-1];

44     return suma;
46 }

48 float suma_iznad(float a[][MAX], int n)
{
50     float suma = 0;
    int i, j;

52     for(i=0; i<n; i++)
54         for(j=i+1; j<n; j++)
            suma += a[i][j];
56 }
```

```

    return suma;
58 }

60 float suma_ispod(float a[][MAX], int n)
{
62     float suma = 0;
    int i, j;

64     for(i=0; i<n; i++)
66         for(j=n-i-1; j>i; j--)
            suma += a[i][j];

68     return suma;
70 }

72 int main()
{
74     float a[MAX][MAX];
    int n;

76     učitavanje(a, &n);

78     printf("Trag je %.3f.\n", trag(a, n));
80     printf("Suma na sporednoj dijagonali je %.3f.\n", suma_sporedna(a,
        n));
    printf("Suma iznad glavne dijagonale je %.3f.\n", suma_iznad(a, n))
    ;
82     printf("Suma ispod sporedne dijagonale je %.3f.\n", suma_ispod(a, n
        ));

84     return 0;
}

```

### Rešenje 3.7.10

```

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

4  #define MAX 50

6  void učitavanje(int mat[][MAX], int* n)
{
8     int i, j;

10     printf("Uneti dimenzije matrice: ");
    scanf("%d", n);

12     if (*n <= 0 || *n > MAX)
14     {
        printf("Neispravna dimenzija matrice\n");
16         exit(EXIT_FAILURE);
    }
}

```

```
    }
18
    printf("Uneti matricu celih brojeva\n");
20
    for(i=0; i<*n; i++)
22        for(j=0; j<*n; j++)
            scanf("%d", &mat[i][j]);
24 }

26 int donje_trougaona(int a[][MAX], int n)
{
28     int i, j;

30     for(i=0; i<n; i++)
        for(j=i+1; j<n; j++)
32             if (a[i][j] != 0)
                return 0;
34
    return 1;
36 }

38 int main()
{
40     int a[MAX][MAX];
    int n;
42
    ucitavanje(a, &n);
44
    if (donje_trougaona(a, n))
46         printf("Matrica jeste donje trougaona.\n");
    else
48         printf("Matrica nije donje trougaona.\n");

50     return 0;
}
```

#### Rešenje 3.7.11

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

4 #define MAX 50

6 void ucitavanje(int mat[][MAX], int* n)
{
8     int i, j;

10     printf("Uneti dimenzije matrice: ");
    scanf("%d", n);
12

    if (*n <= 0 || *n > MAX)
```



```

14 {
15     printf("Neispravna dimenzija matrice\n");
16     exit(EXIT_FAILURE);
17 }
18
19 printf("Uneti matricu celih brojeva\n");
20
21 for(i=0; i<*n; i++)
22     for(j=0; j<*n; j++)
23         scanf("%d", &mat[i][j]);
24 }
25
26 int main()
27 {
28     int a[MAX][MAX];
29     int n, i, j;
30     int max_zbir, trenutni_zbir = 0, indeks_kolone = 0;
31
32     učitavanje(a, &n);
33
34     for(i=0; i<n; i++)
35         trenutni_zbir += a[i][0];
36
37     max_zbir = trenutni_zbir;
38
39     for(j=1; j<n; j++)
40     {
41
42         trenutni_zbir = 0;
43         for(i=0; i<n; i++)
44             trenutni_zbir += a[i][j];
45
46         if (trenutni_zbir > max_zbir)
47         {
48             max_zbir = trenutni_zbir;
49             indeks_kolone = j;
50         }
51     }
52
53     printf("Indeks kolone je: %d\n", indeks_kolone);
54
55     return 0;
56 }

```

### Rešenje 3.7.12

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX 50
5

```

```
void učitavanje(float mat[][MAX], int* n)
7 {
    int i, j;
9
    printf("Uneti dimenzije matrice: ");
11    scanf("%d", n);

13    if (*n <= 0 || *n > MAX)
    {
15        printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
17    }

19    printf("Uneti matricu celih brojeva\n");

21    for(i=0; i<*n; i++)
        for(j=0; j<*n; j++)
23        scanf("%f", &mat[i][j]);
    }

25
27 int main()
{
    float a[MAX][MAX];
29    int n, i, j;
    float gornji_trougao = 0, donji_trougao = 0;
31

    učitavanje(a, &n);
33

    for(i=0; i<n/2; i++)
35        for(j=i+1; j<n-i-1; j++)
            gornji_trougao += a[i][j];
37

    for(i=n/2; i<n; i++)
39        for(j=n-i; j<i; j++)
            donji_trougao += a[i][j];
41

    printf("%f %f\n", gornji_trougao, donji_trougao);
43

    printf("Razlika je: %.2f\n", gornji_trougao - donji_trougao);
45

    return 0;
47 }
```

#### Rešenje 3.7.13

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

4 #define MAX 50

6 void učitavanje(int mat[][MAX], int* m, int* n)
```

```

8   int i, j;

10  printf("Uneti dimenzije matrice: ");
   scanf("%d%d", m, n);

12  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
14  {
       printf("Neispravna dimenzija matrice\n");
       exit(EXIT_FAILURE);
   }

18  printf("Uneti matricu celih brojeva\n");

20  for(i=0; i<*m; i++)
22      for(j=0; j<*n; j++)
           scanf("%d", &mat[i][j]);
24 }

26 int main()
   {
28     int a[MAX][MAX];
       int n, i, j, m, x, y, p, k;
30     int suma;

32     učitavanje(a, &m, &n);

34     printf("Uneti dva cela broja: ");
       scanf("%d%d", &p, &k);

36     printf("Sume podmatrica su: ");
38     for(i=0; i<= m-p; i++)
       for(j=0; j<= n-k; j++)
40     {
           suma = 0;
42         for(x=0; x<p; x++)
           for(y=0; y<k; y++)
44             suma += a[i+x][j+y];

46         printf("%d ", suma);
       }

48     printf("\n");

50     return 0;
52 }

```

### Rešenje 3.7.14

```

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

```

```
3
4 #define MAX 50
5
6 void ucitavanje(int mat[][MAX], int* n)
7 {
8     int i, j;
9
10    printf("Uneti dimenzije matrice: ");
11    scanf("%d", n);
12
13    if (*n <= 0 || *n > MAX)
14    {
15        printf("Neispravna dimenzija matrice\n");
16        exit(EXIT_FAILURE);
17    }
18
19    printf("Uneti matricu celih brojeva\n");
20
21    for(i=0; i<*n; i++)
22        for(j=0; j<*n; j++)
23            scanf("%d", &mat[i][j]);
24 }
25
26 int sortirana_kolona(int mat[][MAX], int n, int j)
27 {
28     int i;
29
30     for(i=0; i<n-1; i++)
31         if (mat[i][j] >= mat[i+1][j])
32             return 0;
33
34     return 1;
35 }
36
37 int sortirani_po_kolonama(int mat[][MAX], int n)
38 {
39     int j;
40
41     for(j=0; j<n; j++)
42         if (!sortirana_kolona(mat, n, j))
43             return 0;
44
45     return 1;
46 }
47
48 int sortirana_vrsta(int mat[][MAX], int n, int i)
49 {
50     int j;
51
52     for(j=0; j<n-1; j++)
53         if (mat[i][j] >= mat[i][j+1])
```

```
55     return 1;
56 }
57
58 int sortirani_po_vrstama(int mat[][MAX], int n)
59 {
60     int i;
61
62     for(i=0; i<n; i++)
63         if (!sortirana_vrstama(mat, n, i))
64             return 0;
65
66     return 1;
67 }
68
69 int sortirana_glavna(int mat[][MAX], int n)
70 {
71     int i;
72
73     for(i=0; i<n-1; i++)
74         if (mat[i][i] >= mat[i+1][i+1])
75             return 0;
76
77     return 1;
78 }
79
80 int sortirana_sporedna(int mat[][MAX], int n)
81 {
82     int i;
83
84     for(i=0; i<n-1; i++)
85         if (mat[i][n-i-1] >= mat[i+1][n-i-2])
86             return 0;
87
88     return 1;
89 }
90
91 int sortirani_po_dijagonalama(int mat[][MAX], int n)
92 {
93     if (!sortirana_glavna(mat, n))
94         return 0;
95
96     if (!sortirana_sporedna(mat, n))
97         return 0;
98
99     return 1;
100 }
101
102 int main()
103 {
104     int mat[MAX][MAX];
105     int n;
```

```
107     učitavanje(mat, &n);
109
110     if (sortirani_po_kolonama(mat, n))
111         printf("Elementi su sortirani po kolonama.\n");
112     else
113         printf("Elementi nisu sortirani po kolonama.\n");
114
115     if (sortirani_po_vrstama(mat, n))
116         printf("Elementi su sortirani po vrstama.\n");
117     else
118         printf("Elementi nisu sortirani po vrstama.\n");
119
120     if (sortirani_po_dijagonalama(mat, n))
121         printf("Elementi su sortirani po dijagonalama.\n");
122     else
123         printf("Elementi nisu sortirani po dijagonalama.\n");
124
125     return 0;
126 }
```

#### Rešenje 3.7.15

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX 10
5
6  void učitavanje(int mat[][MAX], int* n)
7  {
8      int i, j;
9
10     printf("Uneti dimenzije matrice: ");
11     scanf("%d", n);
12
13     if (*n <= 0 || *n > MAX)
14     {
15         printf("Neispravna dimenzija matrice\n");
16         exit(EXIT_FAILURE);
17     }
18
19     printf("Uneti matricu celih brojeva\n");
20
21     for(i=0; i<*n; i++)
22         for(j=0; j<*n; j++)
23             scanf("%d", &mat[i][j]);
24 }
25
26 int suma_kolone(int mat[][MAX], int n, int j)
27 {
28     int suma = 0, i;
```

```

29     for(i=0; i<n; i++)
31         suma += mat[i][j];

33     return suma;
34 }

35 int uredjene_sume(int mat[][MAX], int n)
36 {
37     int suma1, suma2;
38     int j;

41     suma1 = suma_kolone(mat, n, 0);

43     for(j=1; j<n; j++)
44     {
45         suma2 = suma_kolone(mat, n, j);

47         if (suma1 >= suma2)
48             return 0;

49         suma1 = suma2;
51     }

53     return 1;
54 }

55 int main()
56 {
57     int mat[MAX][MAX];
58     int n;

61     učitavanje(mat, &n);

63     if (uredjene_sume(mat, n))
64         printf("Sume jesu uredjenje strogo rastuce.\n");
65     else
66         printf("Sume nisu uredjenje strogo rastuce.\n");
67

69     return 0;
70 }

```

### Rešenje 3.7.16

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX 200
5
6  void učitavanje(int mat[][MAX], int* n)

```

```
7 {
    int i, j;
9
    printf("Uneti dimenzije matrice: ");
11    scanf("%d", &n);
13
    if (*n <= 0 || *n > MAX)
    {
15        printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
17    }
19    printf("Uneti matricu celih brojeva\n");
21
    for(i=0; i<*n; i++)
        for(j=0; j<*n; j++)
23            scanf("%d", &mat[i][j]);
    }
25
    int skalarni_proizvod(int mat[][MAX], int n, int i, int j)
27 {
    int suma = 0, k;
29
    for(k=0; k<n; k++)
31        suma += mat[i][k] * mat[j][k];
33
    return suma;
    }
35
    int ortonormirana(int mat[][MAX], int n)
37 {
    int i, j;
39
    for(i=0; i<n; i++)
41        for(j=0; j<n; j++)
            if (i==j && skalarni_proizvod(mat, n, i, i) != 1)
43                return 0;
            else if (i != j && skalarni_proizvod(mat, n, i, j) != 0)
45                return 0;
47
    return 1;
    }
49
    int main()
51 {
    int mat[MAX][MAX];
53    int n;
55
    ucitavanje(mat, &n);
57
    if (ortonormirana(mat, n))
        printf("Matrica jeste ortonormirana.\n");
```



```
59     else
60         printf("Matrica nije ortonormirana.\n");
61
62     return 0;
63 }
```

### Rešenje 3.7.17

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX 50
5
6  void ucitavanje(int mat[][MAX], int* n)
7  {
8      int i, j;
9
10     printf("Uneti dimenzije matrice: ");
11     scanf("%d", n);
12
13     if (*n <= 0 || *n > MAX)
14     {
15         printf("Neispravna dimenzija matrice\n");
16         exit(EXIT_FAILURE);
17     }
18
19     printf("Uneti matricu celih brojeva\n");
20
21     for(i=0; i<*n; i++)
22         for(j=0; j<*n; j++)
23             scanf("%d", &mat[i][j]);
24 }
25
26 int suma_kolone(int mat[][MAX], int n, int j)
27 {
28     int i, suma = 0;
29
30     for(i=0; i<n; i++)
31         suma += mat[i][j];
32
33     return suma;
34 }
35
36 int suma_vrste(int mat[][MAX], int n, int i)
37 {
38     int j, suma = 0;
39
40     for(j=0; j<n; j++)
41         suma += mat[i][j];
42
43     return suma;
```

```

45 }
46
47 int magicni_kvadrat(int mat[][MAX], int n)
48 {
49     int suma = suma_kolone(mat, n, 0);
50     int i, j;
51
52     for(j=1; j<n; j++)
53         if (suma_kolone(mat, n, j) != suma)
54             return 0;
55
56     for(i=0; i<n; i++)
57         if (suma_vrste(mat, n, i) != suma)
58             return 0;
59
60     return 1;
61 }
62
63 int main()
64 {
65     int mat[MAX][MAX];
66     int n;
67
68     učitavanje(mat, &n);
69
70     if (magicni_kvadrat(mat, n))
71         printf("Matrica jeste magicni kvadrat.\n");
72     else
73         printf("Matrica nije magicni kvadrat.\n");
74
75     return 0;
76 }
```

#### Rešenje 3.7.18

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX 100
5
6 void učitavanje(int mat[][MAX], int* n)
7 {
8     int i, j;
9
10    printf("Uneti dimenzije matrice: ");
11    scanf("%d", n);
12
13    if (*n <= 0 || *n > MAX)
14    {
15        printf("Neispravna dimenzija matrice\n");
16        exit(EXIT_FAILURE);
17    }
18 }
```

```
18     }
19
20     printf("Uneti matricu celih brojeva\n");
21
22     for(i=0; i<*n; i++)
23         for(j=0; j<*n; j++)
24             scanf("%d", &mat[i][j]);
25 }
26
27 int main()
28 {
29     int mat[MAX][MAX];
30     int n;
31     int i, j, k;
32
33     učitavanje(mat, &n);
34
35     for(k=0; k<n; k++)
36     {
37         j = k;
38         i = 0;
39
40         while(j >= 0)
41         {
42             printf("%d ", mat[i][j]);
43             i++;
44             j--;
45         }
46
47         printf("\n");
48     }
49
50     for(k=1; k<n; k++)
51     {
52         i = k;
53         j = n-1;
54
55         while(i < n)
56         {
57             printf("%d ", mat[i][j]);
58             i++;
59             j--;
60         }
61
62         printf("\n");
63     }
64
65     return 0;
66 }
```

#### Rešenje 3.7.19

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

4 #define MAX 50

6 void ucitavanje(int mat[][MAX], int* m, int* n)
{
8     int i, j;

10     printf("Uneti dimenzije matrice: ");
    scanf("%d%d", m, n);

12     if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
    {
14         printf("Neispravna dimenzija matrice\n");
16         exit(EXIT_FAILURE);
    }

18     printf("Uneti matricu celih brojeva\n");

20     for(i=0; i<*m; i++)
22         for(j=0; j<*n; j++)
            scanf("%d", &mat[i][j]);
24 }

26 void ispis(int mat[][MAX], int m, int n)
{
28     int i, j;

30     for(i=0; i<m; i++)
    {
32         for(j=0; j<n; j++)
            printf("%d ", mat[i][j]);
34         printf("\n");
    }
36 }

38 void mnozenje(int a[][MAX], int m, int n, int b[][MAX], int k, int t,
    int c[][MAX])
{
40     int i, j, w;

42     for(i=0; i<m; i++)
        for(j=0; j<t; j++)
        {
44             c[i][j] = 0;
46             for(w=0; w<n; w++)
                c[i][j] += a[i][w] * b[w][j];
48         }
}
```

```

50 int main()
51 {
52     int a[MAX][MAX], b[MAX][MAX], c[MAX][MAX];
53     int m, n;
54     int k, t;
55
56     učitavanje(a, &m, &n);
57     učitavanje(b, &k, &t);
58
59     if (n != k)
60     {
61         printf("Množenje matrica nije moguće.\n");
62         return -1;
63     }
64
65     množenje(a, m, n, b, k, t, c);
66
67     printf("Rezultat množenja je:\n");
68     ispis(c, m, t);
69
70     return 0;
71 }

```

### Rešenje 3.7.20

```

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

#define MAX 50

void učitavanje(double mat[][MAX], int* m, int* n)
{
    int i, j;

    printf("Uneti dimenzije matrice: ");
    scanf("%d%d", m, n);

    if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
    {
        printf("Neispravna dimenzija matrice\n");
        exit(EXIT_FAILURE);
    }

    printf("Uneti matricu celih brojeva\n");

    for(i=0; i<*m; i++)
        for(j=0; j<*n; j++)
            scanf("%lf", &mat[i][j]);
}

```

### 3 Predstavljanje podataka

---

```
26 int main()
27 {
28     double mat[MAX][MAX];
29     int m, n, k, i, j;
30
31     int indeks_kolone;
32     double max_kolone, min_vrste;
33
34     ucitavanje(mat, &m, &n);
35
36     for(i=0; i<m; i++)
37     {
38         min_vrste = mat[i][0];
39         indeks_kolone = 0;
40
41         for(j=1; j<n; j++)
42             if (mat[i][j] < min_vrste)
43             {
44                 min_vrste = mat[i][j];
45                 indeks_kolone = j;
46             }
47
48         max_kolone = mat[0][indeks_kolone];
49
50         for(k=1; k<m; k++)
51             if (mat[k][indeks_kolone] > max_kolone)
52                 max_kolone = mat[k][indeks_kolone];
53
54         if (min_vrste == max_kolone)
55             printf("Sedlo: %d %d %g\n", i, indeks_kolone, min_vrste);
56     }
57
58     return 0;
59 }
60 }
```

#### Rešenje 3.7.21

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX 50
5
6 void ucitavanje(int mat[][MAX], int* m, int* n)
7 {
8     int i, j;
9
10    printf("Uneti dimenzije matrice: ");
11    scanf("%d%d", m, n);
```

```
13  if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
14  {
15      printf("Neispravna dimenzija matrice\n");
16      exit(EXIT_FAILURE);
17  }
18
19  printf("Uneti matricu celih brojeva\n");
20
21  for(i=0; i<*m; i++)
22      for(j=0; j<*n; j++)
23          scanf("%d", &mat[i][j]);
24  }
25
26  int main()
27  {
28      int mat[MAX][MAX];
29      int m, n, brojac, i, j;
30
31      int pravac = 1;
32      int gore_i, dole_i, levo_j, desno_j;
33
34      učitavanje(mat, &m, &n);
35
36      gore_i = 1;
37      dole_i = m-1;
38
39      levo_j = 0;
40      desno_j = n-1;
41
42      i = 0;
43      j = 0;
44
45      for(brojac=0; brojac < m*n; brojac++)
46      {
47          printf("%d ", mat[i][j]);
48
49          switch(pravac)
50          {
51              case 1:
52                  if (j == desno_j)
53                  {
54                      pravac = 2;
55                      desno_j--;
56                      i++;
57                  }
58                  else
59                      j++;
60                  break;
61              case 2:
62                  if (i == dole_i)
63                  {
64                      pravac = 3;
```

### 3 Predstavljanje podataka

---

```
65         dole_i--;
66         j--;
67     }
68     else
69         i++;
70     break;
71 case 3:
72     if (j == levo_j)
73     {
74         pravac = 4;
75         levo_j++;
76         i--;
77     }
78     else
79         j--;
80     break;
81 case 4:
82     if (i == gore_i)
83     {
84         pravac = 1;
85         gore_i++;
86         j++;
87     }
88     else
89         i--;
90 }
91 }
92
93 return 0;
94 }
```

#### Rešenje 3.7.22

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

4 #define MAX 50

6 void učitavanje(int mat[][MAX], int* m, int* n)
{
8     int i, j;

10     printf("Uneti dimenzije matrice: ");
11     scanf("%d%d", m, n);

12     if (*n <= 0 || *n > MAX || *m <= 0 || *m > MAX)
13     {
14         printf("Neispravna dimenzija matrice\n");
15         exit(EXIT_FAILURE);
16     }
17 }
18
```



```

printf("Uneti matricu celih brojeva\n");
20
for(i=0; i<=m; i++)
22     for(j=0; j<=n; j++)
        scanf("%d", &mat[i][j]);
24 }

26 int podmatrica(int a[][MAX], int m, int n, int b[][MAX], int k, int t
    )
{
28     int i, j, x, y;
    int jeste_pod;
30
    for(i=0; i<= m-k; i++)
32         for(j=0; j<= n-t; j++)
            {
34                 jeste_pod = 1;
                for(x=0; x<k && jeste_pod; x++)
36                     for(y=0; y<t && jeste_pod; y++)
                        if (a[i+x][j+y] != b[x][y])
38                             jeste_pod = 0;

40                 if (jeste_pod)
                    return 1;
42             }

44     return 0;
}

46
48 int main()
{
    int a[MAX][MAX], b[MAX][MAX];
50     int m, n;
    int k, t;
52
    učitavanje(a, &m, &n);
54     učitavanje(b, &k, &t);

56     if (podmatrica(a, m, n, b, k, t))
        printf("Druga matrica je sadržana u prvoj matrici.\n");
58     else
        printf("Druga matrica nije sadržana u prvoj matrici.\n");
60 }

```

## 3.9 Struktura

**Zadatak 3.9.1** Definirati strukturu kojom se predstavlja kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja. Napisati program koji za učitana dva kompleksna broja ispituje

### 3 Predstavljanje podataka

---

vrednost zbira, razlike, proizvoda i količnika.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite realni i imaginarni deo prvog broja: 1 2
Unesite realni i imaginarni deo drugog broja: -2 3
Zbir: -1.00+5.00*i
Razlika: 3.00-1.00*i
Proizvod: -8.00-1.00*i
Kolicnik: 0.31-0.54*i
```

[Rešenje 3.9.1]

**Zadatak 3.9.2** Definisati strukturu kojom se predstavlja razlomak. Napisati funkcije koje izračunavaju zbir i proizvod dva razlomka. Napisati program koji za uneti broj  $n$  i unetih  $n$  razlomaka ispisuje njihov zbir i proizvod.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesi broj razlomaka: 5
Uneti razlomke:
1 2
7 8
3 4
5 6
2 9
Suma svih razlomaka je 229/72.
Proizvod svih razlomaka je 35/576.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesi broj razlomaka: 10
Uneti razlomke:
4 3
12 25
3 8
1 3
8 9
2 3
5 6
-24 50
7 18
-7 19
Suma svih razlomaka je 6089/1368.
Proizvod svih razlomaka je 1568/577125.
```

[Rešenje 3.9.2]

**Zadatak 3.9.3** Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vočka* sadrži ime vočke (nisku maksimalne dužine 20 karaktera) i količinu vitamina C u miligramima (realan broj). Napisati program koji učitava podatke o vočkama sve do unosa reči KRAJ i ispisuje ime vočke sa najviše vitamina C. Pretpostaviti da broj vočki neće biti veći od 50.

## 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 51
Unesite ime vočke i njenu količinu vitamina C: kivi 92.7
Unesite ime vočke i njenu količinu vitamina C: banana 8.7
Unesite ime vočke i njenu količinu vitamina C: pomorandža 53.2
Unesite ime vočke i njenu količinu vitamina C: KRAJ
Voce sa najvise C vitamina je: kivi

```

[Rešenje 3.9.3]

**Zadatak 3.9.4** Definirati strukturu **Grad** u kojoj se nalazi ime grada (niska dužine 20 karaktera) i prosečna temperatura u toku decembra (realan broj). Napisati program koji učitava imena  $n$  ( $0 < n < 50$ ) gradova i njihove prosečne temperature, a zatim ispisuje one gradove koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni.

## Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Unesite grad i temperaturu: Beograd 7
Unesite grad i temperaturu: Uzice 1.5
Unesite grad i temperaturu: Subotica 4
Unesite grad i temperaturu: Zrenjanin 9
Gradovi sa idealnom temperaturom za klizanje u decembru:
Beograd
Subotica

```

## Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 2
Unesite grad i temperaturu: Varшава 11
Unesite grad i temperaturu: Prag 2
Gradovi sa idealnom temperaturom za klizanje u decembru:

```

[Rešenje 3.9.4]

**Zadatak 3.9.5** Definirati strukturu **ParReci** koja sadrži reč na srpskom jeziku i odgovarajući prevod na engleski jezik. Napisati program koji do kraja ulaza učitava sve parove reči, a potom za rečenicu koja se zadaje u jednoj liniji ispisati prevod. Ako je reč u rečenici nepoznata umesto nje ispisati odgovarajući broj zvezdica. Maksimalna dužina reči je 50 karaktera, ukupan broj parova reči je maksimalno 100, a maksimalna dužina rečenice je 100 karaktera.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
zima winter
godina year
sreca happiness
programiranje programming
caj tea
Unesite recenicu za prevod:
piti caj zimi je sreca
**** tea **** ** happiness
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
zima winter
pas dog
sreca happiness
prijatelj friend
solja cup
covek man
Unesite recenicu za prevod:
pas je covekov najbolji prijatelj
dog is ***** best friend
```

[Rešenje 3.9.5]

**Zadatak 3.9.6** Cenoteka pomaže kupcima da pronađu najpovoljniju cenu za proizvod koji žele da kupe. Napisati program koji učitava najpre broj različitih prodavnica (ceo broj manji od 50) a zatim i podatke o ceni traženog artikla – zadaje se naziv prodavnice (niske maksimalne dužine 20 karaktera) i cena u toj prodavnici (realan broj). Korisnik zadaje željenu cenu proizvoda, a program ispisuje imena svih onih prodavnica u kojima je cena proizvoda jednaka ili manja od željene. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj prodavnica: 5
idea 58.9
mavi 58.2
roda 55.1
tempo 54.5
interev 57.99
Uneti zeljenu cenu: 57.0
Povoljne prodavnice su:
roda
tempo
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj prodavnica: 4
dm 43.2
lily 45.99
benu_apoteke 43.99
sephora 50.99
Uneti zeljenu cenu: 47.00
Povoljne prodavnice su:
dm
lily
benu_apoteke
```

[Rešenje 3.9.6]

**Zadatak 3.9.7** Statistički zavod Srbije istražuje kako rade obdaništa u Srbiji. Za dato obdanište dobija spisak  $n$  dece sa kolonama: pol (m ili z), broj godina (od 3 do 6) i ocena koju je dete dalo radu obdaništa (od 1 do 5). Maksimalan broj dece u obdaništu je 200. Napisati program koji za decu datog pola i broja godina ispisuje na tri decimale prosečnu ocenu obdaništa. U slučaju neispravnog unosa ispisati odgovarajuću poruku.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 5
Uneti podatke za svako dete, pol,
broj godina i ocenu:
m 3 5
z 3 4
m 4 2
m 5 4
m 3 4
Uneti pol i broj godina: m 3
Prosečna ocena je: 4.500.

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 10
Uneti podatke za svako dete, pol,
broj godina i ocenu:
m 3 5
z 4 4
m 5 4
z 4 3
z 3 2
z 4 5
m 6 5
z 4 4
z 4 5
m 6 3
Uneti pol i broj godina: z 4
Prosečna ocena je: 4.200.

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 15
Uneti podatke za svako dete, pol,
broj godina i ocenu:
m 3 2
z 7 5
Neispravan broj godina.

```

*Primer 4*

```

INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 2
Uneti podatke za svako dete, pol,
broj godina i ocenu:
m 3 2
z 3 5
Uneti pol i broj godina: h 5
Neispravan pol.

```

[Rešenje 3.9.7]

**Zadatak 3.9.8** Definirati strukturu kojom se opisuje student. Student je zadan svojim imenom i prezimenom (oba su maksimalne dužine 30 karaktera), smerom (R, I, V, N, T, O) i prosečnom ocenom. Napisati program koji učitava podatke o  $n$  studenata, zatim učitava smer i ispisiuje imena i prezimena onih studenta koji su sa datog smera. Potom ispisati podatke za studenta koji ima najveći prosek. Ako ima više takvih studenata ispisati sve njih. Maksimalan broj studenata je 2000. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj studenata: 5
Uneti podatke o studentima:
0. student: Kocic Marija R 9.14
1. student: Tanja Mratinkovic R 7.88
2. student: Mihailo Simic N 8.44
3. student: Milena Medar I 9.14
4. student: Ljubica Mihic N 9.00
Uneti smer: R
Studenti sa R smerom:
Kocic Marija
Tanja Mratinkovic
-----
Svi studenti koji imaju maksimalni prosek:
Kocic Marija, R, 9.14
Milena Medar, I, 9.14
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj studenata: 4
Uneti podatke o studentima:
0. student: Djordje Lazarevic N 9.05
1. student: Minja Peric W 7.70
Nekorektan smer.
```

[Rešenje 3.9.8]

**Zadatak 3.9.9** Program učitava podatke o učenicima do kraja unosa. Učenika može biti najviše 30. Za svakog učenika dato je njegovo ime (maksimalne dužine 20 karaktera) i 9 ocena (ocene su celi brojevi od 1 do 5). Ispisati:

- (a) Reč NEDOVOLJNI:, a potom imena nedovoljnih učenika. Učenik je nedovoljan ako ima barem jednu jedinicu.
- (b) Potom ispisati reč ODLICNI:, a potom imena odličnih učenika. Učenik je odličan ako ima prosek ocena veći ili jednak 4.5.

U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti podatke o djaku: Maja 4 5 2 3 4 4 3 3 4
Uneti podatke o djaku: Nikola 5 4 5 5 5 4 4 5 5
Uneti podatke o djaku: Jasmina 2 2 1 1 2 3 3 1 3
Uneti podatke o djaku: Pera 5 4 5 3 5 5 1 5 5
Uneti podatke o djaku: Pavle 4 3 2 4 3 2 4 3 2
Uneti podatke o djaku:

NEDOVOLJNI: Jasmina Pera
ODLICNI: Nikola
```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Uneti podatke o djaku:  Uros 3 4 2 3 4 2 3 4 4
Uneti podatke o djaku:  Nebojsa 4 5 5 5 4 5 5 5 5
Uneti podatke o djaku:  Sreten 2 3 2 4 5 4 4 4 2
Uneti podatke o djaku:

NEDOVOLJNI:
ODLICNI: Nebojsa

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
Uneti podatke o djaku:  Mirko 2 3 4 4 4 3 3 3 4
Uneti podatke o djaku:  Mihailo 2 3 10 5 5 2 3 4 2
Neispravna ocena.

```

[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 (maksimalne dužine 20 karaktera), prezime (maksimalne dužine 30 karaktera) i email adresa (maksimalne dužine 50 karaktera). Napisati program koji učitava ceo broj  $n$  ( $0 < n \leq 50$ ) a zatim podatke o  $n$  osoba. Ispisati imena i prezimena svih osoba koje imaju gmail adresu (čija se email adresa završava sa @gmail.com). U slučaju greške ispisati odgovarajuću poruku. Može se smatrati da je svaka email adresa dobro zadata i sadrži samo jedno pojavljivanje znaka @.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
Uneti broj osoba: 3
Uneti podatke o osobama:
ime, prezime i email.
Dusko Dugousko dusko@yahoo.com
Pink Panther panter@gmail.com
Pera Detlic pd@gmail.com
Vlasnici gmail naloga su:
Pink Panther
Pera Detlic

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
Uneti broj osoba: 3
Uneti podatke o osobama:
ime, prezime i email.
Homer Simpson homer@yahoo.com
Mardz Simpson mardz@matf.bg.ac.rs
Vlasnici gmail naloga su:

```

[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.

### 3 Predstavljanje podataka

Program treba da učitava broj potrošača  $n$  (najviše 100), zatim podatke za  $n$  potrošačkih korpi i da na osnovu učitanih podataka izračuna prosečnu cenu potrošačke korpe. Program ispisuje na dve decimale račune svake potrošačke korpe i na kraju ispisuje prosečnu cenu potrošačke korpe. Možemo pretpostaviti da nije-dan potrošač neće kupiti više od 20 artikala, kao i da naziv svakog artikla sadrži maksimalno 30 karaktera. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj potrosackih korpi: 3
Uneti podatke o korpi:
Broj artikala: 4
Unesi artikal, naziv, kolicinu i cenu: jabuke 10 22.4
Unesi artikal, naziv, kolicinu i cenu: dezodorans 1 120.99
Unesi artikal, naziv, kolicinu i cenu: C_supa 3 36.56
Unesi artikal, naziv, kolicinu i cenu: sunka 1 230.99
Uneti podatke o korpi:
Broj artikala: 2
Unesi artikal, naziv, kolicinu i cenu: Jafa_keks 55.78
Unesi artikal, naziv, kolicinu i cenu: Najlepse_zelje 62.99
Uneti podatke o korpi:
Broj artikala: 3
Unesi artikal, naziv, kolicinu i cenu: prasad_za_ves 1 1199.99
Unesi artikal, naziv, kolicinu i cenu: omeksivac 1 279.99
Unesi artikal, naziv, kolicinu i cenu: protiv_kamenca 1 699.99

Korpa 0:
jabuke 10 22.40
dezodorans 1 120.99
C_supa 3 36.56
sunka 1 230.99
-----
ukupno: 685.66

Korpa 1:
Jafa_keks 55 0.78
Najlepse_zelje 62 0.99
-----
ukupno: 104.28

Korpa 2:
prasad_za_ves 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** Uvesti tip podataka *Sifra* kojim se opisuje način šifrovanja alfanumeričkih karaktera. Svaka šifra se opisuje pozitivnom celobrojnom



vrednošću  $b$  koja određuje broj pozicija pomeranja, kao i karakterom 'L' ili 'D' koji određuje smer pomeranja (levo ili desno).

- (a) Napisati funkciju `char sifruj(char c, Sifra s)` koja transformiše zadati karakter  $c$  po šifri  $s$ . Karakter se šifruje tako što se svako slovo zamenjuje slovom za  $b$  mesta levo ili desno od njega u abecedi, i to ciklično, a isto tako i za cifre. Na primer: za  $b = 2$ , i  $\text{smer} = \text{'D'}$  : a se menja sa c, b sa d, ..., x sa z, y sa a, z sa b, 1 sa 3, ..., 8 sa 0, 9 sa 1. Funkcija vraća novodobijeni karakter.
- (b) Načini šifrovanja se zadaju do kraja unosa i to u obliku 2 D 5 L. Potom se zadaju karakteri do kraja unosa. Izmeniti alfanumeričke karaktere prema svim zadatim šiframa i ispisati dobijeni rezultat. Maksimalan broj karaktera može biti 5000. Maksimalan broj šifri može biti 100. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti sifre u obliku: broj, smer:
23 D
Uneti tekst za sifrovanje:
Temperatura danas je 23 stepena Celzijusova.
Rckncpxrpx bxlqx hc 56 qrcnclx Zcjxghsqmtx.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti sifre u obliku: broj, smer:
3 l 7 a
Neispravan smer.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti sifre u obliku: broj, smer:
23 D 3 L 14 D 20 L 1 L 2 L 5 D
Uneti tekst za sifrovanje:
Temperatura danas je 23 stepena Celzijusova.
Kudguiqkqliq tqeqj zu 89 jkugueq Sucqyzljfmq.
```

[Rešenje 3.9.12]

**Zadatak 3.9.13** Definisati strukturu **Lopta** sa poljima **poluprecnik** (ceo broj u centimetrima) i **boja** (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Zatim učitati informacije o  $n$  lopti ( $0 < n < 50$ ) i ispisati ukupnu zapreminu, kao i broj crvenih lopti. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 4
Unesite dalje poluprecnike i boje lopti
(1-plava, 2-zuta, 3-crvena, 4-zelena):
1.lopta: 4 1
2.lopta: 1 3
3.lopta: 2 3
4.lopta: 10 4
Ukupna zapremina: 4494.57
Broj crvenih lopti: 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 8
Unesite dalje poluprecnike i boje lopti
(1-plava, 2-zuta, 3-crvena, 4-zelena):
1. lopta: 2 1
2. lopta: 30 3
3. lopta: 7 3
4. lopta: 4 1
5. lopta: 5 2
6. lopta: 6 2
7. lopta: 12 3
8. lopta: 14 2
Ukupna zapremina: 134996.34
Ukupno crvenih lopti: 3
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 8
Unesite dalje poluprecnike i boje lopti
(1-plava, 2-zuta, 3-crvena, 4-zelena):
1. lopta: 1 2
2. lopta: 2 10
Nekorektan unos.
```

[Rešenje 3.9.13]

**Zadatak 3.9.14** Napisati program za predstavljanje poligona i izračunavanje njegovog obima i dužine stranica.

- Definisati tip podataka **TACKA** pogodan za predstavljanje tačke Dekartovske ravni (čije su  $x$  i  $y$  koordinate podaci tipa **double**).
- Definisati funkciju **double rastojanje(TACKA a, TACKA b)** koja izračunava rastojanje između dve tačke.
- Definisati funkciju **unsigned ucitaj\_poligon(TACKA\* tacke, unsigned n)** koja učitava 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.
- Definisati funkciju **double obim(TACKA\* poligon, unsigned n)** koja izračunava obim poligona sa  $n$  tačaka u zadatom nizu *NAPOMENA: Prilikom računanja obima ne zaboraviti stranicu koja spaja poslednje i prvo teme.*
- Definisati funkciju **double maksimalna\_stranica(TACKA\* poligon, unsigned n)** koja izračunava dužinu najduže stranice poligona sa  $n$  tačaka u zadatom nizu.

- (f) Napisati funkciju `double povrsina_trougla(TACKA A, TACKA B, TACKA C)` za računanje površine trougla.
- (g) Napisati funkciju `double povrsina(TACKA* poligon, unsigned n)` za računanje površine konveksnog poligona. NAPOMENA: *Zadatak se može rešiti korišćenjem funkcije `povrsina_trougla`.*
- (h) Napisati program koji učitava poligon sa maksimalno  $N$  temena ( $0 < N \leq 1000$ ) i za učitani poligon ispisuje na tri decimale obim, dužinu maksimalnu stranice i površinu. Pretpostaviti da je uneseni poligon konveksan. Poligon mora imati barem 3 temena. U slučaju greške ispisati odgovarajuću poruku.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti maksimalan broj tacaka poligona: 10
0 0
0 6
3 3
Obim poligona je 14.485.
Duzina maksimalne stranice je 6.000.
Povrsina poligona je 9.000.
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Uneti maksimalan broj tacaka poligona: 10
0 0
12 0
13 2
16 5
20 10
18 15
15 20
10 20
8 15
3 4
Obim poligona je 63.566.
Duzina maksimalne stranice je 12.083.
Povrsina poligona je 247.500.
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Uneti maksimalan broj tacaka poligona: 4
0 0
Neispravan broj tacaka poligona.
```

[Rešenje 3.9.14]

\* **Zadatak 3.9.15** Definirati strukturu **IZRAZ** kojom se opisuje numerički izraz nad celim brojevima koji se sastoji od dva celobrojna operanda, numeričke operacije (sabiranje, oduzimanje, množenje ili celobrojno deljenje) nad celim brojevima.

- (a) Napisati funkciju koja ispituje da li je dati izraz korektno zadat i vraća 1 ako jeste a 0 u suprotnom. Podrazumevamo da je izraz korektno zadat ako operacija odgovara  $+$ ,  $-$ ,  $*$  ili  $/$  i u slučaju deljenja drugi operand je različit od 0.

### 3 Predstavljanje podataka

---

- (b) Napisati funkciju koja za dati izraz odredjuje vrednost izraza.
- (c) Napisati funkciju koja učitava dati izraz. Funkcija treba da uči sa standardnog ulaza izlaz koji je zadat prefiksno — prvo operacija, a potom dva operanda. Funkcija vraća 1 ako je učitavanje bilo uspešno, tj. ako je izraz bio korektno zadat ili 0 u suprotnom.
- (d) Napisati funkciju koja štampa dati izraz infiksno, u obliku "*operand<sub>1</sub> operacija operand<sub>2</sub> = vrednost*".

Napisati glavni program koji učitava prirodan broj  $n$ , ( $n < 1000$ ) a zatim  $n$  izraza u prefiksnoj notaciji. Program treba da ispiše maksimalnu vrednost unetih izraza i sve izraze čija vrednost je manja od polovine maksimalne vrednosti.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj izraza: 4
Uneti izraze u prefiksnoj notaciji:
+ 10 4
- 9 2
* 11 2
/ 7 3
Maksimalna vrednost izraza: 22
Izrazi čija je vrednost manja
od polovine maksimalne vrednosti:
9 - 2 = 7
7 / 3 = 2
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj izraza: 10
Uneti izraze u prefiksnoj notaciji:
+ 10 2
- -678 34
* 77 2
+ 1000 -23
+ 102 4
- 200 23
/ 67 12
/ 1000 2
* 44 6
/ 13 1
Maksimalna vrednost izraza: 977
Izrazi 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:
Uneti broj izraza: 3
Uneti izraze u prefiksnoj notaciji:
* 1 2
/ 3 0
Deljenje nulom!
Nekorektan unos
```

[Rešenje 3.9.15]

\* **Zadatak 3.9.16** Definisati strukturu kojom se zadaje polinom. Polinom je dat svojim stepenom (može biti najviše 10) i realnim koeficijentima.

- (a) Napisati funkciju koja učitava jedan polinom dat stepenom i koeficijentima.
- (b) Napisati funkciju koja ispisuje polinom u obliku  $k_0 \pm k_1 * x \pm k_2 * x^2 \pm k_3 * x^3 \pm \dots \pm k_n * x^n$  (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.
- (c) Napisati funkciju koja za dati polinom određuje njegov integral.
- (d) Učitati polinome do kraja ulaza i za svaki učitani polinom odrediti i ispisati integral tog polinoma. Maksimalan broj polinoma je 100.

U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti stepen: 3
Uneti koeficijente polinoma:
1 0 3 1
Uneti stepen: 4
Uneti koeficijente polinoma:
7 9 4 0 4
Uneti stepen:

Integrali su:
1.00*x + 1.00*x^3 + 0.25*x^4
7.00*x + 4.50*x^2 + 1.33*x^3 + 0.80*x^5
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti stepen: 3
Uneti koeficijente polinoma:
1 0 -4 1
Uneti stepen: 2
Uneti koeficijente polinoma:
1 2 -3
Uneti stepen: 1
Uneti koeficijente polinoma:
0 -1
Uneti stepen:

Integrali su:
1.00*x -1.33*x^3 + 0.25*x^4
1.00*x + 1.00*x^2 -1.00*x^3
-0.50*x^2
```

[Rešenje 3.9.16]

## 3.10 Rešenja

### Rešenje 3.9.1

```
1 #include <stdio.h>
3 /* Struktura koja opisuje kompleksni broj obuhvata polje za realni
   * i polje za imaginarni deo broja.
```

```
5  */
   typedef struct Complex {
7
   float re;
   float im;
9  } Complex;
11
   /* Funkcija kojom se izracunava zbir kompleksnih brojeva. */
13  Complex saberi(Complex *a, Complex *b) {

15      Complex c;
      c.re = a->re + b->re;
17      c.im = a->im + b->im;
      return c;
19  }

21  /* Funkcija kojom se izracunava razlika kompleksnih brojeva. */
   Complex oduzmi(Complex *a, Complex *b) {
23
25      Complex c;
      c.re = a->re - b->re;
      c.im = a->im - b->im;
27      return c;
   }
29
   /* Funkcija kojom se izracunava proizvod kompleksnih brojeva. */
31  Complex pomnozi(Complex *a, Complex *b) {

33      Complex c;
      c.re = a->re * b->re - a->im * b->im;
35      c.im = b->re * a->im + a->re * b->im;
      return c;
37  }

39  /* Funkcija kojom se izracunava kolicnik kompleksnih brojeva. */
   Complex podeli(Complex *a, Complex *b) {
41
43      Complex c;
      c.re = (a->re * b->re + a->im * b->im) / (b->re*b->re + b->im*b->im
      );
      c.im = (b->re * a->im - a->re * b->im) / (b->re*b->re + b->im*b->im
      );
45      return c;
   }
47
49  int main() {

51      Complex a, b;
      Complex c;

53      /* Ucitavamo kompleksne brojeve. */
      printf("Unesite realni i imaginarni deo prvog broja: ");
```

```

55 scanf("%f%f", &a.re, &a.im);

57 printf("Unesite realni i imaginarni deo drugog broja: ");
scanf("%f%f", &b.re, &b.im);

59
c = saberi(&a, &b);
61 /* Ukoliko je imaginarni deo negativan,
   * njegov zapis vec ukljucuje znak,
63 * te to treba proveriti.
   * Inace, broj je oblika a+b*i.
65 */
printf("Zbir: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im);

67
c = oduzmi(&a, &b);
69 printf("Razlika: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im)
;

71
c = pomnozi(&a, &b);
printf("Proizvod: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im
);

73
if(b.re != 0 || b.im != 0) {
75     c = podeli(&a, &b);
    printf("Kolicnik: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.
im);
77 }
/* U polju kompleksnih brojeva
79 * nije dozvoljeno deljenje nulom.
   */
81 else
    printf("Kolicnik ne postoji.\n");
83
85 return 0;
}

```

### Rešenje 3.9.2

```

1 #include <stdio.h>

3 typedef struct
{
5     int brojilac;
    int imenilac;
7 }razlomak;

9 int nzd(int a, int b)
{
11     int pom;

13     if (a < b)

```

```
15 {
16     pom = a;
17     a = b;
18     b = pom;
19 }
20
21 while(b != 0)
22 {
23     pom = a % b;
24     a = b;
25     b = pom;
26 }
27 return a;
28 }
29
30 razlomak zbir(razlomak a, razlomak b)
31 {
32     razlomak c;
33     int nzd_razlomka;
34
35     c.brojilac = a.brojilac * b.imenilac + b.brojilac*a.imenilac;
36     c.imenilac = a.imenilac*b.imenilac;
37
38     /* Brojilac i imenilac dobijenog zbira se dele najvećim zajedničkim
39      * deliocom.
40      */
41     nzd_razlomka = nzd(c.brojilac, c.imenilac);
42
43     c.brojilac = c.brojilac/nzd_razlomka;
44     c.imenilac = c.imenilac/nzd_razlomka;
45
46     return c;
47 }
48
49 razlomak proizvod(razlomak a, razlomak b)
50 {
51     razlomak c;
52     int nzd_razlomka;
53
54     c.brojilac = a.brojilac*b.brojilac;
55     c.imenilac = a.imenilac*b.imenilac;
56
57     /* Brojilac i imenilac dobijenog zbira se dele najvećim zajedničkim
58      * deliocom.
59      */
60     nzd_razlomka = nzd(c.brojilac, c.imenilac);
61
62     c.brojilac = c.brojilac/nzd_razlomka;
63     c.imenilac = c.imenilac/nzd_razlomka;
64
65     return c;
```



```

}
67
int main()
69 {
    int n, i;

71     razlomak suma, proizvod_svih, r;

73     printf("Unesi broj razlomaka: ");
75     scanf("%d", &n);

77     suma.brojilac = 0;
79     suma.imenilac = 1;

81     proizvod_svih.brojilac = 1;
    proizvod_svih.imenilac = 1;
83

    printf("Uneti razlomke:\n");
85     for(i=0; i<n; i++)
    {
87         scanf("%d%d", &r.brojilac, &r.imenilac);

89         suma = zbir(suma, r);
        proizvod_svih = proizvod(proizvod_svih, r);
91     }

93     printf("Suma svih razlomaka je %d/%d.\n", suma.brojilac, suma.
        imenilac);
    printf("Proizvod svih razlomaka je %d/%d.\n", proizvod_svih.
        brojilac, proizvod_svih.imenilac);
95

    return 0;
97 }

```

### Rešenje 3.9.3

```

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

4  #define MAX_DUZINA 21
    #define MAX_BR_VOCKI 50

6  typedef struct vocka
8  {
    char ime[MAX_DUZINA];
10     float vitamin;
    } VOCKA;
12

14 int main()

```

### 3 Predstavljanje podataka

---

```
{
16  VOCKA  vocke[MAX_BR_VOCKI];
    int i = 0, n, max_vocka;
18  char ime[MAX_DUZINA];

20  /*
    Program ucitava podatke o vockama i smesta ih u niz
22    sve dok se ne unese rec KRAJ ili ucita MAX_BR_VOCKI vocki.
    */
24  do
    {
26      printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
      scanf("%s",ime);
28      /*
        Kada se unese rec KRAJ prekida se petlja.
        */
30      if(strcmp(ime, "KRAJ") == 0)
32          break;

34      /*
        Inace ucitava se kolicina vitamina
36        i ta vrednost se smesta u vocku na poziciji "i".
        */
38      strcpy(vocke[i].ime,ime);
      scanf("%f",&vocke[i].vitamin);
40      i++;
    }
42  while(i<MAX_BR_VOCKI);

44  n = i;

46  /*
    Pretpostavka je da prva vocka ima najvise vitamina.
48    Petljom se prolazi niz vocki i ukoliko se naidje na vocku koja
    ima vise vitamina
    od one koja trenutno ima najvise, azurira se vrednosti maksimalne
    vocke.

50    Sve vreme se cuva indeks vocke sa najvise vitamina C.
52    */

54  max_vocka = 0;
    for(i=1;i<n;i++)
56        if(vocke[i].vitamin > vocke[max_vocka].vitamin)
        {
58            max_vocka = i;
        }

60  printf("Voce sa najvise C vitamina je: %s\n", vocke[max_vocka].ime)
    ;

62  return 0;
```

64 }

## Rešenje 3.9.4

```

1  #include <stdio.h>
2  #define MAX_DUZINA 20
3  #define MAX_BR_GRADOVA 50
4
5  typedef struct Grad{
6      char ime_grada[MAX_DUZINA+1];
7      float temperatura;
8  }Grad;
9
10
11 int main(){
12     int n, i;
13     Grad grad[MAX_BR_GRADOVA];
14
15     printf("Unesite broj n: ");
16     scanf("%d", &n);
17     if(n<0 || n>MAX_BR_GRADOVA){
18         printf("Greska: pogresan unos!\n");
19         return 0;
20     }
21
22     for(i=0; i<n; i++){
23         printf("Unesite grad i temperaturu: ");
24         scanf("%s %f", grad[i].ime_grada, &grad[i].temperatura);
25     }
26
27     printf("Gradovi sa idealnom temperaturom za klizanje u decembru:\n");
28
29     for(i=0; i<n; i++){
30         if(grad[i].temperatura>=3 && grad[i].temperatura<=8){
31             printf("%s\n", grad[i].ime_grada);
32         }
33     }
34
35     return 0;
36 }

```

## Rešenje 3.9.5

```

1  #include <stdio.h>
2  #include <string.h>
3  #define MAX_DUZINA 21
4  #define MAX_BR_RECIP 100
5
6  typedef struct ParReci{

```

```

    char sr[MAX_DUZINA+1];
8   char en[MAX_DUZINA+1];
}ParReci;

10

12  /*
    Funkcija koja u recniku koji sadrzi n reci trazi prevod reci rec i
    upisuje ga u prevod.
14   Ukoliko se rec ne nalazi u recniku, prevod se sastoji od zvezdica
    pri cemu broj zvezdica odgovara
    duzini nepoznate reci.
16  */

18  void pronadji_prevod(ParReci recnik[], int n, char rec[], char prevod
    []){
    int i;

20

    /* Pretrazuje se recnik i trazi se zadata rec. */
22   for(i=0; i<n; i++){
    {
24     if(strcmp(recnik[i].sr, rec)==0)
    {
26       strcpy(prevod, recnik[i].en);
       return;
28     }
    }
30

    /* Ukoliko rec nije pronadjena, formira se prevod reci koji se
    sastoji od zvezdica. */
32   for(i=0; rec[i]; i++){
    prevod[i]='*';
34   }
    prevod[i]='\0';
36 }

38

int main(){
40   ParReci recnik[MAX_BR_RECII];
    int n;
42   char sr[MAX_DUZINA+1];
    char en[MAX_DUZINA+1];
44   int i, j, k;
    char rec[MAX_DUZINA+1];
46   char prevod[MAX_DUZINA+1];
    char c;

48

    /* Ucitavaju se parovi reci sa standardnog ulaza sve do kraja ulaza
    . */
50   i=0;
    while(scanf("%s %s", sr, en)!=EOF){
52     if(i==MAX_BR_RECII)
        break;
```

```

54     strcpy(recnik[i].sr, sr);
55     strcpy(recnik[i].en, en);
56
57     i++;
58 }
59 /* Broj parova reci se cuva u promenljivoj n. */
60 n=i;
61
62 printf("Unesite recenicu za prevod: \n");
63 do
64 {
65     /* Ucitava se rec po rec date recenice i pronalazi se njen prevod
66     . */
67     scanf("%s", rec);
68
69     pronadji_prevod(recnik, n, rec, prevod);
70     printf("%s ", prevod);
71
72     /* Ukoliko je karakter iza reci znak za novi red, onda se prekida
73     sa unosom, a ako nije
74     * učitava se sledeca recenica.
75     */
76     c = getchar();
77
78 }while(c != '\n');
79
80 putchar('\n');
81
82 return 0;
83 }

```

### Rešenje 3.9.6

```

1  #include <stdio.h>
2
3  #define MAX_PRODAVNICA 50
4  #define DUZINA_RECI 21
5
6  typedef struct
7  {
8      char prodavnica[DUZINA_RECI];
9      double cena;
10 }podatak;
11
12 int main()
13 {
14     podatak niz[MAX_PRODAVNICA];
15     double zeljena;
16     int n, i;
17

```

### 3 Predstavljanje podataka

---

```
19  printf("Uneti broj prodavnica: ");
    scanf("%d", &n);

21  if (n <=0 || n > MAX_PRODAVNICA)
    {
23      printf("Neispravan broj prodavnica.\n");
        return -1;
25  }

27  for(i=0; i<n; i++)
    {
29      scanf("%s%lf", niz[i].prodavnica, &niz[i].cena);

31      if (niz[i].cena <= 0)
        {
33          printf("Neispravna cena.\n");
            return -1;
35        }
    }

37  printf("Uneti zeljenu cenu: ");
39  scanf("%lf", &zeljena);

41  printf("Povoljne prodavnice su:\n");
    for(i=0; i<n; i++)
43      if (niz[i].cena <= zeljena)
        printf("%s\n", niz[i].prodavnica);
45
47  return 0;
}
```

#### Rešenje 3.9.7

```
#include <stdio.h>

2
#define MAX_DECE 200

4
typedef struct
6 {
    char pol;
    int broj_godina;
    int ocena;
8 }dete;
10

12 int main()
    {
14     int n, i, broj_godina;
        dete niz[MAX_DECE];
16     char blanko, pol;
        int suma, broj_dece;
18
```

```
20 printf("Uneti broj dece: ");
   scanf("%d", &n);

22 if (n <= 0 || n > MAX_DECE)
   {
24     printf("Neispravan broj dece.\n");
       return -1;
26 }

28 printf("Uneti podatke za svako dete, pol, broj godina i ocenu:\n");
   for(i=0; i<n; i++)
30 {
       scanf("%c%c%d", &blanko, &niz[i].pol, &niz[i].broj_godina, &niz
           [i].ocena);

32     /* Ispitivanje pogresnog unosa. */
34     if (niz[i].pol != 'm' && niz[i].pol != 'z')
       {
36         printf("Neispravan pol.\n");
           return -1;
38     }
       if (niz[i].broj_godina > 6 || niz[i].broj_godina < 3)
40     {
           printf("Neispravan broj godina.\n");
42         return -1;
       }
       if (niz[i].ocena < 1 || niz[i].ocena > 5)
44     {
           printf("Neispravna ocena.\n");
46         return -1;
       }
48     }
50 }

52 printf("Uneti pol i broj godina: ");
   scanf("%c%c%d", &blanko, &pol, &broj_godina);

54 /* Ispitivanje ispravnosti unetih podataka. */
   if (pol != 'm' && pol != 'z')
56 {
       printf("Neispravan pol.\n");
58     return -1;
   }
60 if (broj_godina > 6 || broj_godina < 3)
   {
62     printf("Neispravan broj godina.\n");
       return -1;
64 }

66 suma = 0;
   broj_dece = 0;

68 for(i=0; i<n; i++)
```

### 3 Predstavljanje podataka

```
70     if (niz[i].pol == pol && niz[i].broj_godina == broj_godina)
71     {
72         suma += niz[i].ocena;
73         broj_dece++;
74     }
75
76     if (broj_dece == 0)
77         printf("Ne postoje deca sa takvim karakteristikama.\n");
78     else
79         printf("Prosecna ocena je: %.3lf.\n", (double)suma/broj_dece);
80
81     return 0;
82 }
```

#### Rešenje 3.9.8

```
#include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAXST 2000
5 #define MAX 31
6
7 typedef struct Student
8 {
9     char ime[MAX];
10    char prezime[MAX];
11    char smer;
12    float prosek;
13 } STUDENT;
14
15 void provera(char smer)
16 {
17     if (smer != 'R' && smer != 'I' && smer != 'V' && smer != 'N' &&
18         smer != 'T' && smer != 'O')
19     {
20         printf("Nekorektan smer.\n");
21         exit(EXIT_FAILURE);
22     }
23 }
24
25 void ucitaj(STUDENT* s)
26 {
27     scanf("%s", s->ime);
28
29     scanf("%s", s->prezime);
30
31     getchar();
32     scanf("%c", &s->smer);
33
34     scanf("%f", &s->prosek);
35 }
```



```

36  /* II */
37  /*
38   Kada neku promenljivu prenosimo u funkciju kao argument, obicno
   je prenosimo po vrednosti (bez pokazivaca), ako se ona nece menjati
   u funkciji
40   ili po adresi (preko pokazivaca), ako ce se njena vrednost
   promeniti u funkciji.

42   Prilikom poziva funkcije, za svaki argument funkcije kreira se
   promenljiva
   koja predstavlja lokalnu kopiju argumenta i koja prestaje da
   postoji po zavrsetku
44   funkcije. S obzirom da se strukture sastoje od vise polja,
   zauzimaju
   vise memorije nego nestrukturne promenljive. Zbog toga je za
   njihovo kopiranje
46   potrebno vise vremena i vise memorijskih resursa nego za kopiranje
   nestrukturnih
   promenljivih.

48   Da bismo ucinili program efikasnijim, korisno je da strukturu uvek
   kao
50   argument funkcije prenosimo po adresi (preko pokazivaca), bez
   obzira
   da li ce se struktura u toj funkciji menjati ili ne. Pokazivac na
   strukturu
52   zauzima manje memorije nego sama struktura pa je izrada njegove
   kopije
   brza a kopija pokazivaca uzima manji memorijski prostor nego kopija
   strukture.

56   Kada prenosimo strukturnu promenljivu u funkciju po adresi (preko
   pokazivaca), tada
   imamo mogucnost da je u funkciji menjamo. Ukoliko zelimo da
   onemogucimo promenu,
58   uz argument dodajemo kljucnu rec const. Ako pokusamo da promenimo
   argument
   funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
   s->smer='X';),
60   kompajler ce prijaviti gresku. Na ovaj nacin obezbedjujemo da
   promenljiva
   koju smo preneli po adresi ne da bismo je promenili vec radi
   povecanja
62   efikasnosti programa, ne bude, cak ni slucajno, izmenjena u
   funkciji.

64  */

66  void ispisi(const STUDENT* s)
67  {
68      printf("%s %s, %c, %.2f\n", s->ime, s->prezime, s->smer, s->prosek);

```

```
70 }
72 float najveći_prosek(STUDENT studenti[], int n)
73 {
74     float m;
75     int i;
76
77     m = studenti[0].prosek;
78     for(i=1; i<n; i++)
79         if(m<studenti[i].prosek)
80             m=studenti[i].prosek;
81     return m;
82 }
84 /*
85     Struktura može da bude povratna vrednost funkcije.
86 */
87 STUDENT prvi_student_sa_najvecim_prosekom(STUDENT studenti[], int n,
88     float m)
89 {
90     STUDENT s;
91     int i;
92     for(i=0; i<n; i++)
93         if(m == studenti[i].prosek)
94         {
95             /*
96                 Na strukture se može primenjivati
97                 naredba dodele.
98             */
99             s = studenti[i];
100             break;
101         }
102     return s;
103 }
104
105 int main()
106 {
107     STUDENT studenti[MAXST];
108     int n;
109     int i;
110     float max_prosek;
111     STUDENT student_sa_max_prosekom;
112     int indeks;
113     char smer;
114
115     printf("Uneti broj studenata: ");
116     scanf("%d", &n);
117
118     if (n<0 || n>MAXST)
119     {
```

```

120     printf("Nekorektan unos\n");
121     return -1;
122 }

124 printf("Uneti podatke o studentima:\n");
125 for(i=0;i<n;i++)
126 {
127     printf("%d. student: ", i);
128     ucitaj(&studenti[i]);
129     provera(studenti[i].smer);
130 }

132 printf("Uneti smer: ");
133 getchar();
134 scanf("%c", &smer);

136 provera(smer);

138 printf("Studenti sa R smerom:\n");
139 for(i=0;i<n;i++)
140     if(studenti[i].smer == smer)
141         printf("%s %s\n", studenti[i].ime, studenti[i].prezime);
142 printf("-----\n");

144 /* Stampamo podatke o svim studentima sa
145    maksimalnim prosekom.
146 */

148 max_prosek = najveći_prosek(studenti, n);
149 printf("Svi studenti koji imaju maksimalni prosek:\n");
150 for(i=0;i<n;i++)
151     if(studenti[i].prosek == max_prosek)
152         ispisi(&studenti[i]);
153
154
156 return 0;
157 }

```

### Rešenje 3.9.9

```

1 #include <stdio.h>
2 #include <stdlib.h>

4 #define IME 21
5 #define OCENE 9
6 #define MAX_DJAKA 30

8 typedef struct
9 {
10     char ime[IME];

```

```
    int ocena[OCENE];
12 }_djak;

14 void provera(int ocena)
{
16     if (ocena < 1 || ocena > 5)
    {
18         printf("Neispravna ocena.\n");
        exit(EXIT_FAILURE);
20     }
}

22
24 int main()
{
    _djak niz[MAX_DJAKA];
26     int i = 0, n, j;
    int suma;
28     float prosek;

30     printf("Uneti podatke o djaku: ");
    while (scanf("%s", niz[i].ime) != EOF && i < MAX_DJAKA)
32     {
        for (j=0; j<9; j++)
34         {
            scanf("%d", &niz[i].ocena[j]);
36             provera(niz[i].ocena[j]);
        }

38         i++;
40         printf("Uneti podatke o djaku: ");
    }

42     n = i;

44     printf("\n\nNEDOVOLJNI: ");
    for (i=0; i<n; i++)
46     for (j=0; j<9; j++)
48     if (niz[i].ocena[j] == 1)
    {
50         printf("%s ", niz[i].ime);
        break;
52     }
    printf("\n");

54

56     printf("ODLICNI: ");
    for (i=0; i<n; i++)
58     {
        suma = 0;
60         for (j=0; j<9; j++)
            suma += niz[i].ocena[j];
62     }
```

```
        prosek = (float)suma/9;
64
        if (prosek >= 4.5)
66            printf("%s ", niz[i].ime);
        }
68    printf("\n");
70    return 0;
}
```

### Rešenje 3.9.10

```
1  #include <stdio.h>
   #include <string.h>
3
   #define IME 21
5   #define PREZIME 31
   #define EMAIL 51
7
   #define MAX_OSOBA 50
9
   typedef struct
11 {
       char ime[IME];
13       char prezime[PREZIME];
       char email[EMAIL];
15 }Osoba;

17 int gmail(char* s)
   {
19       char* deo = strtok(s, "@");
       deo = strtok(NULL, "");
21
       if (strcmp(deo, "gmail.com") == 0)
23           return 1;
       else
25           return 0;
   }
27
   int main()
29 {
       int n, i;
31       Osoba osobe[MAX_OSOBA];

33       printf("Uneti broj osoba: ");
       scanf("%d", &n);
35
       if (n < 0 || n >= MAX_OSOBA)
37       {
           printf("Greska u broju osoba.\n");
39           return -1;
       }
```

### 3 Predstavljanje podataka

```
    }  
41  
    printf("Uneti podatke o osobama, ime, prezime i email.\n");  
43    for(i=0; i<n; i++)  
        scanf("%s%s%s", osobe[i].ime, osobe[i].prezime, osobe[i].email);  
45  
    printf("Vlasnici gmail naloga su:\n");  
47    for(i=0; i<n; i++)  
        if (gmail(osobe[i].email))  
49            printf("%s %s\n", osobe[i].ime, osobe[i].prezime);  
51    return 0;  
}
```

#### Rešenje 3.9.11

```
1  #include <stdio.h>  
3  #define MAXART 20  
   #define MAXPOT 100  
5  #define MAXNAZIV 31  
7  typedef struct artikal  
   {  
9      char naziv[MAXNAZIV];  
      int kolicina;  
11     float cena;  
   } ARTIKAL;  
13  
   typedef struct korpa  
   {  
15     int br_art;  
      ARTIKAL artikli[MAXART];  
17   } KORPA;  
19  
   /*  
21     Funkcija ucitaj_artikal ucitava podatke za jedan  
      artikal i vraca 1 ako je ucitavanje bilo uspesno  
23     a 0 u suprotnom. Ucitavanje je neuspesno ukoliko  
      kolicina nekog artikla ili njegova cena nisu pozitivni  
25     brojevi.  
27  
      S obzirom da funkcija ucitaj_artikal treba da vrati  
      dve vrednosti (ucitanu strukturu i indikator uspesnosti),  
29     strukturu ARTIKAL prenosimo preko pokazivaca a  
      indikator uspesnosti vracamo kao povratnu vrednost.  
31  
   */  
33  
   int ucitaj_artikal(ARTIKAL* a)  
35   {
```

```
printf("Unesi artikal, naziv, kolicinu i cenu: ");
37 scanf("%s", a->naziv);
   scanf("%d", &a->kolicina);
39
   if (a->kolicina<=0)
41   {
       printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina
   );
43       return 0;
   }

   scanf("%f",&a->cena);
47   if (a->cena<0)
   {
49       printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
       return 0;
51   }

53   return 1;
}

55
/*
57 Funkcija izracunaj_racun izracunava racun date
   potrosacke korpe u kojoj su inicijalizovani
59 podaci o broju artikala i o svakom pojedinacnom
   artiklu.
61 */
float izracunaj_racun(const KORPA* k)
63 {
   int i;
   float racun=0;
   for(i=0;i<k->br_art;i++)
65     racun+=k->artikli[i].kolicina * k->artikli[i].cena;
   return racun;
67
69 }

71
/*
73 Pri ucitavanju korpe, zadaje se broj artikala a zatim
   podaci za svaki artikal.

75 Funkcija ucitaj_korpu vraca 1 ako je ucitavanje uspesno
   i 0 u suprotnom. Do neuspesnog ucitavanja moze doci
77 ako broj artikala u korpi nije pozitivan ili ako dodje
   do neuspesnog ucitavanja nekog artikla.
79 */

81 int ucitaj_korpu(KORPA* k)
   {
83     int i;
       printf("Uneti podatke o korpi: \n");
       printf("Broj artikala: ");
85       scanf("%d", &k->br_art);
```

### 3 Predstavljanje podataka

---

```
87     if (k->br_art<=0)
88     {
89         printf("Nekorektan unos za broj artikala: %d\n", k->br_art);
90         return 0;
91     }
92     for(i=0; i<k->br_art;i++)
93         if (ucitaj_artikal(&k->artikli[i])==0)
94             return 0;
95
96     return 1;
97 }
98
99 /*
100 Funkcija ucitaj_niz_korpi ucitava podatke
101 za niz od n potrosackih korpi. Funkcija
102 vraća 1 ako je ucitavanje uspesno i 0 ako
103 nije. Ucitavanje je neuspesno ukoliko ne uspe
104 ucitavanje jedne od korpi.
105 */
106
107 int ucitaj_niz_korpi(KORPA korpe[], int n)
108 {
109     int i,j;
110     for(i=0; i<n; i++)
111         if(ucitaj_korpu(&korpe[i])==0)
112             return 0;
113
114     return 1;
115 }
116
117 /*
118 Funkcija stampaj_racun ispisuje na
119 standardni izlaz racun za datu korpu
120 tako sto za svaki artikal ispise
121 naziv, cenu i kolicinu i na kraju
122 ukupnu cenu za kupljene artikle.
123 */
124
125 void stampaj_racun(const KORPA* k)
126 {
127     int i,j;
128     for(i=0;i<k->br_art;i++)
129         printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
130             kolicina, k->artikli[i].cena);
131     printf("-----\n");
132     printf("\tukupno: %.2f\n", izracunaj_racun(k));
133 }
134
135 /*
136 Funkcija stampaj_racune_za_korpe
```



```
139     ispisuje na standardni izlaz racune
140     za svaku korpu u nizu potrosackih
141     korpi
142 */
143 void stampaj_racune_za_korpe(KORPA korpe[], int n)
144 {
145     int i;
146     for (i=0;i<n;i++)
147     {
148         printf("\nKorpa %d:\n",i);
149         stampaj_racun(&korpe[i]);
150     }
151 }
152
153 /*
154 Funkcija prosek racuna prosechnu cenu
155 potrosacke korpe za dati niz potrosackih
156 korpi
157 */
158 float prosek(KORPA korpe[], int n)
159 {
160     int i;
161     float p;
162
163     for(i=0;i<n;i++)
164         p+=izracunaj_racun(&korpe[i]);
165
166     return p/n;
167 }
168
169 int main()
170 {
171     int n;
172     KORPA korpe[MAXPOT];
173
174     printf("Uneti broj potrosackih korpi:");
175     scanf("%d", &n);
176
177     if(n<0 || n>MAXPOT)
178     {
179         printf("Nekorektan unos broja potrosackih korpi: %d\n",n);
180         return -1;
181     }
182
183     if (ucitaj_niz_korpi(korpe, n)==0)
184         return -1;
185
186     stampaj_racune_za_korpe(korpe,n);
187     printf("Prosečna cena potrosacke korpe: %.2f\n", prosek(korpe, n))
188     ;
```

```
189     return 0;
    }
```

#### Rešenje 3.9.12

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

4 #define BROJ_SIFRI 100
#define BROJ_KARAKTERA 5000

6
8 typedef struct
9 {
10     int b;
11     char smer;
12 }Sifra;

12 char sifruj(char c, Sifra s)
13 {
14     int pomeraaj;

16     if (!isalnum(c))
17         return c;

20     if (s.smer == 'L')
21     {
22         if (isdigit(c))
23         {
24             pomeraaj = s.b % 10;

26             if (pomeraaj > c - '0')
27             {
28                 pomeraaj = pomeraaj - (c - '0') - 1;
29                 c = '9' - pomeraaj;
30             }
31             else
32                 c = c - pomeraaj;

34             return c;
35         }

36         pomeraaj = s.b % 26;

38         if (islower(c))
39         {
40             if (pomeraaj > c - 'a')
41             {
42                 pomeraaj = pomeraaj - (c - 'a') - 1;
43                 c = 'z' - pomeraaj;
44             }
45             else
```

```

    c = c - pomeraj;
48
    return c;
50
}

52
if (pomeraj > c - 'A')
{
54
    pomeraj = pomeraj - (c - 'A') - 1;
    c = 'Z' - pomeraj;
56
}
else
58
    c = c - pomeraj;

60
    return c;
}
62
else
{
64
    if (isdigit(c))
    {
66
        pomeraj = s.b % 10;

68
        if (pomeraj > '9' - c)
        {
70
            pomeraj = pomeraj - ('9' - c) - 1;
            c = '0' + pomeraj;
72
        }
        else
74
            c = c + pomeraj;

76
        return c;
    }
78
    pomeraj = s.b % 26;

80
    if (islower(c))
    {
82
        if (pomeraj > 'z' - c)
        {
84
            pomeraj = pomeraj - ('z' - c) - 1;
            c = 'a' + pomeraj;
86
        }
        else
88
            c = c + pomeraj;

90
        return c;
    }
92
}

94
if (pomeraj > 'Z' - c)
{
96
    pomeraj = pomeraj - ('Z' - c) - 1;
    c = 'A' + pomeraj;
98
}
```

```
100         else
101             c = c + pomeraaj;
102
103         return c;
104     }
105 }
106
107 int main()
108 {
109     char linija[BROJ_KARAKTERA];
110     Sifra sifre[BROJ_SIFRI];
111     char c;
112     int n = 0, j = 0, i;
113
114     printf("Uneti sifre u obliku: broj, smer:\n");
115     while (scanf("%d %c", &sifre[n].b, &sifre[n].smer) != EOF)
116     {
117         if (sifre[n].smer != 'L' && sifre[n].smer != 'D')
118         {
119             printf("Neispravan smer.\n");
120             return -1;
121         }
122
123         if (sifre[n].b < 0)
124         {
125             printf("Neispravan broj za sifrovanje.\n");
126             return -1;
127         }
128
129         n++;
130         if (n == BROJ_SIFRI)
131             break;
132     }
133
134     printf("Uneti tekst za sifrovanje:\n");
135     while ((c = getchar()) != EOF)
136     {
137         for (i=0; i<n; i++)
138             c = sifruj(c, sifre[i]);
139
140         linija[j] = c;
141         j++;
142
143         if (j == BROJ_KARAKTERA)
144             break;
145     }
146
147     linija[j] = 0;
148
149     printf("%s\n", linija);
150
151     return 0;
```

```
}

```

### Rešenje 3.9.13

```

1  #include <stdio.h>
2  #include <math.h>
3
4  #define MAX 50
5
6  typedef struct lopta {
7      int poluprecnik;
8      enum {plava, zuta, crvena, zelena} boja;
9  } LOPTA;
10
11 float zapremina(LOPTA l) {
12     return pow(l.poluprecnik, 3)*4/3*M_PI;
13 }
14
15 float ukupna_zapremina(LOPTA lopte[], int n) {
16
17     int i;
18     float z = 0;
19
20     for(i = 0; i < n; i++)
21         z += zapremina(lopte[i]);
22
23     return z;
24 }
25
26 /*
27  Funkcija je opstija od trazene i broji sve lopte odredjene boje u
28  nizu lopti.
29  U zavisnosti od prosledjene boje funkciji, funkcija vraca
30  odgovarajuci broj.
31 */
32 int broj_lopti_u_boji(LOPTA lopte[], int n, int boja) {
33
34     int br = 0;
35     int i;
36     for(i = 0; i < n; i++)
37         if(lopte[i].boja == boja)
38             br++;
39     return br;
40 }
41
42 int main() {
43     LOPTA lopte[MAX];
44     int n;
45     int i;

```

### 3 Predstavljanje podataka

```
47     int boja;
49     printf("Unesite broj lopti: ");
51     scanf("%d", &n);
53     if(n < 1 || n > MAX) {
55         printf("Nekorektan unos.\n");
57         return 0;
59     }
61     printf("Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta,
63         3-crvena, 4-zelena):\n");
65     for(i = 0; i < n; i++) {
67         printf("%d. lopta: ", i+1);
69         scanf("%d%d", &lopte[i].poluprecnik, &boja);
71         /* U zavisnosti od unetog celog broja,
73         bira se boja lopte.
75         */
77         switch(boja) {
79             case 1: lopte[i].boja = plava; break;
81             case 2: lopte[i].boja = zuta; break;
83             case 3: lopte[i].boja = crvena; break;
85             case 4: lopte[i].boja = zelena; break;
87             default:
89                 printf("Nekorektan unos.\n");
91                 return 0;
93         }
95     }
97     printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));
99     printf("Ukupno crvenih lopti: %d\n", broj_lopti_u_boji(lopte, n,
101         crvena));
103     return 0;
105 }
```

#### Rešenje 3.9.14

```
1 #include <stdio.h>
2 #include <math.h>
4 #define MAX_TACAKA 1000
6 typedef struct
7 {
8     int x, y;
```

```
10 }TACKA;
12 double rastojanje(TACKA a, TACKA b)
13 {
14     return sqrt(pow(a.x - b.x, 2) + pow(a.y - b.y, 2));
15 }
16 unsigned učitaj_poligon(TACKA* tacke, unsigned n)
17 {
18     int i = 0;
19
20     while(i < n && scanf("%d%d", &tacke[i].x, &tacke[i].y) != EOF)
21         i++;
22
23     return i;
24 }
25
26 double obim(TACKA* poligon, unsigned n)
27 {
28     double o = rastojanje(poligon[0], poligon[n-1]);
29     int i;
30
31     for(i=0; i<n-1; i++)
32         o += rastojanje(poligon[i], poligon[i+1]);
33
34     return o;
35 }
36
37 double maksimalna_stranica(TACKA* poligon, unsigned n)
38 {
39     double max = rastojanje(poligon[0], poligon[n-1]);
40     double stranica;
41     int i;
42
43     for(i=0; i<n-1; i++)
44     {
45         stranica = rastojanje(poligon[i], poligon[i+1]);
46         if (stranica > max)
47             max = stranica;
48     }
49
50     return max;
51 }
52
53 double površina_trougla(TACKA A, TACKA B, TACKA C)
54 {
55     double a = rastojanje(B, C);
56     double b = rastojanje(A, C);
57     double c = rastojanje(A, B);
58
59     double s = (a + b + c)/2;
60 }
```

```
        return sqrt(s*(s - a)*(s - b)*(s - c));
62 }

64 double povrsina(TACKA* poligon, unsigned n)
65 {
66     double P = 0;
67     int i;
68
69     for(i=1; i<n-1; i++)
70         P += povrsina_trougla(poligon[0], poligon[i], poligon[i+1]);
71
72     return P;
73 }
74
75 int main()
76 {
77     int N;
78     unsigned m;
79     TACKA poligon[MAX_TACKA];
80
81     printf("Uneti maksimalan broj tacaka poligona: ");
82     scanf("%d", &N);
83
84     if (N < 3 || N > MAX_TACKA)
85     {
86         printf("Neispravan broj tacaka poligona.\n");
87         return -1;
88     }
89
90     m = ucitaj_poligon(poligon, N);
91
92     if (m < 3)
93     {
94         printf("Neispravan broj tacaka poligona.\n");
95         return -1;
96     }
97
98     printf("Obim poligona je %.3lf.\n", obim(poligon, m));
99     printf("Duzina maksimalne stranice je %.3lf.\n",
100           maksimalna_stranica(poligon, m));
101     printf("Povrsina poligona je %.3lf.\n", povrsina(poligon, m));
102
103     return 0;
104 }
```

#### Rešenje 3.9.15

```
1 #include <stdio.h>
2 #define MAX 1000
3
4 typedef struct
```



```

5 {
    char o;
7   int x;
    int y;
9 } IZRAZ;

11
12 int korektan_izraz(const IZRAZ izraz)
13 {
    if(izraz.o != '+' && izraz.o != '-' && izraz.o != '*' && izraz.o
14       != '/')
15     {
        printf("Nedozvoljena operacija!\n");
16         return 0;
17     }
18     if(izraz.o == '/' && izraz.y == 0)
19     {
        printf("Deljenje nulom!\n");
20         return 0;
21     }
22     return 1;
23 }
24 }

25
26 /* Racunanje vrednosti izraza. */
27 int vrednost(const IZRAZ izraz)
28 {
29     int v;
30
31     switch (izraz.o)
32     {
33         case '+':
34             return izraz.x + izraz.y;
35         case '-':
36             return izraz.x - izraz.y;
37         case '*':
38             return izraz.x * izraz.y;
39         case '/':
40             return izraz.x / izraz.y;
41     }
42 }
43 }

44
45 /*
46  Promenljiva izraz ce se promeniti u funkciji
47  ucitaj_izraz tako sto ce njenim neinicijalizovanim
48  poljima o,x,y biti dodeljene vrednosti ucitane
49  sa ulaza. Zbog toga ovu promenljivu
50  funkciji prosledjujemo po adresi, preko pokazivaca.
51
52  S obzirom da ucitavanje karaktera nije prvo
53  ucitavanje koje se obavlja u programu, funkcijom
54  getchar() se ucita karakter kojim se razdvaja

```

### 3 Predstavljanje podataka

---

```
unos karaktera od prethodnog unosa (najcesce blanko
57 znak ili znak za novi red).

59 */

61 int ucitaj_izraz(IZRAZ* izraz)
{
63     getchar();
    scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
65     if (!korektan_izraz(*izraz))
        return 0;
67     return 1;
}

69

71 void stampaj_izraz(const IZRAZ izraz)
{
73     printf("%d %c %d = %d\n", izraz.x, izraz.o, izraz.y, vrednost(
        izraz));
}

75
77 int max_vr(IZRAZ izrazi[], int n)
{
79     int i;
    int max;

81     max=vrednost(izrazi[0]);

83     for(i=1; i<n; i++)
        if(vrednost(izrazi[i])>max)
85             max=vrednost(izrazi[i]);

87     return max;
}

89
91 int main()
{
93     int n;
    IZRAZ izrazi[MAX];
    int max;
    int i;

95

97     printf("Uneti broj izraza: ");
    scanf("%d", &n);
99     if(n<0 || n>MAX)
    {
101         printf("Nekorektna vrednost broja n!\n");
        return -1;
103     }

105     printf("Uneti izraze u prefiksnoj notaciji:\n");
```

```

107     for(i=0; i<n; i++)
109         if(ucitaj_izraz(&izrazi[i])==0)
111             {
113                 printf("Nekorektan unos\n");
115                 return -1;
117             }

119     max = max_vr(izrazi, n);
121     printf("Maksimalna vrednost izraza:%d\n", max);

123     printf("Izrazi cija je vrednost manja od polovine maksimalne
125     vrednosti:\n");

127     for(i=0; i<n; i++)
129         if(vrednost(izrazi[i])<max/2)
131             stampaj_izraz(izrazi[i]);

133     return 0;
135 }

```

### Rešenje 3.9.16

```

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

#define STEPEN 10
#define MAX_POLINOMA 100

typedef struct
{
    int stepen;
    float koef[STEPEN+1];
}_polinom;

int ucitaj(_polinom* p)
{
    int i;

    printf("Uneti stepen: ");
    if (scanf("%d", &(p->stepen)) == EOF)
        return 0;

    if (p->stepen > STEPEN || p->stepen < 0)
    {
        printf("Greska u unosu stepena.\n");
        exit(EXIT_FAILURE);
    }

    printf("Uneti koeficijente polinoma:\n");
    for(i=0; i<=p->stepen; i++)

```

```
        scanf("%f", &(p->koef[i]));
30
    return 1;
32 }

34 int ispis_prvog_monoma(float koef, int stepen)
{
36     if (koef != 0)
    {
38         printf("%.2f", koef);

40         if (stepen == 1)
            printf("*x ");
42         else if (stepen > 1)
            printf("*x^%d ", stepen);
44
        return 1;
46     }
    else
48         return 0;
}

50 void ispis_monoma(float koef, int stepen)
52 {
    if (koef != 0)
54     {
        if (koef > 0)
56         printf("+ ");

58         printf("%.2f", koef);

60         if (stepen == 1)
            printf("*x ");
62         else if (stepen > 1)
            printf("*x^%d ", stepen);
64     }
}

66 void ispis(const _polinom *p)
68 {
    int prvi = 1;
70     int i;

72     for(i=0; i <= p->stepen; i++)
        if (prvi)
74         {
            prvi = !ispis_prvog_monoma(p->koef[i], i);
76         }
        else
78         ispis_monoma(p->koef[i], i);

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

```

}
82
void integral(const _polinom* p, _polinom* integ)
84 {
    int i;
86
    integ->stepen = p->stepen + 1;
88
    integ->koef[0] = 0;
90
    for(i=1; i <= integ->stepen; i++)
92         integ->koef[i] = (float)p->koef[i-1]/i;
94 }
96
int main()
{
98     _polinom p[MAX_POLINOMA], integ;
    int i = 0, j;
100
    while(ucitaj(&p[i]))
102         i++;
104
    printf("\n\nIntegrali su:\n");
    for(j=0; j<i; j++)
106     {
        integral(&p[j], &integ);
108         ispis(&integ);
    }
110
    return 0;
112 }
```



## 4

# Ulaz i izlaz programa

## 4.1 Datoteke

**Zadatak 4.1.1** Napisati program koji prepisuje sadržaj datoteke `ulaz.txt` u datoteku `izlaz.txt` karakter po karakter.

[Rešenje [4.1.1](#)]

**Zadatak 4.1.2** Napisati program koji u datoteci cije se ime navodi kao prvi argument komandne linije odredjuje liniju maksimalne duzine i ispisuje je na standardni izlaz. Ukoliko ima vise takvih linija , ispisati onu koja je leksikografski prva. Mozemo pretpostaviti da datoteka ne sadrzi linije duze od 80 karaktera.

[Rešenje [4.1.2](#)]

**Zadatak 4.1.3** U datoteci cije se ime zadaje kao prvi argument komandne linije nalazi se prirodan broj  $n$  a zatim  $i$   $n$  celih brojeva. Napisati program koji prebrojava koliko  $k$ -tocifrenih brojeva postoji u datoteci , pri cemu se prirodan broj  $k$  zadaje kao drugi argument komandne linije.

[Rešenje [4.1.3](#)]

**Zadatak 4.1.4** U datoteci cije se ime navodi kao prvi argument komandne linije navedena je rec  $r$  i niz linija. Napisati program koji u datoteku cije se ime navodi kao drugi argument komandne linije upisuje sve linije u kojima se rec  $r$  pojavljuje bar  $n$  puta , gde je  $n$  prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu `broj_pojavljivanja : linija`.

[Rešenje 4.1.4]

**Zadatak 4.1.5** Program se pokrece tako sto se navedu nazivi dve datoteke (ulazna i izlazna) i opcije. U datoteci cije se ime navodi kao prvi argument komandne linije nalaze se podaci o razlomcima: u prvom redu se nalazi broj razlomaka, a u svakom sledecem redu brojilac i imenilac jednog razlomka. Potrebno je kreirati strukturu koja opisuje razlomak i učitati niz razlomaka iz datoteke, a potom:

a) ukoliko je navedena opcija x, upisati u datoteku cije je ime drugi argument komandne linije recipročni razlomak za svaki razlomak iz niza (npr. za  $2/3$  treba upisati  $3/2$ )

b) ukoliko je navedena opcija y, upisati u datoteku cije je ime drugi argument komandne linije realnu vrednost recipročnog razlomka svakog razlomka iz niza (npr. za  $2/3$  treba upisati 1.5)

Mozemo pretpostaviti da se u datoteci sa podacima o razlomcima nalazi najviše 100 razlomaka.

[Rešenje 4.1.5]

**Zadatak 4.1.6** Za svaki automobil poznati su marka, model i cena. Iz datoteke cije se ime zadaje sa standardnog ulaza učitava se broj automobila a potom i podaci za svaki automobil. Program treba da:

a) izracuna prosečnu cenu po marki kola

b) za maksimalnu cenu koju je kupac spreman da plati, a koja se zadaje kao argument komandne linije, da ispise automobile u tom cenovnom rangi zajedno sa prosečnom cenom odgovarajuće marke

Mozemo pretpostaviti da se model i marka sastoje od jedne reci i da svaka od njih sadrži najviše 30 karaktera kao i da se u datoteci nalaze podaci za najviše 100 automobila.

[Rešenje 4.1.6]

**Zadatak 4.1.7** Napisati program koji prebrojava mala slova u datoteci *test.txt*.

*Primer 1*

```
TEST.TXT
Abcd EFGH+ijKLMN

IZLAZ:
Broj malih slova je: 5
```

*Primer 2*

```
TEST.TXT
PrograMiranje

IZLAZ:
Broj malih slova je: 11
```



[Rešenje 4.1.7]

**Zadatak 4.1.8** Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*.

*Primer 1*

```

|| ULAZ.TXT
||   Volim programiranje.
|| IZLAZ.TXT
||   Vipgmae

```

[Rešenje 4.1.8]

**Zadatak 4.1.9** Kao argumenti komandne linije se zadaju ime datoteke i ceo broj  $k$ . Napisati program koji na standardni izlaz ispisuje sve linije zadate datoteke čija je dužina veća od  $k$ . Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera.

*Primer 1*

```

|| POKRETANJE: ./a.out test.txt 7
|| TEST.TXT
||   Teme koje su obradjuivane:
||   Petlje
||   Funkcije
||   Nizovi
||   Strukture
||
|| IZLAZ:
||   Teme koje su obradjuivane:
||   Funkcije
||   Strukture

```

*Primer 2*

```

|| POKRETANJE: ./a.out test.txt
||
|| IZLAZ:
||   Greska: Pogresan broj argumenata!

```

[Rešenje 4.1.9]

**Zadatak 4.1.10** Napisati program koji prebrojava koliko se linija datoteke *ulaz.txt* završava niskom  $s$  koja se učitava sa standardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera, kao i da dužina niske  $s$  neće biti veća od 20 karaktera.

## 4 Ulaz i izlaz programa

---

### Primer 1

```
ULAZ.TXT
abcde abcde
abcde aab
abcde abcde abcde
abcde abcde Aab
abcde abcde ab
abcde abcde abcde abcde

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 3
```

### Primer 2

```
ULAZ.TXT
abcde abcde
abcde
abcde abcde AB

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 0
```

[Rešenje 4.1.10]

**Zadatak 4.1.11** Napisati program koji pronalazi maksimum brojeva zapisanih u datoteci *brojevi.txt*.

### Primer 1

```
BROJEVI.TXT
2.36 -16.11 5.96 8.88
-265.31 54.96 38.4

IZLAZ:
Najveci broj je: 54.96
```

[Rešenje 4.1.11]

**Zadatak 4.1.12** U datoteci *studenti.txt* se nalaze informacije o studentima: prvo broj studenata, a zatim u pojedinačnim linijama korisničko ime i pet poslednjih ocena koje je student dobio. Napisati program koji pronalazi studenta koji je ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da broj studenata neće biti veći od 100.

### Primer 1

```
STUDENTI.TXT
mr15239 10 9 9 8 10
mi14005 8 8 9 8 10
ml15112 9 8 8 7 10
mr15007 10 10 10 10 10
mn13208 7 7 9 6 10

IZLAZ:
korisnicko ime: mr15007, prosek ocena: 10.00
```

[Rešenje 4.1.12]

**Zadatak 4.1.13** U datoteci *tacke.txt* se nalazi prvo broj tačaka, a zatim u pojedinačnim linijama  $x$  i  $y$  koordinate tačke. Napisati program koji u datoteku *rastojanja.txt* upisuje rastojanje svake od pročitanih tačaka od koordinatnog početka, a na standardni izlaz koordinate tačke koja je najudaljenija. Koristiti strukturu *Tacka* sa poljima  $x$  i  $y$ , kao i funkciju kojom se računa rastojanje. Pretpostaviti da broj tačaka u datoteci neće biti veći od 50.

*Primer 1*

```
TACKE.TXT
4
11 -2
3 5
8 -8
0 4

RASTOJANJA.TXT
11.18
5.29
11.31
4.00

IZLAZ:
Najudaljenija je tačka: 8 -8
```

*Primer 1*

```
TACKE.TXT
-2
0 0
9 -8

IZLAZ:
Greska: Nedozvoljen broj tacaka!
```

[Rešenje 4.1.13]

**Zadatak 4.1.14** Napisati program koji za reč  $s$  maksimalne dužine 20 karaktera koja se zadaje sa standardnog ulaza u datoteku *rotacije.txt* upisuje sve rotacije reči  $s$ .

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite rec: abcde

ROTACIJE.TXT
abcde
bcdea
cdeab
deabc
eabcd
```

[Rešenje 4.1.14]

**Zadatak 4.1.15** Napisati program koji linije koji se učitavaju sa standardnog ulaza sve do kraja ulaza prepisuje u datoteku *izlaz.txt* i to, ako je prilikom pokretanja zadata opcija  $-v$  ili  $-V$  samo one linije koje počinju velikim slovom, ako je zadata opcija  $-m$  ili  $-M$  samo one linije koje počinju malim slovom, a ako

## 4 Ulaz i izlaz programa

---

je opcija izostavljena sve linije. Pretpostaviti da linije neće biti duže od 80 karaktera.

### Primer 1

```
POKRETANJE: ./a.out -m
INTERAKCIJA SA PROGRAMOM:
Unesite recenice:
programiranje u C-u je zanimljivo
Volim programiranje!
Kada porastem bicu programer!
u slobodno vreme programiram

IZLAZ.TXT
programiranje u C-u je zanimljivo
u slobodno vreme programiram
```

### Primer 2

```
POKRETANJE: ./a.out -V
INTERAKCIJA SA PROGRAMOM:
Unesite recenice:
programiranje u C-u je zanimljivo
Volim programiranje!
Kada porastem bicu programer!
u slobodno vreme programiram

IZLAZ.TXT
Volim programiranje!
Kada porastem bicu programer!
```

### Primer 3

```
POKRETANJE: ./a.out -k
INTERAKCIJA SA PROGRAMOM:
Greska: Pogresno pokretanje programa!
```

[Rešenje 4.1.15]

**Zadatak 4.1.16** Sa standardnog ulaza učitavaju se imena dve tekstualne datoteke i jedan karakter. Napisati program koji prepisuje datoteku čije se ime navodi kao prvo u datoteku čije ime se navodi kao drugo. Ukoliko je učitani karakter u program prilikom prepisivanja treba da zamenjuje sva mala slova velikim, a ukoliko je učitani karakter 1 sva velika slova se zamenjuju malim. U slučaju greske ispisati -1. Greška može biti neuspešno otvaranje datoteke ili pogrešno zadat karakter. Maksimalna dužina naziva datoteka je 20 karaktera.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
ulaz.txt izlaz.txt u
ULAZ.TXT
danas je lep dan
i Ja zelim
da postanem programer
IZLAZ.TXT
DANAS JE LEP DAN
I JA ZELIM
DA POSTANEM PROGRAMER
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
prva.dat druga.dat l
PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120
DRUGA.DAT
cena soka je 30
cena vina je 150
cena limunade je 200
cena sendvica je 120
```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
  primer.c prazna.txt V
PRIMER.C
#include <stdio.h>
int main()
{
}
PRAZNA.TXT

IZLAZ:
-1

```

[Rešenje 4.1.33]

**Zadatak 4.1.17** Sastaviti program koji sa standardnog ulaza prima ime datoteke koju treba otvoriti. Ispisati (na standardnom izlazu) koja cifra (meu svim ciframa koje se pojavljuju u datoteci) ima najveći broj pojavljivanja. U slučaju greške pri otvaranju datoteke ispisati -1. Ukoliko nema cifara u datoteci ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

*Primer 1*

```

INTERAKCIJA SA PROGRAMOM:
  ulaz.txt
ULAZ.TXT
  danas je lep dan
  i Ja zelim
  da postanem programer
IZLAZ:
-1

```

*Primer 2*

```

INTERAKCIJA SA PROGRAMOM:
  prva.dat druga.dat l
PRVA.DAT
  Cena soka je 30
  Cena vina je 150
  Cena limunade je 200
  Cena sendvica je 120
IZLAZ:
0

```

*Primer 3*

```

INTERAKCIJA SA PROGRAMOM:
  primer.c
PRIMER.C
#include <stdio.h>
int main()
{
}
PRAZNA.TXT

IZLAZ:
-1

```

[Rešenje 4.1.33]

**Zadatak 4.1.18** Prvi red datoteke `matrice.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice A. Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji pronalazi sve elemente matrice A koji su jednaki zbiru svih svojih susednih elemenata i štampa ih u obliku

(broj vrste, broj kolone, vrednost elementa).

U slučaju greške prilikom otvaranja datoteke ispisati -1. Pretpostaviti da je sadržaj datoteke ispravan.

### Primer 1

```
MATRICE.TXT
1 2 3 4
7 2 15 -3
-1 3 1 3
IZLAZ:
(1, 0, 7)
(1, 2, 15)
```

[Rešenje 4.1.33]

**Zadatak 4.1.19** Napisati program koji za dve datoteke čija su imena data kao prvi i drugo na standardnom ulazu, radi sledeće: za cifru u prvoj datoteci, u drugu datoteku se upisuje 0, za slovo se upisuje 1, a za sve ostale karaktere se upisuje 2. Maksimalna dužina naziva datoteka je 20 karaktera.

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
prva.dat druga.dat
PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120
DRUGA.DAT
11112111121120021111211112112000211112111111112112000211112111111112112000
```

[Rešenje 4.1.33]

**Zadatak 4.1.20** Ako je data tekstualna datoteka `plain.txt` napraviti tekstualnu datoteku `sifra.txt` tako što se svako slovo zamenjuje svojim prethodnikom (ciklično) suprotne velicine 'b' sa 'A', 'B' sa 'a', 'a' sa 'Z', 'A' sa 'z', itd. Podrazumevati da se na sistemu koristi tabela karaktera ASCII.

[Rešenje 4.1.33]

**Zadatak 4.1.21** Sa standardnog ulaza se učitava ime tekstualne datoteke i prirodan broj  $k$ . Podrazumeva se da zadata datoteka sadrži samo slova i beline i da je svaka reč iz datoteke dužine najviše 100. Program treba da učitava reči iz datoteke, da svaku reč rotira za  $k$  mesta i da tako dobijenu reč upiše u datoteku čije je ime `rotirano.txt`. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.1.33]

**Zadatak 4.1.22** Napisati program koji u datoteku `izlaz.txt` prepisuje sve reči iz datoteke `ulaz.txt` čiji je zbir ascii kodova slova strogo veći od 1000. Reči su odvojene prazninama i nisu duže od 200 karaktera.

*Primer 1*

```

|| ULAZ.TXT
|| Sa standardnog ulaza unosi se neoznaceni
|| ceo broj. Formirati novi broj koji se dobija
|| izbacivanjem svake druge cifre iz polaznog
|| broja.
|| IZLAZ.TXT
|| standardnog izbacivanjem

```

*Primer 2*

```

|| ULAZ.TXT
|| i sada jedan kratak primer
|| p1: 1234567890
|| p2: ABCDEFGHIJ
|| p3: abcdefghij
|| IZLAZ.TXT
|| abcdefghij

```

*Primer 3*

```

|| ULAZ.TXT
|| konstruisanje test-primera sa
|| i dugackim recima kao prestolonaslednik
|| brojevima1234567890
|| IZLAZ.TXT
|| konstruisanje test-primera
|| prestolonaslednik
|| brojevima1234567890

```

*Primer 4*

```

|| ULAZ.TXT
|| ima jos dugackih reci: predskazanje,
|| potom
|| nelogicnosti, zanemarivati, odugovlaciti, a ima
|| i i malih reci koje su kratke
|| predosecaj
|| IZLAZ.TXT
|| predskazanje, nelogicnosti,
|| zanemarivati, odugovlaciti,
|| predosecaj

```

[Rešenje 4.1.33]

**Zadatak 4.1.23** U datoteci `razno.txt` nalazi se tekst. U datoteku `palindromi.txt` prepisati sve reči iz datoteke `razno.txt` koje su palindromi. Reč je palindrom ako se čita isto sa leve i desne strane. Za reč smatramo niz karaktera koji se nalazi između belina i koji nije duži od 200 karaktera. Dozvoljeno je korišćenje specifikatora za čitanje reči. Maksimalan broj reči nije poznat. U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

### Primer 1

```
RAZNO.TXT
Ana i melem su primeri palindroma.
PALINDROMI.TXT:
Ana i melem
```

### Primer 2

```
RAZNO.TXT
jabuka neven pomorandza kuk
Oko kapAk pero radar caj
PALINDROMI.TXT:
neven kuk\datoteka{Oko kapAk radar}
```

[Rešenje 4.1.33]

**Zadatak 4.1.24** U datoteci čije se ime navodi na standardnom ulazu programa nalazi se broj  $n$ , a zatim i  $n$  reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

- (a) ispisuje ga [3],
- (b) iz njega uklanja sve duplikate i u datoteku `rez.txt` ispisuje transformisani niz [4]

U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
dat1.txt
DAT1.TXT
12 jha14 hahaha deda mraz deda
mraz deda deda jase konj konj konj
IZLAZ:
jha14 hahaha deda mraz deda mraz deda
deda jase konj konj konj
REZ.TXT:
jha14 hahaha deda mraz jase konj
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
dat2.txt
DAT2.TXT
14
so secer supa so ljuto secer kiselo slatko
ljuto
paprika, ljuta paprika, ljuto dete
IZLAZ:
so secer supa so ljuto secer kiselo slatko
ljuto paprika, ljuta paprika, ljuto dete
REZ.TXT:
so secer supa ljuto kiselo slatko
paprika, ljuta dete
```

[Rešenje 4.1.33]

**Zadatak 4.1.25** U datoteci čije se ime navodi na standardnom ulazu programa nalazi se broj  $n$ , a zatim i  $n$  reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

- (a) ispisuje ga, [3]
- (b) u datoteku `rez.txt` upisuje sve reči koje sadrže prvu reč i podvlaku. [4]



U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT
  7 rec Opet _rec Reci rec_enica
  Dva recica_
IZLAZ:
  rec Opet _rec Reci rec_enica
  Dva recica_
REZ.TXT:
  _rec rec_enica recica_
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  dat2.txt
DAT2.TXT
  11 Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
  suncanica.
IZLAZ:
  Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
  suncanica.
REZ.TXT:
  Sunce_Moje Sunce123_123
```

[Rešenje 4.1.33]

**Zadatak 4.1.26** Imena dve datoteke se zadaje na standardnom ulazu. U prvoj datoteci navedena je rec **r** i niz linija. Napisati program koji u drugu datoteku upisuje sve linije u kojima se reč **r** pojavljuje bar **n** puta, gde je **n** prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu broj\_pojavljivanja: linija. Linije brojati počevši od 1. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.1.33]

**Zadatak 4.1.27** Napisati program koji poredi dva fajla i ispisuje redni broj linija u kojima se fajlovi razlikuju. Imena fajlova se zadaju kao argumenti komandne linije. U slučaju neuspešnog otvaranja datoteka ispisati poruku o grešci. Pretpostaviti da je maksimalna dužina reda u datoteci 200 karaktera. Ukoliko nisu zadati potrebni argumenti komadne linije ispisati poruku o grešci. Linije brojati počevši od 1.

#### Primer 1

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt
ULAZ.TXT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ.TXT:
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ:
```

#### Primer 2

```
POKRETANJE: ./a.out primer1.dat primer2.dat
PRIMER1.DAT
  danas vezbamo
  analizu
  ovo je primer kad
  su datoteke razlicite
PRIMER2.DAT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke razlicite
IZLAZ:
  2 3 4
```

### Primer 3

```
POKRETANJE: ./a.out prva.dat
IZLAZ:
greska
```

### Primer 2

```
POKRETANJE: ./a.out prva.dat druga.dat
PRVA.DAT
ovo je primer
kada su
datoteke
razlicite duzine
DRUGA.DAT
kada su
programiranje
datoteke
razlicite
duzine
i kada treba ispisati broj
tih redova
IZLAZ:
1 4 5 6 7
```

[Rešenje 4.1.33]

### Zadatak 4.1.28 Definirati strukturu

```
typedef struct{
    unsigned int a, b;
    char ime[5];
}_pravougaonik;
```

kojom se opisuje pravougaonik dužinama svojih stranica i imenom. Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava pravougaonike (nepoznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine među pravougaonikima koji nisu kvadrati. U slučaju unosa nekorektnih dužina stranica pravougaonika ili nekorektnih vrednosti broja *n*, ispisati -1 i odmah prekinuti izvršavanje programa. Maksimalan broj pravougaonika je 200.

### Primer 1

```
POKRETANJE: ./a.out pravougaonici.dat
PRAVOUGAONICI.DAT
2 4 p1
3 3 p2
1 6 p3
IZLAZ:
p2 8
```

### Primer 2

```
POKRETANJE: ./a.out dva.dat
DVA.DAT
5 2 pm
4 7 pv
IZLAZ:
28
```

*Primer 3*

```

|| POKRETANJE: ./a.out tri.dat
|| TRI.DAT
|| 5 5 m
|| 3 3 s
|| 8 8 xl
|| IZLAZ:
|| m s xl

```

*Primer 4*

```

|| POKRETANJE: ./a.out primerx.dat
|| PRIMERX.DAT
|| 9 7 p
|| IZLAZ:
|| 63

```

*Primer 5*

```

|| POKRETANJE: ./a.out prazna.dat
|| PRAZNA.DAT
|| IZLAZ:

```

[Rešenje 4.1.33]

**Zadatak 4.1.29** Ime datoteke dato je kao argument komandne linije. U datoteci se nalaze otvorene i zatvorene zagrade i još nekakav tekst. Proveriti da li su zagrade pravilno uparene. Npr. `ab( cd) ..` odgovor je `jesu`, a `..)ba()` odgovor je `nisu`. Ukoliko nisu zadati svi argumenti komadne linije ispisati poruku o grešci.

*Primer 1*

```

|| POKRETANJE: ./a.out
|| zagrade.txt
|| ZAGRADE.TXT
|| ab( cd) ..
|| ((3+4)*5+1)*9
|| IZLAZ:
|| jesu

```

*Primer 2*

```

|| POKRETANJE: ./a.out
|| primer2.dat
|| PRIMER2.DAT
|| (7+8
|| nisu(
|| uparene
|| IZLAZ:
|| nisu

```

*Primer 3*

```

|| POKRETANJE: ./a.out
|| primer3.dat
|| PRIMER3.DAT
|| )) 7 + 6 ((
|| IZLAZ:
|| nisu

```

*Primer 4*

```

|| POKRETANJE: ./a.out
|| IZLAZ:
|| greska

```

[Rešenje 4.1.33]

**Zadatak 4.1.30** Napraviti strukturu `STUDENT` koja sadrži:

- `ime` (u polju se čuva ime i prezime studenta, napr. "Marko Markovic", maksimalna dužina polja je 100 karaktera),

- `oc` (sadrži najviše 10 ocena studenta)
- `br_ocena` (ukupan broj ocena za studenata)
- `pr_oc` (prosečna ocena)

U datoteci se nalaze podaci o studentima. Za svakog studenta unosi se ime i prezime razdvojeno razmakom (uputstvo: može se korisiti `strcat` da spoji ime i prezime koji se mogu pročitati sa specifikatorom `%s`), a potom ocene koje se završavaju sa 0. Pronaći studenta koji ima najveći prosek i ispisati sve njegove podatke (prosek ispisati na 2 decimale). Maksimalan broj studenta je 100. Ime datoteke se zadaje kao argument komandne linije.

<i>Primer 1</i>	<i>Primer 2</i>
<pre>   POKRETANJE: ./a.out       studenti.txt    STUDENTI.TXT       Marko Markovic 5 6 7 8 9 0       Jelena Jankovic 10 10 10 0       Filip Viskovic 10 9 8 7 6 0       Jana Peric 10 10 9 9 8 8 7 7       0    IZLAZ:       Jelena Jankovic 10 10 10 0       10.00</pre>	<pre>   POKRETANJE: ./a.out       IZLAZ:       greska</pre>

[Rešenje 4.1.33]

### Zadatak 4.1.31

- Napisati C funkciju `int unesiSkup(char *s, FILE* f)` kojom se unosi skup elemenata iz datoteke `F`. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiđe na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspešno učitani.
- Napisati funkciju `void prebroj(char *s, int *br_slova, int *br_cifara)` kojom se određuje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.
- Napisati glavni program gde se unose podaci o skupu elemenata. Ime datoteke se zadaje kao argument komandne linije. Na standardni izlaz ispisati informacije o broju slova i cifara (koristiti funkcije pod a) i b)).

*Primer 1*

```
POKRETANJE: ./a.out skup.txt
SKUP.TXT
  abc56ighj9012hjFGHH
IZLAZ:
  broj slova: 13
  broj cifara: 6
```

*Primer 2*

```
POKRETANJE: ./a.out skup2.txt
SKUP2.TXT
  ovdeimamo$dolar
IZLAZ:
  broj slova: 9
  broj cifara: 0
```

*Primer 3*

```
POKRETANJE: ./a.out skup3.txt
SKUP3.TXT
  broJ3
  broj5
IZLAZ:
  broj slova: 4
  broj cifara: 1
```

*Primer 4*

```
POKRETANJE: ./a.out
IZLAZ:
  greska
```

[Rešenje 4.1.33]

**Zadatak 4.1.32** Definirati strukturu

```
typedef struct{
    int x;
    int y;
    int z;
} vektor;
```

kojom se opisuje trodimenzioni vektor. U datoteci `vektori.txt` nalazi se nepoznati broj vektora (maksimalno ih može biti 200). Učitati ih u niz i ispisuje na standardnom izlazu koordinate vektora sa najvećom dužinom. Dužina vektora se izračunava po formuli:

$$|v| = \sqrt{x^2 + y^2 + z^2}$$

U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

*Primer 1*

```
VEKTORI.TXT
  2
  4 -1 7
  3 1 2
IZLAZ:
  4 -1 7
```

*Primer 2*

```
VEKTORI.TXT
  67
IZLAZ:
  -1
```

*Primer 3*

```
VEKTORI.TXT
  3
  0 0 0
  0 1 0
  1 0 0
IZLAZ:
  0 1 0
```

### Primer 4

```
VEKTORI.TXT
4
3 0 1
4 5 2
1 0 0
2 -1 2
IZLAZ:
4 5 2
```

[Rešenje 4.1.33]

**Zadatak 4.1.33** Prvi red datoteke `ulaz.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice A. Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji nalazi i štampa sve četvorke oblika  $(A(i,j), A(i+1,j), A(i,j+1), A(i+1,j+1))$  u kojima su svi elementi međusobno različiti.

[Rešenje 4.1.33]

## 4.2 Rešenja

### Rešenje 4.1.1

```
2  /*
   Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
   datoteku izlaz.txt karakter po karakter.
4  */
6  #include <stdio.h>
   #include <stdlib.h>
8
10 int main()
   {
12     int c;
       FILE *ulaz, *izlaz;
14
       /*
16         Promenljive ulaz i izlaz predstavljaju
           pokazivace na ugradjenu strukturu FILE.
           Unutar ove strukture nalaze se polja neophodna
18         za rad sa datotekama.
20
           Kada zelimo da radimo sa nekom datotekom,
```

```
22     moramo je prvo otvoriti. Ugradjena funkcija
23     fopen(dat, mode) otvara datoteku sa nazivom
24     dat. Datoteka moze biti otvorena za citanje,
25     pisanje ili nadovezivanje, sto odredjuje
26     argument mode koji moze imati vrednost "r" (read),
27     "w"(write) ili "a"(append).
28 */
29
30     ulaz=fopen("ulaz.txt","r");
31
32     /*
33     Do neuspesnog otvaranja datoteke moze doci
34     ukoliko ne postoji datoteka sa datim nazivom
35     ili je putanja do datoteke pogresna. U tom
36     slucaju, funkcija fopen vraca pokazivac na NULL
37     i tada treba prijaviti gresku. Datoteka stderr
38     predstavlja standardnu datoteku u koju se upisuju
39     greske. Stderr je podrazumevano postavljen
40     na standardni izlaz.
41
42     Ugradjena funkcija exit prouzrokuje zavrsetak programa.
43     Argument ove funkcije je jedna od konstanti definisanih
44     u biblioteci stdlib.h koje pokazuju da li se program
45     zavrrio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).
46
47     */
48     if(ulaz==NULL)
49     {
50         fprintf(stderr,"error fopen(): Neuspelo otvaranje datoteke ulaz
51         .txt za citanje.\n");
52         exit(EXIT_FAILURE);
53     }
54
55     izlaz= fopen("izlaz.txt", "w");
56     if(izlaz==NULL)
57     {
58         fprintf(stderr,"error fopen(): Neuspelo otvaranje datoteke
59         izlaz.txt za citanje.\n");
60         exit(EXIT_FAILURE);
61     }
62
63     /*
64     Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
65     Povratna vrednost ove funkcije je ascii kod unetog
66     karaktera.
67
68     Funkcija fputc ispisuje karakter c u datoteku izlaz.
69
70     */
71
72     while((c=fgetc(ulaz))!=EOF)
73         fputc(c,izlaz);
```

## 4 Ulaz i izlaz programa

---

```
72      /*
       * Nakon zavrsetka rada sa datotekama, neophodno ih je
       * zatvoriti pomocu ugradjene funkcije fclose.
       */
74      fclose(ulaz);
76      fclose(izlaz);
       return 0;
78 }
```

### Rešenje 4.1.2

```
/*
2   Napisati program koji u datoteci cije se ime navodi kao prvi
   argument komandne linije odredjuje liniju maksimalne duzine i
4   ispisiuje je na standardni izlaz. Ukoliko ima vise takvih linija,
   ispisiati onu koja je leksikografski prva. Mozemo pretpostaviti
6   da datoteka ne sadrzi linije duze od 80 karaktera.
   */
8   #include <stdio.h>
   #include <stdlib.h>
10  #include <string.h>
   #define MAX_LEN 81
12
   int main(int argc, char* argv[])
14  {
       char linija[MAX_LEN];
       char max_linija[MAX_LEN];
       int duzina;
       int max_duzina;
18
       FILE *ulaz, *izlaz;
20
       /*
       * Proveravamo da li poziv programa ima dovoljan broj argumenata.
       */
24      if(argc!=2)
       {
26          fprintf(stderr, "Greska: program se pokrece sa: %s
           ime_ulazne_datoteke\n", argv[0]);
           exit(EXIT_FAILURE);
28      }
30
       ulaz=fopen(argv[1], "r");
       if(ulaz==NULL)
       {
34          fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke %s
           za citanje.\n", argv[1]);
           exit(EXIT_FAILURE);
36      }
38      /*
```



```

    Funkcija fgets učitava jednu liniju teksta maksimalne duzine
    MAX_LEN
40     iz datoteke ulaz u string linija. Ukoliko učitavanje ne uspe (
    na primer,
    zato sto smo dosli do kraja datoteke), povratna vrednost ove
    funkcije
42     bice prazan pokazivac (NULL).
    */
44
    max_duzina=0;
46     while(fgets(linija, MAX_LEN, ulaz)!=NULL)
    {
48         duzina = strlen(linija);
        /*
50         Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u
        petlju.
        Ovu promenljivu menjamo kada je duzina učitana linije
52         veca od max_duzina ili kada su jednake, ali je učitana
        linija
        leksikografski ispred trenutne linije sa maksimalnom duzinom
        .

54         Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii
        kodova prva dva
        razlicita karaktera stringova s1 i s2 na istim indeksima,
56         ukoliko oni
        postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga
        osetljiva
58         na mala i velika slova (npr 'D' je leksikografski ispred 'p
        ').

60         */

62         if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija,
        max_linija)<0))
        {
64             strcpy(max_linija, linija);
            max_duzina=duzina;
66         }
    }
68
    /*
70     Funkcija fputs ispisuje string koji je njen prvi argument u
    datoteku
    koja je njen drugi argument. Sve funkcije za učitavanje iz
    datoteka i
72     upis u datoteke (fgetc, fputc, fgets, fputs, ...) mozemo
    koristiti
    i kada radimo sa standardnim ulazom i standardnim izlazom. Kao
    nazive
74     datoteka tada navodimo stdin i stdout.
    */

```

## 4 Ulaz i izlaz programa

---

```
76     fputs(max_linija, stdout);
78     fclose(ulaz);
    return 0;
80 }
```

### Rešenje 4.1.3

```
/*
2   U datoteci cije se ime zadaje kao prvi argument komandne linije
   nalazi se
   prirodan broj n a zatim i n celih brojeva. Napisati program koji
   prebrojava
4   koliko k-tocifrenih brojeva postoji u datoteci, pri cemu se
   prirodan broj k
   zadaje kao drugi argument komandne linije.
6 */

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

12 /*
   Funkcija ucitaj_i_prebroj ucitava brojeve
14   iz datoteke na koju pokazuje f i prebrojava
   koliko je medju njima k-tocifrenih brojeva
16 */
int ucitaj_i_prebroj(FILE* f, int k)
18 {
   int n;
20   int x;
   int i;
22   int br;

24   /* U datoteci je prvo naveden ukupan broj brojeva. */
   fscanf(f, "%d", &n);

26   /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
   */
28   if(n<=0)
   {
30       fprintf(stderr, "Greska: broj n mora biti prirodan\n");
       exit(EXIT_FAILURE);
32   }

34   br=0;
   for(i=0; i<n; i++)
36   {
       fscanf(f, "%d", &x);
38       if(broj_cifara(x)==k)
           br++;
   }
```

```
40     }
41
42     return br;
43 }
44
45 int broj_cifara(int x)
46 {
47     int br_c;
48
49     br_c=0;
50
51     /*
52      Do while petlja je pogodnija od petlji sa preduslovom
53      jer tacno racuna broj cifara i za broj 0.
54     */
55     do
56     {
57         br_c++;
58         x/=10;
59     } while(x);
60
61     return br_c;
62 }
63
64 int main(int argc, char* argv[])
65 {
66     int n;
67     int k;
68     FILE* f;
69     int br;
70
71     if(argc!=3)
72     {
73         fprintf(stderr, "Greska: program se pokrece sa: %s
74         naziv_datoteke k \n", argv[0]);
75         exit(EXIT_FAILURE);
76     }
77
78     f=fopen(argv[1], "r");
79
80     if(f==NULL)
81     {
82         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
83         \n", argv[1]);
84         exit(EXIT_FAILURE);
85     }
86
87     /* Argumenti komandne linije su stringovi. Da bismo konvertovali
88        string
89        u ceo broj koristimo ugradjenu funkciju atoi. */
90     k = atoi(argv[2]);
```

## 4 Ulaz i izlaz programa

```
90     if (k<=0)
91     {
92         fprintf(stderr, "Greska: broj k mora biti prirodan\n");
93         exit(EXIT_FAILURE);
94     }
95
96     printf("Broj %d-cifrenih brojeva u datoteci: %d\n", k,
97           ucitaj_i_prebroj(f,k));
98
99     fclose(f);
100    return 0;
101 }
```

### Rešenje 4.1.4

```
1  /*
2   U datoteci cije se ime navodi kao prvi argument komandne
3   linije navedena je rec r i niz linija. Napisati
4   program koji u datoteku cije se ime navodi kao
5   drugi argument komandne linije upisuje sve linije
6   u kojima se rec r pojavljuje bar n puta, gde je
7   n prirodan broj koji se unosi sa standardnog ulaza. Ispis
8   treba da bude u formatu broj_pojavljivanja: linija.
9  */
10
11 #include <stdio.h>
12 #include <stdlib.h>
13 #define MAXL 81
14 #define MAXR 31
15
16 /*
17  Funkcija broj_pojavljivanja broji koliko
18  se puta pojavio string t u stringu s
19  */
20 int broj_pojavljivanja(char s[], char t[])
21 {
22     int br;
23     int i,j;
24     /*
25      i - indeks karaktera u s
26      j - indeks karaktera u t
27      br - brojac koliko se puta t javlja u s
28     */
29     br=0;
30     for(i=0;s[i];i++)
31     {
32         for(j=0;t[j];j++)
33             if(s[i+j]!=t[j]) /* Ako naidjemo na razlicite karaktere, */
34                 break;      /* prekidamo petlju. */
35
36         /*
37          Do prekida petlje moze doci ili zbog toga sto su pronadjeni
38          svi karakteri t u s
39         */
34         br++;
35     }
36     return br;
37 }
```

```
37         razliciti karakteri i usledio je break ili zbog toga sto
38         je prestao da vazi uslov petlje, odnosno karakter t[j] je
39         jednak '\0'. Ako vazi drugi slucaj, to znaci da se string
        t nalazi u stringu s pocev od indeksa i i potrebno je
        uvecati
41         brojac br.
        */
43         if (t[j]=='\0')
            br++;
45     }

47     return br;
}
49 int main(int argc, char* argv[])
{
51     char rec[MAXR];
52     char linija[MAXL];
53     FILE* in, *out;
54     int n;
55     int br;

57     if(argc!=3)
    {
59         fprintf(stderr, "Greska: program se pokrece sa: %s
        ime_ulazne_datoteke ime_izlazne_datoteke\n", argv[0]);
        exit(EXIT_FAILURE);
61     }

63     in= fopen(argv[1], "r");
64     if(in==NULL)
    {
65         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", argv[1] );
66         exit(EXIT_FAILURE);
67     }

69     out= fopen(argv[2], "w");
70     if(out==NULL)
    {
71         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", argv[2] );
72         exit(EXIT_FAILURE);
73     }

75     printf("Unesi n:");
76     scanf("%d", &n);

77     if(n<=0)
    {
78         fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
79         exit(EXIT_FAILURE);
80     }
81 }
```

```
85     fscanf(in, "%s", rec);
87
89     while(fgets(linija, MAXL, in) != NULL)
90     {
91         br = broj_pojavljivanja(linija, rec);
92         if (br >= n)
93             fprintf(out, "%d: %s\n", br, linija);
94     }
95     fclose(in);
96     fclose(out);
97     return 0;
98 }
```

### Rešenje 4.1.5

```
1  /* Program se pokrece tako sto se navedu nazivi dve datoteke(ulazna i
   izlazna) i opcije.
   U datoteci cije se ime navodi kao prvi argument komandne linije
   nalaze se podaci o razlomcima:
3  u prvom redu se nalazi broj razlomaka, a u svakom sledecem redu
   brojilac i imenilac jednog razlomka.
   Potrebno je kreirati strukturu koja opisuje razlomak i učitati niz
   razlomaka
5  iz datoteke, a potom:
   a) ukoliko je navedena opcija x, upisati u datoteku cije je ime
   drugi argument komandne linije
7   recipročni razlomak za svaki razlomak iz niza (npr. za 2/3
   treba upisati 3/2)
   b) ukoliko je navedena opcija y, upisati u datoteku cije je ime
   drugi argument komandne linije
9   realnu vrednost recipročnog razlomka svakog razlomka iz niza
   (npr. za 2/3 treba upisati 1.5)
   Mozemo pretpostaviti da se u datoteci sa podacima o razlomcima
   nalazi najviše 100 razlomaka.
11 */
13 /*
   Prilikom pokretanja programa se, pored naziva ulazne i izlazne
   datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
   obe opcije, sto znaci da je minimalni broj argumenata 3.
17
   Moguci nacini pokretanja:
19 ./a.out ulaz.txt izlaz.txt -x
   ./a.out ulaz.txt izlaz.txt -y
21 ./a.out ulaz.txt izlaz.txt -yx
   ./a.out ulaz.txt izlaz.txt -xy
23
25 */
#include <stdio.h>
```

```
27 #include <stdlib.h>
28 #include <ctype.h>
29
30 #define MAX 100
31
32 typedef struct razlomak
33 {
34     int br;
35     int im;
36 } RAZLOMAK;
37
38 /*
39 Funkcija ucitaj_razlomke učitava razlomke iz datoteke
40 na koju pokazuje f u niz. Dimenzija niza, na koju
41 pokazuje pokazivac dim, nije poznata. Prva vrednost
42 u datoteci je ukupan broj razlomaka i tu vrednost
43 učitavamo u promenljivu dim.
44
45 Funkcija fscanf se koristi isto kao i funkcija scanf
46 uz dodatni prvi argument koji predstavlja naziv
47 datoteke iz koje se vrsi učitavanje.
48
49 */
50 int ucitaj_razlomke(RAZLOMAK niz[], int* dim, FILE* f)
51 {
52     int i;
53
54     fscanf(f, "%d", dim);
55     for (i=0; i<*dim; i++)
56     {
57         fscanf(f, "%d %d", &niz[i].br, &niz[i].im);
58         if (niz[i].im==0)
59             return 0;
60     }
61     return 1;
62 }
63
64 RAZLOMAK reciprocni(RAZLOMAK* r)
65 {
66     RAZLOMAK rec;
67     rec.im = r->br;
68     rec.br = r->im;
69     return rec;
70 }
71
72 float vrednost(RAZLOMAK* r)
73 {
74     return 1.0*r->br/r->im;
75 }
76
77 int main(int argc, char* argv[])
78 {
```

```
79 FILE *in, *out;
   char c;
81 int i;
   int j;
83 int xoption=0;
   int yoption=0;
85 int dim;
   RAZLOMAK razlomci[MAX];
87 RAZLOMAK r;

89 /*
   Prilikom pokretanja programa se, pored naziva ulazne i izlazne
91 datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
   obe opcije, sto znaci da je minimalni broj argumenata 3.

93
   Moguci nacini pokretanja:
95 ./a.out ulaz.txt izlaz.txt -x
   ./a.out ulaz.txt izlaz.txt -y
97 ./a.out ulaz.txt izlaz.txt -yx
   ./a.out ulaz.txt izlaz.txt -xy
99
101 */
   if(argc!=4)
103 {
       fprintf(stderr, "Greska: program se pokrece sa: %s
       ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n", argv[0]);
105       exit(EXIT_FAILURE);
   }

107
109 in= fopen(argv[1], "r");
   if(in==NULL)
111 {
       fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
       .\n", argv[1] );
113       exit(EXIT_FAILURE);
   }

115
   out= fopen(argv[2], "w");
   if(out==NULL)
117 {
       fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
       .\n", argv[2] );
119       exit(EXIT_FAILURE);
   }
121

123 /* Ispitujemo da li je treca opcija u trazenom formatu. Prvi
   karakter mora biti '-' */

125 if (argv[3][0] != '-')
   {
```



```

127     fprintf(stderr, "Greska u zadavanju opcija: program se pokrece
sa: %s ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n",
argv[0]);
    exit(EXIT_FAILURE);
129 }

131 /* Ostali karakteri mogu biti 'x' ili 'y'. U zavisnosti od date
opcije, postavljamo
vrednosti indikatorskih promenljivih xoption i yoption. */
133
134 for(j=1;argv[3][j]!='\0';j++)
135     switch(argv[3][j])
136     {
137         case 'x': xoption=1;
138                 break;
139         case 'y': yoption=1;
140                 break;
141         default:
142             fprintf(stderr, "Greska: nedozvoljeni karakter\n"
143             );
144             exit(EXIT_FAILURE);
145     }

146 if(ucitaj_razlomke(razlomci, &dim, in)==0)
147 {
148     fprintf(stderr, "Greska pri zadavanju razlomaka\n");
149     exit(EXIT_FAILURE);
150 }
151
152 /*
153  U zavisnosti od datih opcija, vrsimo upis reciprocnih
154  razlomaka u trazenom formatu.
155
156  Funkcija fprintf se koristi na isti nacin kao
157  funkcija printf uz dodatni prvi argument koji
158  oznacava naziv datoteke u koju se vrsi upis.
159  */
160 for (i=0; i<dim;i++)
161 {
162     /*
163      Ukoliko je brojilac razlomka jednak nuli,
164      nema smisla traziti njegovu reciprocnu vrednost
165      */
166     if (razlomci[i].br==0)
167         continue;
168
169     r = reciprocn(&razlomci[i]);
170
171     if (xoption)
172         fprintf(out,"%d/%d ", r.br, r.im);

```

## 4 Ulaz i izlaz programa

```
175     if (yoption)
176         fprintf(out, "%f ", vrednost(&r));
177
178     fprintf(out, "\n");
179 }
180
181 fclose(in);
182 fclose(out);
183
184 return 0;
185 }
```

### Rešenje 4.1.6

```
1  /*
2   * Za svaki automobil poznati su marka, model i cena. Iz datoteke cije
3   * se ime zadaje sa standardnog ulaza učitava se broj automobila a
4   * potom
5   * i podaci za svaki automobil. Program treba da:
6   * a) izracuna prosečnu cenu po marki kola
7   * b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
8   *   zadaje
9   * kao argument komandne linije, da ispise automobile u tom cenovnom
10  *   rangu zajednu sa prosečnom cenom odgovarajuće marke
11  *
12  * Mozemo pretpostaviti da se model i marka sastoje od jedne reci i
13  * da svaka od njih sadrži najviše 30 karaktera kao i da se u datoteci
14  * nalaze podaci za najviše 100 automobila.
15  */
16
17 #include <stdio.h>
18 #include <stdlib.h>
19 #include <string.h>
20 #define MAX 31
21 #define MAXA 100
22
23 typedef struct automobil
24 {
25     char marka[MAX];
26     char model[MAX];
27     float cena;
28 } AUTOMOBIL;
29
30 /*
31 * Struktura INFO sadrži naziv
32 * marke automobila, prosek cena
33 * za tu marku i broj automobila
34 * te marke
35 */
36 typedef struct info
```

```

{
37     char marka[MAX];
    float vrednost;
39     int n;
} INFO;

41 int učitaj_podatke(FILE* f, AUTOMOBIL a[], int* pn, int max)
43 {
    int i;
45
    fscanf(f, "%d", pn);
47     if (*pn <= 0 || *pn > max)
    {
49         printf("Nekorektan unos dimenzije niza automobila\n");
        return 0;
51     }
    for(i=0; i<*pn; i++)
53         fscanf(f, "%s %s %f", a[i].marka, a[i].model, &a[i].cena);

55     return 1;
}

57 /*
59 Funkcija sadrži ispituje da li se u nizu proseka po marki
    nalazi prosek za marku m. Posto podatak o marki automobila
61 predstavlja string, poredjenje vrsimo pomocu funkcije strcmp.

63 Povratna vrednost ove funkcije je indeks pojavljivanja, ukoliko
    se marka m pojavljuje u nizu proseka, ili -1 u suprotnom.
65 */

67 int sadrzi(INFO p[], int n, char m[])
{
69     int i;
    for(i=0; i<n; i++)
71         if(strcmp(p[i].marka, m) == 0)
            return i;

73     return -1;
75 }

77 /*
79 Funkcija informacije_o_markama za niz automobila a dimenzije n
    racuna proseke cena automobila po markama i smesta ih u niz
    p. Na dimenziju niza p pokazuje pokazivac pn.

81 Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
83 sumu cena automobila te marke (koju cemo smestiti u polje vrednost
    strukture
    INFO), i broj automobila te marke (koju cemo smestiti u polje
85 n strukture INFO) i da na kraju podelimo ove dve promenljive
    i tako dobijemo prosečnu vrednost cene.

```

```
87      Za svaki automobil a[i] proveravamo da li se njegova marka vec
89      nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
91      vredost cene automobila a[i] i uvecavamo broj automobila sa
93      tom markom. U suprotnom, dodajemo novi element u niz p. Posto
      ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
      na koju pokazuje pokazivac *pn.
93  */
95  void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
96  {
97      int i,j;
98      int ind;
99      for(i=0;i<n;i++)
100      {
101          /* Proveravamo da li se marka automobila a[i] vec nalazi u
102             nizu p (niz proseka po markama) */
103          ind = sadrzi(p,*pn1,a[i].marka);
104          if(ind!=-1) /* Ako se ne nalazi, uvodimo novi element niza na
105             kraj, na poziciju *pn. */
106          {
107              strcpy(p[*pn1].marka, a[i].marka);
108              p[*pn1].vrednost = a[i].cena;
109              p[*pn1].n = 1;
110              (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
111             */
112          }
113          else /* Ako se nalazi, azuriramo polja strukture. */
114          {
115              p[ind].vrednost+=a[i].cena;
116              p[ind].n++;
117          }
118      }
119      /* Na osnovu sume cena i broja automobila racunamo proseчну
120         vrednost. */
121      for(i=0;i<*pn1;i++)
122          p[i].vrednost = p[i].vrednost/p[i].n;
123  }
124
125  void stampaj_informacije(INFO p[], int n)
126  {
127      printf("Informacije o broju automobila i prosečnoj ceni po markama
128         :\n");
129      int i;
130      for(i=0;i<n;i++)
131          printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
132  }
133
134  /*
135  Funkcija stampa automobile cija je cena manja od maksimalne
136  cene koju je korisnik naveo u komandnoj liniji da je spreman
```

```

135     da plati, zajedno sa prosecom cenom za tu marku automobila
136     */
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
    n1)
138 {
139     /*
140         S obzirom da je niz p formiran na osnovu niza a, marka svakog
141         automobila iz niza a se sigurno nalazi u nizu p. Zbog toga
142         nije neophodno proveravati da li je povratna vrednost funkcije
143         sadrzi razlicita od -1.
144     */
145     int i;
146     printf("Kola u vasem cenovnom rangu:\n");
147     for(i=0;i<n;i++)
148         if(a[i].cena<g)
149             printf("%s %s %.2f\n", a[i].marka, a[i].model, p[sadrzi(p,n1
    ,a[i].marka)].vrednost);
150 }
151
152 int main(int argc, char* argv[])
153 {
154     AUTOMOBIL kola[MAXA];
155     FILE* f;
156     char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
    ulaza. */
157     float granica; /* Maksimalna cena koju je korisnik spreman da
    plati.
158
159                     Zadaje se kao argument komandne linije.
160     */
161     INFO infos[MAXA];
162     int dim_kola,dim_infos;
163     int i;
164
165     if(argc!=2)
166     {
167         fprintf(stderr,"Greska: program se pokrece sa: %s
    gornja_granica_cene \n", argv[0]);
168         exit(EXIT_FAILURE);
169     }
170
171     /* Argumenti komandne linije su stringovi. Da bismo od stringa
    dobili
172     realan broj, koristimo ugradjenu funkciju atof. */
173     granica = atof(argv[1]);
174
175     printf("Unesi naziv datoteke:");
176     scanf("%s", dat);
177
178     f=fopen(dat, "r");
179
180     if(f==NULL)
181     {

```

## 4 Ulaz i izlaz programa

```
181     fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s\n", dat);
182     exit(EXIT_FAILURE);
183 }
184
185 if (ucitaj_podatke(f,kola,&dim_kola,MAXA)==0)
186 {
187     fprintf(stderr, "Greska pri ucitavanju podataka\n");
188     exit(EXIT_FAILURE);
189 }
190
191 informacije_o_markama(kola, dim_kola, infos, &dim_infos);
192
193 stampaj_informacije(infos,dim_infos);
194
195 stampaj_kandidate(kola, dim_kola, granica, infos, dim_infos);
196
197 fclose(f);
198 return 0;
199 }
```

### Rešenje 4.1.7

```
1 /* Napisati program koji prebrojava mala slova u datoteci test.txt */
2
3 #include<stdio.h>
4
5 int main(){
6
7     FILE* in;
8     int c, broj_malih=0;
9
10    /*Otvaramo datoteku test.txt za citanje i proveravamo da li smo je
11       uspesno otvorili*/
12    in = fopen("test.txt", "r");
13    if(in == NULL){
14        printf("Greska!");
15        return 0;
16    }
17
18    /*Citamo karakter po karakter, i ukoliko je procitani
19       *karakter malo slovo, uvecevamo brojac*/
20    while((c=fgetc(in))!=EOF){
21        if(islower(c))
22            broj_malih++;
23    }
24
25    /*Ispisujemo rezultat*/
26    printf("Broj malih slova je: %d\n", broj_malih);
27 }
```

```
27  /*Zatvaramo datoteku*/  
    fclose(in);  
29  
    return 0;  
31 }
```

### Rešenje 4.1.8

```
/* Napisati program koji prepisuje svaki treci karakter datoteke ulaz  
:txt u datoteku izlaz.txt */  
2  
#include<stdio.h>  
4  
int main(){  
6  
    FILE *in, *out;  
8    int c;  
    int rbr_karaktera;  
10  
12    /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je  
        uspesno otvorili*/  
    in = fopen("ulaz.txt", "r");  
14    if(in == NULL){  
        printf("Greska!");  
16    return 0;  
    }  
18  
    /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo dali smo je  
        uspesno otvorili*/  
20    out = fopen("izlaz.txt", "w");  
    if(out == NULL){  
22        printf("Greska!");  
        return 0;  
24    }  
26  
    /* Inicijalizujemo redni broj karaktera koji se cita */  
    rbr_karaktera=0;  
28  
    /* Citamo karakter po karakter iz datoteke sve dok ne stignemo do  
        kraja datoteke */  
30    while((c=fgetc(in)) != EOF){  
32        /* Ukoliko je procitani karakter na poziciji koja je deljiva sa 3  
            prepisujemo ga */  
        if(rbr_karaktera%3==0)  
34            fputc(c, out);  
36  
        /* Uvecavamo redni broj karaktera */  
        rbr_karaktera++;  
38    }
```

```
40  /*Zatvaramo obe datoteke koje smo otvorili*/
    fclose(out);
42  fclose(in);
    return 0;
44 }
```

### Rešenje 4.1.9

```
/* Kao argumenti komandne linije se zadaju ime datoteke i ceo broj k.
   Napisati program koji na standardni izlaz
2  ispisuje sve linije zadate datoteke cija je duzina veca od k. Moze se
   pretpostaviti da duzina linije nece biti veca
   od 80 karaktera */
4
#include<stdio.h>
6  #include<string.h>
8  #define MAXL 81
10 int main(int argc, char* argv[]){
12     FILE* in;
    char linija[MAXL];
14     int k;

16     /*Proveravamo da li je program ispravno pozvan*/
    if(argc!=3){
18         printf("Greska: pogresan broj argumenata!");
        return 0;
20     }

22     /*Otvaramo za citanje datoteku koja se navodi kao prvi argument
        komandne linije*/
    in = fopen(argv[1], "r");
24     if(in == NULL){
        printf("Greska: neuspesno otvaranje datoteke!");
26         return 0;
    }

28     /*Uzimamo brojevnu vrednost drugog argumenta komandne linije*/
30     k = atoi(argv[2]);

32     /*Citamo liniju po liniju i sve linije duze od k ispisujemo na
        standardni izlaz*/
    while(fgets(linija, MAXL, in) != NULL){
34         if(strlen(linija) > k)
            printf("%s", linija);
36     }
    printf("\n");
38 }
```



```

/*Zatvaramo datoteku*/
40 fclose(in);
    return 0;
42 }

```

### Rešenje 4.1.10

```

/* Napisati program koji prebrojava koliko se linija datoteke ulaz.
   txt završava niskom s koja se učitava sa stan-
2 dardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća
   od 80 karaktera, kao i da dužina niske s
   ne će biti veća od 20 karaktera */
4
#include<stdio.h>
6 #include <string.h>
#define MAXL 81
8 #define MAXS 21

/* Funkcija brojLinija proverava koliko linija u datoteci in se
   završava niskom s.
   Funkcija radi tako što čita jednu po jednu liniju iz datoteke,
12 i zatim kraj te linije poredi sa niskom s.*/
int brojLinija(FILE* in, char* s){
14
    char linija[MAXL];
16 int broj_linija = 0;
    int dužina_s = strlen(s);
18 int dužina_linije;

    while(fgets(linija, MAXL, in) != NULL){
20 dužina_linije = strlen(linija);

22
    /* Ukoliko je znak za novi red bio indikacija kraja linije,
       uklanjamo ga kako bi mogli da izvršimo
24 *ispravno poredjenje (jer niska s nema novi red na kraju) */
    if(linija[dužina_linije-1]=='\n'){
26 linija[dužina_linije-1] = '\0';
        dužina_linije--;
28 }

30
    /*linija + dužina_linije će nas odvesti na kraj tog stringa, a kada
       oduzmemo dužinu stringa s,
       a kada od toga oduzmemo dužinu niske s, dobićemo bas onoliko
       poslednjih karaktera, koliko
32 nam i treba. U primeru uspravna crta (|) označava pokazivac
       s                ab
34 dužina_s                2
       Linija:            aaabbbdfssab

36 |
       Linija + dužina linije aaabbbdfssab
38 |

```

```

        Linija + duzina linije - 2 aaabbbdfssab
        |
        kada kazemo strcmp(linija + duzina_linije - duzina_s, s), mi
        cemo u nasem primeru zaista porediti "ab" i "ab".
42    */
    if(strcmp(linija + duzina_linije - duzina_s, s) == 0)
44        broj_linija++;
    }
46    return broj_linija;
}
48
49 int main(){
50
51     FILE* in;
52     char s[MAXS];
53
54     /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
        uspesno otvorili*/
    in = fopen("ulaz.txt", "r");
56    if(in == NULL){
        printf("Greska: neuspesno otvaranje datoteke!\n");
58        return 0;
    }
60
61     /*Ucitavamo nisku*/
62     printf("Unesite nisku s: ");
        scanf("%s", s);
64
65     /*Ispisujemo koliko linija iz datoteke se zavrшава sa niskom s*/
66     printf("Broj linija: %d\n", brojLinija(in, s));
67
68     /*Zatvaramo datoteku*/
        fclose(in);
70
71     return 0;
72 }
```

### Rešenje 4.1.11

```

/* Napisati program koji pronalazi maksimum brojeva zapisanih u
   datoteci brojevi.txt */
2
#include<stdio.h>
4
int main(){
6
    FILE* in;
    float broj, max_broj;
8
10    /*Otvaramo datoteku brojevi.txt za citanje i proveravamo da li smo
        je uspesno otvorili*/
    in = fopen("brojevi.txt", "r");
    if(in == NULL){
        printf("Greska: neuspesno otvaranje datoteke!\n");
        return 0;
    }
    while(fscanf(in, "%f", &broj) != EOF){
        if(broj > max_broj)
            max_broj = broj;
    }
    printf("Maksimum broja je: %f\n", max_broj);
    fclose(in);
    return 0;
}
```

```

12  in = fopen("brojevi.txt", "r");
13  if(in == NULL){
14      printf("Greska pri otvaranju datoteke!");
15      return 0;
16  }
17
18  /*
19   Kako bismo inicijalizovali promenljivu max_broj,
20   citamo jedan broj iz datoteke i smestamo ga u
21   ovu promenljivu. */
22  fscanf(in, "%f", &max_broj);
23
24  /*U petlji citamo sve ostale brojeve i poredimo ih sa trenutnim
25   maksimumom.*/
26  while(fscanf(in, "%f", &broj) != EOF){
27      if(broj > max_broj)
28          max_broj = broj;
29  }
30
31  /*Ispisujemo rezultat*/
32  printf("Najveci broj je: %.2f\n", max_broj);
33
34  /*Zatvaramo datoteku brojevi.txt*/
35  fclose(in);
36
37  return 0;
38  }

```

### Rešenje 4.1.12

```

1  /* U datoteci studenti. txt se nalaze informacije o studentima: prvo
2   broj studenata, a zatim u pojedinacnim linijama
3   korisnicko ime i pet poslednjih ocena koje je student dobio. Napisati
4   program koji pronalazi studenta koji je
5   ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da
6   broj studenata nece biti veci od 100. */
7
8  #include<stdio.h>
9
10 #define MAXS 100
11
12 /*Definisemo strukturu za cuvanje studenata*/
13 typedef struct st{
14     char korisnicko_ime[8];
15     float prosek;
16 }STUDENT;
17
18 int main(){
19
20     FILE *ulaz;
21     STUDENT studenti[MAXS];

```

```
19  int ocena1,ocena2,ocena3,ocena4,ocena5, i=0, i_max_prosek;
21  float max_prosek = 0;

23  /*Otvaramo datoteku studenti.txt za citanje*/
    ulaz = fopen("studenti.txt", "r");
25  if(ulaz == NULL){
    printf("Greska pri otvaranju datoteke!\n");
27  return 0;
    }

29  /*Ucitavamo liniju po liniju iz datoteke, sve dok ne dodjemo do
    kraja.
31  Korisnicko ime smestamo u niz, a ocene ucitavamo u pomocne
    promenljive ocena1,...ocena5.
    Zatim, na osnovu ocena racunamo prosek.

33  Ovde paralelno sa ucitavanjem pronalazimo i studenta sa najvećim
    prosekom i
35  pamtimo njegov prosek i njegovu poziciju u nizu studenata,
    Nismo morali ovako. Mogli smo i prvu da ucitamo sve studente, a
    zatim da prodjemo
37  jednom kroz niz i da nadjemo onog sa najvećim prosekom.

39  */
    while(fscanf(ulaz, "%s%d%d%d%d", studenti[i].korisnicko_ime, &
        ocena1, &ocena2, &ocena3, &ocena4, &ocena5) != EOF){
41  studenti[i].prosek = (ocena1 + ocena2 + ocena3 + ocena4 + ocena5)
        /5.0;

43  if(studenti[i].prosek > max_prosek){
        max_prosek= studenti[i].prosek;
45  i_max_prosek = i;
    }
47  i++;
    }

49  /*Ispisujemo rezultat*/
51  printf("korisnicko ime: %s, prosek ocena: %.2f\n", studenti[
    i_max_prosek].korisnicko_ime, studenti[i_max_prosek].prosek);

53  /*Zatvaramo datoteku*/
    fclose(ulaz);

55  return 0;
57 }
```

### Rešenje 4.1.13

### Rešenje 4.1.14

```
1  /* Napisati program koji za rec s maksimalne duzine 20 karaktera koja
   se zadaje sa standardnog ulaza u datoteku
   rotacije.txt upisuje sve rotacije reci s */
3
4  #include<stdio.h>
5  #include<string.h>
6
7  #define MAXS 21
8
9  /*Funkcija rotira nisku za jedno mesto u desno.
   Duzina niske n nije obavezan argument. Mogli smo
11 i da je racunamo u okviru funkcije, ali kako ce sve niske
   sa kojima radimo biti iste duzine, efikasnije je da jednom
13 izracunamo tu duzinu u glavnom programu,
   pa da je prosledjujemo kao argument.*/
15 void rotiraj_zal(char* s, int n){
    int i;
17     char c = s[0];
    for(i=0; i<n-1; i++){
19         s[i] = s[i+1];
    }
21     s[n-1] = c;
    }
23
24 int main(){
25
26     char s[MAXS];
27     int n, i;
28     FILE * izlaz;
29
30     /*Otvaramo datoteku rotacije.txt za pisanje i proveravamo da li smo
       je uspesno otvorili*/
31     izlaz = fopen("rotacije.txt", "w");
    if(izlaz == NULL){
33         printf("Greska pri otvaranju fajla!");
        return 0;
35     }
36
37     /*Sa standardnog ulaza učitavamo rec koju treba da rotiramo*/
    scanf("%s", s);
39
40     /*Racunamo njenu duzinu*/
41     n = strlen(s);
42
43     /*U petlji, ispisujemo tu rec u datoteku, pa je rotiramo za jedno
       mesto u desno.*/
    for(i=0; i<n; i++){
45         fprintf(izlaz, "%s\n", s);
        rotiraj_zal(s,n);
47     }
```

## 4 Ulaz i izlaz programa

---

```
49  /*Zatvaramo datoteku rotacije.txt*/
    fclose(izlaz);
51
    return 0;
53
}
```

### Rešenje 4.1.15

```
1  /* 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,
3  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.
5  */

7  #include<stdio.h>
   #include<string.h>
9  #include<ctype.h>

11 #define MAXL 81

13 int main(int argc, char* argv[]){

15     char linija[MAXL];
     FILE* izlaz;

17

   /*Indikatori koji označavaju koja opcija je navedena kao argument
     komandne linije
19     vind - ispisuju se rečenice koje počinju velikim slovom
     mind - ispisuju se rečenice koje počinju malim slovom
21     */
     int vind=0, mind = 0;

23

   /*Proveravamo da li je program ispravno pozvan*/
25     if(argc > 2){
         printf("Greska pri pozivanju programa!\n");
27         return 0;
     }

29

   /*Ako opcije nisu zadate, onda treba da se ispisuju sve rečenice,
     pa postavljamo oba indikatora na 1*/
31     if(argc == 1){
         vind = mind = 1;
33     }else{

35         /*Proveravamo da li je postavljena neka od opcija -v,-V,-m, -M
           Ako jeste, postavljamo odgovarajući indikator
           Ako nije, onda ispisujemo poruku o gresci*/
37     }
```

```
39     if(strcmp(argv[1], "-v") == 0 || strcmp(argv[1], "-V") == 0)
        vind = 1;
41     else if(strcmp(argv[1], "-m") == 0 || strcmp(argv[1], "-M") == 0)
        mind = 1;
43     else{
        printf("Greska pri zadavanju opcije!\n");
        return 0;
45     }
47 }

49 /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo da li smo je
    uspesno otvorili*/
    izlaz = fopen("izlaz.txt", "w");
51     if(izlaz == NULL){
        printf("Greska pri otvaranju datoteke!\n");
53     return 0;
    }

55     /*Citamo liniju po liniju sa standardnog ulaza i ispisujemo je u
        datoteku.
57     Liniju ispisujemo ukoliko je ispunjen neki od dva uslova:
        1. Izabrana je opcija za ispis malih slova i linija pocinje malim
            slovom
59     2. Izabrana je opcija za velika slova i linija pocinje velikim
            slovom
        NAPOMENA: Kada dodje do kraja ulaza, funkcija fgets vraca NULL
61     */
    while(fgets(linija, MAXL, stdin) != NULL){
63     if( mind && islower(linija[0]) || vind && isupper(linija[0]) ||
        mind && vind)
        fputs(linija, izlaz);
65     }

67     /*Zatvaramo datoteku izlaz.txt*/
    fclose(izlaz);
69
    return 0;
71 }
```

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje 4.1.33

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)

Rešenje [4.1.33](#)