

## PROGRAMIRANJE 1



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

**PROGRAMIRANJE 1**  
**Zbirka zadataka sa rešenjima**

**Beograd  
2016.**

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 sa rešenjima

# Sadržaj

<b>1</b>	<b>Kontrola toka</b>	<b>1</b>
1.1	Uvodni zadaci . . . . .	1
1.2	Naredbe grananja . . . . .	5
1.3	Petlje . . . . .	12
1.4	Funkcije . . . . .	20
1.5	Rešenja . . . . .	28
<b>2</b>	<b>Predstavljanje podataka</b>	<b>101</b>
2.1	Nizovi . . . . .	101
2.2	Pokazivači i argumenti komandne linije . . . . .	109
2.3	Niske . . . . .	113
2.4	Višedimenzioni nizovi . . . . .	117
2.5	Strukture . . . . .	117
2.6	Rešenja . . . . .	121
<b>3</b>	<b>Datoteke</b>	<b>189</b>
3.1	Rešenja . . . . .	192
<b>4</b>	<b>Razni zadaci</b>	<b>209</b>
4.1	Rešenja . . . . .	209
<b>A</b>	<b>Ispitni rokovi</b>	<b>211</b>
A.1	Ispitni rokovi MNVRLA . . . . .	211
A.1.1	Kvalifikacioni zadaci . . . . .	211
A.1.2	Praktični deo ispita, jun ... . . . .	211
A.2	Ispitni rokovi I smer . . . . .	211
A.3	Rešenja . . . . .	211



# Predgovor

U okviru kursa *Programiranje 1* na Matematičkom fakultetu vežbaju se zadaci koji imaju za cilj da studente nauče ...

*Autori*





# 1

## Kontrola toka

### 1.1 Uvodni zadaci

**Zadatak 1.1** Tekst

[Rešenje [1.1](#)]

**Zadatak 1.2** Tekst

[Rešenje [1.2](#)]

**Zadatak 1.3** Tekst

[Rešenje [1.3](#)]

**Zadatak 1.4** Tekst

[Rešenje [1.4](#)]

**Zadatak 1.5** Tekst

[Rešenje [1.5](#)]

**Zadatak 1.6** Tekst

[Rešenje [1.6](#)]

**Zadatak 1.7** Tekst

[Rešenje [1.7](#)]

### Zadatak 1.8 Tekst

[Rešenje 1.8]

### Zadatak 1.9 Tekst

[Rešenje 1.9]

### Zadatak 1.10 Tekst

[Rešenje 1.10]

**Zadatak 1.11** Napisati program koji omogućava korisniku da unese ceo broj, a zatim ispisuje njegov kvadrat i kub.

#### Primer 1

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

[Rešenje 1.11]

**Zadatak 1.12** Napisati program koji za unete stranice pravougaonika ispisuje njegov obim i površinu.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite duzine stranica pravougaonika: 2 8  
|| Obim: 20  
|| Povrsina: 16
```

[Rešenje 1.12]

**Zadatak 1.13** Napisati program koji za unete stranice trougla ispisuje njegov obim i površinu.

#### Primer 1

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

[Rešenje 1.13]

**Zadatak 1.14** Napisati program koji za unete dimenzije sobe u metrima (dužinu, širinu i visinu) ispisuje koju površinu treba da okreči moler. Uračunati da na vrata i prozore otpada oko 20%. Omogućiti i unos cene usluge po kvadratnom metru i izračunati zaradu koju ostvaruje moler.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenzije sobe: 4 4 3
Unesite cenu po kvadratnom metru: 500
Moler treba da okreči 51.2 kvadratna metra
Cena krecenja je 25600
```

[Rešenje 1.14]

**Zadatak 1.15** Napisati program koji za unetu količinu jabuka u kilogramima i unetu cenu po kilogramu ispisuje ukupan iznos koji treba platiti.

*Primer 1*

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

[Rešenje 1.15]

**Zadatak 1.16** Napisati program koji pomaže kasirki da obračuna kusur tako što od nje traži da unese cenu artikla, količinu artikla i iznos koji je dobila od kupca.

*Primer 1*

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

[Rešenje 1.16]

**Zadatak 1.17** Napisati program koji prirodnom četvorocifrenom broju koji se unosi sa standardnog ulaza:

- izračunava proizvod cifara
- izračunava razliku sume krajnjih i srednjih cifara
- izračunava sumu kvadrata cifara
- određuje broj koji se dobija ispisom cifara u obrnutom poretku

## 1 Kontrola toka

---

- određuje broj koji se dobija zamenom cifre jedinice i cifre stotine

### 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
```

[Rešenje 1.17]

**Zadatak 1.18** Napisati program koji izbacuje cifru desetica datom prirodnim broju.

### Primer 1

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

[Rešenje 1.18]

**Zadatak 1.19** Napisati program koji u datom prirodnom broju  $x$  ubacuje cifru  $c$  na poziciju  $p$  i rezultat ispisuje na standardni izlaz. Brojevi  $x$ ,  $c$  i  $p$  se unose sa standardnog ulaza. Podrazumeva se da je broj  $p$  manji od ukupnog broja cifara broja  $i$  da numeracija cifara počinje od 1. Uputstvo: koristiti funkciju *pow* iz *math.h* biblioteke.

### Primer 1

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

[Rešenje 1.19]

**Zadatak 1.20** Napisati program koji:

- unetu dužinu u miljama konvertuje u kilometre ( $1 \text{ mi} = 1.609344 \text{ km}$ )
- unetu težinu u funtama konvertuje u kilograme ( $1 \text{ lb} = 0.45359237 \text{ kg}$ )
- unetu temperaturu u celzijusima konvertuje u farenhajte ( $F = \frac{9 \cdot C}{5} + 32$ )

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite duzinu u miljama: 1.8
Vrednost duzine u kilometrima je: 2.896819
Unesite tezinu u funtama: 10
Vrednost tezine u kilogramima je: 4.535923
Unesite temperaturu u celzjusima: 37.2
Vrednost temperature u farenhajtima je: 98.960007
```

[Rešenje [1.20](#)]

**Zadatak 1.21** Napisati program koji učitava sa standardnog ulaza vreme poletanja i vreme sletanja aviona, a potom ispisuje dužinu trajanja leta. Možemo pretpostaviti da su poletanje i sletanje u istom danu.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite vreme poletanja: 8 5 0
Unesite vreme sletanja: 12 41 30
Duzina trajanja leta: 4 h 36 min 30 sec
```

[Rešenje [1.21](#)]

## 1.2 Naredbe grananja

**Zadatak 1.22** Tekst

[Rešenje [1.42](#)]

**Zadatak 1.23** Tekst

[Rešenje [1.43](#)]

**Zadatak 1.24** Tekst

[Rešenje [1.44](#)]

**Zadatak 1.25** Tekst

[Rešenje [1.45](#)]

**Zadatak 1.26** Tekst

[Rešenje [1.46](#)]

**Zadatak 1.27** Tekst

[Rešenje [1.47](#)]

**Zadatak 1.28** Tekst

[Rešenje [1.48](#)]

**Zadatak 1.29** Tekst

[Rešenje [1.49](#)]

**Zadatak 1.30** Tekst

[Rešenje [1.50](#)]

**Zadatak 1.31** Tekst

[Rešenje [1.51](#)]

**Zadatak 1.32** Tekst

[Rešenje [1.52](#)]

**Zadatak 1.33** Tekst

[Rešenje [1.53](#)]

**Zadatak 1.34** Tekst

[Rešenje [1.54](#)]

**Zadatak 1.35** Tekst

[Rešenje [1.55](#)]

**Zadatak 1.36** Tekst

[Rešenje [1.36](#)]

**Zadatak 1.37** Tekst

[Rešenje [1.37](#)]

**Zadatak 1.38** Tekst

[Rešenje [1.38](#)]

**Zadatak 1.39** Tekst

[Rešenje [1.39](#)]

**Zadatak 1.40** Tekst

[Rešenje [1.40](#)]

**Zadatak 1.41** Tekst

[Rešenje [1.41](#)]

**Zadatak 1.42** Tekst

[Rešenje [1.42](#)]

**Zadatak 1.43** Tekst

[Rešenje [1.43](#)]

**Zadatak 1.44** Tekst

[Rešenje [1.44](#)]

**Zadatak 1.45** Tekst

[Rešenje [1.45](#)]

**Zadatak 1.46** Tekst

[Rešenje [1.46](#)]

**Zadatak 1.47** Tekst

[Rešenje [1.47](#)]

**Zadatak 1.48** Tekst

[Rešenje [1.48](#)]

**Zadatak 1.49** Tekst

[Rešenje [1.49](#)]

**Zadatak 1.50** Tekst

[Rešenje [1.50](#)]

**Zadatak 1.51** Tekst

[Rešenje [1.51](#)]

**Zadatak 1.52** Tekst

[Rešenje [1.52](#)]

**Zadatak 1.53** Tekst

[Rešenje [1.53](#)]

**Zadatak 1.54** Tekst

[Rešenje [1.54](#)]

**Zadatak 1.55** Tekst

[Rešenje [1.55](#)]

- (a) Sa standardnog ulaza se unosi ceo četvorocifren broj. Napisati program koji ispisuje njegovu najveću cifru na standardni izlaz.

*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!
```

- (b) Napisati program koji za dati trocifren broj proverava da li je Armstrongov. Broj je Armstrongov ako je jednak zbiru kubova svojih cifara.

*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!
```

- (c) Za ceo broj  $k$  između 1 i 189 koji se unosi sa standardnog ulaza, odrediti cifru koja se nalazi na  $k$ -toj poziciji niza 12345678910111213....9899 u kom



su redom ispisani brojevi od 1 do 99.

### 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.
```

- (d) Sa standardnog ulaza se unosi četvorocifreni pozitivan broj. Napisati program koji računa i ispisuje proizvod parnih cifara datog broja. Ukoliko uneti broj nije pozitivna četvorocifrena vrednost ispisati poruku *Greska!*.

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 3579  
| Proizvod parnih cifara: 0
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 288  
| Greska!
```

- (e) Sa standardnog ulaza unosi se 5 karaktera. Proveriti da li je prvi karakter veliko ili malo slovo *a*. Ako jeste, ispisati karaktere obrnutim redosledom, a ako nije, ništa ne ispisivati.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite karaktere: A u E f h  
| h f E u A
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite karaktere: k L M 9 o
```

- (f) Sa standardnog ulaza unosi se jedan karakter. Ako je u pitanju malo slovo, zameniti ga odgovarajućim velikim slovom i ispisati na standardni izlaz. Ako je u pitanju veliko slovo, zameniti ga odgovarajućim malim slovom i ispisati ga na standardni izlaz. Ako je u pitanju cifra ispisati poruku *cifra*. Ako je u pitanju bilo koji drugi karakter, onda ga ispisati na standardni izlaz između dveju zvezdica.

### 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: >
|| *>
```

- (g) Sa standardnog ulaza se unosi 5 karaktera. Ispisati na izlazu broj unetih 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
```

- (h) Sa standardnog ulaza se unosi četvorocifren ceo broj. Napisati program koji datom broju razmenjuje najmanju i najveću cifru. Dobijeni broj ispisati na standardni izlaz. Ako uneti broj nije četvorocifren ispisati poruku *Greska!*.

### Primer 1

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

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 247
|| Greska!
```

- (i) Sa standardnog ulaza se unose tri neoznačena trocifrena broja. Spojiti dva najveća u šestocifren broj. Spajanje izvršiti tako da najveći od trocifrenih brojeva bude na početku šestocifrenog broja. Dobijeni šestocifreni broj ispisati na izlazu. Ako neki od unetih brojeva nije trocifren, ispisati poruku *Greska!*.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 185 247 311
|| Trazeni broj je: 311247
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 865 11 298
|| Greska!
```

- (j) Sa standardnog ulaza se učitavaju realni koeficijenti  $A$  i  $B$  linearne jednačine  $Ax + B = 0$ . Napisati program koji ispisuje rešenja ove jednačine - ukoliko jednačina nema rešenja ili ukoliko ima više od jednog rešenja ispisati odgovarajuće poruke.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koeficijente A i B: 2 -5
|| x=2.5
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite koeficijente A i B: 0 18.5
|| Jednacina nema resenja.
```

- (k) Napisati program koji za dva data intervala realne prave  $(a_1, b_1)$  i  $(a_2, b_2)$  određuje:

- 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]
```

- (l) Data je funkcija  $f(x) = 2 \cdot \cos(x) - x^3$ . Sa standardnog ulaza se unosi realan broj  $x$  i broj  $k$  koje može biti 1, 2 ili 3. Napisati program koji izračunava  $F(k, x) = f(f(f(\dots f(x))))$  gde je funkcija  $f$  primenjena  $k$ -puta.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom x i k: 2.31 2
|| F(2.31, 2)=2557.516602
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom x i k: 12 1
|| F(12, 1)=-1726.312256
```

- (a) Napisati program koji za uneti broj  $n$  ( $1 \leq n \leq 7$ ) koji predstavlja redni broj dana u nedelji ispisuje ime 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: nedozvoljni unos!
```

- (b) Sa standardnog ulaza se učitavaju dva cela broja i jedan od karaktera +, -, \*, / ili % koji predstavlja operaciju koju treba izvršiti nad unetim brojevima. Napisati program koji korišćenjem *switch* naredbe analizira o kom

## 1 Kontrola toka

---

karakteru je reč i na standardni izlaz ispisuje rezultat. 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!
```

- (c) 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!
```

- (d) 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.
```

- (e) 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.
```

## 1.3 Petlje

### Zadatak 1.56 Tekst

[Rešenje 1.56]

**Zadatak 1.57** Tekst

[Rešenje [1.57](#)]

**Zadatak 1.58** Tekst

[Rešenje [1.58](#)]

**Zadatak 1.59** Tekst

[Rešenje [1.59](#)]

**Zadatak 1.60** Tekst

[Rešenje [1.60](#)]

**Zadatak 1.61** Tekst

[Rešenje [1.61](#)]

**Zadatak 1.62** Tekst

[Rešenje [1.62](#)]

**Zadatak 1.63** Tekst

[Rešenje [1.63](#)]

**Zadatak 1.64** Tekst

[Rešenje [1.64](#)]

**Zadatak 1.65** Tekst

[Rešenje [1.65](#)]

**Zadatak 1.66** Tekst

[Rešenje [1.66](#)]

**Zadatak 1.67** Tekst

[Rešenje [1.67](#)]

**Zadatak 1.68** Tekst

[Rešenje [1.68](#)]

### Zadatak 1.69 Tekst

[Rešenje 1.69]

- (a) Sa standardnog ulaza unosi se 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: -1 1 0 3
-1
```

#### Primer 3

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

- (b) Sa standardnog ulaza unosi se realan broj  $m$ , ceo pozitivan broj  $n$  i  $n$  realnih brojeva. Izračunati i ispisati koliko je brojeva među unetima manje od zadatog broja  $m$ .

#### 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
```

- (c) Sa standardnog ulaza unosi se ceo pozitivan broj  $n$ , a potom i  $n$  karaktera. Za svaki od samoglasnika ispisati koliko puta se pojavio među unetim karakterima. Prilikom implementacije koristiti *switch* naredbu. Ne praviti razliku između malih i velikih slova.

#### Primer 1

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

#### Primer 2

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

- (d) Sa standardnog ulaza unosi se ceo neoznačen broj. Napisati program koji

proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu ili ne.

*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!
```

- (e) Napisati program koji unetom broju uklanja nule sa desne strane. Novodobijeni broj ispisati na standardni izlaz.

*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
```

- (f) Napisati program koji uneti neoznačeni ceo broj transformiše tako što svaku parnu cifru u zapisu broja uveća za 1.

*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
```

- (g) Sa standardnog ulaza unosi se neoznačen ceo broj. Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre polaznog broja. Cifre se posmatraju sa desna na levo.

*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
```

- (h) Sa standardnog ulaza unose se realan broj  $x$  i ceo neoznačen broj  $n$ . Napisati program koji izračunava  $x^n$ .

### 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
```

- (i) Sa standardnog ulaza unose se realan broj  $x$  i ceo broj  $n$ . Napisati program koji izračunava  $x^n$ .

### 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
```

- (j) Sa standardnog ulaza unose se realan broj  $x$  i ceo neoznačen broj  $n$ . Napisati program koji izračunava 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
```

- (k) Sa standardnog ulaza unose se realan broj  $x$  i ceo neoznačen broj  $n$ . Napisati program koji izračunava 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
```



- (l) Napisati program koji sa zadatom tačnošću izračunava sumu  $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$

*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
```

- (m) Napisati program koji sa zadatom tačnošću izračunava sumu  $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} \dots$

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite x: 3
|| Unesite tacnost eps: 0.001
|| S=0.049997
```

*Primer 2*

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

- (n) Sa standardnog ulaza unosi se neoznačen ceo broj. Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem cifara koje su jednake zbiru svojih suseda. Cifre se posmatraju sa desna na levo.

*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: 242
|| 22
```

- (o) Napisati program koji proverava da li je dati prirodan broj palindrom. Broj je palindrom ako se isto čita i sa leve i sa desne strane.

*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!
```

- (p) Sa standardnog ulaza se unosi 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 n brojeva: 18 365 25 1 78
78
```

- (q) Sa standardnog ulaza se unosi 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
365
```

*Primer 2*

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

- (r) Sa standardnog ulaza se unosi ceo pozitivan broj  $n$ , a zatim i  $n$  celih brojeva. Napisati program koji ispisuje broj sa najvećom vodećom cifrom. Vodeća cifra je prva cifra iz zapisa broja. Ukoliko ima više takvih, ispisati prvi.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: 8 964 32 511 27
964
```

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
Unesite n brojeva: 41 669 8
8
```

- (s) Sa standardnog ulaza se unose celi pozitivni brojevi  $n$  ( $n > 1$ ) 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  $d(x, y) = |y - x|$ . Rezultat ispisati na standardni izlaz.

*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
```

- (t) Sa standardnog ulaza se unosi ceo broj  $n$ , a zatim i  $n$  karaktera. Napisati program koji proverava da li se od unetih karaktera može napisati reč *Zima*.

*Primer 1*

```

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

```

*Primer 2*

```

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

```

- (u) Sa standardnog ulaza se unose celi brojevi sve do unosa broja 0. Napisati program koji izračunava i ispisuje razliku najvećeg i najmanjeg unetog broja.

*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

```

- (v) Sa standardnog ulaza se unose realni brojevi sve do unosa broja 0. Napisati program koji izračunava i ispisuje aritmetičku sredinu unetih brojeva.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 8 5.2 6.11 3 0
|| Aritmeticka sredina: 5.5775

```

- (w) Napisati program koji za uneti ceo broj  $n$  iscrtava rub kvadrata dimenzije  $n$ .

*Primer 1*

```

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

```

*Primer 2*

```

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

```

- (x) Napisati program koji za uneti ceo broj  $n$  i karakter  $c$  iscrtava rub jednakostranog pravouglog trougla čije su katete dužine  $n$ .

### Primer 1

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

### Primer 2

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

- (y) Napisati program koji za uneti ceo broj  $n$  iscrtava *krstiće* dimenzije  $n$ .

### Primer 1

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

### Primer 2

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

- (z) Napisati program koji za uneti ceo broj  $n$  iscrtava strelice dimenzije  $n$ .

### Primer 1

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

### Primer 2

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

## 1.4 Funkcije

**Zadatak 1.70** Tekst

[Rešenje [1.70](#)]

**Zadatak 1.71** Tekst

[Rešenje [1.71](#)]

**Zadatak 1.72** Tekst

[Rešenje [1.72](#)]

**Zadatak 1.73** Tekst

[Rešenje [1.73](#)]

**Zadatak 1.74** Tekst

[Rešenje [1.74](#)]

**Zadatak 1.75** Tekst

[Rešenje [1.75](#)]

**Zadatak 1.76** Tekst

[Rešenje [1.76](#)]

**Zadatak 1.77** Tekst

[Rešenje [1.77](#)]

**Zadatak 1.78** Tekst

[Rešenje [1.78](#)]

**Zadatak 1.79** Tekst

[Rešenje [1.79](#)]

**Zadatak 1.80** Tekst

[Rešenje [1.80](#)]

**Zadatak 1.81** Tekst

[Rešenje [1.81](#)]

**Zadatak 1.82** Tekst

[Rešenje [1.82](#)]

**Zadatak 1.83** Tekst

[Rešenje [1.83](#)]

## 1 Kontrola toka

---

- (a) Napisati funkciju *int min(int x, int y, int z)* koja izračunava minimum tri broja. Napisati program koji sa standardnog ulaza 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
```

- (b) Napisati funkciju *unsigned int apsolutna\_vrednost(int x)* koja izračunava apsolutnu vrednost broja *x*. Napisati program koji sa standardnog ulaza 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
```

- (c) Napisati funkciju *float razlomljeni\_deo(float x)* koja izračunava razlomljeni deo broja *x*. Napisati program koji sa standardnog ulaza 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
```

- (d) 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. U slučaju pogrešnog unosa, ispisati poruku o grešci.

### Primer 1

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

### Primer 2

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

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -5  
|| Greska: pogresna dimenzija!
```

- (e) Napisati funkciju *void grafikon\_h(int a, int b, int c, int d)* koja vrši horizontalno prikazivanje 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
*****
**
**
*****
```

- (f) Napisati funkciju *void grafikon\_v(int a, int b, int c, int d)* koja vrši vertikalno prikazivanje 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
*
* *
* *
****
****
```

- (g) Napisati funkciju *int prestupna(int godina)* koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati 1 ako je godina prestupna

## 1 Kontrola toka

---

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!
```

- (h) Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja  $n$ . Napisati program koji sa standardnog ulaza 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!
```

- (i) Napisati funkciju `int ukloni_stotine(int n)` koja modifikuje zadati broj tako što iz njegovog zapisa uklanja cifru stotina (ako postoji). Napisati program koji za brojeve koji se unose sa standardnog ulaza sve do pojave broja 0 ispisuje rezultat primene funkcije.

### 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
```

- (j) Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sa standard-



nog ulaza sve do pojave broja 0 ispisuje rezultat primene funkcije.

#### 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
```

- (k) Napisati funkciju *float aritmeticka\_sredina(int n)* koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji testira rad napisane funkcije. Rezultat ispisivati 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
```

- (l) 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, odnosno 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!
```

- (m) Napisati funkciju *int faktorijel(int n)* koja računa faktorijel broja *n*. Napisati i program koji učitava dva cela broja *x* i *y* ( $0 \leq x, y \leq 12$ ) i ispisuje

## 1 Kontrola toka

---

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
```

- (n) 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 rezultat primene funkcije.

### 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!
```

- (o) Broj je Armstrongov ako je jednak sumi nekih stepena svojih cifara.
- Napisati funkciju *int stepen(int x, int n)* koja izračunava  $n$ -ti stepen broja  $x$ .
  - Napisati funkciju *int armstrong(int x)* koja vraća 1 ako je broj Armstrongov, odnosno 0 ako nije.
  - Napisati program koji za ceo broj koji se unosi sa standardnog ulaza proverava da li je Armstrongov (koristeci funkciju *armstrong*).

### 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!

```

- (p) 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!

```

- (q) Napisati funkciju *int prebrojavanje(float x)* koja prebrojava koliko puta se broj  $x$  pojavljuje u nizu brojeva koji se unose sa standardnog ulaza sve do pojave nule. 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

```

- (r) Napisati funkciju *long int fibonaci(int n)* koja računa  $n$ -ti element Fibonačijevog niza. Fibonačijev niz je niz za koji važi:  $F_0 = 1$ ,  $F_1 = 1$ ,  $F_{n+2} = F_{n+1} + F_n$  za  $n \geq 0$ . 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!

```

- (s) 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 iza njega u abecedi. U suprotnom karakter ostaje nepromenjen. Šifrovanje treba da bude kružno, što znači da je, na primer, šifra za  $c='b'$  i  $k=2$  karakter  $'z'$ . Napisati program koji učitava karakter po karakter do kraja ulaza (do pojave EOF koji se generiše kombinacijom CTRL+D) 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
```

## 1.5 Rešenja

### Rešenje 1.1

```
1  /*
   Napisati program koji na standardni izlaz ispisuje tekst "Zdravo
   svete!"
3  */
   #include<stdio.h>
5
   int main()
7  {
   /* printf - funkcija pomocu koje se vrši ispis */
   /* oznaka \n : prelazak u novi red */
9  printf("Zdravo svete!\n");
11
   /* naredne dve naredbe ispisace reci Zdravo i svete u istom redu*/
13  printf("Zdravo ");
   printf("svete \n");
15
   /* naredne dve naredbe ispisace reci Zdravo i svete u posebnim
   redovima */
17  /* jer se u prvoj printf naredbi na kraju oznakom \n prelazi u novi
   red */

19  printf("Zdravo \n");
   printf("svete \n");
21
```

```
23     return 0;  
    }
```

## Rešenje 1.2

```
1  /*  
   Napisati program koji poziva korisnika da unese dva cela broja sa  
   standardnog ulaza,  
3  a zatim ispisuje:  
   1) unete vrednosti  
5  2) njihov zbir  
   3) njihovu razliku  
7  4) njihov proizvod  
   5) ceo deo pri deljenju jednog broja drugim brojem  
9  6) ostatak pri deljenju jednog broja drugim brojem  
11 */  
13 #include<stdio.h>  
15 int main()  
16 {  
17     int x;  
18     int y;  
19     int rezultat;  
21     printf("Unesi vrednost celobrojne promenljive x:");  
    scanf("%d", &x); /* "%d" - specifikator tipa koji treba uneti (%d  
        za int)  
23                     &x    - adresa promenljive x  
                        */  
25     printf("Unesi vrednost celobrojne promenljive y:");  
27     scanf("%d", &y);  
29     /* 1) ispis unetih vrednosti */  
    printf("x=%d, y=%d\n", x,y); /* umesto prvog %d bice ispisana  
        vrednost promenljive x */  
31                                /* umesto drugog %d bice ispisana  
        vrednost promenljive y */  
33     /* 2) ispis zbira */  
    rezultat = x+y; /* dodelimo vrednost promenljivoj rezultat */  
35     printf("Zbir je %d\n", rezultat);  
37     /* 3) ispis razlike */  
    printf("Razlika je %d\n",x-y); /* mozemo ispisivati direktno  
        vrednost izraza x-y i bez */  
39                                /* njegovog dodeljivanja posebno  
        promenljivoj */
```

## 1 Kontrola toka

```
41      /* 4) ispis proizvoda */
43      printf("%d*d=%d\n",x,y,x*y);

45      /* 5) ispis kolicnika */
      rezultat = x/y;
47      printf("celobrojno deljenje: %d/%d=%d\n",x,y,rezultat); /*
      promenljiva rezultat je celobrojna (int) */                                /* ona ne
                                                                                       /* ukoliko
49      moze sadrzati realan broj */
                                                                                       /*
      je x=7, a y=2, tada ce nakon naredbe */
                                                                                       /*
      rezultat=x/y; promenljiva rezultat imati vrednost 2 */
                                                                                       /* a ne
51      2.5 */

53      printf("ostatak pri celobrojnem deljenju: %d %% %d=%d\n",x,y,x%y);
                                                                                       /*
      operator % izracunava ostatak pri celobrojnem deljenju */
55      vrednost 1 (jer je 7=3*2+1) */
                                                                                       /* 7%2 ima
                                                                                       /* oznaku
      % u naredbi printf pisemo %% */
57      return 0;
}
```

### Rešenje 1.3

```
/*
2      Napisati program koji sa standardnog ulaza ucitava realnu vrednost
      izrazenu
      u incima, konvertuje tu vrednost u centimetre i ispisuje je na
      standardni izlaz
4      zaokruzenu na dve decimale.
*/
6      #include <stdio.h>

8      int main()
      {
10         float in;
          float cm;

12

          printf("Unesi broj inca: ");
14         scanf("%f", &in);
                                                                                       /* "%f" specifikator za unos
          /ispis float promenljivih */

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

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

```
20     return 0;
    }
```

### Rešenje 1.4

```
1  /*
   * Napisati program koji sa standardnog ulaza ucitava duzinu
   * poluprecnika kruga
   * i na standardni izlaz ispisuje njegov obim i povrsinu
   */
5
7  #include <stdio.h>
8  #include <math.h> /* biblioteka matematickih funkcija; za prevodjenje
   * je neophodno ukljuciti opciju -lm
   * npr. gcc primer.c -lm */
9  int main()
10 {
11     int r;
12     float O;
13     float P;
14     printf("Unesi poluprecnik kruga:");
15     scanf("%d", &r);
16
17     O=2*r*M_PI; /* M_PI - konstanta pi koja se nalazi u biblioteci math
   * .h */
18     P=r*r*M_PI;
19
20     printf("Obim: %f, povrsina: %f\n",O,P);
21
22     return 0;
23 }
```

### Rešenje 1.5

```
1  /*
   * Napisati program koji ucitava trocifreni broj koji se
   * unosi sa standardnog ulaza i ispisuje njegove cifre na
   * standardni izlaz.
   */
5
7  #include <stdio.h>
8  int main()
9  {
10     int x;
11     int cifra_jedinice;
12     int cifra_desetice;
13     int cifra_stotine;
14
15     printf("Unesi trocifreni broj:");
```

## 1 Kontrola toka

---

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

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

21     printf("Cifre unetog broja su %d,%d,%d\n", cifra_jedinice,
        cifra_desetice, cifra_stotine);

23     /*
    2. nacin, bez uvođenja dodatnih promenljivih cifra_jedinice,
        cifra_desetice i cifra_stotine:

25     printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10, x/100)
        ;
27     */
    return 0;
29 }
```

### Rešenje 1.6

```
1  /*
    Napisati program koji učitava trocifreni broj koji se
3  unosi sa standardnog ulaza i ispisuje broj dobijen obrtanjem
    njegovih cifara.
5  */
#include <stdio.h>
7  int main()
{
9      int x;
    int obrnuto_x;

11

    int cifra_jedinice;
13    int cifra_desetice;
    int cifra_stotine;

15

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

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

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

25    printf("Obrnuto x: %d\n", obrnuto_x);

27    return 0;
}
```



## Rešenje 1.7

```
2  /*
   Napisati program koji za unetu duzinu stranice jednakostranicnog
   trougla
   ispisuje njegovu površinu.
4  */

6  #include <stdio.h>
   #include <math.h>
8  int main()
   {
10     unsigned int a;
       float P;

12     printf("Unesi duzinu stranice jednakostranicnog trougla:");
14     scanf("%d",&a);

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

18     printf("Povrsina jednakostranicnog trougla stranice %d je %f\n",a
       ,P);
       return 0;
20 }
```

## Rešenje 1.8

```
2  /*
   Napisati program koji za unetu cenu proizvoda ispisuje najmanji
   broj
   novcanica koje je potrebno izdvojiti da bi se proizvod platio. Na
   raspolaganju su novcanice od 1000,100,50,10 i 1 dinar. Na primer,
   za unetu cenu 5178, program na standardni izlaz treba da ispise:
   5178=5*1000+ 1*100 +1*50 +2*10 +8*1
6  */

8  #include <stdio.h>

10 int main()
   {
12     int x;
       printf("Unesi cenu:");
       scanf("%d", &x);

14     printf("%d=%d*1000+ ", x,x/1000);
       x=x%1000;
16     printf("%d*100 +", x/100);
       x=x%100;
18     printf("%d*50 +",x/50);
       x=x%50;
20     printf("%d*10 +",x/10);
       x=x%10;
22     printf("%d*1 +",x/1);
       x=x%1;
       printf("= ");
       while(x>0)
       {
           if(x/1000>0) printf("5*1000+ ");
           if(x/100>0) printf("1*100+ ");
           if(x/50>0) printf("1*50+ ");
           if(x/10>0) printf("2*10+ ");
           if(x/1>0) printf("8*1");
           x=x/1000;
           x=x%1000;
           x=x/100;
           x=x%100;
           x=x/50;
           x=x%50;
           x=x/10;
           x=x%10;
           x=x/1;
           x=x%1;
       }
   }
```

## 1 Kontrola toka

---

```
24     printf("%d*10 +", x/10);  
26     x=x%10;  
    printf("%d*1\n", x);  
    return 0;  
}
```

### Rešenje 1.9

```
1  /*  
   Napisati program koji za tri cela broja koja se unose sa standardnog  
   ulaza  
3  ispisi njihovu aritmeticku sredinu na standardni izlaz.  
   */  
5  
   #include<stdio.h>  
7  
   int main()  
9   {  
       int a, b, c;  
11      float as;  
  
13      printf("Unesi tri cela broja:");  
       scanf("%d%d%d",&a,&b,&c);  
15  
       as=(a+b+c)/3.0; /* da bismo dobili kolicnik, jedan argument mora da  
           bude realan broj */  
17  
       /*  
19      moguće je i:  
       as=1.0*(a+b+c)/3;  
21      ili  
       as=((float)(a+b+c))/3;  
23      */  
  
25      printf("Aritmeticka sredina unetih brojeva je %f\n", as);  
       return 0;  
27  }
```

### Rešenje 1.10

```
1  /*  
   Napisati program koji poziva korisnika da unese dve celobrojne  
   vrednosti,  
3  smesta ih u promenljive x i y, zamenjuje vrednosti tih  
   promenljivih i  
   stampa ih na standardni izlaz.  
5  */  
   #include<stdio.h>  
7   int main()
```

```
{
9   int x,y;
   int t;
11  printf("Unesi dve celobrojne vrednosti:");
   scanf("%d%d",&x,&y);
13  printf("x=%d, y=%d\n",x,y);
   t=x; /* promenljiva t dobija vrednost promenljive x */
15  x=y; /* promenljiva x dobija vrednost promenljive y */
   y=t; /* promenljiva y dobija vrednost promenljive t */
17  printf("nakon zamene, x=%d, y=%d\n",x,y);
   return 0;
19 }
```

Rešenje 1.11

Rešenje 1.12

Rešenje 1.13

Rešenje 1.14

Rešenje 1.15

Rešenje 1.16

Rešenje 1.17

Rešenje 1.18

Rešenje 1.19

Rešenje 1.20

Rešenje 1.21

Rešenje 1.42

```
/*
2  Napisati program koji 10 puta ispisuje tekst "We love C programming".
   */
4
```

```
#include<stdio.h>

6
int main()
8
{
    int i=0;          /* promenljiva i kontrolise koliko puta ce se petlja
        izvorsiti */
10    while (i<10) /* pre ulaska u telo petlje proverava se da li je */
    {                /* ispunjen uslov petlje */
12        printf("We love C programming\n");
        i++; /* operator ++ uvecava i promenljivu za 1
14            i++; ima isti efekat kao i=i+1;
                ili i+=1;

16            ukoliko ne bismo menjali vrednost promenljive i doslo
        bi
18            do beskonacne petlje!
                */
20    }

22    /*
        brojanje u while petlji smo mogli realizovati i preko uslova:
24
        i=1;
26        while(i<=10)
        {
28            ...
        }

        ili
32
        i=2;
34        while(i<=11)
        {
36            ...
        }

        ili
40
        i=3;
42        while(i<13)
        {
44            ...
        }

        Brojanje pocев od 0 uz koriscenje stroge nejednakosti
48        je u duhu programskog jezika C i zato cemo ovaj nacin
        brojanja najcesce koristiti

50    */
    return 0;
52 }
```

## Rešenje 1.43

```
2  /*
4  Napisati program koji poziva korisnika da unese pozitivan ceo broj n
   a potom ispisuje brojeve od 0 do n-1.
6  */
8  #include<stdio.h>
10 int main()
12 {
14     int x;
16     int n;
18     printf("Unesi pozitivan ceo broj:");
20     scanf("%d", &n);
22     x=0;
    while (x<n)
    {
        printf("%d\n", x);
        x++;
    }
    return 0;
}
```

## Rešenje 1.44

```
2  /*
4  Napisati program koji za uneti pozitivan ceo broj
   izracunava njegov faktorijel. Testirati program
   za razlicite vrednosti promenljive x. Obratiti paznju
   da pocev od 23! dolazi do prekoracenja.
6  */
8  #include<stdio.h>
10 int main()
12 {
14     int x;
16     unsigned long f;
18     int i;
20     int original;
22     printf("Unesi x>=0:");
    scanf("%d",&x);
    original=x;
    f=1;
    if (x<0)
        printf("Nekorektan unos\n");
}
```

## 1 Kontrola toka

```
24  else
25  {
26      while (x>1)
27      {
28          f=f*x; /* vrednost izraza sa desne strane naredbe dodele
                        dodeljujemo promenljivoj sa leve strane naredbe
30          dodele
                        */
31          x--; /* operator -- umanjuje vrednost promenljive x za 1
32              naredba x--; ima isti efekat kao x-=1;
33              ili x=x-1;
34              */
35      }
36      printf("%d! = %lu\n",x,f);          /* nekorektno: vrednost
promenljive x je unistena */
37      printf("%d! = %lu\n",original,f); /* korektno: promenljiva
original sadrzi vrednost promenljive x pre ulaska u petlju */
38
39  }
40
41
42  return 0;
43  }
```

### Rešenje 1.45

```
1  /*
2     Napisati program koji poziva korisnika da unese pozitivan ceo broj
3     n,
4     a zatim za unetih n celih brojeva ispisuje sumu pozitivnih i sumu
5     negativnih brojeva.
6
7  */
8
9  #include<stdio.h>
10
11  int main()
12  {
13      int n;
14      int x;
15      int suma_poz;
16      int suma_neg;
17      int i;
18
19      printf("Unesi pozitivan ceo broj:");
20      scanf("%d",&n);
21
22      suma_poz=0; /* promenljivim koje ce sadržati sumu se pre ulaska u
petlju */
23      suma_neg=0; /* dodeljuje se 0 (neutral za sabiranje) */
24      i=0;
```

```
25 while(i<n)
26 {
27     printf("Unesi ceo broj:");
28     scanf("%d", &x);
29
30     if (x<0)
31         suma_neg+=x;
32     else
33         suma_poz+=x;
34
35     i++;
36 }
37
38 printf(" Suma pozitivnih: %d\n Suma negativnih: %d\n", suma_poz,
39        suma_neg);
40 return 0;
41 }
```

## Rešenje 1.46

```
2  /*
3   Napisati program koji omogućava korisniku da unosi cele brojeve dok
4   ne unese nulu. Nakon toga ispisati proizvod onih unetih brojeva
5   koji
6   su pozitivni.
7  */
8
9  #include <stdio.h>
10
11 int main()
12 {
13     int x;
14     int p;
15
16     p=1;
17     while (1) /* izraz 1 je konstantan; razlicit je od nule sto znaci
18                da ga tumacimo kao tacnog */
19     {
20         printf("Unesi jedan ceo broj:");
21         scanf("%d", &x);
22         if (x==0) /* ukoliko je uneta nula */
23             break; /* break prekidamo petlju; izvršavanje se nastavlja
24                    od prve naredbe nakon petlje */
25
26         if (x<0) /* ukoliko je unet negativan broj, tu vrednost ne
27                  zelimo da pomnozimo sa ukupnim proizvodom p; zato moramo
28                  nastaviti dalje */
29             continue; /* sa izvršavanjem petlje; continue prekida
30                        trenutnu iteraciju petlje tako sto preskace sve naredbe
31                        koje nakon njega slede; izvršavanje se
32                        nastavlja od provere uslova petlje */
33     }
34 }
```

## 1 Kontrola toka

---

```
24     p=p*x;
    }
26     printf("Proizvod unetih brojeva je %d\n",p);
28     return 0;
30 }
```

### Rešenje 1.47

```
/*
2   Napisati program koji za uneti ceo broj ispisuje njegove cifre
   u obrnutom poretku.
4 */
6 #include<stdio.h>
7 #include<stdlib.h>
8 int main()
9 {
10     int x;
11     char cifra;
12     printf("Unesi ceo broj:");
13     scanf("%d", &x);
14
15     x = abs(x); /* pretvaranje u apsolutnu vrednost se vrsi za slucaj
   kada je unet
16                 negativan broj kako bismo osigurali da ce nam
   izdvojene cifre
17                 biti pozitivne
18                 */
19
20     while(x>0)
21     {
22         cifra=x%10; /* izdvajamo poslednju cifru broja x */
23         printf("%d\n", cifra);
24         x/=10; /* ako je npr x=1582, x%10 ce biti 2,
   a x/10 ce biti 158;
25                 npr x=5, x%10 ce biti 5
   a x/10 ce biti 0 */
26     }
27
28     return 0;
29 }
30
```

### Rešenje 1.48

```
1 /*
   Napisati program koji omogućava korisniku da unosi karaktere dok ne
   zada tacku i ukoliko je karakter malo slovo,
```



```

3  ispisuje odgovarajuće veliko, ukoliko je karakter veliko slovo
   ispisuje odgovarajuće malo, a u suprotnom ispisuje
   isti karakter kao i uneti.
5  */
7  #include <stdio.h>
9  int main()
10 {
11     int c;
12
13     /* funkcija getchar učitava jedan karakter.
       naredbom dodele (c=getchar()) promenljivoj c bice dodeljena
       vrednost
       ascii koda unetog karaktera
       obratiti paznju na zagrade!
14    */
15    while((c=getchar())!='.')
16    {
17        if (c>='A' && c<='Z')
18            putchar(c+'a'-'A'); /* Razlika izmedju ascii koda svakog malog
19                               i odgovarajućeg velikog slova
                                   je konstanta koja se može sračunati
                                   izrazom 'a'-'A' (i iznosi 32) */
20        else if (c>='a' && c<='z')
21            putchar(c-'a'+'A');
22        else
23            putchar(c);
24    }
25    return 0;
26 }

```

### Rešenje 1.49

```

/*
2  Napisati program koji omogućava korisniku da unosi karaktere dok
   ne zada EOF a potom ispisuje broj velikih slova, broj malih slova
   ,
4  broj cifara, broj belina i zbir cifara.
   */
6
7  #include <stdio.h>
8
9  int main()
10 {
11     /* promenljivoj c dodelicemo povratnu vrednost funkcije getchar()
12     funkcija getchar() učitava jedan karakter sa standardnog ulaza
       i vraća njegov ascii kod; povratna vrednost funkcije getchar je
13     int, pa i promenljiva c mora biti tipa int
14     */

```

## 1 Kontrola toka

```
16  int c;

18  /* brojacji moraju biti inicijalizovani na 0 */
19  int br_v=0;
20  int br_m=0;
21  int br_c=0;
22  int br_b=0;
23  int br_k=0;
24  int suma=0;

26  while((c=getchar())!=EOF)          /* petlja se završava kada
    korisnik ne unese karakter, već zada konstantu EOF */
    {                                  /* ova konstanta se zadaje
    kombinacijom tastera CTRL+D. U tom slučaju, getchar() vraća -1*/
28      if (c>='A' && c<='Z')
          br_v++; /* <=> br_v = br_v+1; */
30      else if (c>='a' && c<='z')
          br_m++;
32      else if (c>='0' && c<='9')
          {
34              br_c++;
              suma=suma+c-'0';          /* funkcija getchar() vraća ascii
    kod unetog karaktera; ascii kodovi cifara 0,1,...,9
36                                          su redom 48,49,...,57; Na primer,
    za unetu 1
                                          promenljiva c će imati vrednost
    49. Zbog toga bi bilo pogrešno računati
38              zbir kao zbir=zbir+c. Promenljivu zbir zato
    računamo kao zbir=zbir+(c-'0')
                                          jer c-'0' će za unetu 0 proizvesti 48-'0' što je
    0,
40              za unetu 1 49-'0' što je 1, za unetu 2 50-'0' što
    je 2, ...*/
          }
42      else if (c=='\t' || c=='\n' || c==' ')
          br_b++;
44
          br_k++;
46      }

48      printf("velika: %d, mala: %d, cifre: %d, beline: %d, svi: %d\n",
    br_v, br_m, br_c, br_b, br_k);
    printf("suma cifara: %d\n", suma);
50
    return 0;
52 }
```

### Rešenje 1.50

```
/* Niz prirodnih brojeva formira se na sledeci nacin:
2 an+1 = an/2 ako je an parno
```

```

an+1 = (3*an+1)/2 ako je an neparno
4 Napisati program koji za uneti pocetni clan niza a0 stampa niz
   brojeva sve do prvog clana jednakog
1.
6 */
#include<stdio.h>
8 int main()
{
10     int a0;
    int an,an1;
12
    printf("Unesi pocetni clan niza brojeva:");
14     scanf("%d",&a0);

16     if (a0>0)
    {
18         printf("%d\n", a0);

20         an=a0;
        while(an!=1)
22         {
            if (an%2) /* ukoliko je vrednost izraza an%2 razlicita od nule,
24             /* izraz se tumaci kao tacan i izvrsavaju se naredbe
            iz if grane */
                an1=(3*an+1)/2;
26         }
            else /* u suprotnom, ukoliko je vrednost izraza an%2 jednaka
            nuli, izraz */
28         { /* se tumaci kao netacan i izvrsavaju se naredbe iz else
            grane */
                an1=an/2;
30         }
            printf("%d\n",an1);
32         an=an1;
        }
34     }
    else
36         printf("Nekorektan unos\n");

38     return 0;
}

```

### Rešenje 1.51

```

1 /*
   Napisati program koji za uneti ceo broja n ispisuje n puta tekst
3   "We love C programming" koriscenjem while, for i do while petlje.
   Obratiti paznju
   na rezultat kada je n<=0.
5 */

```

```
7  #include <stdio.h>

9  int main()
10 {
11
12     int n,m;
13     int i;

14
15     printf("Unesi ceo broj:");
16     scanf("%d",&n);

17
18     /* 1. nacin - while petlja */
19     printf("while: ");

20
21     i=0;
22     while (i<n)          /* uslov petlje se proverava pre ulaska u
23         telo petlje */
24     {
25         printf("We love C programming\n");
26         i++;
27     }

28     printf("\n");

29
30     /* 2. nacin - for petlja */
31     printf("for: ");

32         /* naredba i=0 se izvsava jednom, pre prve
33         iteracije */
34     for(i=0;i<n;i++)      /* uslov petlje i<=m se proverava pre svake
35         iteracije */
36         printf("We love C programming\n"); /* naredba i++ se izvsava
37         nakon svake iteracije */

38     printf("\n");

39
40     /* 3. nacin - do while petlja */
41     printf("do while: "); /* uslov petlje se proverava na kraju svake
42         iteracije */

43         /* zbog toga se do while petlja izvsava
44         bar jednom, cak i u slucaju */
45         /* da uslov petlje nikada nije ispunjen */

46     i=0;
47     do                      /* petlja se zapocinje bez
48         provere uslova */
49     {
50         printf("We love C programming\n"); /* stampa se dati tekst */
51         i++;                               /* uvecava se vrednost
52         promenljive i */
53     }
54     while(i<n);              /* proverava se uslov i
55         ukoliko je ispunjen, nastavlja se sa sledecom iteracijom */
56 }
```

```
49      /* u suprotnom, petlja se
      završava i program se nastavlja od prve naredbe koja sledi za
      petljom */
      printf("\n");
51
      return 0;
53
  }
```

## Rešenje 1.52

```
/*
2   Napisati program koji za uneta dva cela broja n i m ispisuje sve
   cele brojeve
   iz intervala [n,m] koriscenjem while, for i do while petlje.
   Obratiti paznju
4   na rezultat kada je n>m.
*/
6
#include <stdio.h>
8
int main()
10 {
12     int n,m;
13     int i;
14
15     printf("Unesi dva cela broja:");
16     scanf("%d%d",&n,&m);
17
18     /* 1. nacin - while petlja */
19     printf("while: ");
20
21     i=n;
22     while (i<=m)          /* uslov petlje se proverava pre ulaska u
23         telo petlje */
24     {
25         printf("%d ", i);
26         i++;
27     }
28
29     printf("\n");
30
31     /* 2. nacin - for petlja */
32     printf("for: ");
33
34     /* naredba i=n se izvsava jednom, pre prve
       iteracije */
35     for(i=n;i<=m;i++)     /* uslov petlje i<=m se proverava pre svake
       iteracije */
36     {
37         printf("%d ", i); /* naredba i++ se izvsava nakon svake
       iteracije */
38     }
```

```
36     printf("\n");
38     /* 3. nacin - do while petlja */
    printf("do while: "); /* uslov petlje se proverava na kraju svake
        iteracije */
40                                /* zbog toga se do while petlja izvršava
        bar jednom, čak i u slučaju */
                                /* da uslov petlje nikada nije ispunjen */
42     i=n;
    do                                /* petlja se zapocinje bez provere uslova */
44     {
        printf("%d ",i); /* stampa se vrednost promenljive i */
46         i++;           /* uvecava se vrednost promenljive i */
    }
48     while(i<=m); /* proverava se uslov i ukoliko je ispunjen,
        nastavlja se sa sledecom iteracijom */
                                /* u suprotnom, petlja se završava i program
        se nastavlja od prve naredbe koja sledi za petljom */
50     printf("\n");
    return 0;
52 }
```

### Rešenje 1.53

```
/*
2  Program izracunava minimum n unetih brojeva.
  Npr. za n=4 i brojeve 3 8 2 9 program ispisuje 2
  */
#include <stdio.h>
6  int main()
  {
8      int n, i;
        float x, min;
10
12     printf("Unesi n>0:");
        scanf("%d", &n);
14     if (n<=0)                                /* ako je unos neispravan */
        {
16         printf("Neispravan unos\n");
            return -1; /* prekidamo izvršavanje
                programa pomocu naredbe return */
18     }                                /* u slučaju greske kao što je
        neispravan unos vraćamo vrednost -1 */
        printf("Unesi realan broj:");
20     scanf("%f", &x); /* prvi broj je unet izvan petlje */
        min=x;         /* kako bi bio njegova vrednost bila
        dodeljena promenljivoj min */
22                                /* neophodno je da promenljiva min
        bude inicijalizovana pre ulaska u petlju */
    }
```

```

/* da bi uslov x<min mogao da bude
ispitan u prvoj iteraciji */
24 i=0;
while(i<(n-1))
26 {
    printf("Unesi realan broj:");
    scanf("%f", &x);
    if(x<min)
30     min=x;
    i++;
32 }
printf("Minimum je: %f\n", min);
34 return 0;
}

```

## Rešenje 1.54

```

/*
2   a) Napisati program za uneti pozitivan ceo broj n ispisuje zbir
   s = 1+2^3+3^3+...+n^3. Na primer, za n=4, izlaz iz programa
4   treba da bude:
   Suma kubova od 1 do 4 je 100
6   b) Modifikovati program tako da ispisuje zbir s = 1+2^3+3^3+...+k
   ^3
   za svako i od 1 do n. Na primer, za n=4, izlaz iz programa
   treba da
8   bude:
i=1, n=1
10 i=2, n=9
i=3, n=36
12 i=4, n=100

14 */

16 #include <stdio.h>

18 int main()
{
20     int n;
    int i;
22     int s;

24     printf("Unesite jedan pozitivan ceo broj:");
26     scanf("%d", &n);

28     if (n<0)
        return -1;

30     i=1;
32     s=0; /* inicijalizacija promenljive u kojoj se cuva suma kubova */

```

## 1 Kontrola toka

---

```
34  for(i=1;i<=n;i++)
35  {
36      s+=i*i*i;
37      /* b) */
38      printf("i=%d, s=%d\n", i, s);
39  }
40  /* a) */
41  printf("Suma kubova od 1 do %d: %d\n", n, s);
42  return 0;
43  }
```

### Rešenje 1.55

```
1  /*
2  Napisati program koji ispisuje sve prave delioce unetog pozitivnog
3  celog broja.
4
5  */
6
7  #include<stdio.h>
8  #include<math.h>
9  int main()
10 {
11     int x;
12     int i;
13
14     printf("Unesi x>0:");
15     scanf("%d", &x);
16
17     if (x<=0)
18     {
19         printf("Neispravan unos\n");
20         return -1;
21     }
22
23     /* 1. nacin */
24     printf("----- 1. nacin ----- \n");
25     for(i=2;i<x;i++)
26     {
27         printf("proveravam za %d...\n",i);
28         if (x%i==0)
29             printf("\t delilac:%d \n",i);
30     }
31     /* 2. nacin (brzi) */
32     printf("----- 2. nacin ----- \n");
33     for(i=2;i<=sqrt(x);i++)
34     {
35         printf("proveravam za %d...\n",i);
36         if (x%i==0)
```



```
        if (i==x/i) /* u slucaju kada je delilac koren broja, npr 4
za 16, ispisujemo ga jednom */
37         printf("\t delilac:%d \n",i);
        else /* u suprotnom, npr 2 za 16, ispisujemo i 2 i 8
*/
39         printf("\t delioci:%d %d \n",i,x/i);
    }
41     return 0;
}
```

## Rešenje 1.36

```
/*
2  Napisati program koji ispituje da li je uneti broj prost.
*/
4
#include<stdio.h>
6  #include<math.h>
int main()
8  {
    int x;
10   int i;
    int indikator;

12   printf("Unesi x>0:");
14   scanf("%d", &x);

16   if (x<=0)
    {
18       printf("Neispravan unos\n");
        return -1;
20   }

22   indikator=0;
    for(i=2;i<x;i++)
24   {
        printf("proveravam za %d\n",i);
        if (x%i==0) /* cim pronadjemo prvog delioca, znamo
da broj nije prost. prekidamo petlju */
26         {
            indikator=1;
30             break;
        }
32   }

34   if (indikator==0)
        printf("jeste\n");
36   else
        printf("nije\n");
38 }
```

## 1 Kontrola toka

---

```
    return 0;
40 }
```

### Rešenje 1.37

```
/*
2  Napisati program koji ispituje da li je uneti broj prost.
*/
4
6  #include<stdio.h>
7  #include<math.h>
8  int main()
9  {
10     int x;
11     int i;
12     int indikator;
13
14     printf("Unesi x>0:");
15     scanf("%d", &x);
16
17     if (x<=0)
18     {
19         printf("Neispravan unos\n");
20         return -1;
21     }
22
23     indikator=0;
24     for(i=2;i<=sqrt(x);i++) /* dovoljno je da ispitamo delioce do <=
sqrt(x) */
25     {
26         printf("proveravam za %d\n",i);
27         if (x%i==0) /* cim pronadjemo prvog delioca, znamo
da broj nije prost. prekidamo petlju */
28         {
29             indikator=1;
30             break;
31         }
32     }
33
34     if (indikator==0)
35         printf("jeste\n");
36     else
37         printf("nije\n");
38
39     return 0;
40 }
```

### Rešenje 1.38

```
2  /*
   * Napisati program koji ispituje da li je uneti broj prost.
   */
4
6  #include<stdio.h>
7  #include<math.h>
8  int main()
9  {
10     int x;
11     int i;
12     int indikator;
13
14     printf("Unesi x>0:");
15     scanf("%d", &x);
16
17     if (x<=0)
18     {
19         printf("Neispravan unos\n");
20         return -1;
21     }
22
23     if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
24         indikator=0;
25     else if (x%2==0) /* parni brojevi nisu prosti */
26         indikator=1;
27     else /* za neparne brojeve ispituje da li imaju
28         delioce */
29     {
30         indikator=0;
31         for(i=3;i<=sqrt(x);i+=2) /* kandidati za delioce neparnih
32         brojeva su neparni brojevi */
33         {
34             printf("proveravam za %d\n",i);
35             if (x%i==0) /* cim pronadjemo prvog delioca, znamo da
36             broj nije prost. prekidamo petlju */
37             {
38                 indikator=1;
39                 break;
40             }
41         }
42     }
43
44     if (indikator==0)
45         printf("jeste\n");
46     else
47         printf("nije\n");
48
49     return 0;
50 }
```

### Rešenje 1.39

```
1  /*
   Napisati C program koji učitava sa standardnog ulaza cele brojeve dok
   ih je manje od 10 ili
3  dok ne naidje na nulu. Unetu nulu tretiramo kao validan unos. Program
   treba da ispise na standardni
   izlaz broj sa maksimalnom poslednjom cifrom.
5  Ako ima vise takvih brojeva neka ispise:
   a) prvi unet broj sa maksimalnom poslednjom cifrom
7  b) poslednji unet broj sa maksimalnom poslednjom cifrom
   */
9
11 #include <stdio.h>
12 #include <stdlib.h>
13
14 int main()
15 {
16     int n;
17     int x;
18     int x_sa_max_cifrom;
19     char max_cifra;
20     char cifra;
21
22     max_cifra = -1;
23     /*
24      S obzirom da su cifre brojevi iz konacnog skupa
25      {0,1,2,3,4,5,6,7,8,9},
26      maksimalnu cifru inicijalizujemo na -1; u prvom prolazu kroz petlju
27      ,
28      max_cifra ce promeniti vrednost (jer je svaka cifra >-1)
29      */
30
31     for(n=0;n<10;n++)
32     {
33         printf("Unesi jedan ceo broj:");
34         scanf("%d",&x);
35
36         cifra=abs(x)%10;
37
38         printf("%d\n",cifra);
39         /* a) */
40         /* if(cifra>max_cifra) */
41         /* b) */
42         if(cifra>=max_cifra)
43         {
44             x_sa_max_cifrom=x;
45             max_cifra = cifra;
46         }
47
48         if (x==0)
49             break;
50     }
51 }
```

```

47 }
49 printf("Broj sa najvecom poslednjom cifrom: %d\n", x_sa_max_cifrom)
   ;
   return 0;
51 }

```

### Rešenje 1.40

```

1  /*
   Napisati program koji za uneto n>0 ispisuje:
3  a) brojeve od 1 do n*n po n u redu
   b) tablicu mnozenja od 1 do n*n
5  c) tabelu koja za n=10 izgleda ovako:
   1  2  3  4  5  6  7  8  9  10
7  2  3  4  5  6  7  8  9  10  1
   3  4  5  6  7  8  9  10  1  2
9  4  5  6  7  8  9  10  1  2  3
   5  6  7  8  9  10  1  2  3  4
11 6  7  8  9  10  1  2  3  4  5
   7  8  9  10  1  2  3  4  5  6
13 8  9  10  1  2  3  4  5  6  7
   9  10  1  2  3  4  5  6  7  8
15 10 1  2  3  4  5  6  7  8  9
   d) kvadrat nxn zvezdica; za n=10:
17 *****
   *****
19 *****
   *****
21 *****
   *****
23 *****
   *****
25 *****
   *****
27 e) donjetrougaonu matricu zvezdica; za n=10:
   *
29 **
   ***
31 ****
   *****
33 *****
   *****
35 *****
   *****
37 *****
   f) gornjetrougaona matricu zvezdica; za n=10:
39 *****
   *****
41 *****
   *****

```

```

43      *****
44          *****
45              ****
46                  ***
47                      **
48                          *
49      g) klin; za n=10:
50  * * * * *
51  * * * * *
52      * * * * *
53  * * * * *
54      * * * * *
55  * * * * *
56      * * * *
57          * * *
58              * *
59                  *
60  */
61  #include <stdio.h>
62  int main()
63  {
64      int n;
65      int i,j;
66      printf("Unesi prirodan broj:");
67      scanf("%d",&n);
68
69      /*
70       U svim varijantama zadatka potrebno je prikazati tabelu.
71       Svaka tabela se sastoji iz vrsta i kolona. Tabela se stampa
72       vrstu po vrstu, a unutar vrste stampa se jedno po jedno polje.
73
74       Zadatak resavamo pomocu dvostrukih petlji. Brojacka promenljiva
75       u spoljasnjoj petlji je i, a u unutrasnjoj j. U svakoj iteraciji
76       petlje stampamo polje koje se nalazi u i-toj vrsti i j-toj
77       koloni. Koja ce se vrednost nalaziti u tom polju, zavisi od
78       varijante zadatka.
79
80       Prelazak u novi red stampamo na kraju svake vrste.
81      */
82
83      /* a */
84      printf("brojevi od 1 do %d, po %d u redu \n",n*n,n);
85      for (i=1;i<=n;i++) /* imamo n vrsta */
86      {
87          for (j=1; j<=n; j++) /* u svakoj vrsti imamo n elemenata*/
88              printf("%3d ", (i-1)*n+j); /* za i=1: 1,2,3,...,n */
89                                          /* za i=2: 10,11,12,...,10+n */
90                                          /* ... */
91
92          printf("\n"); /* prethodna for petlja stampa polja jednog reda
93                           tabele
94
95                           na kraju svakog reda tabele neophodno je preci

```

```

        u novi red
        */
95     }
97
98     /* b */
99     printf("tablica mnozenja od 1 do %d \n",n);
100     for (i=1;i<=n;i++)
101     {
102         for (j=1; j<=n; j++)
103             printf("%3d ", i*j);    /* u tablici mnozenja vrednost svakog
104                                     polja je proizvod
105                                     vrste i kolone u kojoj se nalazi
106                                     */
107         printf("\n");
108     }
109
110     /* c */
111     printf("brojevi od 1 do %d rotirani ulevo \n",n);
112     for (i=1;i<=n;i++)
113     {
114         /* i oznacava broj vrste
115            svaka vrsta je zarotirana za i-1 mesta ulevo;
116            npr. za n=10, 3. red ce biti
117            3 4 5 6 7 8 9 10 1 2
118            */
119
120         /*
121            na pocetku svakog reda stampamo vrednosti od i do n
122            npr. za n=10, u 3. redu prvo stampamo
123            3 4 5 6 7 8 9 10
124            */
125         for (j=i; j<=n; j++)
126             printf("%3d ", j);
127
128         /*
129            nakon toga, dopunimo red vrednostima koje nedostaju
130            npr. za n=10, u 3. redu to su
131            1 2
132            */
133         for (j=1; j<i; j++)
134             printf("%3d ", j);
135
136         printf("\n");
137     }
138
139     /* d */
140     printf("kvadrat \n");
141     for (i=0; i<n; i++)
142     {
143         /*
144            kvadrat predstavlja tabelu sa n vrsta

```

```
143         gde svaka vrsta sadrzi n polja, a svako
           polje je isto i predstavlja karakter *
145     */
           for (j=0; j<n; j++)
147         printf("*");
           printf("\n");
149     }

151     /* e */
           printf("donjetrougaona matrica\n");
153     for (i=0; i<n; i++)
           {
155         /*
           umesto cele tabele kvadrata zvezdica, stampamo
157         samo zvezdice koje se nalaze ispod glavne dijagonale,
           ukljucujuci i glavnu dijagonalu;

           za njihovu poziciju (i,j) u tabeli vazi da je
161         redni broj vrste i veci ili jednak rednom broju kolone j

163         */
           for (j=0; j<n; j++)
165             if (i>=j)
               printf("*");
           printf("\n");
167     }

169     /* f */
           printf("gornjetrougaona matrica\n");
171     for (i=0; i<n; i++)
           {
173         /*
           umesto cele tabele kvadrata zvezdica, stampamo
175         samo zvezdice koje se nalaze iznad glavne dijagonale,
           ukljucujuci i glavnu dijagonalu;

177         za njihovu poziciju (i,j) u tabeli vazi da je
           redni broj vrste i manji ili jednak rednom broju kolone j

179         S obzirom da ispis zvezdica ne pocinje od pocetka reda,
           nephodno je da u slucaju da ne stampamo zvezdicu
183         odstampamo razmak kako bismo pravilno pozicionirali
           pocetak stampanja reda

185         */
           for (j=0; j<n; j++)
187             if (i<=j)
               printf("*");
           else
191             printf(" ");
           printf("\n");
193     }
```



```

195  /* g */
197  printf("klin \n");
199  for (i=0; i<n; i++)
201  {
202      /*
203       * kod klina, i-ti red pocinje sa i razmaka
204       * nakon cega se n-i puta stampaju zajedno zvezdica
205       * i razmak ("* ")
206       */
207      for (j=0; j<i; j++)
208          printf(" ");
209      for (j=0; j<n-i; j++)
210          printf("* ");
211      printf("\n");
212  }
213  return 0;
214  }

```

### Rešenje 1.41

```

/*
2  Sa standardnog ulaza unosi se broj n. Napisati program koji ispisuje
   brojeve od 1 do n, zatim
   od 2 do n - 1, 3 do n - 2, itd. i na kraju ispisuje njihovu sumu. Za
   neispravan unos, program ispisuje broj -1.
4  Na primer, za n=7, program treba da ispise
   1 2 3 4 5 6 7 2 3 4 5 6 3 4 5 4
   S=64
6  */
8
#include<stdio.h>
10
11 int main()
12 {
13     int n;
14     int i,j;
15     int S=0;
16
17     printf("Unesi jedan prirodan broj:");
18     scanf("%d",&n);
19
20     if (n<=0)
21     {
22         printf("Neispravan unos\n");
23         return -1;
24     }
25
26     for(i=0;i<n;i++)

```

## 1 Kontrola toka

---

```

    for (j=i+1;j<n-i+1;j++)
28     {
        printf("%d ",j);
30         S+=j;
    }
32     printf("\n");
    printf("S=%d\n",S);
34     return 0;
}
```

### Rešenje 1.42

```

1  /*
    Napisati program koji za uneto vreme ispisuje koliko je sati i
    minuta ostalo
3  do ponoci.
    */
5  #include<stdio.h>
7  int main()
    {
9      int sati;
      int minuti;
11     int preostali_sati;
      int preostali_minuti;
13
      printf("Unesi vreme (broj sati u intervalu [0,24), broj minuta u
          intervalu [0,60]):");
15     scanf("%d%d",&sati,&minuti);

17     preostali_sati = 24-sati-1;
      preostali_minuti = 60-minuti;
19     if (preostali_minuti==60)
    {
21         preostali_sati++;
        preostali_minuti=0;
23     }

25     printf("Do ponoci je ostalo %d sati i %d minuta\n", 24-sati-1, 60-
        minuti);
      return 0;
27 }
```

### Rešenje 1.43

```

1  /*
    Napisati program koji za uneti ceo broj ispisuje njegovu reciprocnu
    vrednost.

```

```

3  Ukoliko je uneti broj jednak nuli, ispisati poruku "Nedozvoljeno
   deljenje nulom".
   */
5
   #include <stdio.h>
7
   int main()
9   {
       int x;
       float rx;
11
       printf("Unesi jedan ceo broj:");
       scanf("%d",&x);
13
       /*
15      obratiti paznju:
17      x==0 - relacija jednakosti (da li je vrednost promenljive x
          jednaka nuli)
19      x=0 - naredba dodele (promenljiva x dobija vrednost nula)
       */
21
       if (x==0)
23         printf("Nedozvoljeno deljenje nulom\n");
       else
25       {
           rx = 1.0/x;
27         printf("Reciprocna vrednost unetog broja:%f\n",rx);
       }
29
       return 0;
31 }

```

### Rešenje 1.44

```

1  #include <stdio.h>
   /*
3  Napisati program koji za uneti ceo broj x ispisuje da li je jednak
   nuli,
   manji od nule ili veci od nule.
   */
5
   int main()
7   {
       int x;
9       printf("Unesi ceo broj:");
       scanf("%d",&x);
11
       /*
13      obratiti paznju:
          x==0 - relacija jednakosti (da li je vrednost promenljive x
          jednaka nuli)
15      x=0 - naredba dodele (promenljiva x dobija vrednost nula)
       */

```

## 1 Kontrola toka

---

```
17  */
    if (x==0)
        printf("Broj je jednak nuli\n");
19  else if (x<0)
        printf("Broj je manji od nule\n");
21  else
        printf("Broj je veci od nule\n");
23
    return 0;
25 }
```

### Rešenje 1.45

```
1  /*
   Napisati program koji za godinu koja se unosi sa standardnog ulaza
   na standardni izlaz
3  ispisuje da li je prestupna.
   */
5
   #include <stdio.h>
7
   int main()
9   {
       int x;
11      printf("Unesi godinu:");
       scanf("%d",&x);
13
       if ((x%4==0 && x%100!=0) || x%400==0)
15          printf("Godina je prestupna\n");
       else
17          printf("Godina nije prestupna\n");
       return 0;
19   }
```

### Rešenje 1.46

```
1  /*
   Napisati program koji za 2 cela broja uneta sa standardnog ulaza
3  ispisuje njihov minimum na standardni izlaz.
   */
5
   #include <stdio.h>
7   int main()
   {
9       int a,b;
       int min1;
11      int min2;
       int min3;
13  }
```

```

15     scanf("%d%d",&a,&b);

17     /* 1. nacin */
18     if (a<b)
19         min1=a;
20     else
21         min1=b;

23     printf("Minimum unetih brojeva (1.nacin) je %d\n",min1);

25     /* 2. nacin */
26     min2 = (a<b) ? a : b;
27     printf("Minimum unetih brojeva (2.nacin) je %d\n",min2);

29     /* 3. nacin */
30     min3=a;
31     if (b<a)
32         min3 = b;
33     printf("Minimum unetih brojeva (3.nacin) je %d\n",min3);

35     return 0;
}

```

### Rešenje 1.47

```

/*
2  a) Napisati program koji za 3 cela broja uneta sa standardnog ulaza
3  ispisuje njihov minimum na standardni izlaz.
4  b) Neka uneti brojevi predstavljaju cene artikla. Ukoliko se
5  najjeftiniji
6  artikal dobija za 1 dinar, napisati kolika je ukupna cena, kao i
7  koliko
8  dinara se uštedi zahvaljujuci popustu.
9  */

10 #include <stdio.h>
11 int main()
12 {
13     int a,b,c;
14     int min;
15     int min1;
16     int min2;
17     int cena_bez_popusta, cena_sa_popustom;

18     scanf("%d%d%d",&a,&b,&c);

19
20     if (a<b)
21         if (a<c) /* poredak: a<b,a<c => a,b,c ili a,c,b */
22             min=a;

```

```
24     else      /* poredak: a<b, a>=c => a<b, c<=a => c,a,b */
        min=c;
26     else      /* b<=a */
        if (b<c) /* poredak: b<=a,b<c => b,a,c ili b,c,a */
            min=b;
28     else      /* poredak: b<=a, c<=b => c,b,a */
        min=c;
30
32     printf("Minimum unetih brojeva (1.nacin) je %d\n",min);
34
    /* 2. nacin */
    /* najpre odredimo minimum brojeva a,b*/
36     if (a<b)
        min1=a;
38     else
        min1=b;
40
    if (c<min1)
42         min1=c;
    printf("Minimum unetih brojeva (2.nacin) je %d\n",min1);
44
    /* 3. nacin */
46     min2=a;
    if(min2>b)
48         min2=b;
    if(min2>c)
50         min2=c;
52
    printf("Minimum unetih brojeva (3.nacin) je %d\n",min2);
54
    cena_bez_popusta=a+b+c;
    cena_sa_popustom = cena_bez_popusta - min2 + 1;
56
    printf("Cena sa popustom: %.2f\n Cena bez popusta: %d\n Usteda:
    %.2f\n", cena_sa_popustom, cena_bez_popusta, cena_bez_popusta-
    cena_sa_popustom);
58
    return 0;
60 }
```

### Rešenje 1.48

```
/*
2  Napisati program koji za koeficijente kvadratne jednacine
   koji se unose sa standardnog ulaza na standardni izlaz
4  ispisuje koliko realnih resenja jednacina ima i ako ih ima, ispisuje
   resenja jednacine
   zaokružena na dve decimale.
6  */
   #include <stdio.h>
8  #include <math.h>
```

```

10 int main()
11 {
12     float a,b,c;
13     float D;
14     float x1,x2;
15     printf("Unesi koeficijente kvadratne jednacine:");
16     scanf("%f%f%f",&a,&b,&c);
17
18     /* proveravamo da li je kvadratna jednacina korektno zadata */
19     if (a==0)
20         if (b==0)
21             if (c==0) /* slucaj a==0 && b==0 && c==0 */
22                 printf("Jednacina ima beskonacno mnogo resenja\n");
23             else /* slucaj a==0 && b==0 && c!=0 */
24                 printf("Jednacina nema resenja\n");
25         else /* slucaj a!=0 && b!=0 */
26         {
27             x1=-c/b;
28             printf("Jednacina ima jedinstveno realno resenje %.2f\n",x1);
29         }
30     else /* slucaj a!=0 */
31     {
32         D=b*b-4*a*c; /* funkcija sqrt nalazi se u biblioteci math.h (
33         prevodjenje sa -lm opcijom) */
34         if (D<0)
35             printf("Jednacina nema realnih resenja\n");
36         else if (D>0)
37         {
38             x1 = (-b+sqrt(D))/(2*a);
39             x2 = (-b-sqrt(D))/(2*a);
40             printf("Jednacina ima dva razlicita realna resenja %.2f i %.2
41             f\n",x1,x2);
42         }
43         else
44         {
45             x1 = (-b)/(2*a);
46             printf("Jednacina ima jedinstveno realno resenje %.2f\n",x1);
47         }
48     }
49     return 0;
50 }

```

### Rešenje 1.49

```

1  /*
2
3  Napisati program koji za karakter koji ucitava jedan karakter i :
4  - u slucaju da je uneta cifra, ispisuje nju i njen ascii kod

```

## 1 Kontrola toka

```
5 - u slucaju da je uneto malo slovo, ispisuje njega, njegov ascii kod,
   odgovarajuće veliko slovo i njegov ascii kod
- u slucaju da je uneto veliko slovo, ispisuje njega, njegov ascii
  kod, odgovarajuće malo slovo i njegov ascii kod
7 - u ostalim slucajevima, ispisuje uneti karakter i njegov ascii kod
  */
9 #include <stdio.h>
int main()
11 {
    char c;
13     printf("Unesi jedan karakter:");
    scanf("%c", &c);

15     if (c>='0' && c<='9')
17         printf("cifra:%c ascii:%d\n",c,c);
    else if (c>='A' && c<='Z')
19         printf("veliko slovo:%c ascii:%d odgovarajuće malo:%c, ascii:%d
        \n",c,c,c-'A'+ 'a',c-'A'+ 'a'); /* Razlika izmedju ascii koda
        svakog malog i odgovarajućeg velikog slova

                                   je konstanta koja se moze
        sracunati izrazom 'a'-'A' (i iznosi 32) */
21     else if (c>='a' && c<='z')
        printf("malo slovo:%c ascii:%d odgovarajuće veliko:%c, ascii:%d
        \n",c,c,c-'a'+ 'A',c-'a'+ 'A');
23     else
25         printf("karakter:%c ascii:%d\n",c,c);

    return 0;
27 }
```

### Rešenje 1.50

```
1 /*
3 Napisati program koji ucitava tri cela broja i ispisuje zbir onih
   unetih brojeva
   koji su pozitivni.
5
   */
7 #include<stdio.h>
int main()
9 {
    int a,b,c;
11     int s;
    printf("Unesi prvi ceo broj:");
13     scanf ("%d",&a);
    printf("Unesi drugi ceo broj:");
15     scanf ("%d",&b);
    printf("Unesi treci ceo broj:");
17     scanf ("%d",&c);
```



```

19  s=0; /* inicijalizujemo promenljivu s na nulu */
21  if (a>0)
    s=s+a; /* naredba dodele: vrednost izraza a desne strane znaka
jednakosti
23      dodeljujemo promenljivoj sa leve strane znaka
jednakosti.
        Staru vrednost promenljive s saberemo sa vrednoscu
promenljive a
25      i dobijenu vrednost upisemo u promenljivu s */

27  if (b>0)
    s+=b; /* operator +=
29      s+=b je skraceni zapis za s=s+b
        */

31  if (c>0)
33      s+=c;

35  printf("Suma unetih pozitivnih brojeva: %d\n",s);
    return 0;
37 }

```

### Rešenje 1.51

```

1  /*
3  Napisati program koji za realan broj unet sa standardnog ulaza
    ispisuje njegovu apsolutnu vrednost.
5
6  */
7
9  #include<stdio.h>
10 #include<math.h>
11 #include<stdlib.h>
12 int main()
13 {
14     float x;
15     float y;
16
17     printf("Unesi jedan realan broj:");
18     scanf("%f",&x);
19
20     /* 1. nacin */
21     if (x>0)
22         y=x;
23     else
24         y=-x;
25     printf("Apsolutna vrednost broja %f je %f\n",x,y);

```

## 1 Kontrola toka

---

```
/* 2. nacin */
27 y=x;
   if (y<0)
29     y=-y;

31 printf("Apsolutna vrednost broja %f je %f\n",x,y);

33 /* 3. nacin - pogresan!*/
y=abs(x); /* funkcija abs vraća ceo broj! za računanje apsolutne
vrednosti realnog broja treba koristiti funkciju fabs */
35 /* funkcija abs se nalazi u zaglavlju stdlib.h */
printf("Apsolutna vrednost broja %f je %f\n",x,y);

37

/* 4. nacin */
39 y=fabs(x); /* funkcija fabs se nalazi u zaglavlju math.h */
printf("Apsolutna vrednost broja %f je %f\n",x,y);
41 return 0;
}
```

### Rešenje 1.52

```
2 /*
   Napisati program koji poziva korisnika da unese jedan karakter i
   ispisuje
4   da li je uneti karakter samoglasnik.
   */

6 #include <stdio.h>

8

10 int main()
{
   char c;
12   printf("Unesi jedan karakter:");
   scanf("%c", &c);
14   switch(c)
   {
16     case 'A' :
18     case 'E' :
20     case 'I' :
22     case 'O' :
24     case 'U' :
26     case 'a' :
28     case 'e' :
30     case 'i' :
32     case 'o' :
34     case 'u' : printf("Uneli ste samoglasnik\n");
36               break;
38     default : printf("Niste uneli samoglasnik\n");
39               break;
40   }
}
```

```
30     return 0;
32 }
```

### Rešenje 1.53

```
1  /*
2  Napisati program koji za uneti dan i mesec ispisuje godisnje doba kom
3  pripadaju. Mozemo podrazumevati da je unos korektan.
4  */
5
6  #include <stdio.h>
7
8  int main()
9  {
10     int d,m;
11     printf("Unesi dan i mesec");
12     scanf("%d%d",&d,&m);
13
14     switch(m) /* argument u naredbi switch mora biti celobrojna
15                promenljiva */
16     {
17         case 1: /* argument u naredbi case mora biti celobrojna
18                  konstanta */
19             case 2: /* ispitujemo da li je m==2 */
20                 printf("zima\n");
21                 break;
22             case 3:
23                 if (d<21)
24                     printf("zima\n");
25                 else
26                     printf("prolece\n");
27                 break;
28             case 4:
29             case 5:
30                 printf("prolece\n");
31                 break;
32             case 6:
33                 if (d<21)
34                     printf("prolece");
35                 else
36                     printf("leto");
37                 break;
38             case 7:
39             case 8:
40                 printf("leto");
41                 break;
42             case 9:
43                 if (d<23)
44                     printf("leto\n");
45                 else
```

```
        printf("jesen\n");
45     break;
    case 10:
47     case 11:
        printf("jesen\n");
49     break;
    case 12:
51     if (d<22)
        printf("jesen\n");
53     else
        printf("zima\n");
55 }
    return 0;
57 }
```

### Rešenje 1.54

```
1  /*
   Napisati program koji od korisnika zahteva da unese
3  cetvorocifreni broj. Program za taj broj proverava
   da li su cifre uredjene rastuce, opadajuce ili nisu
5  uredjene i stampa odgovarajucu poruku na standardni
   izlaz. Voditi racuna o nekorektnim unosima. Na primer,
7  pokretanje programa moze da izgleda ovako:

9  Unesi jedan cetvorocifreni broj: -1357
   Cifre su mu uredjene neopadajuce.

11 ili ovako

13 Unesi jedan cetvorocifreni broj: 9952
   Cifre su mu uredjene nerastuce.

17 ili ovako

19 Unesi jedan cetvorocifreni broj: 9572
   Cifre su mu nisu uredjene.

21 Unesi jedan cetvorocifreni broj: 123
23 Uneti broj nije cetvorocifren.

25 */

27 #include <stdio.h>
29 #include <stdlib.h>

31 int main()
   {
33     int x;
        char c1;    /* cifre su brojevi {0,1,2,3,4,5,6,7,8,9} */
```

```

35 char c10;
36 char c100;
37 char c1000;

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

41 x=abs(x); /* u slucaju da je broj negativan, uzimamo njegovu
         apsolutnu vrednost
42         kako ne bismo za cifre dobili negativne brojeve */
43         /* funkcija abs nalazi se u zaglavlju stdlib.h */

45 if (x<1000 || x>9999)
46     printf("Uneti broj nije cetvorocifren\n");
47 else
48 {
49     c1 = x%10;
50     c10 = (x/10)%10;
51     c100 = (x/100)%10;
52     c1000 = (x/1000)%10;

53     printf("Cifre broja: %d,%d,%d,%d\n",c1000,c100,c10,c1);

54     if (c1000<=c100 && c100<=c10 && c10<=c1)
55         printf("Cifre su uredjene neopadajuće \n");
56     else if (c1000>=c100 && c100>=c10 && c10>=c1)
57         printf("Cifre su uredjene nerastuće \n");
58     else
59         printf("Cifre nisu uredjene\n");
60 }
61 return 0;
62 }

```

### Rešenje 1.55

```

1 /*
2 Sa standardnog ulaza unose se jedan karakter i 8 realnih brojeva koji
   predstavljaju
3 koordinate cetiri tacke: A(x1, y1), B(x2, y2), C(x3, y3), D(x4, y4).
   Na osnovu unetog karaktera
4 ispisuje se odgovarajuca poruka na standardni izlaz:
5 k - proverava da li su date tacke temena pravougaonika cije su
   stranice paralelne koordinatnim osama i
   u slucaju da jesu, ispisuje obim datog pravougaonika; mozemo
   podrazumevati da ce korisnik koordinate tacaka
6 unosi redom A,B,C,D, pri cemu ABCD opisuje pravougaonik cije su
   stranice AB,BC,CD i DA, a dijagonale AC i BD
7 na primer, tacke (1,1),(2,1),(2,2),(1,2) cine pravougaonik cije
   su stranice paralelne koordinatnim osama i ciji je obim 4
8 a tacke (1,1),(2,2),(3,3),(4,4) ne cine pravougaonik
9

```

```

11 h - proverava da li su unete tacke kolinearne i ukoliko jesu,
    ispisati jednacinu prave kojoj pripadaju
    na primer, tacke (1,2),(2,3),(3,4),(4,5) su kolinearne i
    pripadaju pravoj y=x+1
    tacke (1,1),(1,2),(1,3),(1,4) su kolinearne i pripadaju pravoj x
    =1
13 a tacke (1,1),(2,1),(2,2),(1,2) nisu kolinearne
j - Kramerovim pravilom proverava da li je dati sistem jednacina
15 x1 * p + x2 * q = x4 - x3
y1 * p + y2 * q = y4 - y3
17 odredjen, neodredjen ili nema resenja, i u slucaju da je odredjen
    ispisati resenja.
    na primer, za unete koordinate (1,1),(1,1),(1,0),(2,2) sistem
    nema resenja
19 za unete koordinate (1,1),(1,1),(1,1),(1,1) sistem je
    neodredjen ili nema resenja
    za unete koordinate (6,1),(8,3),(10,-4),(9,1) sistem
    ima jedinstveno resenje 4.30, 3.10
21
22 */
23
24 #include<stdio.h>
25 #include<math.h>
26 int main()
27 {
28     char c;
29     float x1,y1,x2,y2,x3,y3,x4,y4;
30     float kab,kbc,kad;
31     float dab,dad;
32     float delta, deltap, deltaq;
33     float 0;
34     float k,n;
35
36     printf("Unesi jedan karakter:");
37     scanf("%c",&c);
38
39     printf("Unesi realne koordinate 4 tacke:");
40     scanf("%f%f%f%f%f%f%f", &x1, &y1, &x2, &y2, &x3, &y3, &x4, &y4);
41
42     switch (c)
43     {
44         case 'k':
45             if (y1==y2 && y3==y4 && x1==x4 && x2==x3)
46             {
47                 dab = sqrt(pow(x1-x2,2)+pow(y1-y2,2)); // funkcija pow(x
48                 ,y) racuna vrednost stepene funkcije x^y
49                 dad = sqrt(pow(x1-x4,2)+pow(y1-y4,2)); // x i y su
50                 realne vrednosti
51                 0 = 2*dab + 2*dad;
52                 printf("Obim pravougaonika je %f\n",0);
53             }
54             else

```

```

53     printf("Tacke ne cine pravougaonik sa stranicama koje su
    paralelne koordinatnim osama\n");
    break;
55     case 'h':
        if ((x1-x2)!=0) // ukoliko se tacke A(x1,y1) i B(x2,y2) ne
            nalaze na pravoj koja je paralelna x osi
57         {
            k = (y1-y2)/(x1-x2); //izracunamo k,n za pravu odredjenu
            tackama A(x1,y1) i B(x2,y2)
59             n = y1-k*x1;

            if (y3==x3*k+n && y4==x4*k+n) // proverimo da li tacke
            C(x3,y3) i D(x4,y4) nalaze na toj pravoj
                printf("Tacke su kolinearne, pripadaju pravoj y=%f*x
61                +%f\n",k,n);
            else
63                printf("Tacke nisu kolinearne\n");
65        }
        else // ukoliko se A i B nalaze na pravoj koja je paralelna
            x osi
67            if (x3==x1 && x4==x1) // proverimo da li tacke C(x3,y3)
            i D(x4,y4) nalaze na toj pravoj
                printf ("Tacke su kolinearne, pripadaju pravoj x=%f\n
69                ",x1);
            else
                printf("Tacke nisu kolinearne\n");
71            break;
        case 'j':
73            delta = x1*y2-x2*y1;
            deltap = x2*(y4-y3)-y2*(x4-x3);
75            deltaq = x1*(y4-y3)-y1*(x4-x3);
            if (delta!=0)
77                printf("Sistem ima jedinstveno resenje %.2f, %.2f\n",
            deltap/delta, deltaq/delta);
            else if (deltap==0 && deltaq==0)
79                printf("Sistem je neodredjen ili nema resenja.\n");
            else
81                printf("Sistem nema resenja\n");
            break;
83            default:
                printf("Nekorektan unos\n");
85        }
        return 0;
87    }

```

### Rešenje 1.56

```

1  /*
    Napisati program koji 10 puta ispisuje tekst "We love C programming".
3  */

```

## 1 Kontrola toka

---

```
5 #include<stdio.h>

7 int main()
{
9     int i=0;          /* promenljiva i kontrolise koliko puta ce se petlja
    izvorsiti */
11    while (i<10) /* pre ulaska u telo petlje proverava se da li je */
    {                /* ispunjen uslov petlje */
13        printf("We love C programming\n");
        i++; /* operator ++ uvecava i promenljivu za 1
15              i++; ima isti efekat kao i=i+1;
              ili i+=1;

17              ukoliko ne bismo menjali vrednost promenljive i doslo
        bi
19              do beskonacne petlje!
              */
    }

21    /*
23        brojanje u while petlji smo mogli realizovati i preko uslova:

25        i=1;
        while(i<=10)
27        {
            ...
29        }

31        ili

33        i=2;
        while(i<=11)
35        {
            ...
37        }

39        ili

41        i=3;
        while(i<13)
43        {
            ...
45        }

47        Brojanje pocev od 0 uz koriscenje stroge nejednakosti
49        je u duhu programskog jezika C i zato cemo ovaj nacin
        brojanja najcesce koristiti

        */
51    return 0;
}
```



## Rešenje 1.57

```
2  /*
   Napisati program koji poziva korisnika da unese pozitivan ceo broj n
   a potom ispisuje brojeve od 0 do n-1.
4  */

6  #include<stdio.h>

8  int main()
   {
10     int x;
12     int n;

14     printf("Unesi pozitivan ceo broj:");
16     scanf("%d", &n);
18     x=0;
20     while (x<n)
22     {
        printf("%d\n", x);
        x++;
    }
    return 0;
}
```

## Rešenje 1.58

```
2  /*
   Napisati program koji za uneti pozitivan ceo broj
   izracunava njegov faktorijel. Testirati program
   za razlicite vrednosti promenljive x. Obratiti paznju
   da pocev od 23! dolazi do prekoracenja.
4  */

6  #include<stdio.h>

8  int main()
   {
10     int x;
12     unsigned long f;
14     int i;
16     int original;

18     printf("Unesi x>=0:");
20     scanf("%d",&x);

22     original=x;
    f=1;
    if (x<0)
        printf("Nekorektan unos\n");
}
```

## 1 Kontrola toka

```
24  else
25  {
26      while (x>1)
27      {
28          f=f*x; /* vrednost izraza sa desne strane naredbe dodele
                        dodeljujemo promenljivoj sa leve strane naredbe
30          dodele
                        */
31          x--; /* operator -- umanjuje vrednost promenljive x za 1
32              naredba x--; ima isti efekat kao x-=1;
33              ili x=x-1;
34              */
35      }
36      printf("%d! = %lu\n",x,f);          /* nekorektno: vrednost
promenljive x je unistena */
37      printf("%d! = %lu\n",original,f); /* korektno: promenljiva
original sadrzi vrednost promenljive x pre ulaska u petlju */
38
39  }
40
41
42  return 0;
43  }
```

### Rešenje 1.59

```
1  /*
2     Napisati program koji poziva korisnika da unese pozitivan ceo broj
3     n,
4     a zatim za unetih n celih brojeva ispisuje sumu pozitivnih i sumu
5     negativnih brojeva.
6
7  */
8
9  #include<stdio.h>
10
11  int main()
12  {
13      int n;
14      int x;
15      int suma_poz;
16      int suma_neg;
17      int i;
18
19      printf("Unesi pozitivan ceo broj:");
20      scanf("%d",&n);
21
22      suma_poz=0; /* promenljivim koje ce sadržati sumu se pre ulaska u
petlju */
23      suma_neg=0; /* dodeljuje se 0 (neutral za sabiranje) */
24      i=0;
```

```

25     while(i<n)
26     {
27         printf("Unesi ceo broj:");
28         scanf("%d", &x);
29
30         if (x<0)
31             suma_neg+=x;
32         else
33             suma_poz+=x;
34
35         i++;
36     }
37
38     printf(" Suma pozitivnih: %d\n Suma negativnih: %d\n", suma_poz,
39           suma_neg);
40     return 0;
41 }

```

### Rešenje 1.60

```

/*
2   Napisati program koji omogućava korisniku da unosi cele brojeve dok
   ne unese nulu. Nakon toga ispisati proizvod onih unetih brojeva
   koji
4   su pozitivni.
*/
6
#include <stdio.h>
8
int main()
{
10     int x;
11     int p;
12
13     p=1;
14     while (1) /* izraz 1 je konstantan; razlicit je od nule sto znaci
   da ga tumacimo kao tacnog */
15     {
16         printf("Unesi jedan ceo broj:");
17         scanf("%d", &x);
18         if (x==0) /* ukoliko je uneta nula */
19             break; /* break prekidamo petlju; izvršavanje se nastavlja
   od prve naredbe nakon petlje */
20
21         if (x<0) /* ukoliko je unet negativan broj, tu vrednost ne
   zelimo da pomnozimo sa ukupnim proizvodom p; zato moramo
   nastaviti dalje */
22             continue; /* sa izvršavanjem petlje; continue prekida
   trenutnu iteraciju petlje tako sto preskace sve naredbe
   koje nakon njega slede; izvršavanje se
   nastavlja od provere uslova petlje */

```

## 1 Kontrola toka

---

```
24     p=p*x;
    }
26     printf("Proizvod unetih brojeva je %d\n",p);
28     return 0;
30 }
```

### Rešenje 1.61

```
/*
2   Napisati program koji za uneti ceo broj ispisuje njegove cifre
   u obrnutom poretku.
4 */
6 #include<stdio.h>
7 #include<stdlib.h>
8 int main()
9 {
10     int x;
11     char cifra;
12     printf("Unesi ceo broj:");
13     scanf("%d", &x);
14
15     x = abs(x); /* pretvaranje u apsolutnu vrednost se vrsi za slucaj
   kada je unet
16                 negativan broj kako bismo osigurali da ce nam
   izdvojene cifre
17                 biti pozitivne
18                 */
19
20     while(x>0)
21     {
22         cifra=x%10; /* izdvajamo poslednju cifru broja x */
23         printf("%d\n", cifra);
24         x/=10; /* ako je npr x=1582, x%10 ce biti 2,
   a x/10 ce biti 158;
25                 npr x=5, x%10 ce biti 5
26                 a x/10 ce biti 0 */
27     }
28
29     return 0;
30 }
```

### Rešenje 1.62

```
1 /*
   Napisati program koji omogucava korisniku da unosi karaktere dok ne
   zada tacku i ukoliko je karakter malo slovo,
```

```

3  ispisuje odgovarajuće veliko, ukoliko je karakter veliko slovo
   ispisuje odgovarajuće malo, a u suprotnom ispisuje
   isti karakter kao i uneti.
5  */
7  #include <stdio.h>
9  int main()
10 {
11     int c;
13     /* funkcija getchar učitava jedan karakter.
       naredbom dodele (c=getchar()) promenljivoj c bice dodeljena
       vrednost
       ascii koda unetog karaktera
       obratiti paznju na zagrade!
15     */
17     while((c=getchar())!='.')
19     {
21         if (c>='A' && c<='Z')
           putchar(c+'a'-'A'); /* Razlika izmedju ascii koda svakog malog
                                i odgovarajućeg velikog slova
                                je konstanta koja se može sračunati
                                izrazom 'a'-'A' (i iznosi 32) */
23         else if (c>='a' && c<='z')
           putchar(c-'a'+'A');
25         else
           putchar(c);
27     }
29     return 0;
30 }

```

### Rešenje 1.63

```

/*
2  Napisati program koji omogućava korisniku da unosi karaktere dok
   ne zada EOF a potom ispisuje broj velikih slova, broj malih slova
   ,
4  broj cifara, broj belina i zbir cifara.
   */
6
8  #include <stdio.h>
10
12 int main()
13 {
14     /* promenljivoj c dodelicemo povratnu vrednost funkcije getchar()
       funkcija getchar() učitava jedan karakter sa standardnog ulaza
       i vraća njegov ascii kod; povratna vrednost funkcije getchar je
       int, pa i promenljiva c mora biti tipa int
       */

```

## 1 Kontrola toka

```
16  int c;

18  /* brojacji moraju biti inicijalizovani na 0 */
19  int br_v=0;
20  int br_m=0;
21  int br_c=0;
22  int br_b=0;
23  int br_k=0;
24  int suma=0;

26  while((c=getchar())!=EOF)          /* petlja se završava kada
    korisnik ne unese karakter, vec zada konstantu EOF */
    {                                  /* ova konstanta se zadaje
    kombinacijom tastera CTRL+D. U tom slucaju, getchar() vraca -1*/
28      if (c>='A' && c<='Z')
          br_v++; /* <=> br_v = br_v+1; */
30      else if (c>='a' && c<='z')
          br_m++;
32      else if (c>='0' && c<='9')
          {
34              br_c++;
              suma=suma+c-'0';          /* funkcija getchar() vraca ascii
    kod unetog karaktera; ascii kodovi cifara 0,1,...,9
36                                          su redom 48,49,...,57; Na primer,
    za unetu 1
                                          promenljiva c ce imati vrednost
    49. Zbog toga bi bilo pogresno racunati
38              zbir kao zbir=zbir+c. Promenljivu zbir zato
    racunamo kao zbir=zbir+(c-'0')
                                          jer c-'0' ce za unetu 0 proizvesti 48-'0' sto je
    0,
40              za unetu 1 49-'0' sto je 1, za unetu 2 50-'0' sto
    je 2, ...*/
          }
42      else if (c=='\t' || c=='\n' || c==' ')
          br_b++;
44
          br_k++;
46  }

48  printf("velika: %d, mala: %d, cifre: %d, beline: %d, svi: %d\n",
    br_v, br_m, br_c, br_b, br_k);
    printf("suma cifara: %d\n", suma);
50
    return 0;
52 }
```

### Rešenje 1.64

```
/* Niz prirodnih brojeva formira se na sledeci nacin:
2 an+1 = an/2 ako je an parno
```

```

an+1 = (3*an+1)/2 ako je an neparno
4 Napisati program koji za uneti pocetni clan niza a0 stampa niz
   brojeva sve do prvog clana jednakog
1.
6 */
#include<stdio.h>
8 int main()
{
10     int a0;
    int an,an1;
12
    printf("Unesi pocetni clan niza brojeva:");
14     scanf("%d",&a0);
16
    if (a0>0)
    {
18         printf("%d\n", a0);
20
        an=a0;
        while(an!=1)
22         {
            if (an%2) /* ukoliko je vrednost izraza an%2 razlicita od nule,
24             /* izraz se tumaci kao tacan i izvrsavaju se naredbe
            iz if grane */
            an1=(3*an+1)/2;
26         }
            else /* u suprotnom, ukoliko je vrednost izraza an%2 jednaka
            nuli, izraz */
28         { /* se tumaci kao netacan i izvrsavaju se naredbe iz else
            grane */
            an1=an/2;
30         }
            printf("%d\n",an1);
32         an=an1;
        }
34     }
    else
36         printf("Nekorektan unos\n");
38     return 0;
}

```

### Rešenje 1.65

```

1 /*
   Napisati program koji za uneti ceo broja n ispisuje n puta tekst
3   "We love C programming" koriscenjem while, for i do while petlje.
   Obratiti paznju
   na rezultat kada je n<=0.
5 */

```

```
7  #include <stdio.h>

9  int main()
10 {
11
12     int n,m;
13     int i;

14
15     printf("Unesi ceo broj:");
16     scanf("%d",&n);

17
18     /* 1. nacin - while petlja */
19     printf("while: ");

20
21     i=0;
22     while (i<n)          /* uslov petlje se proverava pre ulaska u
23         telo petlje */
24     {
25         printf("We love C programming\n");
26         i++;
27     }

28     printf("\n");

29
30     /* 2. nacin - for petlja */
31     printf("for: ");

32         /* naredba i=0 se izvsava jednom, pre prve
33         iteracije */
34     for(i=0;i<n;i++)      /* uslov petlje i<=m se proverava pre svake
35         iteracije */
36         printf("We love C programming\n"); /* naredba i++ se izvsava
37         nakon svake iteracije */

38     printf("\n");

39
40     /* 3. nacin - do while petlja */
41     printf("do while: "); /* uslov petlje se proverava na kraju svake
42         iteracije */

43         /* zbog toga se do while petlja izvsava
44         bar jednom, cak i u slucaju */
45         /* da uslov petlje nikada nije ispunjen */

46     i=0;
47     do                      /* petlja se zapocinje bez
48         provere uslova */
49     {
50         printf("We love C programming\n"); /* stampa se dati tekst */
51         i++;                               /* uvecava se vrednost
52         promenljive i */
53     }
54     while(i<n);              /* proverava se uslov i
55         ukoliko je ispunjen, nastavlja se sa sledecom iteracijom */
```



```

49                                     /* u suprotnom, petlja se
    završava i program se nastavlja od prve naredbe koja sledi za
    petljom */
    printf("\n");
51
    return 0;
53
}
```

### Rešenje 1.66

```

/*
2   Napisati program koji za uneta dva cela broja n i m ispisuje sve
    cele brojeve
    iz intervala [n,m] koriscenjem while, for i do while petlje.
    Obratiti paznju
4   na rezultat kada je n>m.
*/
6
#include <stdio.h>
8
int main()
10 {
12     int n,m;
    int i;
14
    printf("Unesi dva cela broja:");
16     scanf("%d%d",&n,&m);

    /* 1. nacin - while petlja */
    printf("while: ");
20
    i=n;
22     while (i<=m)          /* uslov petlje se proverava pre ulaska u
        telo petlje */
    {
24         printf("%d ", i);
        i++;
26     }

    printf("\n");
28

    /* 2. nacin - for petlja */
    printf("for: ");
30
32                                     /* naredba i=n se izvršava jednom, pre prve
        iteracije */
    for(i=n;i<=m;i++)        /* uslov petlje i<=m se proverava pre svake
        iteracije */
34         printf("%d ", i);    /* naredba i++ se izvršava nakon svake
        iteracije */

```

```
36     printf("\n");
38     /* 3. nacin - do while petlja */
    printf("do while: "); /* uslov petlje se proverava na kraju svake
        iteracije */
40                                /* zbog toga se do while petlja izvršava
        bar jednom, čak i u slučaju */
                                /* da uslov petlje nikada nije ispunjen */
42     i=n;
    do                                /* petlja se zapocinje bez provere uslova */
44     {
        printf("%d ",i); /* stampa se vrednost promenljive i */
46         i++;           /* uvecava se vrednost promenljive i */
    }
48     while(i<=m);           /* proverava se uslov i ukoliko je ispunjen,
        nastavlja se sa sledecom iteracijom */
                                /* u suprotnom, petlja se završava i program
        se nastavlja od prve naredbe koja sledi za petljom */
50     printf("\n");
    return 0;
52 }
```

### Rešenje 1.67

```
/*
2  Program izracunava minimum n unetih brojeva.
  Npr. za n=4 i brojeve 3 8 2 9 program ispisuje 2
  */
#include <stdio.h>
6  int main()
  {
8      int n, i;
        float x, min;
10
12     printf("Unesi n>0:");
        scanf("%d", &n);
14     if (n<=0)                                /* ako je unos neispravan */
        {
16         printf("Neispravan unos\n");
            return -1;                          /* prekidamo izvršavanje
        programa pomocu naredbe return */
18     }                                /* u slučaju greske kao što je
        neispravan unos vraćamo vrednost -1 */
        printf("Unesi realan broj:");
20     scanf("%f", &x);                        /* prvi broj je unet izvan petlje */
        min=x;                                /* kako bi bio njegova vrednost bila
        dodeljena promenljivoj min */
22                                /* neophodno je da promenljiva min
        bude inicijalizovana pre ulaska u petlju */
    }
```

```

/* da bi uslov x<min mogao da bude
ispitan u prvoj iteraciji */
24 i=0;
while(i<(n-1))
26 {
    printf("Unesi realan broj:");
    scanf("%f", &x);
    if(x<min)
30     min=x;
    i++;
32 }
printf("Minimum je: %f\n", min);
34 return 0;
}

```

## Rešenje 1.68

```

/*
2   a) Napisati program za uneti pozitivan ceo broj n ispisuje zbir
   s = 1+2^3+3^3+...+n^3. Na primer, za n=4, izlaz iz programa
4   treba da bude:
   Suma kubova od 1 do 4 je 100
6   b) Modifikovati program tako da ispisuje zbir s = 1+2^3+3^3+...+k
   ^3
   za svako i od 1 do n. Na primer, za n=4, izlaz iz programa
   treba da
8   bude:
i=1, n=1
10 i=2, n=9
i=3, n=36
12 i=4, n=100

14 */

16 #include <stdio.h>

18 int main()
{
20     int n;
    int i;
22     int s;

24     printf("Unesite jedan pozitivan ceo broj:");
26     scanf("%d", &n);

28     if (n<0)
        return -1;
30
    i=1;
32     s=0; /* inicijalizacija promenljive u kojoj se cuva suma kubova */

```

## 1 Kontrola toka

---

```
34  for(i=1;i<=n;i++)
    {
36      s+=i*i*i;
        /* b) */
38      printf("i=%d, s=%d\n", i, s);
    }
40  /* a) */
    printf("Suma kubova od 1 do %d: %d\n", n, s);
42  return 0;
}
```

### Rešenje 1.69

```
1  /*
    Napisati program koji ispisuje sve prave delioce unetog pozitivnog
        celog broja.
3
    */
5
    #include<stdio.h>
7  #include<math.h>
    int main()
9  {
        int x;
11       int i;

13       printf("Unesi x>0:");
        scanf("%d", &x);

15
        if (x<=0)
17         {
            printf("Neispravan unos\n");
19         return -1;
        }

21
        /* 1. nacin */
23       printf("----- 1. nacin ----- \n");
        for(i=2;i<x;i++)
25         {
            printf("proveravam za %d...\n",i);
27             if (x%i==0)
                printf("\t delilac:%d \n",i);
29         }

        /* 2. nacin (brzi) */
31       printf("----- 2. nacin ----- \n");
        for(i=2;i<=sqrt(x);i++)
33         {
            printf("proveravam za %d...\n",i);
35             if (x%i==0)
```

```

        if (i==x/i) /* u slucaju kada je delilac koren broja, npr 4
za 16, ispisujemo ga jednom */
37     printf("\t delilac:%d \n",i);
        else /* u suprotnom, npr 2 za 16, ispisujemo i 2 i 8
*/
39     printf("\t delioci:%d %d \n",i,x/i);
    }
41     return 0;
}

```

## Rešenje 1.70

```

1  #include <stdio.h>

3  int kvadrat(int x)
{
5     /* promenljive u listi argumenata funkcije, kao i one
    deklarisanе u samoj funkciji, lokalne su za tu funkciju
7     sto znaci da se promenljive x i y neće "videti" nigde izvan
    funkcije kvadrat (ni u funkciji main ni u funkciji kub)
9     */

11    int y;
    y = x*x;
13    return y;
}

15 int kub(int a)
17 {
    /*
19    u listi argumenata funkcije mozemo, a ne moramo, imati
    promenljivu
    istog naziva kao promenljiva koja je deklarisanа u main
    funkciji
21    (u ovom slucaju promenljiva a); ova promenljiva se razlikuje
    od promenljive a deklarisanе u main funkciji i vidljiva je
23    samo unutar funkcije kub
    */
25    return a*a*a;
}

27 int main()
29 {
    int a,kv, kb;
31    printf("Unesi ceo broj:");
    scanf("%d",&a);
33

    kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
    funkcije kvadrat */
35    kb = kub(a); /* promenljivoj kb dodeljujemo povratnu vrednost
    funkcije kub */

```

```
37     printf("Kvadrat broja %d je %d, a njegov kub je %d\n", a, kv, kb);
    return 0;
39 }
```

### Rešenje 1.71

```
1  /*
   Napisati program koji za uneti realan broj x i ceo broj n ispisuje
   vrednost stepena x^n. Unosenje promenljivih, racunanje stepena i
   ispis promenljivih realizovati u posebnim funkcijama.
   */
7  #include <stdio.h>
   #include <stdlib.h>
9
   float stepen(float a, int b)
11 {
    float s=1;
13     int i;

15     for(i=0;i<abs(b);i++)
        s=s*a;

17     return b>0 ? s : 1/s;    /* ukoliko je izlozilac b negativan,
        izracunamo a^|b| i vracamo reciprocnu vrednost
19                                izracunatog stepena */
21 }

23 int main()
   {
25     int n;
    float x;
27     float s;

29
    printf("Unesi jedan realan i jedan ceo broj:");
31     scanf("%f%d",&x,&n);

33     s = stepen(x,n);

35
    printf("%f^%d=%f\n",x,n,s);
37
    return 0;
39 }
```

### Rešenje 1.72

```
1  /*
2  Napisati funkciju koja za dva data cela broja odredjuje
3  najveći zajednički delilac. Napisati potom glavni program
4  koji testira ovu funkciju.
5  */
6
7  #include <stdio.h>
8
9  int euklid(int x, int y)
10 {
11     int r;
12     /* Euklidov algoritam */
13     while(y) /* algoritam se zaustavlja kada vrednost */
14     { /* promenljive y postane nula */
15         r=x%y;
16         x=y;
17         y=r;
18     }
19
20     return x; /* nzd je sacuvan u promenljivoj x */
21 }
22
23 int main()
24 {
25     int a,b;
26     int nzd;
27
28     printf("unesi dva cela broja:");
29     scanf("%d%d", &a,&b);
30
31     nzd = euklid(a,b); /* promenljivoj nzd dodeljujemo povratnu
32     vrednost funkcije euklid */
33
34     printf("najveći zajednički delilac za %d i %d je %d\n", a,b,nzd);
35
36     return 0;
37 }
```

### Rešenje 1.73

```
1  /*
2  Napisati funkciju koja za dato n vraca zbir reciprocnih vrednosti
3  brojeva od 1 do n.
4  Napisati program koji omogucava korisniku da unese prirodan broj n, a
5  potom ispisuje zbir reciprocnih
6  vrednosti brojeva od 1 do n koristeći funkciju float zbir_reciprocnih
7  (int n). Rezultat zaokruziti
8  na dve decimale.
9  */
10
```

## 1 Kontrola toka

```
8  #include <stdio.h>

10 float zbir_reciprocnih(int n)
11 {
12     float z=0;
13     int i;
14     for(i=1;i<=n;i++)
15         z+=1.0/i; /* da bismo dobili reciprocnu vrednost broja, vazno je
16                   da izbegnemo celobrojno deljenje dva cela broja */
17     return z; /* tako sto ce npr deljenik biti 1.0 umesto 1 */
18 }

19
20 int main()
21 {
22     int n;
23     printf("Unesi jedan pozitivan ceo broj:\n");
24     scanf("%d", &n);
25     printf("Zbir reciprocnih vrednosti brojeva od 1 do %d je %.2f\n", n
26           , zbir_reciprocnih(n));
27     /* povratna vrednost funkcije zbir_reciprocnih je float; funkciju
28        mozemo pozvati u okviru
29        naredbe printf i umesto specifikatora %.2f bice ispisana
30        povratna vrednost funkcije
31        zbir_reciprocnih zaokruzena na dve decimale */
32     return 0;
33 }
```

### Rešenje 1.74

```
1  /*
2   Napisati funkciju koja racuna aritmeticku sredinu cifara datog celog
3   broja.
4   Napisati potom glavni program koji omogucava korisniku da unese ceo
5   broj
6   i racuna aritmeticku sredinu njegovih cifara primenom napisane
7   funkcije. Ispisati
8   izracunatu vrednost zaokruzenu na dve decimale.
9   */
10
11 #include<stdio.h>
12 #include<stdlib.h>
13
14 float aritmeticka_sredina(int x)
15 {
16     int zbir_cifara=0;
17     int broj_cifara=0;
18     char cifra;
19
20     if (x==0) /* u slucaju da je uneta 0 */
21         return 0; /* aritmeticka sredina cifara iznosi 0 i tu vrednost
22                   vratamo */
23 }
```



```

19
21  x=abs(x); /* uzimamo apsolutnu vrednost broja za slucaj da je
    negativan */
23  while(x)
    {
25      cifra=x%10;

27      broj_cifara++;
      zbir_cifara+=cifra;

29      x/=10;
31  }

33  return (0.0+zbir_cifara)/broj_cifara; /* posto su zbir_cifara i
    broj_cifara celobrojne vrednosti,
                                         neophodno je da bar
    jednu od njih konvertujemo u realnu
35                                         kako bismo izbegli
    celobrojno deljenje */
}

37 int main()
39 {
    int x;
41    printf("Unesi jedan ceo broj:");
    scanf("%d",&x);
43    printf("Aritmeticka sredina cifara broja %d iznosi %.2f\n", x,
        aritmeticka_sredina(x));
    return 0;
45 }

```

### Rešenje 1.75

```

1  /*
    Napisati funkciju koja za dva realna broja x i y i jedan neoznaceni
    ceo broj n
3  ispisuje vrednosti funkcije sin u n ravnomerno rasporedjenih tacaka
    intervala [x,y].
    Napisati potom glavni program koji omogucava korisniku da unese
    potrebne vrednosti
5  i poziva napisanu funkciju.
    */
7
9  #include <stdio.h>
    #include <math.h>

11 void ispis(float x, float y, int n) /* funkcija nema povratnu
    vrednost; zbog toga je povratni tip void */
    {

```

## 1 Kontrola toka

---

```
13  float i;
    float korak=(y-x)/(n-1);
15
    for(i=x;i<=y;i+=korak)
17         printf("sin(%.4f)=%.4f\n", i,sin(i));
19 }

21 int main()
    {
23         float a,b;
        int n;
25         float t;
        printf("Unesi dva realna broja:");
27         scanf("%f%f",&a,&b);
        printf("Unesi jedan ceo broj > 1:");
29         scanf("%u",&n);

31         if (n<=1 || a==b)
            {
33                 printf("Nekorektan unos\n");
                    return -1;
35             }
        if (b<a) /* u slucaju da je desni kraj intervala manji od levog */
37             { /* zamениmo im mesta */
                    t=a;
39                 a=b;
                    b=t;
41             }

43
45         ispis(a,b,n);

47         return 0;
    }
```

### Rešenje 1.76

```
/*
2  Napisati funkciju koja broji neparne cifre u zapisu datog celog broja
    . Napisati
    potom glavni program koji unosi cele brojeve dok se ne unese nula, i
        ispisuje
4  broj neparnih cifara svakog unetog broja koriscenjem napisane
        funkcije.
    */
6
    #include<stdio.h>
8  #include<stdlib.h>
```

```

10 int broj_ncifara(int x)
11 {
12     int s=0;
13     char cifra;
14     x = abs(x);
15
16     while(x)
17     {
18         cifra = x%10;
19         s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
20 neparna,                a 0 kada je cifra parna */
21         x/=10;
22     }
23
24     return s;
25 }
26
27 int main()
28 {
29     int x;
30     do
31     {
32         scanf("%d",&x);
33         printf("Broj neparnih cifara u zapisu broja %d: %d\n", x,
34 broj_ncifara(x));
35     } while(x!=0);
36
37     return 0;
38 }

```

### Rešenje 1.77

```

1  /*
2  Napisati funkciju koja ispituje da li je dati ceo broj prost.
3  Funkcija treba
4  da vrati 1 ako je broj prost i 0 u suprotnom. Napisati potom glavni
5  program
6  koji za uneti ceo broj n ispisuje prvih n prostih brojeva.
7  */
8
9  #include <stdio.h>
10 #include <math.h>
11
12 int prost (int x) /* 1-broj je prost, 0-broj nije prost */
13 {
14     int i;
15
16     if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
17         return 1;
18 }

```

## 1 Kontrola toka

```
17  if (x%2==0)          /* parni brojevi nisu prosti */
    return 0;
19
21  for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
    if (x%i==0) /* ako je pronadjen, to znaci da broj nije prost */
        return 0; /* završavamo funkciju */
23
    /* ukoliko izvršavanje funkcije dodje do poslednje naredbe return,
25    to znaci da broj nije ispunio nijedan od prethodnih uslova
    (nije ni 2, ni 3, ni paran, niti ima ijednog delioca), odakle
27    sledi da je prost i zbog toga vratamo 1
    */
29    return 1;
}

31
33  int main()
34  {
    int n;
35    scanf("%d",&n);
    int i,j;
37
    i=1; /* kandidat za prost broj */
39    j=0; /* brojac prostih brojeva */
    while(j<n)
    {
41        if (prost(i))          /* ako je broj prost */
42        {
            printf("%d\n", i); /* stampamo ga i */
43            j++;               /* uvecavamo brojac prostih brojeva */
44        }
45        i++; /* bilo da je i prost ili ne, uvecavamo ga za 1 i
        nastavljamo sa sledecom iteracijom */
46    }
47
49    return 0;
51 }
```

### Rešenje 1.78

```
1  /*
   Napisati funkciju koja ispituje da li se cifra c nalazi u zapisu
   celog broja x.
3  Napisati potom glavni program koji za uneti ceo broj i unetu cifru
   poziva
   napisanu funkciju i ispisuje odgovarajucu poruku.
5  */

7  #include<stdio.h>
   #include<stdlib.h>
9
   int sadrzi(int x, int c)
```

```

11 {
    char cifra;
13     x=abs(x);
    while(x)
15     {
        cifra = x%10;
17         if (cifra==c)
            return 1;
19         x/=10;
    }
21     return 0;
}
23 int main()
{
25     int x;
    int c;
27     printf("Unesi jedan ceo broj i jednu cifru:");
    scanf("%d%d",&x,&c);
29     if (sadrzi(x,c))
        printf("Cifra %d se nalazi u zapisu broja %d\n",c,x);
31     else
        printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
33     return 0;
}

```

### Rešenje 1.79

```

/*
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 vraca 1 ako su sve cifre broja
    parne i 0 u suprotnom*/
24 {
    char d;
26 x=abs(x);          /* uzimamo apsolutnu vrednost broja za slucaj da je
    broj negativan */
    while (x>0)
28 {
        d=x%10;          /* izdvajamo cifru broja */

30        if (d%2==1)     /* u slucaju da je neparna, to znaci da nisu sve
        cifre broja parne */
32            return 0;    /* vracamo 0 */

34        x/=10;          /* "uklanjamo" poslednju cifru broja celobrojnim
        deljenjem sa 10 */
    }

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;
67     int d;
68
69     printf("unesi ceo broj:");
70     scanf("%d", &x);
71
72     if (sve_parne_cifre(x))
73         printf("Sve cifre broja su parne\n");
74     else
75         printf("Broj sadrzi bar jednu neparnu cifru\n");
76
77     if (sve_cifre_jednake(x))
78         printf("Sve cifre broja su jednake\n");
79     else
80         printf("Broj sadrzi razlicite cifre \n");
81 }

```

### Rešenje 1.80

```

/*
2  Napisati funkciju koja za dva uneta neoznacena broja x i n utvrđuje
   da li je x neki stepen
   broja n. Ukoliko jeste, funkcija vraća izlozilac stepena, a u
   suprotnom vraća -1. Napisati
4  potom glavni program koji testira ovu funkciju.
   */
6
#include <stdio.h>
8
int je_stepen(unsigned x, unsigned n) /* funkcija vraća izlozilac
   stepena ukoliko broj x jeste neki stepen broja n */
10 {
11     int i=1;
12     int s=n;
13
14     while(s<x)
15     {
16         s=s*n;
17         i++;
18     }
19
20     if (s==x)
21         return i;
22
23     return -1;
24 }
25
int main()
26 {
27     unsigned x;

```

```
    unsigned n;
30    int st;

32    scanf("%u%u",&x,&n);

34    st = je_stepen(x,n);

36    if (st!=-1)
        printf("%u=%u^%d\n",x,n,st);
38    else
        printf("%u nije stepen broja %u\n",x,n);
40
    return 0;
42 }
```

### Rešenje 1.81

```
/*
2
    Napisati funkciju
4
    double e_na_x(double x, double eps)
6
    koja racuna vrednost  $e^x$  kao parcijalnu sumu reda
    suma( $x^n/n!$ ), gde indeks n ide od
8    od 0 do beskonacno, pri cemu se sumiranje vrši dok
    je razlika sabiraka u redu po apsolutnoj vrednosti
10   manja od eps. Napisati potom program koji omogucuje
    korisniku da unese jedan realan broj x i ispisuje
12   vrednost  $e^x$ .

14
*/

16 #include<stdio.h>
18 #include<math.h>

20 double e_na_x(double x, double eps)
{
22     double s=1;
    double clan=1;
24     int n=1;

26     /*
        parcijalnu sumu formiramo tako sto u svakoj iteraciji petlje
28     promenljivoj s dodamo jedan sabirak sume oblika  $(x^n)/n!$  koji
        cuvamo u promenljivoj clan

30
        svaki sabirak mozemo da dobijemo na osnovu prethodnog tako sto
32     ga pomnozimo sa x i podelimo sa n, koje predstavlja redni broj
        sabirka u sumi

34
    */
}
```



```

36     prvi sabirak (kome odgovara n=0) iznosi 1; zbog toga promenljive
      s i clan inicijalizujemo na vrednost 1

38     sumiranje se sprovodi dogod je sabirak po apsolutnoj vrednosti
      veci od trazene tacnosti eps

40     */

42     do
    {
44         clan = (clan*x)/n;
      s += clan;
46         n++;
    } while(fabs(clan)>eps);

48     return s;
50 }

52 int main()
    {
54     double x,eps;
      printf("x=");
56     scanf("%lf", &x);
      printf("eps=");
58     scanf("%lf", &eps);

60     printf("e~%f=%f\n", x, e_na_x(x,eps));
      return 0;
62 }

```

### Rešenje 1.82

```

/*
2  Za dati broj moze se formirati niz tako da je svaki sledeci clan niza
   dobijen
   kao suma cifara prethodnog clana niza. Broj je srecan ako se dati niz
   završava sa
4  jedinicom. Napisati program koji za uneti broj odredjuje da li je
   srecan.
   Na primer:
6  - broj 1234 je srecan jer je zbir njegovih cifara 10, dalje zbir
   cifara broja 10 je 1.
   - broj 999 nije srecan jer je njegov zbir cifara 27, zbir cifara
   broja 27 je 9.
8  - broj 991 je srecan, zbir njegovih cifara je 19, zbir cifara broja
   19 je 10, zbir cifara
   broja 10 je 1.
10 - broj 372 nije srecan, zbir njegovih cifara je 12, zbir cifara broja
   12 je 3

12 Napisati funkciju koja vraca 1 ako je broj srecan, a 0 u suprotnom.

```

## 1 Kontrola toka

---

```
14  Napisati program koji omogućava korisniku da unese prirodan broj,
    poziva funkciju
    i ispisuje da li je dati broj srecan. Potom traziti od korisnika da
    unese prirodan
16  broj n i ispisati sve srecne brojeve od 1 do n.
    */
18
19  #include<stdio.h>
20
21  int zbir_cifara(int x)
22  {
23      int s=0;
24      char cifra;
25      while(x)
26      {
27          cifra = x%10;
28          s+=cifra;
29          x/=10;
30      }
31      return s;
32  }
33
34  int srecan(int x)
35  {
36      int s; /* promenljiva s sadrzi sumu cifara */
37
38      do
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 1.83

```
2  /*
3  . a) Napisati funkciju
4
5      int konverzija (int c)
6
7  koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.
8
9  b) Napisati program koji omogućava korisniku da unese niz karaktera
10 sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
11 Na primer, za uneti tekst "Kolokvijum iz Progi je 1.12." program
12 treba da ispise "kOLOVKIJUM IZ pROG1 JE 1.12."
13
14 */
15 #include <stdio.h>
16
17 int konverzija(int c)
18 {
19     /* ključna rec return vraća povratnu vrednost funkcije (ako je ima)
20      */
21     /* i završava izvršavanje funkcije */
22
23     if (c>='A' && c<='Z')
24         return c+'a'-'A';
25
26     if (c>='a' && c<='z')
27         return c-'a'+'A';
28
29     return c;
30 }
31
32 int main()
33 {
34     int c;
35
36     while((c=getchar())!=EOF) /* korisnik unosi karakter po karakter
37                               do konstante EOF */
38         putchar(konverzija(c)); /* funkcija putchar ispisuje jedan
39                                karakter na standardni izlaz */
40
41     return 0;
42 }
```



## 2

# Predstavljanje podataka

## 2.1 Nizovi

**Zadatak 2.1** Tekst

[Rešenje [2.1](#)]

**Zadatak 2.2** Tekst

[Rešenje [2.2](#)]

**Zadatak 2.3** Tekst

[Rešenje [2.3](#)]

**Zadatak 2.4** Tekst

[Rešenje [2.4](#)]

**Zadatak 2.5** Tekst

[Rešenje [2.5](#)]

**Zadatak 2.6** Tekst

[Rešenje [2.6](#)]

**Zadatak 2.7** Tekst

[Rešenje [2.7](#)]

## 2 Predstavljanje podataka

---

### Zadatak 2.8 Tekst

[Rešenje 2.8]

### Zadatak 2.9 Tekst

[Rešenje 2.9]

### Zadatak 2.10 Tekst

[Rešenje 2.10]

- (a) Sa standardnog ulaza se unosi dimenzija niza (broj manji od 100), a zatim i njegovi elementi. Napisati program koji kvadrira sve negativne elemente niza i ispisuje rezultujući niz.

#### 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
```

- (b) Sa standardnog ulaza se učitava dimenzija niza (broj manji od 100), elemente niza i jedan ceo broj  $k$ . Napisati program koji štampa indekse elemenata koji su deljivi sa  $k$ .

#### 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
```

- (c) Napisati program koji sa standardnog ulaza učitava dimenziju niza (broj manji od 100) i elemente niza, a zatim štampa niz u kojem su najveći i najmanji element niza razmenili mesta.

*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: pogresan unos!
```

- (d) Napisati program koji učitava karaktere sa ulaza (najviše njih 100) sve do pojave karaktera \*, a zatim ih 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
```

- (e) Napisati program koji za dva cela broja  $x$  i  $y$  koja se učitavaju sa standardnog ulaza proverava da li se zapisuju pomoću istih cifara. Napomena: iskoristiti niz za čuvanje broja pojavljivanja svake od cifara.

## 2 Predstavljanje podataka

---

### 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!
```

- (f) Sa standardnog ulaza se učitava dimenzija niza (broj manji od 100), zatim i elementi dvaju nizova  $a$  i  $b$ . 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: pogresan unos!
```

- (g) Sa standardnog ulaza se unosi dimenzija niza  $a$  (broj manji od 100), a zatim i njegovi elementi. Napisati program koji od datog niza formira niz  $b$  u koji ulaze elementi niza  $a$  koji se pojavljuju tačno 3 puta.

### 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:
```

- (h) Sa standardnog ulaza se, redom, učitavaju dimenzija i elementi dvaju ni-



zova  $a$  i  $b$ . Napisati program koji određuje njihovu uniju, presek i razliku (redosled prikaza elemenata nije bitan). Pretpostaviti da će nizovi imati manje od 100 elemenata.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 5
Unesite elemente niza a: 2 8 1 5 2
Unesite broj elemenata niza b: 3
Unesite elemente niza b: 5 7 8
Unija: 2 8 1 5 2 5 7 8
Presek: 5
Razlika: 2 1 2
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 3
Unesite elemente niza a: 11 4 4
Unesite broj elemenata niza b: 2
Unesite elemente niza b: 18 9
Unija: 11 4 4 18 9
Presek:
Razlika: 11 4 4
```

*Primer 3*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 6
Unesite elemente niza a: 12 7 9 12 5 1
Unesite broj elemenata niza b: 4
Unesite elemente niza b: 1 12 22 12
Unija: 12 7 9 12 5 1 1 12 22 12
Presek: 12 12 1
Razlika: 7 9 5
```

- (i) Napisati program koji učitava dimenziju niza (broj manji od 100) i elemente niza, a zatim formira i ispisuje niz koji se dobija izbacivanjem svih neparnih elemenata niza. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza.

*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
```

## 2 Predstavljanje podataka

---

- (j) Napisati program koji učitava dimenziju niza (broj manji od 100) i elemente niza, a zatim formira i ispisuje niz koji se dobija izbacivanjem svih elemenata koji su prosti brojevi. Zadatak rešiti na dva načina: korišćenjem pomoćnog niza i transformacijom polaznog niza. Napomena: brojeve -1 i 1 smatrati prostim.

### 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
```

- (k) Napisati funkciju *int prebrojavanje(int a[], int n)* koja izračunava broj elemenata niza celih brojeva *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
```

- (l) Napisati funkciju *int prebrojavanje(int a[], int n)* koja izračunava broj parnih elemenata niza celih brojeva *a* dužine *n* koji prethode maksimalnom elementu niza. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 4
|| Unesite elemente niza: 11 2 4 9
|| 0

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 7
|| Unesite elemente niza: 7 2 1 14 65 2 8
|| 2

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 5
|| Unesite elemente niza: 25 18 29 30 14
|| 1

```

- (m) Napisati funkciju *int prebrojavanje\_cifre(char s[], int n)* koja izračunava broj cifara u nizu karaktera *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: 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

```

- (n) 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.

## 2 Predstavljanje podataka

---

### 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
```

- (o) 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
```

- (p) Napisati funkciju *void kvadriranje(float a[], int n)* koja kvadrira elemente realnog niza *a* dužine *n* koji se nalaze na parnim pozicijama. 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
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
```

## 2.2 Pokazivači i argumenti komandne linije

**Zadatak 2.11** Tekst

[Rešenje [2.11](#)]

**Zadatak 2.12** Tekst

[Rešenje [2.12](#)]

**Zadatak 2.13** Tekst

[Rešenje [2.13](#)]

**Zadatak 2.14** Tekst

[Rešenje [2.14](#)]

**Zadatak 2.15** Tekst

[Rešenje [2.15](#)]

**Zadatak 2.16** Tekst

[Rešenje [2.16](#)]

- (a) Napisati program koji ispisuje zbir numeričkih argumenata komandne linije. Napomena: može se koristiti funkcija *atoi*.

### Primer 1

```
|| POKRETANJE: ./a.out 5 mkp 9 -2 11 a 4 2  
|| INTERAKCIJA SA PROGRAMOM:  
|| Zbir numerickih argumenata: 29
```

### Primer 2

```
|| POKRETANJE: ./a.out ab u f hj  
|| INTERAKCIJA SA PROGRAMOM:  
|| Zbir numerickih argumenata: 0
```

### Primer 3

```
|| POKRETANJE: ./a.out 33 1 p 44  
|| INTERAKCIJA SA PROGRAMOM:  
|| Zbir numerickih argumenata: 78
```

### Primer 4

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
||   Zbir numerickih argumenata: 0
```

- (b) 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:
```

- (c) Napisati program koji ispisuje broj argumenata komandne linije koji sadrže slovo z.

### Primer 1

```
|| POKRETANJE: ./a.out zvezda grozd jesen kisa
|| INTERAKCIJA SA PROGRAMOM:
||   2
```

### Primer 2

```
|| POKRETANJE: ./a.out AZBUKA deda mraz
|| INTERAKCIJA SA PROGRAMOM:
||   2
```

### Primer 3

```
|| POKRETANJE: ./a.out japan caj
|| INTERAKCIJA SA PROGRAMOM:
||   0
```

### Primer 4

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
||   0
```

- (d) Napisati program koji na osnovu broja  $n$  koji se zadaje kao argument komandne linije ispisuje cele brojeve iz intervala  $[-n, n]$ .

## 2.2 Pokazivači i argumenti komandne linije

---

### 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!
```

- (e) 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.
```

- (f) 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 i program koji testira rad napisane funkcije.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 123abc789XY
Modifikovana niska je: 123ABC789XY
Broj modifikacija je: 3
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: zimA
Modifikovana niska je: ZIMA
Broj modifikacija je: 3
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: SNEG
Modifikovana niska je: SNEG
Broj modifikacija je: 0
```

- (g) Napisati funkciju `void interpunkcija(int * br_tacaka, int * br_zareza)` koja za tekst koji se unosi sa standardnog ulaza sve do kraja ulaza prebrojava broj tačaka i zareza. Napisati zatim program koji testira napisanu funkciju.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
a.b.c.d
a,b,,c,d,e
Broj tacaka: 3
Broj zareza: 5
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
.....789.....
Broj tacaka: 10
Broj zareza: 0
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
sunce
Broj tacaka: 0
Broj zareza: 0
```

- (h) 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

```

- (i) Napisati funkciju `void min_max(float a[], int n, float * min, float * max)` koja izračunava minimalni i maksimalni element niza `a` dužine `n`. Napisati zatim i program koji učitava niz realnih brojeva maksimalne dužine 50 i ispisuje vrednosti minimuma i maksimuma na tri decimale.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 5
|| Unesite elemente niza:
|| 24.16 -32.11 999.25 14.25 11
|| Minimum: -32.110
|| Maksimum: 999.250

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 4
|| Unesite elemente niza:
|| -5.126 -18.29 44 29.268
|| Minimum: -18.290
|| Maksimum: 44.000

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 1
|| Unesite elemente niza:
|| 4.16
|| Minimum: 4.160
|| Maksimum: 4.160

```

## 2.3 Niske

**Zadatak 2.17** Tekst

[Rešenje [2.17](#)]

**Zadatak 2.18** Tekst

[Rešenje [2.18](#)]

**Zadatak 2.19** Tekst

[Rešenje [2.19](#)]

**Zadatak 2.20** Tekst

[Rešenje 2.20]

**Zadatak 2.21** Tekst

[Rešenje 2.21]

**Zadatak 2.22** Tekst

[Rešenje 2.22]

**Zadatak 2.23** Tekst

[Rešenje 2.23]

**Zadatak 2.24** Tekst

[Rešenje 2.24]

- (a) a) Napisati funkciju *int samoglasnik(char c)* koja proverava da li je zadati karakter samoglasnik. Funkcija treba da vrati vrednost 1 ako karakter *c* jeste samoglasnik, odnosno 0 ako nije.
- b) Napisati funkciju *int samoglasnik\_na\_kraju(char s[])* koja proverava da li se niska *s* završava samoglasnikom (koristiti funkciju iz tačke a)).
- c) Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje da li završava samoglasnikom ili ne.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: abcde  
|| Niska se 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!
```

- (b) Napisati funkciju *void kopiraj\_n(char t[], char s[], int n)* koja kopira najviše *n* karaktera niske *s* u nisku *t*. Napisati i program koji učitava nisku maksimalne dužine 20 karaktera i jedan ceo broj i testira rad napisane funkcije.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: abcdef
|| Unesite broj n: 3
|| Rezultujuca niska: abc

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: programiranje
|| Unesite broj n: 5
|| Rezultujuca niska: progr

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: abc
|| Unesite broj n: 15
|| Rezultujuca niska: abc

```

- (c) Napisati funkciju *void dupliranje(char t[], char s[])* koja na osnovu niske *s* formira nisku *t* tako što duplira svaki karakter niske *s*. Napisati i program koji učitava nisku maksimalne dužine 20 karaktera i testira rad napisane funkcije.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: zima
|| zziimmaa

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: A+B+C
|| AA++BB++CC

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: C
|| CC

```

- (d) Napisati funkciju *int heksa\_broj(char s[])* koja proverava da li je niskom *s* zadat korektan heksadekadni broj. Heksadekadni broj je korektno zadat ako počinje prefiksom *0x* ili *0X* i ako sadrži samo cifre i mala ili velika slova *A*, *B*, *C*, *D*, *E* i *F*. Funkcija treba da vrati vrednost 1 ako je niska korektan heksadekadni broj, odnosno 0 ako nije. Napisati i program koji učitava nisku maksimalne dužine 7 karaktera i ispisuje rezultat rada funkcije.

*Primer 1*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0x12EF
|| Korektan heksadekadni broj!

```

*Primer 2*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0X22af
|| Korektan heksadekadni broj!

```

*Primer 3*

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: 0aErA9
|| Nekorektan heksadekadni broj!

```

## 2 Predstavljanje podataka

---

- (e) Napisati funkciju `int hekso_broj(char s[])` koja izračunava dekadnu vrednost heksadekadnog broja zadatog niskom `s`. Napisati i program koji učitava nisku maksimalne dužine 7 karaktera i ispisuje rezultat rada funkcije. Pretpostaviti da je uneta niska korektan heksadekadni broj.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: 0x2A34  
|| 10804
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: 0xff2  
|| 4082
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: 0xE1A9  
|| 57769
```

- (f) Napisati funkciju `int podniska(char s[], char t[])` koja proverava da li je niska `t` podniska niske `s`. Napisati i program koji učitava dve niske maksimalne dužine 10 karaktera i testira rad napisane funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku s: abcde  
|| Unesite nisku t: bcd  
|| t je podniska niske s!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku s: abcde  
|| Unesite nisku t: bCd  
|| t nije podniska niske s!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku s: abcde  
|| Unesite nisku t: def  
|| t nije podniska niske s!
```

- (g) Napisati funkciju `void modifikacija(char* s)` koja modifikuje nisku `s` tako što svaki drugi karakter zameni zvezdicom. Pretpostaviti da niska `s` neće biti duža od 20 karaktera. Napisati i program koji testira rad napisane funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: 123abc789XY  
|| Modifikovana niska je: 1*3*b*7*9*Y
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite nisku: zima  
|| Modifikovana niska je: z*m*
```

## Primer 3

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: SNEG
|| Modifikovana niska je: S*E*

```

- (h) Napisati funkciju *int strspn\_klon(char \* t, char \* s)* koja izračunava dužinu prefiksa niske *t* sastavljenog od karaktera niske *s*. Napisati zatim i program koji učitava dve niske maksimalne dužine 20 karaktera i ispisuje rezultat poziva napisane funkcije.

## Primer 1

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku t: programiranje
|| Unesite nisku s: opqr
|| 3

```

## Primer 2

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku t: aaiioo124
|| Unesite nisku s: aeioo
|| 6

```

## Primer 3

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku t: 5296abc
|| Unesite nisku s: 0123456789
|| 4

```

- (i) Napisati implementaciju funkcije *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*. Učitati potom jednu nisku maksimalne dužine 20 karaktera i jedan dodatni karakter i testirati rad napisane funkcije.

## Primer 1

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: programiranje
|| Unesite karakter c: a
|| Karakter se nalazi u niski!

```

## Primer 2

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: 123456789
|| Unesite karakter c: y
|| Karakter se ne nalazi u niski!

```

## 2.4 Višedimenzioni nizovi

## 2.5 Strukture

### Zadatak 2.25 Tekst

[Rešenje 2.25]

### Zadatak 2.26 Tekst

[Rešenje 2.26]

### Zadatak 2.27 Tekst

[Rešenje 2.27]

### Zadatak 2.28 Tekst

[Rešenje 2.28]

### Zadatak 2.29 Tekst

[Rešenje 2.29]

- (a) Definirati strukturu kojom se predstavlja kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja, a zatim i program koji učitava dva kompleksna broja i ispisuje vrednost zbira, razlike, proizvoda i količnika.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite realni i imaginarni deo prvog broja: 1 2
Unesite realni i imaginarni deo drugog broja: -2 3
Zbir: -1.00+5.00*i
Razlika: 3.00-1.00*i
Proizvod: -8.00-1.00*i
Kolicnik: 0.31-0.54*i
```

- (b) Definirati strukturu *Lopta* sa poljima *poluprecnik* (ceo broj u centimetrima) i *boja* (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Zatim učitati informacije o  $n$  lopti ( $0 < n < 50$ ) i ispisati ukupnu zapreminu, kao i broj crvenih lopti. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 4
Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta, 3-crvena, 4-zelena):
1.lopta: 4 1
2.lopta: 1 3
3.lopta: 2 3
4.lopta: 10 4
Ukupna zapremina: 4494.57
Broj crvenih lopti: 2
```

- (c) Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vocka* sadrži ime vočke (nisku maksimalne dužine 20 karaktera) i količinu vitamina C u miligramima (realan broj). Napisati program koji sa standardnog ulaza 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. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

## 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: pomorandza 53.2
Unesite ime vočke i njenu količinu vitamina C: KRAJ
Voce sa najvise C vitamina je: kivi
```

- (d) Deda Mraz planira kupovinu poklona za studente koji su vredno učili C u toku godine. Na njegovoj listi se nalazi ime i prezime studenta (niske dužina do 50 karaktera) i njegova želja (niska maksimalne dužine 100 karaktera). Napisati program koji će služiti Deda Mrazu kao podsetnik: na osnovu liste koju je napravio, Deda Mraz može da unese ime i prezime studenta i da proveriti njegovu želju. Ako ima više studenata sa istim imenom i prezime-  
nom ispisati sve želje. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

## Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Ime i prezime studenta:
Pera Peric
Njegova zelja:
privezak za kljuceve
Jos vrednih studenata (da/ne)?
da
Ime i prezime studenta:
Zika Zikic
Njegova zelja:
stap za pecanje
Jos vrednih studenata (da/ne)?
da
Ime i prezime studenta:
Mara Maric
Njegova zelja:
komplet Knutovih knjiga
Jos vrednih studenata (da/ne)?
ne
Za podsecanje uneti ime i prezime:
Pera Peric
Novogodisnja zelja: privezak za kljuceve
```

## 2 Predstavljanje podataka

---

- (e) 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. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
Unesite grad i temperaturu: Beograd 7
Unesite grad i temperaturu: Uzice 1.5
Unesite grad i temperaturu: Subotica 4
Unesite grad i temperaturu: Zrenjanin 9
Gradovi sa idealnom temperaturom za klizanje u decembru:
Beograd
Subotica
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 2
Unesite grad i temperaturu: Varsava 11
Unesite grad i temperaturu: Prag 2
Gradovi sa idealnom temperaturom za klizanje u decembru:
```

- (f) Definirati strukturu *ParReci* koja sadrži reč na srpskom jeziku i odgovarajući prevod na engleski jezik. Zatim sa standardnog ulaza sve do kraja ulaza učitavati parove reči i, posebno, za rečenicu koja se zadaje sa ulaza ispisati prevod - ako je reč u rečenici nepoznata umesto nje ispisati odgovarajući broj zvezdica. Reči neće biti duže od 50 karaktera, ukupan broj parova reči neće biti veći od 100, a ukupna dužina rečenice neće biti veća od 100 karaktera. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
zima winter
godina year
sreca happiness
programiranje programming
caj tea
Unesite recenicu za prevod:
piti caj zimi je sreca
**** tea **** ** happiness
```



## 2.6 Rešenja

### Rešenje 2.1

```
1  /*
   Napisati program koji racuna skalarni proizvod dva vektora. Svaki
   vektor
3  je zadat kao celobrojni niz sa najvise 100 elemenata. Program treba
   da
   ucita dimenziju nizova (oba niza su iste dimenzije), zatim jedan po
5  jedan element niza i da ispise njihov skalarni proizvod na
   standardni
   izlaz.
7  */

9  #include <stdio.h>
   #define MAX 100

11 /*
13 Pretprocesorskom direktivom define uvode se simbolicka imena (u ovom
   slucaju
15 MAX) kojima se pridruzuje nekakav tekst (u ovom slucaju 100). Pre
   kompilacije,
   sva pojavljivanja simbolickog imena MAX bice zamenjena pridruzenim
   tekstom
17 100. MAX nije promenljiva i za nju se tokom izvršavanja programa ne
   izdvaja
   memorijski prostor.

19 MAX se u ovom zadatku koristi kao maksimalni broj elemenata niza.
   Ukoliko bismo zeleli
21 da izmenimo ovu vrednost, npr. da povecamo sa 100 na 200, sve
   sto bi bilo neophodno uraditi je da izmenimo tekst sa 100 na 200. Sa
   druge
23 strane, da nismo koristili pretprocesorsku direktivu i da smo svaki
   put
   umesto MAX direktno navodili vrednost 100, morali bismo da je
   izmenimo na svakom
25 mestu u kodu.

27 */
   int main()
29 {
   int a[MAX];
   int b[MAX];
31 int n;
   int i;
33 int s;
35
```

## 2 Predstavljanje podataka

```
37 printf("Unesi dimenziju niza:");
   scanf("%d", &n);
39
   if (n<1 || n>100)
41   {
       printf("Neispravan unos\n");
43       return -1;
   }
45
   /*
47    prvi element niza ima indeks 0, a poslednji n-1,
       gde je n broj elemenata niza; elementima niza pristupamo
49    preko indeksa; na primer, ako niz a ima 5 elemenata, mozemo
       im pristupiti pomocu
51    a[0], a[1], a[2], a[3], a[4]
53
   */
55   for (i=0; i<n; i++)
   {
57       printf("a[%d]=",i);
       scanf("%d", &a[i]);
59   }
61   for (i=0; i<n; i++)
   {
63       printf("b[%d]=",i);
       scanf("%d", &b[i]);
65   }
67   s=0;
69   for (i=0; i<n; i++)
       s = s + a[i]*b[i];
71   printf("Skalarni proizvod: %d\n",s);
73   return 0;
}
```

### Rešenje 2.2

```
/*
2   Napisati program koji učitava broj elemenata niza (n<=100),
       zatim učitava elemente niza i ispisuje:
4   a) elemente niza koji se nalaze na parnim indeksima
       b) parne elemente niza
6
   */
8
#include <stdio.h>
```

```

10 #define MAX 100
12 int main()
13 {
14     int a[MAX];
15     int n;
16     int i;
17
18     printf("Unesi dimenziju niza:");
19     scanf("%d", &n);
20
21     if (n<1 || n>MAX)
22     {
23         printf("Nekorektan unos\n");
24         return -1;
25     }
26
27
28
29
30     for (i=0; i<n; i++)
31     {
32         printf("a[%d]=",i);
33         scanf("%d", &a[i]); /* učitavamo jedan po jedan element niza */
34     }
35
36     printf("Elementi sa parnim indeksima:\n");
37     for (i=0; i<n; i+=2)
38         printf("a[%d]=%d\n",i,a[i]);
39
40     printf("Parni elementi:\n");
41     for (i=0; i<n; i++)
42         if (a[i]%2==0)
43             printf("a[%d]=%d\n",i,a[i]);
44
45     return 0;
46 }

```

### Rešenje 2.3

```

1 /*
2     Napisati program koji učitava jedan ceo broj a zatim ispisuje
3     koliko puta koja cifra ucestvuje
4     u zapisu tog broja. Nije potrebno ispisivati da se neka cifra
5     pojavila 0 puta.
6
7     Na primer, za uneti broj 4611, izlaz treba da bude:
8
9     U zapisu broja 4611, cifra 1 se pojaviljuje 2 puta
10    U zapisu broja 4611, cifra 4 se pojaviljuje 1 puta

```

## 2 Predstavljanje podataka

---

```
9      U zapisu broja 4611, cifra 6 se pojaviljuje 1 puta
11     A za uneti broj -252
13     U zapisu broja -252, cifra 2 se pojaviljuje 2 puta
14     U zapisu broja -252, cifra 5 se pojaviljuje 1 puta
15
16     */
17
18     #include<stdio.h>
19     #include<stdlib.h>
20     #define MAX 100
21
22     int main()
23     {
24         int x;
25         int brojac[10];
26         char cifra;
27         int original;
28         int i;
29
30         printf("Unesi jedan ceo broj:");
31         scanf("%d",&x);
32
33         /*
34          svaki element niza brojac predstavlja
35          brojac za jednu cifru:
36          brojac[0] sadrzi broj nula
37          brojac[1] sadrzi broj jedinica
38          ...
39          brojac[9] sadrzi broj devetki
40
41          brojac se inicijalizuju na vrednost 0
42         */
43
44         for(i=0;i<10;i++)
45             brojac[i]=0;
46
47         /*
48          vrednost promenljive x ce biti unistena
49          u while petlji jer je u svakom koraku delimo
50          sa 10; njenu vrednost cuvamo u promenljivoj
51          original kako bismo mogli da je iskoristimo
52          na kraju prilikom ispisa
53         */
54         original = x;
55
56         /*
57          Uzimamo apsolutnu vrednost broja za slucaj
58          da je uneti broj negativan
59         */
60         x=abs(x);
```

```

61      /* Izdvajanje cifara broja */
62      do
63      {
64          cifra = x%10;
65          brojaci[cifra]++; /* Uvecavamo brojac odgovarajuće cifre */
66          x/=10;
67      } while(x);
68
69      /* Ispis brojaca koji su razliciti od nule */
70      for(i=0;i<10;i++)
71          if(brojaci[i])
72              printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n",
73                  original, i, brojaci[i]);
74
75      return 0;
76  }

```

## Rešenje 2.4

```

1  /* Napisati program koji ucitava karakter po karakter do EOF i
   ispisuje koliko se puta
   u unetom tekstu pojavila svaka cifra, svako malo slovo i svako
   veliko slovo. Ispisati
3  broj pojavljivanja samo za ona mala slova, velika slova i cifre
   koji su se u unetom
   tekstu pojavili >0 puta.
5  */
7  #include <stdio.h>
9
10 int main()
11 {
12     /* Za svaku dekadnu cifru definisemo jedan brojac (tj. imamo niz
13        od 10 brojaca): brojaci[0] broji koliko se puta pojavio karakter
14        '0', brojaci[1] broji koliko se puta pojavio karakter '1' i tako
15        dalje. Svi brojaci se inicijalizuju nulama.
16     */
17     int cifre[10];
18     int mala[26];
19     int velika[26];
20
21     int c, i;
22
23     for(i=0;i<10;i++)
24         cifre[i]=0;
25
26     for(i=0;i<26;i++)
27     {

```

## 2 Predstavljanje podataka

```
29     mala[i]=0;
    velika[i]=0;
31 }
33 while((c = getchar()) != EOF)
34 {
35     if (c>='A' && c<='Z')
36         velika[c-'A']++;
37     else if (c>='a' && c<='z')
38         mala[c-'a']++;
39     else if(c >='0' && c <= '9') /* Ako je karakter c dekadna cifra
... */
        cifre[c-'0']++;          /* Uvecavamo odgovarajuci brojac za
1 */
41
42     /*
43     Izraz c - '0' ce u slucaju da je c dekadna cifra imati
44     upravo
45     vrednost 0, 1, ..., 9 za karaktere '0', '1', ..., '9' respektivno,
46     a to su upravo indeksi u nizu brojac (jer niz ima 10 elemenata,
47     pa su indeksi od 0 do 9). Time postizemo da brojac[0] broji
48     karaktere '0', itd. Isto vazi i za brojace za mala i velika slova.
49     */
50 }
51
52 /* Prikazujemo elemente niza, tj. vrednosti brojaca: */
53 for(i = 0; i < 10; i++)
54     if (cifre[i]!=0)
55         printf("Karakter %c se pojavljuje %d puta\n", '0' + i,
56             cifre[i]);
57
58 for(i = 0; i < 26; i++)
59     if (mala[i]!=0)
60         printf("Karakter %c se pojavljuje %d puta\n", 'a' + i,
61             mala[i]);
62
63     for(i = 0; i < 26; i++)
64         if (velika[i]!=0)
65             printf("Karakter %c se pojavljuje %d puta\n", 'A' + i,
66                 velika[i]);
67
68 return 0;
69 }
```

### Rešenje 2.5

```
2 /*
   Napisati program koji učitava dimenziju n dva celobrojna niza a i b
   (oba niza su iste dimenzije),
```

```

    zatim ucitava elemente oba niza i formira treci niz c tako sto
    naizmenicno rasporedjuje
4   elemente nizova a i b unutar njega: a_0,b_0,a_1,b_1,...,a_(n-1),b_(
    n-1). Program treba
    da ispise elemente novog niza c na standardni izlaz. Mozemo
    pretpostaviti da je maksimalni
6   broj elemenata u nizovima a i b 100.
    */
8
    #include <stdio.h>
10  #define MAX 100

12  int main()
    {
14      int a[MAX];
        int b[MAX];
16      int c[2*MAX];

18      int n;
        int i,j;
20

        printf("Unesi dimenziju niza:");
22      scanf("%d", &n);

24      if (n<1 || n>MAX)
        {
26          printf("Neispravan unos\n");
            return -1;
28      }

30

        printf("\nUnesi elemente niza a:\n");
32      for(i=0;i<n;i++)
        {
34          printf("a[%d]=",i);
            scanf("%d", &a[i]);
36      }

38      printf("\nUnesi elemente niza b:\n");
        for(i=0;i<n;i++)
        {
40          printf("b[%d]=",i);
            scanf("%d", &b[i]);
42      }

44      /*
46      Koristimo dva indeksa:
        1. i, sa kojim pristupamo
48          elementima niza a i b, i koji uvecavamo za 1
            nakon svake iteracije,
        2. j, sa kojim pristupamo
50          elementima niza c; s obzirom da u svakoj

```

## 2 Predstavljanje podataka

---

```
52         iteraciji dodeljujemo vrednost za dva
           elementa niza c (c[j] i c[j+1]), indeks
54         j uvecavamo za 2 nakon svake iteracije
           */
56     for(i=0,j=0;i<n;i++,j+=2)
           {
58         c[j]=a[i];
           c[j+1]=b[i];
60     }

62     printf("\nNiz c:\n");
           for(i=0;i<2*n;i++)
64         printf("c[%d]=%d\n",i,c[i]);

66     return 0;
}
```

### Rešenje 2.6

```
1  /*
   Napisati program koji ucitava dimenziju n celobrojnog niza a i
3  njegove elemente, a zatim iz niza a izbacuje sve elemente
   koji nisu deljivi svojom poslednjom cifrom, izuzev elemenata
5  cija je poslednja cifra 0 koji treba zadržati. Program treba da
   ispise
   izmenjeni niz na standardni izlaz. Mozemo pretpostaviti da niz a
7  sadrzi najvise 100 elemenata.
   */
9
11 #include <stdio.h>
12 #define MAX 100
13
14 int main()
15 {
16     int a[MAX];
17
18     int n;
19     int i,j;
20     char poslednja_cifra;
21     int novo_n;

22     printf("Unesi dimenziju niza:");
23     scanf("%d", &n);

24     if (n<1 || n>MAX)
25     {
26         printf("Neispravan unos\n");
27         return -1;
28     }
29 }
31
```



```

33 printf("\nUnesi elemente niza a:\n");
34 for(i=0;i<n;i++)
35 {
36     printf("a[%d]=",i);
37     scanf("%d", &a[i]);
38 }
39
40
41 /*
42  Dodatni indeks j se uvecava u slucaju da element na indeksu
43  i treba da ostane u nizu, tj da je deljiv svojim
44  indeksom i; u suprotnom, j se nece uvecati i
45  element i ce u narednoj iteraciji biti zamenjen elementom koji
46  jeste deljiv svojim indeksom
47 */
48
49 for(i=0,j=0;i<n;i++)
50 {
51     poslednja_cifra = a[i]%10;
52
53     /*
54      zbog lenjog izracunavanja, ako je prvi uslov
55      u disjunkciji tacan, drugi se nece ispitivati
56      (jer ce tada disjunkcija biti tacna bez obzira
57      da li je drugi uslov tacan ili ne)
58     */
59     if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
60     {
61         a[j]=a[i];
62         j++;
63     }
64 }
65 /*
66  Izbacivanjem elemenata dimenzija niza se menja, odnosno
67  smanjuje se za broj izbacenih elemenata
68 */
69 novo_n=j;
70
71 printf("Nakon izmena:\n");
72 for(i=0;i<novo_n;i++)
73     printf("a[%d]=%d\n",i,a[i]);
74
75 return 0;
76 }

```

## Rešenje 2.7

```

/*
2  a) Napisati funkciju koja ucitava sadrzaj niza.
   b) Napisati funkciju koja stampa sadrzaj niza.

```

## 2 Predstavljanje podataka

---

```
4      c) Napisati funkciju koja racuna sumu elemenata niza.
5      d) Napisati funkciju koja racuna prosechnu vrednost elemenata niza.
6      e) Napisati funkciju koja izracunava minimum elemenata niza.
7      f) Napisati funkciju koja izracunava poziciju maksimalnog elementa
8          u nizu.
9      g) Napisati program koji testira prethodne funkcije.
10
11 */
12 #include <stdio.h>
13 #define MAX 100
14
15 /* a) */
16 void ucitaj(int a[], int n)
17 {
18     int i;
19     for(i=0;i<n;i++)
20     {
21         printf("Unesi element na poziciji %d:",i);
22         scanf("%d",&a[i]);
23     }
24 }
25
26 /* b) */
27 void stampaj(int a[], int n)
28 {
29     int i;
30     for(i=0;i<n;i++)
31         printf("%d\t",a[i]);
32     printf("\n");
33 }
34
35 /* c) */
36 int suma(int a[], int n)
37 {
38     int i;
39     int s=0;
40     for(i=0;i<n;i++)
41         s+=a[i];
42     return s;
43 }
44
45 /* d) */
46 float prosek(int a[], int n)
47 {
48     int i;
49     int s = suma(a,n);
50     return (float) s/n;
51 }
52
53
54
```

```
/* e) */
56 int minimum (int a[],int n)
{
58     int m;
    int i;
60     m = a[0];

62     /*
        minimum inicijalizujemo na prvi element niza (a[0])
64     u svakom koraku poredimo vrednost minimuma
        sa jednim elementom niza, iduci redom; s obzirom
66     da je minimum inicijalizovan na a[0], nema potrebe
        porediti a[0] sa a[0] i zbog toga indeksiranje kreće
68     od 1
    */

70     for(i=1;i<n;i++)
72         if (m>a[i])
            m = a[i];

74     return m;
76 }

78 /* f) */
80 int max_pozicija (int a[],int n)
{
82     int m;
    int m_poz;
84     int i;
    m = a[0];
86     m_poz=0;

88     for(i=1;i<n;i++)
90         if (m<a[i])
            {
92             m = a[i];
            m_poz=i;
94         }

96     return m_poz;
98 }

100 int main()
102 {
    int a[MAX];
    int n;
104     printf("Unesi dimenziju niza:");
106     scanf("%d",&n);
```

## 2 Predstavljanje podataka

```
108     if (n<1 || n>MAX)
109     {
110         printf("Nekorektan unos\n");
111         return -1;
112     }
113
114     ucitaj(a,n);
115     printf("Ucitani niz:");
116     stampaj(a,n);
117
118     printf("Suma elemenata niza: %d\n", suma(a,n));
119     printf("Prosecna vrednost elemenata niza: %.2f\n", prosek(a,n));
120     printf("Minimumalni element niza: %d\n", minimum(a,n));
121     printf("Indeks maksimalnog elementa niza: %d\n", max_pozicija(a,n)
122           );
123     return 0;
124 }
```

### Rešenje 2.8

```
/*
2   a) Napisati funkciju koja ucitava sadrzaj niza.
3   b) Napisati funkciju koja stampa sadrzaj niza.
4   c) Napisati funkciju koja proverava da li niz sadrzi neku vrednost
5       m.
6   d) Napisati funkciju koja vraca vrednost prve pozicije na kojoj se
7       nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
8       nizu.
9   e) Napisati funkciju koja vraca vrednost poslednje pozicije na
10      kojoj se
11      nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
12      nizu.
13      f) Napisati funkciju koja proverava da li elementi niza cine
14          palindrom.
15      g) Napisati funkciju koja proverava da li su elementi niza
16          uredjeni
17          neopadajuce.
18      h) Napisati funkciju koja izracunava najduzu uzastopnu seriju
19          jednakih
20          elemenata u nizu. Na primer, za uneti niz 1 2 3 4 4 4 5 6 7 8 9 9
21          funkcija
22          treba da vrati 3.
23      i) Napisati program koji testira prethodne funkcije.
24 */
#include <stdio.h>
#define MAX 100

/* a) */
void ucitaj(int a[], int n)
```

```
22 {
23     int i;
24     for(i=0;i<n;i++)
25     {
26         printf("Unesi element na poziciji %d:",i);
27         scanf("%d",&a[i]);
28     }
29 }
30
31 /* b) */
32 void stampaj(int a[], int n)
33 {
34     int i;
35     for(i=0;i<n;i++)
36         printf("%d\t",a[i]);
37     printf("\n");
38 }
39
40 /* c) */
41 int sadrzi(int a[], int n, int m)
42 {
43     int i;
44     /*
45      * poredimo jedan po jedan element niza a sa datim m; ukoliko
46      * ustanovimo jednakost, to znaci da niz sadrzi element jednak
47      * m i vracamo 1
48      */
49     for(i=0;i<n;i++)
50         if (a[i]==m)
51             return 1;
52
53     /*
54      * ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
55      * ispunjen,
56      * to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
57      * vrati 0
58      */
59     return 0;
60 }
61
62 /* d) */
63 int prvo_pojavljivanje(int a[], int n, int m)
64 {
65     int i;
66     /*
67      * poredimo jedan po jedan element niza a sa datim m; ukoliko
68      * ustanovimo jednakost, vracamo indeks elementa niza a koji
69      * je jednak sa m
70      */
71     for(i=0;i<n;i++)
72         if (a[i]==m)
```

```
72         return i;

74     /*
       ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
       ispunjen,
76     to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
       vrati -1
       */
78     return -1;
   }

80
   /* e) */
82   int poslednje_pojavljivanje(int a[], int n, int m)
   {
84       int i;
86       /*
       krecemo od indeksa poslednjeg elementa, n-1
       */
88       for(i=n-1; i>=0; i--)
           if (a[i]==m)
90               return i;

92       return -1;
   }

94
   /* f) */
96   int palindrom(int a[], int n)
   {
98
100       int i, j;

102       /*
       uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
       uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
104       .
       .
       i tako redom dok je prva pozicija manja od druge
       */
106       for(i=0, j=n-1; i<j; i++, j--)
           if (a[i]!=a[j])
108               return 0;

110       return 1;
   }

112
114   /* g) */
116   int neopadajuci(int a[], int n)
   {
118       int i;

120       /*
       Funkcija neopadajuci proverava da li je dati niz sortiran
```

```

122     neopadajuće i vraća
123     1 ako jeste, a 0 u suprotnom

124     Sortiranost proveravamo na sledeći način: za svaki par susednih
        elemenata
125     a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
        proveravamo
126     da li vazi da je drugi član para manji od prvog. Ako naidjemo na
        par za koji
        to ne vazi, niz nije sortiran i funkcija vraća 0. Ukoliko se
        petlja završi
128     a da pritom uslov a[i]<a[i-1] nije nijednom bio ispunjen, to znaci
        da je
        niz sortiran i funkcija vraća 1

130
131     */
132     for(i=1; i<n; i++)
133         if (a[i]<a[i-1])
134             return 0;

136     return 1;
137 }

138 /* h) */
140 int najduza_konstanta(int a[], int n)
141 {
142     int i; /* indeks niza */
143     int j; /* dužina intervala */
144     int dužina;
145     int max_dužina=0;

146
147     for(i=0, j=0; i<n-1; i++)
148     {
149         if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
150         {
151             j++;          /* uvecavamo dužinu konstantnog intervala */

152
153             /*
154              * ako se niz završava konstantnim intervalom (nalazimo se u
155              * poslednjoj
156              * iteraciji petlje i tada je i==n-2), ispitujemo da li je
157              * taj konstantni
158              * interval maksimalne dužine
159              */
160             if(i==n-2)
161             {
162                 j++;
163                 if(j>max_dužina)
164                     max_dužina=j;
165             }
166         }
167     }

```

```
166     else
167     {
168         /*
169             izasli smo iz konstantnog intervala
170
171             ukoliko smo imali bar dva elementa u konstantnom
172             intervalu,
173             vrednost promenljive j ce biti 1, a duzina tog intervala
174             je 2;
175             zbog toga je neophodno takve (pozitivne) j uvecati za 1;
176
177             sa druge strane, ako su a[i] i a[i+1] razliciti,
178             duzina tog intervala je 0
179         */
180
181         if (j>0)
182             j++;
183
184         /* azuriramo maksimalnu duzinu uspona */
185         if(j>max_duzina)
186             max_duzina=j;
187         /*
188             duzina uspona se postavlja na nulu
189             kako bi mogli da je iskoristimo
190             za naredni uspon
191         */
192         j=0;
193     }
194
195 }
196
197 return max_duzina;
198 }
199
200 int main()
201 {
202     int a[MAX];
203     int n;
204     int m;
205     int i;
206
207     printf("Unesi dimenziju niza:");
208     scanf("%d",&n);
209
210     if (n<1 || n>MAX)
211     {
212         printf("Nekorektan unos\n");
213         return -1;
214     }
215 }
```



```

216     ucitaj(a,n);
218     printf("Ucitani niz:");
219     stampaj(a,n);
220
221     printf("Unesi jedan ceo broj:");
222     scanf("%d",&m);
223
224
225     if(sadrzi(a,n,m))
226         printf("Niz sadrzi element cija je vrednost %d\n", m);
227     else
228         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
229
230     i = prvo_pojavljivanje(a,n,m);
231     if(i!=-1)
232         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
233             prvog pojavljivanja u nizu je %d\n", m,i);
234     else
235         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
236
237     i = poslednje_pojavljivanje(a,n,m);
238     if(i!=-1)
239         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
240             poslednjeg pojavljivanja u nizu je %d\n", m,i);
241     else
242         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
243
244     if(palindrom(a,n))
245         printf("Elementi niza cine palindrom\n");
246     else
247         printf("Elementi niza ne cine palindrom\n");
248
249     if(neopadajuci(a,n))
250         printf("Niz je sortiran neopadajuće\n");
251     else
252         printf("Niz nije sortiran neopadajuće\n");
253
254     printf("Duzina najduzeg konstantnog intervala: %d\n",
255         najduza_konstanta(a,n));
256
257     return 0;
258 }

```

## Rešenje 2.9

```

1  /*
2
3  a) Napisati funkciju koja ucitava sadrzaj niza.
4  b) Napisati funkciju koja stampa sadrzaj niza.

```

## 2 Predstavljanje podataka

---

```

    c) Napisati funkciju koja proverava da li niz sadrzi neku vrednost
       m.
5   d) Napisati funkciju koja vraća vrednost prve pozicije na kojoj se
    nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
       nizu.
7   e) Napisati funkciju koja vraća vrednost poslednje pozicije na
    kojoj se
    nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
       nizu.
9   f) Napisati funkciju koja proverava da li elementi niza cine
    palindrom.
    g) Napisati funkciju koja proverava da li su elementi niza
       uredjeni
11  neopadajuće.
    h) Napisati funkciju koja izracunava najduzu uzastopnu seriju
       jednakih
13  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.
15  i) Napisati program koji testira prethodne funkcije.
    */
17  #include <stdio.h>
    #define MAX 100
19
    /* a) */
21  void ucitaj(int a[], int n)
    {
23      int i;
      for(i=0;i<n;i++)
25      {
          printf("Unesi element na poziciji %d:",i);
27          scanf("%d",&a[i]);
      }
29  }

    /* b) */
31  void stampaj(int a[], int n)
33  {
      int i;
      for(i=0;i<n;i++)
35          printf("%d\t",a[i]);
      printf("\n");
37  }
39

    /* c) */
41  int sadrzi(int a[], int n, int m)
43  {
      int i;
45      /*
       poredimo jedan po jedan element niza a sa datim m; ukoliko
47      ustanovimo jednakost, to znaci da niz sadrzi element jednak

```

```

    m i vratamo 1
49 */
    for(i=0;i<n;i++)
51         if (a[i]==m)
                return 1;
53
    /*
55     ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
        ispunjen,
        to znači da se broj m ne nalazi u nizu a i da funkcija treba da
        vrati 0
57 */
    return 0;
59 }

61 /* d) */
int prvo_pojavljivanje(int a[], int n, int m)
63 {
    int i;
65    /*
        poredimo jedan po jedan element niza a sa datim m; ukoliko
        ustanovimo jednakost, vratamo indeks elementa niza a koji
        je jednak sa m
69 */
    for(i=0;i<n;i++)
71         if (a[i]==m)
                return i;
73
    /*
75     ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
        ispunjen,
        to znači da se broj m ne nalazi u nizu a i da funkcija treba da
        vrati -1
77 */
    return -1;
79 }

81 /* e) */
int poslednje_pojavljivanje(int a[], int n, int m)
83 {
    int i;
85    /*
        krecemo od indeksa poslednjeg elementa, n-1
87 */
    for(i=n-1;i>=0;i--)
89         if (a[i]==m)
                return i;
91
    return -1;
93 }

95 /* f) */
```

## 2 Predstavljanje podataka

---

```
int palindrom(int a[], int n)
{
    int i,j;

    /*
        uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
        uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
        .
        .
        i tako redom dok je prva pozicija manja od druge
    */
    for(i=0,j=n-1;i<j;i++,j--)
        if(a[i]!=a[j])
            return 0;

    return 1;
}

/* g) */
int neopadajuci(int a[], int n)
{
    int i;

    /*
        Funkcija neopadajuci proverava da li je dati niz sortiran
        neopadajuće i vraća
        1 ako jeste, a 0 u suprotnom

        Sortiranost proveravamo na sledeći način: za svaki par susednih
        elemenata
        a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
        proveravamo
        da li važi da je drugi član para manji od prvog. Ako nađemo na
        par za koji
        to ne važi, niz nije sortiran i funkcija vraća 0. Ukoliko se
        petlja završi
        a da pritom uslov a[i]<a[i-1] nije nijednom bio ispunjen, to znači
        da je
        niz sortiran i funkcija vraća 1
    */
    for(i=1; i<n; i++)
        if (a[i]<a[i-1])
            return 0;

    return 1;
}

/* h) */
int najduza_konstanta(int a[], int n)
{
```

```
143 int i; /* indeks niza */
144 int j; /* duzina intervala */
145 int duzina;
146 int max_duzina=0;

147
148 for(i=0,j=0;i<n-1;i++)
149 {
150     if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
151     {
152         j++;          /* uvecavamo duzinu konstantnog intervala */
153
154         /*
155         ako se niz završava konstantnim intervalom (nalazimo se u
156 poslednjoj
157 iteraciji petlje i tada je i==n-2), ispitujemo da li je
158 taj konstantni
159 interval maksimalne duzine
160 */
161         if(i==n-2)
162         {
163             j++;
164             if(j>max_duzina)
165                 max_duzina=j;
166         }
167     }
168     else
169     {
170         /*
171         izašli smo iz konstantnog intervala
172
173         ukoliko smo imali bar dva elementa u konstantnom
174 intervalu,
175 vrednost promenljive j ce biti 1, a duzina tog intervala
176 je 2;
177 zbog toga je neophodno takve (pozitivne) j uvecati za 1;
178
179 sa druge strane, ako su a[i] i a[i+1] razliciti,
180 duzina tog intervala je 0
181 */
182         if (j>0)
183             j++;
184
185         /* azuriramo maksimalnu duzinu uspona */
186         if(j>max_duzina)
187             max_duzina=j;
188
189         /*
190         duzina uspona se postavlja na nulu
191 kako bi mogli da je iskoristimo
192 za naredni uspon
193 */
194     }
```

```

        j=0;
191     }
193
195 }
197     return max_duzina;
199 }
201 int main()
202 {
203     int a[MAX];
204     int n;
205     int m;
206     int i;
207
208     printf("Unesi dimenziju niza:");
209     scanf("%d",&n);
210
211     if (n<1 || n>MAX)
212     {
213         printf("Nekorektan unos\n");
214         return -1;
215     }
216
217     ucitaj(a,n);
218     printf("Ucitani niz:");
219     stampaj(a,n);
220
221     printf("Unesi jedan ceo broj:");
222     scanf("%d",&m);
223
224
225     if(sadrzi(a,n,m))
226         printf("Niz sadrzi element cija je vrednost %d\n", m);
227     else
228         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
229
230     i = prvo_pojavljivanje(a,n,m);
231     if(i!=-1)
232         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
233         prvog pojavljivanja u nizu je %d\n", m,i);
234     else
235         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
236
237     i = poslednje_pojavljivanje(a,n,m);
238     if(i!=-1)
239         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
240         poslednjeg pojavljivanja u nizu je %d\n", m,i);

```

```

241     else
242         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
243
244     if(palindrom(a,n))
245         printf("Elementi niza cine palindrom\n");
246     else
247         printf("Elementi niza ne cine palindrom\n");
248
249     if(neopadajuci(a,n))
250         printf("Niz je sortiran neopadajuće\n");
251     else
252         printf("Niz nije sortiran neopadajuće\n");
253
254     printf("Duzina najduzeg konstantnog intervala: %d\n",
255           najduza_konstanta(a,n));
256
257     return 0;
258 }

```

### Rešenje 2.10

```

1  /*
2  a) Napisati funkciju koja sve vrednosti niza uvecava za vrednost m.
3  b) Napisati funkciju koja obrće vrednosti elementima niza.
4  c) Napisati funkciju koja rotira niz ciklicno za jedno mesto u levo
5  d) Napisati funkciju koja rotira niz ciklicno za k mesta u levo.
6  e) Napisati program koji testira prethodne funkcije.
7
8  Napisati potom glavni program koji testira ovu funkciju.
9  */
10
11 #include<stdio.h>
12 #define MAX 100
13
14 void ucitaj(int a[], int n)
15 {
16     int i;
17     for(i=0;i<n;i++)
18     {
19         printf("Unesi element na poziciji %d:",i);
20         scanf("%d",&a[i]);
21     }
22 }
23
24 void stampaj(int a[], int n)
25 {
26     int i;
27     for(i=0;i<n;i++)
28         printf("%d\t",a[i]);

```

```
29     printf("\n");
30 }
31
32 void uvecaj(int a[], int n, int m)
33 {
34     int i;
35     for(i=0;i<n;i++)
36         a[i]+=m;
37 }
38
39
40 void obrni(int a[], int n)
41 {
42     int t;
43     int i,j;
44     /*
45      Niz obrcemo tako sto razmenimo vrednosti elemenata na pozicijama
46      0 i n-1,
47      zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
48      druge
49      */
50
51     for(i=0,j=n-1;i<j;i++, j--)
52     {
53         t = a[i];
54         a[i] = a[j];
55         a[j] = t;
56     }
57 }
58
59 void rotiraj1(int a[], int n)
60 {
61     int i;
62     int tmp;
63     tmp=a[0]; /* izdvajamo prvi element */
64     for(i=0;i<n-1;i++)
65         a[i]=a[i+1]; /* pomeramo preostale elemente */
66     a[n-1] = tmp; /* poslednjem elementu dodeljujemo
67                  sacuvanu vrednost prvog elementa */
68 }
69
70 void rotirajk(int a[], int n, int k)
71 {
72     int i;
73     /*
74      k puta rotiramo niz za jednu poziciju
75      ulevo
76      */
77     for(i=0;i<k;i++)
```



```
79         rotiraj1(a,n);
80     }
81
82     int main()
83     {
84         int a[MAX];
85         int n;
86         int i;
87         int k;
88         int m;
89
90         printf("Unesi dimenziju niza:");
91         scanf("%d",&n);
92
93         if (n<1 || n>MAX)
94         {
95             printf("Nekorektan unos\n");
96             return -1;
97         }
98
99         ucitaj(a,n);
100
101         printf("Unesi jedan ceo broj:");
102         scanf("%d", &m);
103
104         uvecaj(a,n,m);
105         printf("Elementi niza nakon uvecanja za %d:\n",m);
106         stampaj(a,n);
107
108         obrni(a,n);
109         printf("Elementi niza nakon obrtanja:\n");
110         stampaj(a,n);
111
112         printf("Unesi jedan pozitivan ceo broj:");
113         scanf("%d",&k);
114
115         if (k<=0)
116         {
117             printf("Nekorektan unos\n");
118             return -1;
119         }
120
121         rotiraj1(a,n);
122         printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
123         stampaj(a,n);
124
125         rotirajk(a,n,k);
126         printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n",k);
127         stampaj(a,n);
128
129         return 0;
130     }
```

### Rešenje 2.11

```
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 učitane
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 zavrsi.
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);
69      printf("main :: &a=%p, &b=%p\n", &a, &b);
70      uredi_pogresno(a,b);
71      printf("main :: nakon uredi_pogresno, a=%d, b=%d\n", a, b);
72
73      /*
74       Funkcija uredi_tacno kao argument ima dve pokazivacke
75       promenljive
76       (int*,int*). Zbog toga joj je u pozivu funkcije neophodno
77       proslediti
78       adrese promenljivih koje zelimo da uredimo rastuce, &a i &b.
79      */
80
81      uredi_tacno(&a, &b);
82      printf("main :: nakon uredi_tacno, a=%d, b=%d\n", a, b);
83
84      return 0;
85  }

```

## Rešenje 2.12

```

1  /*
2   Napisati funkciju koja za boju datu u rgb formatu
3   racuna cmy format po formulama:
4   C = 1 - ( R / 255 )
5   M = 1 - ( G / 255 )
6   Y = 1 - ( B / 255 )
7
8   Napisati program koji ucitava boju u rgb formatu,
9   primenjuje odgovarajucu funkciju i ispisuje boju u cmy formatu.
10
11  */
12
13  #include <stdio.h>

```

```
15 #include <math.h>
16
17 void rgb_to_cmy(float* a, float* b, float* c)
18 {
19     /* Zgrade su neophodne jer aritmetickie operacije
20        imaju veci prioritet od operatora dereferenciranja (*).
21     */
22     *a=1-(*a)/255;
23     *b=1-(*b)/255;
24     *c=1-(*c)/255;
25
26     /*
27     Pomocu return ne mozemo vratiti vise od jedne vrednosti.
28
29     Ceste greske:
30     return a,b,c;          return vraca samo jednu vrednost
31     return a; return b; return c; return ce vratiti samo a
32
33     Zato je neophodno da promenljive ciju vrednost
34     zelimo da promenimo prenesemo preko pokazivaca.
35     */
36 }
37
38 int rgb_korektno(float a)
39 {
40     if(a<0 || a>255)
41         return 0;
42     return 1;
43 }
44
45 int main()
46 {
47     float a,b,c;
48
49     /*
50     Argumenti funkcije rgb_to_cmy su
51     pokazivaci na float. Njima prosledjujemo
52     adrese promenljivih a, b i c.
53     */
54
55     printf("Unesi boju u rgb formatu (vrednosti izmedju 0 i 255:");
56     scanf("%f%f%f",&a,&b,&c);
57
58     if(rgb_korektno(a) && rgb_korektno(b) && rgb_korektno(c))
59         rgb_to_cmy(&a,&b,&c);
60     else
61     {
62         printf("Nekorektan unos\n");
63         return -1;
64     }
65 }
```

```

67     printf("Nakon konverzije: %.2f,%.2f,%.2f\n", a,b,c);
69     return 0;
}

```

### Rešenje 2.13

```

/*
2   Napisati funkciju koja za dve prave date svojim koeficijentima
   pravca i slobodnim clanovima odredjuje njihovu tacku preseka.
4   Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju
   tacku preseka (ako su paralelne). Napisati glavni program
6   koji ucitava podatke o pravama, poziva napisanu funkciju i
   ispisuje odgovarajucu poruku.
8 */

10 #include<stdio.h>

12 /*
   Funkcija presek treba da izracuna tri vrednosti:
14   1. indikator da li su koeficijenti pravca jednaki ili ne
   2. prvu koordinatu preseccne tacke (ukoliko prave nisu paralelne)
16   3. drugu koordinatu preseccne tacke (ukoliko prave nisu paralelne)

18   Indikator funkcija vraca kao povratnu vrednost, preko kljucne reci
   return.

20   Koordinate preseccne tacke (ako postoji) funkcija vraca preko
22   liste argumenata, zbog cega promenljive kojima ce koordinate
   biti dodeljene prenosimo preko pokazivaca (promenljive px i py)
24
   Promenljive koje sadrze podatke o pravama (k1,n1,k2,n2) se ne
26   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;

```

## 2 Predstavljanje podataka

---

```
44     float x,y;

46     printf("Unesi k i n za prvu pravu:");
    scanf("%f%f",&k1,&n1);

48     printf("Unesi k i n za drugu pravu:");
    scanf("%f%f",&k2,&n2);

50

52     if(presek(k1,n1,k2,n2,&x,&y))
        printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
    else
54         printf("Prave su paralelne\n");

56     return 0;
}
```

### Rešenje 2.14

```
1  /*
    Napisati program koji ispisuje broj navedenih argumenata komandne
    linije,
3   a zatim i same argumente i njihove redne brojeve.
    */

5  #include <stdio.h>

7

9  /*
    Argumenti komandne linije cuvaju se u nizu niski pod nazivom
    argv. Svaki element tog niza odgovara jednom argumentu komandne
11   linije pri cemu prvi element predstavlja naziv programa koji
    pokrecemo. Celobrojna promenljiva argc predstavlja ukupan
13   broj argumenata komandne linije ukljucujuci i argument koji
    odgovara nazivu programa.
15  */

17  int main(int argc, char *argv[])
    {
19         int i;

21         printf("Broj argumenata je: %d\n",argc);

23         for(i=0; i<argc; i++)
            printf("%d: %s\n",i,argv[i]);

25         return 0;
27     }
```

### Rešenje 2.15

```
1  /*
2     Napisati funkciju koja za dva data stringa str i
3     accept odredjuje koliko se uzastopnih karaktera stringa str
4     nalazi u stringu accept pocev od pocetka niza str. Napisati
5     potom program koji testira napisanu funkciju za dva stringa
6     koji se unose kao argumenti komandne linije. Primeri upotrebe:
7
8     1:
9     ./a.out aladin bal
10    3
11
12    2:
13    ./a.out aladin lad
14    4
15
16    3:
17    ./a.out Aladin ala
18    0
19
20 */
21
22 #include <stdio.h>
23 #include <string.h>
24
25 /*
26    Funkcija strspn(str,accept) je ugradjena funkcija koja vraca broj
27    karaktera
28    stringa str koji se nalaze u stringu accept, pocev od pocetka
29    stringa str.
30
31    Funkcija strspn se nalazi u zaglavlju string.h.
32
33    Funkcija strspn_klon je jedna implementacija funkcije strspn.
34
35    U zadacima cemo uvek koristiti ugradjenu funkciju strspn osim ako
36    u tekstu zadatka
37    nije naglaseno da se ona ne sme koristiti. Funkcija strspn_klon
38    sluzi da pokaze na koji
39    nacin radi ugradjena funkcija strspn.
40
41    Ugradjena funkcija strspn poziva se na isti nacin kao funkcija
42    strspn_klon:
43    strspn(s1,s2)
44
45 */
46
47 int strspn_klon(char str[], char accept[])
48 {
49     int br=0;
50     int i;
```

## 2 Predstavljanje podataka

```
47     for(i=0; str[i];i++)
48         if(strchr(accept, str[i])!=NULL)
49             br++;
50     else        /* ako pronadjemo karakter u stringu str koji nije */
51         break; /* u stringu accept, prekidamo petlju */
52
53     return br;
54 }
55
56 int main(int argc, char* argv[])
57 {
58
59     int br;
60
61     if(argc<3)
62     {
63         printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
64             arg2\n");
65         return -1;
66     }
67
68     br = strspn_klon(argv[1],argv[2]);
69     printf("Broj karaktera stringa %s koji se nalaze u stringu %s,
70         pocev od pocetka stringa %s: %d\n", argv[1],argv[2],argv[1],br);
71     return 0;
72 }
```

### Rešenje 2.16

```
/*
2   Napisati funkciju void sifruj(char s[], char c, int k) koja
   sifruje
   string s na sledeci nacin: svako malo i veliko slovo stringa s
   konvertuje u
4   slovo koje je u abecedi od njega udaljeno k pozicija, i to
   k pozicija ulevo, ako je karakter c jednak karakteru 'L' ili
   udesno
6   ako je karakter c jednak karakteru 'D'. Sifrovanje treba da bude
   kruzno. Ako string
   s sadrzi karakter koji nije alfanumericki, ostaviti ga
   nesifriranog.
8
   Napisati potom glavni program koji testira napisanu funkciju za
   string i prirodan
10  broj koji se unose kao argumenti komandne linije dok se pravac
   sifrovanja unosi
   kao opcija -p koja moze imati vrednosti 'L' ili 'D'. Ukoliko
   opcija -p nije
12  navedena, podrazumevani pravac je udesno.
14
   Mozemo podrazumevati da string sadrzi najvise 30 karaktera.
```



```
16     Primeri upotrebe:
18
19     1:
20     ./a.out abcd 2
21     cdef
22
23     2:
24     ./a.out abcd 2 -p D
25     cdef
26
27     3:
28     ./a.out abcd 2 -p L
29     yzab
30
31     4:
32     ./a.out abcd -3 -p L
33     Nekorektan unos
34
35     5:
36     ./a.out abcd 3 -p X
37     Nekorektan unos
38
39     6:
40     ./a.out ab12cd 2 -p D
41     cd12ef
42 */
43
44 #include <stdio.h>
45 #include <string.h>
46 #include <stdlib.h>
47 #define MAX 31
48
49 void sifruj(char s[], char c, int k)
50 {
51     int i;
52     int znak;
53     char t;
54
55     /*
56      S obzirom da ce korektnost unosa podataka
57      biti ispitana pre poziva funkcije, promenljiva
58      c ce imati vrednost 'L' ili 'D'.
59
60      Promenljiva znak ima vrednost 1 ili -1
61      i sluzi kao pomocna promenljiva u slucaju
62      da prilikom sifriranja konvertovani
63      karakter izadje iz opsega malih ili velikih slova.
64
65      */
66     znak=1;
```

```
68     if (c=='L')
        znak = -1;

70

72     for(i=0; s[i];i++)
        if(isalpha(s[i]))
        {
74             /*
                Promenljiva t predstavlja sifrirani karakter s[i].
                Ako je promenljiva t izvan opsega malih ili velikih slova
76             ,
                dodajemo joj ili oduzimamo ukupan broj slova u abecedi
                (26),
78             u zavisnosti od pravca sifriranja, kako bismo omogucili
                kruzno sifriranje.
80             */
            t = s[i]+znak*k;
82             if((islower(s[i]) && (t<'a' || t>'z')) || (isupper(s[i]) &&
                (t<'A' || t>'Z'))))
                s[i]=t-znak*26;
84             else
                s[i]=t;
86         }
    }

88
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
132 /* Korektnost unosa je ispitana, sto znaci da
133     argc moze biti 3 ili 5 */
134
135 if (argc==3) /* Ako je argc 3: */
136     pravac='D';
137 else /* Ako argc nije 3, tada je sigurno 5, jer je */
138 { /* korektnost unosa ispitana, a unos je korektan
139     jedino za argc==3 ili argc==5 */
140     /*
141         Ispitujemo korektnost prethodnog argumenta koji mora da
142         bude u formatu "-p".
143         Ovaj argument je string argv[3]. Njegovom prvom karakteru (
144         koji treba
145         da bude '-') pristupamo sa argv[3][0] a drugom sa argv
146         [3][1].
147     */
148     if (argv[3][0] != '-')
149     {
150         printf("Nekorektan unos: pri zadavanju opcija prvi karakter
151         mora biti '-' \n");
152         return -1;
153     }
154
155     if (argv[3][1]!='p')
156     {
157         printf("Nekorektan unos: nedozvoljena opcija\n");
158         return -1;
159     }
160
161     /*
162         Nakon argumenta -p sledi argument koji zadaje vrednost ove
163         opcije. To je
164         poslednji argument kome pristupamo sa argv[4]. Ovaj argument
165         treba
166         da sadrzi samo jedan karakter - 'L' ili 'D' i njemu
167         pristupamo sa
```

```
160         argv[4][0].
    */
162     if(argv[4][0]=='L' || argv[4][0]=='D')
        pravac=argv[4][0];
164     else
    {
166         printf("Nekorektan unos: pravac moze biti L ili D\n");
        return -1;
168     }
}

170 strcpy(rec, argv[1]);
172 sifruj(rec,pravac,k);
174 printf("Sifrovana rec: %s\n", rec);
176 return 0;
}
```

### Rešenje 2.17

```
/*
2   Napisati funkciju koja konvertuje dati string tako sto
   mala slova menja u velika a velika u mala. Napisati
4   potom glavni program koji ucitava string, poziva napisanu
   funkciju i ispisuje konvertovani string. Mozemo pretpostaviti
6   da string ne sadrzi vise od 10 karaktera.
*/
8
#include <stdio.h>
10 #include <ctype.h>

12 /*
   Kada je niz argument funkcije, dodatni argument je obavezno
14   njegova dimenzija. Kod stringova to nije slucaj jer svaki string
   ima isti poslednji element - terminirajucu nulu - i to je oznaka
16   kraja stringa.
*/
18 void konvertuj(char s[])
{
20     int i;

22     for(i=0; s[i]!='\0'; i++)
        if (s[i]>='a' && s[i]<='z')
24         s[i] = toupper(s[i]); /* toupper - konvertuje malo slovo u
        odgovarajuce veliko */
        else if (s[i]>='A' && s[i]<='Z')
26         s[i] = tolower(s[i]); /* tolower - konvertuje veliko slovo
        u odgovarajuce malo */
    /*
28     Funkcije toupper i tolower se nalaze u zaglavlju ctype.h.
    */
}
```

```

30     Konverzija malog slova u veliko bez upotrebe funkcije toupper:
        s[i] = s[i] - 'a' + 'A';
32     Konverzija velikog slova u malo bez upotrebe funkcije tolower:
        s[i] = s[i] + 'a' - 'A';
34
35     */
36 }
37
38 int main()
39 {
40     /*
41      * Poslednji karakter svakog stringa je terminirajuca
42      * nula '\0', specijalni karakter ciji je ASCII kod 0.
43
44      * Ukoliko pretpostavljamo da string sadrzi najviše 30
45      * karaktera, neophodno je deklarirati niz od 31 karaktera,
46      * pri čemu se dodatni izdvaja za terminirajuću nulu.
47
48      */
49     char s[31];
50     printf("Unesi string:");
51
52     /*
53      * Za razliku od nizova koji se učitavaju i štampaju
54      * element po element, stringovi se mogu učitati i
55      * odštampati pomoću jedne scanf/printf naredbe korišćenjem
56      * specifikatora %s.
57
58      * Funkcija scanf učitava string do prvog pojavljivanja razmaka.
59      */
60     scanf("%s", s);
61
62     konvertuj(s);
63
64     printf("Konvertovani string: %s\n", s);
65
66     return 0;
67 }

```

### Rešenje 2.18

```

1  /*
2   * Napisati funkciju skрати koja uklanja beline sa
3   * kraja datog stringa.
4
5   * Napisati glavni program koji testira napisanu
6   * funkciju na stringu "rep belina".
7
8  */

```

```
10 #include <stdio.h>
11 #include <ctype.h>
12
13 /*
14 Funkcija koja racuna duzinu niza
15 ne racunajuci '\0'.
16
17 U biblioteci string.h definisan je veliki
18 broj funkcija za rad sa stringovima,
19 ukljucujuci i funkciju strlen koja racunana
20 duzinu stringa.
21
22 Funkcija strlen_klon predstavlja jednu
23 implementaciju funkcije strlen.
24
25 U zadacima cemo uvek koristiti ugradjenu
26 funkciju strlen osim ako u tekstu zadatka
27 nije naglaseno da se ona ne sme koristiti.
28 Funkcija strlen_klon sluzi da pokaze na koji
29 nacin radi ugradjena funkcija strlen.
30
31 Ugradjena funkcija strlen poziva se na
32 isti nacin kao funkcija strlen_klon:
33 strlen(s1)
34
35 */
36 int strlen_klon(char s[])
37 {
38     int i=0;
39     while(s[i]) /* ASCII kod karaktera '\0' je jednak nuli. */
40         i++;
41
42     return i;
43 }
44
45 void skрати(char s[])
46 {
47     /*
48     Poslednji karakter stringa s(ne racunajuci '\0') ima
49     indeks strlen_klon(s)-1; ideja je da pocnemo od poslednjeg
50     karaktera stringa i da smanjujemo indeks dokle god
51     je karakter na poziciji i blanko znak.
52
53     */
54     int i;
55     for(i=strlen_klon(s)-1;i>=0;i--)
56         if (!isspace (s[i])) /* Ako s[i] nije blanko znak, prekidamo
57             petlju. */
58             break;
59
60     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
    vraca 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:
76     */
77     char s[]="rep belina";
78     /* U ovom slucaju nije neophodno navoditi dimenziju stringa vec
        ce ona biti automatski postavljena na broj karaktera u stringu +
        1 za
80     terminirajucu nulu. */
82
83     printf("Pre skracivanja: %s*\n", s);
84     skрати(s);
85     printf("Posle skracivanja: %s*\n", s);
86
87     return 0;
88 }

```

### Rešenje 2.19

```

/*
2   Napisati program koji učitava string src i formira string dst
    trostrukim nadovezivanjem stringa src. Program treba da ispise
4   string dst. Na primer, za uneti string "dan", string dst treba
    da bude "dandandan". Pretpostaviti da string src nije duzi od
6   30 karaktera.
7
*/
8
9   #include <stdio.h>
10  #include <string.h>
11
12  #define MAX 30
13  /*
14     Na stringove ne mozemo primeniti naredbu dodele.
        Ukoliko zelimo da jedan string "dodelimo" drugom,
16     mozemo koristiti ugradjenu funkciju strcpy(s,t)

```

```
18     koja kopira karaktere stringa t
    u string s zajedno za terminirajucom nulom.

20     Funkcija strcpy se nalazi u biblioteci string.h.

22     Funkcija strcpy_klon predstavlja jednu
    implementaciju funkcije strcpy.

24

26     Karakteri stringa original se, jedan po jedan,
    kopiraju u string kopija. Nakon kopiranja,
    na kraj stringa kopija dodaje se terminalna
28     nula.

30     U zadacima cemo uvek koristiti ugradjenu
    funkciju strcpy osim ako u tekstu zadatka
32     nije naglaseno da se ona ne sme koristiti.
    Funkcija strcpy_klon služi da pokaze na koji
34     nacin radi ugradjena funkcija strcpy.

36     Ugradjena funkcija strcpy poziva se na
    isti nacin kao funkcija strcpy_klon:
38     strcpy(dst,src)
    gde karaktere stringa src kopiramo
40     u string dst.

42
*/
44 void strcpy_klon(char kopija[], char original[])
{
46     int i;
    for(i=0; original[i]; i++)
48         kopija[i]=original[i];

50     kopija[i] = '\0';
}

52
int main()
54 {
    char src[MAX+1];    /* src, skraceno od source (izvor, odnosno sta
        kopiramo) */
56     char dst[3*MAX+1]; /* dst, skraceno od destination (odrediste,
        odnosno gde kopiramo) */

58     /*
        Vazno je izdvojiti dovoljno memorijskog prostora
60     za string dst: on treba da bude tri puta veci od
        maksimalne duzine stringa src + jedan karakter za
62     terminirajucu nulu.
    */

64     printf("Unesi jedan string:");
66     scanf("%s", src);
```



```

68 strcpy_klon(dst,src);

70 /*
   Funkcija strcat(s,t) nadovezuje karaktere stringa
72   t na kraj stringa s i novi string terminira
   karakterom '\0' .

74   Funkcija strcat se nalazi u biblioteci string.h.
76 */
78 strcat(dst,src);
   strcat(dst,src);

80 printf("Kada nadovezemo string %s triput: %s\n",src,dst);

82 return 0;
}

```

## Rešenje 2.20

```

1  /*
   Napisati funkciju int ucitaj_liniju(char s[], int n)
3  koja ucitava liniju maksimalne duzine n u string s
   i vraca duzinu ucitane 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
   pretpostviti da svaka linija sadrzi najvise 80 karaktera,
11  zajedno sa \n.

13 */

15 #include<stdio.h>
   #include<string.h>
17 #define MAX 81

19 /*
   Ukoliko zelimo da ucitamo string koji sadrzi beline
21   (npr liniju teksta), ne mozemo koristiti funkciju
   scanf jer ona ucitava string do prvog blanko znaka.

23   Zbog toga je neophodno napisati funkciju koja ucitava
   string karakter po karakter.

25   Ova funkcija ne dopusta unosenje vise karaktera od
   unapred odredjene granice (argument n).

27   U standardnoj biblioteci stdio.h postoji definisana
   funkcija char *gets(char *s) koja ucitava karaktere
31

```

```
33     dok se ne pojavi novi red ili EOF. Ova funkcija
34     dopusta unosenje vise karaktera nego sto string
35     s sadrzi, sto moze dovesti do neocekivanog ponasanja
36     programa.
37
38     Pored funkcije gets, koja vrsi ucitavanje sa standardnog
39     ulaza, u standardnoj biblioteci stdio.h postoji
40     i ugradjena funkcija fgets koja vrsi ucitavanje iz
41     datoteke. Nju cemo koristiti za nekoliko casova
42     kada budemo radili datoteke. Prototim funkcije fgets je
43     ovakav:
44
45     char *fgets(char *s, int size, FILE *stream);
46
47     Argumenti funkcije fgets su:
48     s - string u koji vrsimo ucitavanje
49     size - maksimalna duzina unetog stringa
50     stream - datoteka iz koje vrsimo ucitavanje
51
52     Funkcija fgets, za razliku od funkcije gets, ne dopusta
53     unos vise karaktera od date vrednosti size. Zbog toga
54     je ona sigurnija nego funkcija gets. Funkciju fgets
55     mozemo koristiti i za unos sa standardnog ulaza
56     ukoliko kao treci argument navedemo stdin.
57
58     */
59     int ucitaj_liniju(char s[], int n)
60     {
61         int i=0;
62         int c;
63
64         while((c=getchar())!='\n' && i<n-2 && c!=EOF)
65         {
66             s[i] = c;
67             i++;
68         }
69
70         /* Ucitavamo najvise n-2 karaktera jer na kraju dodajemo jos
71            dva: '\n' i '\0' */
72
73         s[i]='\n';
74         s[i+1]='\0';
75
76         return i;
77     }
78
79     int main()
80     {
81         char linija[MAX];
82         char najduza_linija[MAX];
83         int max_duzina=0;
```

```

85     int duzina;

86     /*
87      Petlja se zavrsava ukoliko je promenljiva duzina
88      jednaka nuli, sto cemo postici zadavanjem linije koja ne sadrzi
89      nijedan karakter osim EOF.
90     */

91     while ((duzina=ucitaj_liniju(linija, MAX))>0)
92     {
93         /*
94          Proveravamo da li je uneta linija duza od trenutnog
95          maksimuma i azuriramo promenljive max_duzina i najduza_linija
96          .
97         */
98         if (max_duzina<duzina)
99         {
100             max_duzina = duzina;
101             strcpy(najduza_linija,linija);
102         }
103     }

104     printf("Najduza linija: %s duzine: %d\n", najduza_linija,
105           max_duzina);

106     return 0;
107 }

```

### Rešenje 2.21

```

2  /*
3  Napisati program koji pretvara nisku u ceo broj.
4  Npr. za ulaz "-1238" se generise rezultat -1238
5  Pogledati funkcije atoi i atof koje postoje u biblioteci stdlib.h
6  */

7  #include <stdio.h>
8  #include <ctype.h>
9  #define MAX 10
10 /*
11 String b se sastoji od karaktera koji
12 cine jedan ceo broj, onim redom kojim
13 se karakteri pojavljuju u zapisu broja.
14
15 Ako je prvi karakter stringa b '-',
16 to znaci da je broj negativan i
17 funkcija znak_broja vraca -1
18
19 U suprotnom, broj je pozitivan i
20 funkcija znak_broja vraca 1

```

```
22 */
24 int znak_broja(char b[])
25 {
26     if(b[0]=='-')
27         return -1;
28     return 1;
29 }
30
31 /*
32 Funkcija formiraj_broj na osnovu
33 karaktera koji cine broj iz stringa
34 b vraca ceo broj koji odgovara
35 zapisu datom u stringu b.
36
37 Ako su cifre broja a,b,c i d, tada
38 broj mozemo kreirati kao:
39  $a \cdot 10^3 + b \cdot 10^2 + c \cdot 10^1 + d \cdot 10^0$ 
40
41 Medjutim, efikasnije je koristiti
42 Hornerovu semu:
43
44  $10 \cdot (10 \cdot (10 \cdot (10 \cdot 0 + a) + b) + c) + d$ 
45
46 */
47
48 int formiraj_broj(char b[])
49 {
50     int i;
51     int n=0;
52     int znak = znak_broja(b);
53
54     /*
55     Ako je broj negativan, cifre u nizu b
56     pocinju od indeksa 1
57     */
58
59     i=0;
60     if(znak==-1)
61         i=1;
62
63     /*
64     Funkcija isdigit proverava da li je broj
65     cifra. Nalazi se u biblioteci ctype.h
66
67 Proveravamo da li je karakter u zapisu
68 broja cifra kako bismo se osigurali
69 od nekorektnog unosa, npr ako korisnik
70 unese -123abc. Ovaj unos je moguc jer
71 se vrsi sa scanf("%s",broj), gde unosimo
72 karaktere do prvog blanko znaka
```

```

74     Ako naidjemo na karakter koji nije cifra,
75     prekidamo petlju

76
77     */
78     for(; b[i]!='\0'; i++)
79         if(isdigit(b[i]))
80             n = n*10 + b[i] - '0';
81         else
82             break;
83
84     /* Formirani broj mnozimo znakom: */
85
86     n*=znak;
87     return n;
88
89 }
90
91 int main()
92 {
93     char broj[MAX];
94     int n;
95
96     /* Ucitavamo broj: */
97     scanf("%s", broj);
98
99     /* Ispisujemo rezultat: */
100    printf("Broj zapisan kao int: %d\n", formiraj_broj(broj));
101
102    return 0;
103 }

```

## Rešenje 2.22

```

/*
2   Napisati program koji pretvara zadatu broj u nisku.
   Npr. za broj -453 treba generisati nisku "-453"
4   */

6   #include <stdio.h>
7   #include <string.h>
8   #define MAX 10
9   /*
10
11   Funkcija transformisi_negativan vraca
12   1 ako je broj negativan i 0 u suprotnom, a
13   uz to, ako broj jeste negativan, funkcija
14   treba da ga konvertuje u njegovu apsolutnu
15   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
19      2. apsolutnu vrednost broja ce vratiti
20      preko liste argumenata, zbog cega broj
21      prenosimo preko pokazivaca
22
23  */
24  int transformisi_negativan(int* pn)
25  {
26      if(*pn<0)
27      {
28          *pn = -(*pn);
29          return 1;
30      }
31      return 0;
32  }
33
34  int formiraj_niz_cifara(int n, char b[], int neg)
35  {
36      int i=0;
37      char cifra;
38
39      do
40      {
41          cifra = n%10;
42
43          /* Promenljiva b predstavlja string.
44             Da bismo na neku poziciju u stringu
45             upisali karakter koji odgovara nekoj
46             cifri, npr '2', neophodno je da
47             odgovarajucoj poziciji dodelimo vrednost
48             ASCII koda te cifre, konkretno za '2'
49             ASCII kod je '0'+2.
50
51             Greska bi bila navesti b[i]=2
52             jer 2 nije ASCII kod koji odgovara karakteru
53             '2'.
54
55          */
56          b[i]=cifra+'0';
57
58          n/=10;
59          i++;
60      } while(n);
61
62      /* Ako je broj negativan, dodajemo znak minus: */
63      if(neg)
64      {
65          b[i]='-';
66          i++;
67      }
68
69      /* Svaki string se zavrшава terminirajucom nulom: */
70      b[i]='\0';
```

```
70 }
72 void obrni(char s[])
73 {
74     char t;
75     int i,j;
76     /*
77      *Karaktere stringa obrcemo tako sto razmenimo karaktere na
78      *pozicijama 0 i n-1,
79      *zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
80      *druge
81      */
82     for(i=0,j=strlen(s)-1;i<j;i++, j--)
83     {
84         t = s[i];
85         s[i] = s[j];
86         s[j] = t;
87     }
88 }
89
90 void broj_u_niz_cifara(int n, char broj[])
91 {
92     int negativan;
93
94     /* Odredjujemo znak broja: */
95     negativan=transformisi_negativan(&n);
96
97     /* Izdvajamo cifre broja i smestamo ih u niz: */
98     formiraj_niz_cifara(n, broj, negativan);
99
100     /* S obzirom da cifre izdvajamo sa kraja broja, u nizu ce biti u
101     obrnutom redosledu.
102     Na primer, za broj 234 niz ce sadrzati cifre 4 3 2. */
103     obrni(broj);
104 }
105
106 int main()
107 {
108     int n;
109     char broj[MAX];
110     int negativan;
111
112     /* Ucitavamo broj: */
113     scanf("%d", &n);
114
115     /* Kreiramo broj na osnovu niza cifara: */
116     broj_u_niz_cifara(n,broj);
117
118     /* Ispisujemo rezultat: */
```

## 2 Predstavljanje podataka

---

```
120     printf("Broj zapisan kao string: %s\n", broj);
122     return 0;
}
```

### Rešenje 2.23

```
2  /*
   Napisati program koji učitava dva stringa i ispituje najpre da li
   su jednaki. Ako jesu, program
   treba da izda odgovarajucu poruku, a ako nisu, treba da ispita da
   li je drugi podstring
4  prvog. Ukoliko jeste, program treba da ispise pocev od kog indeksa
   prvog
   stringa pocinje drugi string. U suprotnom, ispisati odgovarajucu
   poruku. Mozemo
6  pretpostaviti da stringovi ne sadrze vise od 20 karaktera.
   */
8
10 #include <stdio.h>
12 #include <string.h>
14
16 /*
   Funkcija strcmp(s,t) je ugradjena funkcija koja utvrdjuje da
   li su strinovi
18   s i t jednaki. Ukoliko jesu, vraca 0, a u suprotnom vraca
   razliku
   ASCII kodova prva dva razlicita karaktera na istim pozicijama
16   (npr strcmp("aa","ab") ce vratiti -1 a strcmp("ab","aa") 1).
   Funkcija strcmp se nalazi u zaglavlju string.h.
20   Funkcija strcmp_klon je jedna implementacija funkcije strcmp.
   U zadacima cemo uvek koristiti ugradjenu funkciju strcmp osim
22   ako u tekstu zadatka
   nije naglaseno da se ona ne sme koristiti. Funkcija
   strcmp_klon sluzi da pokaze na koji
24   nacin radi ugradjena funkcija strcmp.
   Ugradjena funkcija strcmp poziva se na isti nacin kao funkcija
26   strcmp_klon:
   strcmp(s1,s2)
28   gde poredimo stringove s1 i s2.
   */
30
32 int strcmp_klon(char s1[], char s2[])
34 {
   int i;
   for(i=0; s1[i]==s2[i];i++)
```



```

36     if (s1[i]=='\0')
37         return 0;
38
39     return s1[i] - s2[i];
40 }
41
42 int main()
43 {
44     char s1[21];
45     char s2[21];
46     char* p;
47
48     printf("Unesi dva stringa:");
49     scanf("%s%s",s1,s2);
50
51     /*
52      Funkcija strstr(s,t) je ugradjena funkcija koja utvrđuje da
53      li je string t
54      podstring stringa s i ako jeste, vraca pokazivac (char*) na
55      karakter
56      stringa s odakle pocinje prvo pojavljivanje stringa t, a NULL
57      u suprotnom.
58
59      NULL je pokazivac koji ne pokazuje ni na sta, odnosno ne
60      sadrzi adresu
61      nijedne promenljive.
62
63      Podsetimo se veze nizova(a time i stringova) i pokazivaca:
64      ako je string deklarisan sa s1[21], tada je njegov naziv s1
65      ekvivalentan adresi prvog karaktera stringa:
66      s1 <=> &s1[0]
67      i nadalje redom:
68      s1+1 <=> &s1[1]
69      ...
70      u opstem slucaju:
71      s1+i <=> &s1[i]
72
73      To znaci da se indeks elementa na koji pokazuje s1+i moze
74      dobiti tako sto od s1+i oduzmemo pokazivac na pocetak niza:
75      s1+i-s1 <=> i. Ovako od pokazivaca na karakter u stringu
76      dobijamo njegov indeks u stringu.
77
78      */
79
80     p = strstr(s1,s2);
81
82     if (strcmp_klon(s1,s2)==0)
83         printf("Uneti stringovi su jednaki\n");
84     else if (p!=NULL)
85         printf("%s jeste podstring od %s pocev od pozicije : %d\n", s2,
86             s1, p-s1);
87     else

```

## 2 Predstavljanje podataka

---

```
84     printf("%s NIJE podstring od %s\n", s2,s1);
86     return 0;
}
```

### Rešenje 2.24

```
2  /*
4  Napisati program koji za uneti string s i karakter c utvrđuje
6  da li se c pojavljuje u stringu s i ukoliko se pojavljuje,
8  ispisuje indeks prvog pojavljivanja a u suprotnom ispisuje
10 odgovarajucu poruku. Mozemo pretpostaviti da string ima najviše
12 20 karaktera.
14 */
16 #include <stdio.h>
18 #include <string.h>
20
22 int main()
24 {
26     char s[21];
28     char c;
30     char* p;
32
34     printf("Unesi karakter:");
36     c=getchar();
38     printf("Unesi string:");
40     scanf("%s", s);
42
44     /*
46      Da smo ucitavali obrnutim redom (prvo string pa karakter)
48      to bismo realizovali na sledeci nacin:
50     printf("Unesi string:");
52     scanf("%s",s);
54     getchar();
56     printf("Unesi karakter:");
58     c=getchar();
60
62     Dodatni getchar() bi sluzio da "pokupi" karakter kojim
64     razdvajamo unos stringa i karaktera (razmak, novi red ili
66     slicno).
68
70     */
72
74     /*
76      Funkcija strchr(s,c) je ugradjena funkcija koja vraca pokazivac
78      na prvi karakter u stringu s koji je jednak karakteru c, ako
80      takav
82      postoji, a NULL u suprotnom.
84     */
86 }
```

```

    Indeks od pokazivaca dobijamo na isti nacin kao u prethodnom
    zadatku
44     sa strstr.
    */
46
    p = strchr(s,c);
48     if(p!=NULL)
        printf("%c se pojavljuje u %s na poziciji %d\n", c, s, p-s);
50     else
        printf("%c se NE pojavljuje u %s\n",c, s);
52
    return 0;
54 }

```

### Rešenje 2.25

```

/*
2
    Data je struktura koja opisuje koordinate
    tacke u ravni:
4
6     typedef struct point
    {
8         float x;
        float y;
10    } POINT;

12    U glavnom programu date su dve tacke: tacka
    A sa fiksiranim koordinatama (1,2) i tacka B
14    cije koordinate zadaje korisnik. Napisati
    funkcije:
16    a) za racunanje rastojanja izmedju dve date tacke
    b) za odredjivanje tacke koja se nalazi na
18    sredini duzi odredjene dvema datim tackama

20    Testirati napisane funkcije u glavnom programu.

22 */

24 #include <stdio.h>
    #include <math.h>
26
    typedef struct point
28 {
        float x;
30        float y;
    } POINT;
32

/*
34    Poljima strukture pristupamo pomocu
    operatora .

```

```
36     Ako je promenljiva a tipa POINT,
38     njenim koordinatama pristupamo
        pomocu a.x i a.y
40 */
42 float rastojanje (POINT a, POINT b)
43 {
44     return sqrt(pow(a.x-b.x,2)+pow(a.y-b.y,2));
45 }
46
47 POINT sredina (POINT a, POINT b)
48 {
49     POINT s;
50     s.x = (a.x+b.x)/2;
51     s.y = (a.y+b.y)/2;
52     return s;
53 }
54
55 int main()
56 {
57
58     POINT a = {1,2};
59     POINT b;
60     POINT sredina_a_b;
61
62     /* Ispisujemo koordinate tacke a. */
63     printf("Tacka a ima koordinate %.2f,%.2f\n", a.x, a.y);
64
65     /* Ucitavamo koordinate tacke b. */
66     printf("Unesi prvu koordinatu tacke: ");
67     scanf("%f", &b.x);
68     printf("Unesi drugu koordinatu tacke: ");
69     scanf("%f", &b.y);
70     printf("Tacka b ima koordinate %.2f,%.2f\n", b.x, b.y);
71
72     /* Strukture kao argumenti funkcije - prenos po vrednosti. */
73     printf("Rastojanje izmedju tacaka a i b je %.2f\n", rastojanje(a,b)
74         );
75
76     /* Struktura kao povratna vrednost funkcije. */
77     sredina_a_b=sredina(a,b);
78     printf("Tacka na sredini izmedju tacaka a i b je %.2f,%.2f\n",
79         sredina_a_b.x, sredina_a_b.y);
80
81     return 0;
82 }
```

### Rešenje 2.26

```

1  /*
   Data je struktura
3   typedef struct Student
   {
5       char ime[MAX];
       char prezime[MAX];
7       char smer;
       float prosek;
9   } STUDENT;

11 I   Napisati funkciju koja ucitava sa standardnog ulaza podatke o
       studentu. Mozemo pretpostaviti da
       ime i prezime studenta ne sadrže više od 30 karaktera.
13 II  Napisati funkciju koja ispisuje podatke o studentu na standardni
       izlaz.
       III Ucitati niz od n studenata i :
15     a) ispisati imena i prezimena onih koji su na smeru R
       b) ispisati podatke za studenta sa najvećim prosekom; ako ima
       više takvih studenata, ispisati
17     1) sve njih
       2) prvog
19     3) poslednjeg
   */
21
   #include <stdio.h>
23   #define MAXST 100
   #define MAX 31
25
   typedef struct Student
27   {
       char ime[MAX];
29       char prezime[MAX];
       char smer;
31       float prosek;
   } STUDENT;
33
   /*
35   I

37   Ako je dat pokazivac na strukturnu promenljivu s,
       poljima ove strukture pristupamo sa
39   (*s).ime,(*s).prezime, itd.

41   Zgrade su neophodne zbog prioriteta operatora:
       operator * ima veći prioritet nego operator . .

43   Operator -> pruža skraćeni zapis za prethodno
       navedeni pristup poljima:
45   s->ime je skraćeno za (*s).ime
       s->prezime je skraćeno za (*s).prezime
47   itd.

```

```
49  */
51  void ucitaj(STUDENT* s)
52  {
53      /* printf("Ime:"); */
54      scanf("%s", s->ime);
55      /* printf("Prezime:"); */
56      scanf("%s", s->prezime);
57      getchar();
58      /* printf("Smer:"); */
59      scanf("%c", &s->smer);
60      /* printf("Prosek:"); */
61      scanf("%f", &s->prosek);
62  }
63
64  /* II */
65  /*
66   Kada neku promenljivu prenosimo u funkciju kao argument, obicno
67   je prenosimo po vrednosti (bez pokazivaca), ako se ona nece menjati
68   u funkciji
69   ili po adresi (preko pokazivaca), ako ce se njena vrednost
70   promeniti u funkciji.
71
72   Prilikom poziva funkcije, za svaki argument funkcije kreira se
73   promenljiva
74   koja predstavlja lokalnu kopiju argumenta i koja prestaje da
75   postoji po zavrsetku
76   funkcije. S obzirom da se strukture sastoje od vise polja,
77   zauzimaju
78   vise memorije nego nestrukturane promenljive. Zbog toga je za
79   njihovo kopiranje
80   potrebno vise vremena i vise memorijskih resursa nego za kopiranje
81   nestrukturnih
82   promenljivih.
83
84   Da bismo ucinili program efikasnijim, korisno je da strukturu uvek
85   kao
86   argument funkcije prenosimo po adresi (preko pokazivaca), bez
87   obzira
88   da li ce se struktura u toj funkciji menjati ili ne. Pokazivac na
89   strukturu
90   zauzima manje memorije nego sama struktura pa je izrada njegove
91   kopije
92   brza a kopija pokazivaca uzima manji memorijski prostor nego kopija
93   strukture.
94
95   Kada prenosimo strukturnu promenljivu u funkciju po adresi (preko
96   pokazivaca), tada
97   imamo mogucnost da je u funkciji menjamo. Ukoliko zelimo da
98   onemogucimo promenu,
99   uz argument dodajemo kljucnu rec const. Ako pokusamo da promenimo
100  argument
```

```
87     funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
      s->smer='X');),
      kompajler ce prijaviti gresku. Na ovaj nacin obezbedujemo da
      promenljiva
89     koju smo preneli po adresi ne da bismo je promenili vec radi
      povecanja efikasnosti programa,
      ne bude, cak ni slucajno, izmenjena u funkciji.
91
    */
93
    void ispisi(const STUDENT* s)
95    {
      printf("%s %s, %c, %.2f\n",s->ime, s->prezime, s->smer, s->prosek);
97    }

99
    float najveci_prosek(STUDENT studenti[], int n)
101    {
      float m;
103      int i;
      /* Pretpostavimo da student sa indeksom 0 ima
105         maksimalni prosek. */
      m = studenti[0].prosek;
107      for(i=1;i<n;i++)
          if(m<studenti[i].prosek) /* Ako student sa indeksom i ima veci
109             prosek od maksimalnog, */
              m=studenti[i].prosek; /* menjamo maksimalni prosek */
      return m;
111    }

113
    /*
      Struktura moze da bude povratna vrednost funkcije.
115
    */
    STUDENT prvi_student_sa_najvecim_prosekom(STUDENT studenti[], int n,
      float m)
117    {
      STUDENT s;
119      int i;
      for(i=0;i<n;i++)
121          if(m==studenti[i].prosek) /* Ako naidjemo na studenta sa
              maksimalnim prosekom, prekidamo petlju. */
              {
123                  /*
                      Na strukture se moze primenjivati
125                      naredba dodele.
                  */
                  s = studenti[i];
127                  break;
129              }
      return s;
131    }
```

## 2 Predstavljanje podataka

---

```
133  /*
    Strukturu mozemo preneti u funkciju preko pokazivaca. Strukture se
    obavezno
135  prenose preko pokazivaca ukoliko je neophodno promeniti vrednosti
    njihovih
    polja u funkciji.
137  */
void poslednji_student_sa_najvecim_prosekom(STUDENT studenti[], int n
    , float m, STUDENT* s)
139  {
    int i;
141  for(i=0;i<n;i++)
        if(m==studenti[i].prosek)
143            *s = studenti[i];
    }
145
    /*
147  Napomena: funkcije
        1)prvi_student_sa_najvecim_prosekom
149        2)poslednji_student_sa_najvecim_prosekom
    odredjuju studenta sa najvecim prosekom po odredjenom kriterijumu.
151  Funkcija su realizovane na razlicite nacine kako bi ilustrovale:
    - strukturu kao povratnu vrednost
153  - prenos strukture preko pokazivaca u funkciju, s obzirom da ce se
        promeniti u funkciji

155  Prilikom izrade zadataka moze biti izabran bilo koji od opisanih
        nacina rada, osim
        ako neki nacin nije posebno naglasen u tekstu zadatka.

157
    */
159  int main()
    {
161      STUDENT studenti[MAXST];
        int n;
163        int i;
        float max_prosek;
165        STUDENT student_sa_max_prosekom;
        int indeks;
167

169  /* printf("Unesi broj studenata:"); */
        scanf("%d", &n);

171

        if (n<0 || n>MAXST)
173        {
            printf("Nekorektan unos\n");
175            return -1;
        }
177

179  /* printf("Unesi podatke o studentima:"); */
        for(i=0;i<n;i++)
```



```

181 {
182     /* printf("%d. student:\n", i); */
183     ucitaj(&studenti[i]);
184 }
185
186 printf("Studenti sa R smer:\n");
187 for(i=0;i<n;i++)
188     if(studenti[i].smer == 'R')
189         ispisi(&studenti[i]);
190 printf("-----\n");
191
192 /* b)1)
193
194     Stampamo podatke o svim studentima sa
195     maksimalnim prosekom.
196 */
197
198 max_prosek = najveći_prosek(studenti, n);
199 printf("Svi studenti koji imaju maksimalni prosek:");
200 for(i=0;i<n;i++)
201     if(studenti[i].prosek==max_prosek)
202         ispisi(&studenti[i]);
203
204 /* b)2) */
205 student_sa_max_prosekom = prvi_student_sa_najvecim_prosekom(
206     studenti,n,max_prosek);
207
208 printf("Prvi student u nizu sa najvecim prosekom: ");
209 ispisi(&student_sa_max_prosekom);
210
211 /* b)3) */
212 poslednji_student_sa_najvecim_prosekom(studenti,n,max_prosek,&
213     student_sa_max_prosekom);
214
215 printf("Poslednji student u nizu sa najvecim prosekom: ");
216 ispisi(&student_sa_max_prosekom);
217
218 return 0;
219 }

```

### Rešenje 2.27

```

1 /*
2
3 Napisati program koji ucitava reci sa standardnog ulaza dok korisnik
4 ne zada EOF i ispisuje
5 ih na standardni izlaz svaku u posebnom redu, poravnatu udesno u
6 odnosu
7 na poslednji karakter najduze reci. Koristiti
8 strukturu typedef struct rec

```

## 2 Predstavljanje podataka

---

```
7      {
9          char s[21];
          int duzina;
          }REC;
11 Na primer, ako su unesene sledece reci:
    Danas imamo ispit iz programiranja1.
13 Nadam se da nece biti tesko!
    onda ispis izgleda ovako:
15         Danas
            imamo
17         ispit
            iz
19 programiranja1.
        Nadam
21         se
            da
23         nece
            biti
25         tesko!

27 Program realizovati kroz sledece funkcije:
a) Funkciju za učitavanje jedne reci u strukturu REC.
29 b) Funkciju za učitavanje niza struktura koja vraca dimenziju niza
c) Funkciju koja odredjuje maksimalnu duzinu reci u datom nizu
31 d) Funkciju koja ispisuje reci u trazenom formatu

33 Mozemo pretpostaviti da nijedna rec ne sadrzi vise od 30 karaktera i
    da nece biti
    uneto vise od 1000 reci.

35 */
37 #include<stdio.h>
39 #include<string.h>
    #define MAXRECI 100
41 #define MAX 31

43 typedef struct rec
    {
45         char s[MAX];
            int duzina;
47     }REC;

49 void ucitaj_rec(REC* rec)
51 {
    scanf("%s", rec->s);
53     rec->duzina = strlen(rec->s);
    }

55 /*
57     U funkciji ucitaj_niz_reci argument n oznacava broj
```

```
59      elemenata niza reci, koji ce biti poznat tek po
60      zavrsetku funkcije. Ova promenljiva ce dobiti svoju
61      vrednost u funkciji i zbog toga mora biti prenesena
62      preko pokazivaca.
63  */
64
65  void ucitaj_niz_reci(REC reci[], int* pn, int granica)
66  {
67      int i=0;
68      do
69      {
70          ucitaj_rec(&reci[i]);
71          i++;
72      }
73      while(reci[i-1].duzina>0 && (i-1)<granica);
74
75      /*
76       S obzirom da se promenljiva i ucitava
77       pre ispitivanja uslova, uslov ispitujemo
78       za rec sa indeksom i-1
79      */
80
81      *pn = i-1;
82
83      /*
84       S obzirom da se vrednost promenljive i
85       ucitava i kada je unesen EOF, dimenzija
86       niza odgovarace vrednosti i-1
87      */
88  }
89
90  int max_duzina(REC reci[], int n)
91  {
92      int najveca_duzina;
93      int i;
94
95      /*
96       Najvecu duzinu inicijalizujemo na duzinu
97       prve reci.
98      */
99      najveca_duzina = reci[0].duzina;
100
101      for(i=1;i<n;i++)
102          if(reci[i].duzina>najveca_duzina) /* Ukoliko u nizu naidjemo
103              na rec duzine vece od najvece duzine, */
104              najveca_duzina = reci[i].duzina; /* menjamo vrednost
105              promenljive najveca_duzina. */
106
107      return najveca_duzina;
108  }
```

## 2 Predstavljanje podataka

---

```
109  /*
110     Da bismo realizovali ispis u trazenom formatu, pre
111     svake reci ispisujemo onoliko razmaka koliko iznosi
112     razlika maksimalne duzine i duzine date reci.
113  */
114
115  void ispis(REC reci[], int n, int max_d)
116  {
117      int i,j;
118
119      for(i=0;i<n;i++)
120      {
121          for(j=0;j<max_d-reci[i].duzina;j++)
122              printf(" ");
123          printf("%s\n", reci[i].s);
124      }
125  }
126
127  int main(int argc, char* argv[])
128  {
129      REC reci[MAXRECI];
130      int najveca_duzina;
131      int n;
132
133      ucitaj_niz_reci(reci, &n, MAXRECI);
134      najveca_duzina = max_duzina(reci,n);
135      ispis(reci, n, najveca_duzina);
136
137      return 0;
138  }
```

### Rešenje 2.28

```
2  /*
3     Napisati program koji izracunava prosečnu cenu jedne potrosacke
4     korpe. Potrosacka korpa se sastoji od broja kupljenih artikala i
5     niza kupljenih artikala. Svaki artikal odredjen je svojim nazivom,
6     kolicinom i cenom. Program treba da ucita broj potrosaca n (
7     najvise 100),
8     zatim podatke za n potrosackih korpi i da na osnovu ucitanih
9     podataka
10    izracuna prosečnu cenu potrosacke korpe. Ucitavanje se vrši sa
11    standardnog
12    ulaza pri čemu se prvo zada broj artikala, a zatim za svaki
13    artikal naziv,
14    kolicina i cena.  Mozemo pretpostaviti da nijedan
15    potrosac neće kupiti više od 20 artikala, kao i da naziv svakog
16    artikla
17    sadrži maksimalno 30 karaktera.
18  */
```

```
14  */
15  #include <stdio.h>
16  #define MAXART 20
17  #define MAXPOT 100
18  #define MAXNAZIV 31
19
20  typedef struct artikal
21  {
22      char naziv[MAXNAZIV];
23      int kolicina;
24      float cena;
25  } ARTIKAL;
26
27  typedef struct korpa
28  {
29      int br_art;
30      ARTIKAL artikli[MAXART];
31  } KORPA;
32
33  /*
34   Funkcija ucitaj_artikal ucitava podatke za jedan
35   artikal i vraca 1 ako je ucitavanje bilo uspesno
36   a 0 u suprotnom. Ucitavanje je neuspesno ukoliko
37   kolicina nekog artikla ili njegova cena nisu pozitivni
38   brojevi.
39
40   S obzirom da funkcija ucitaj_artikal treba da vrati
41   dve vrednosti (ucitanu strukturu i indikator uspesnosti),
42   strukturu ARTIKAL prenosimo preko pokazivaca a
43   indikator uspesnosti vracamo kao povratnu vrednost.
44  */
45
46  int ucitaj_artikal(ARTIKAL* a)
47  {
48
49      scanf("%s", a->naziv);
50      scanf("%d", &a->kolicina);
51
52      if (a->kolicina<=0)
53      {
54          printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina
55          );
56          return 0;
57      }
58
59      scanf("%f",&a->cena);
60      if (a->cena<0)
61      {
62          printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
63          return 0;
```

```
64     }
65
66     return 1;
67 }
68
69 /*
70 Funkcija izracunaj_racun izracunava racun date
71 potrosacke korpe u kojoj su inicijalizovani
72 podaci o broju artikala i o svakom pojedinacnom
73 artiklu.
74 */
75 float izracunaj_racun(const KORPA* k)
76 {
77     int i;
78     float racun=0;
79     for(i=0; i<k->br_art; i++)
80         racun+=k->artikli[i].kolicina * k->artikli[i].cena;
81     return racun;
82 }
83
84 /*
85 Pri ucitavanju korpe, zadaje se broj artikala a zatim
86 podaci za svaki artikal.
87
88 Funkcija ucitaj_korpu vraca 1 ako je ucitavanje uspesno
89 i 0 u suprotnom. Do neuspesnog ucitavanja moze doci
90 ako broj artikala u korpi nije pozitivan ili ako dodje
91 do neuspesnog ucitavanja nekog artikla.
92 */
93
94 int ucitaj_korpu(KORPA* k)
95 {
96     int i;
97     scanf("%d", &k->br_art);
98     if (k->br_art<=0)
99     {
100         printf("Nekorektan unos za broj artikala: %d\n", k->br_art);
101         return 0;
102     }
103     for(i=0; i<k->br_art; i++)
104         if (ucitaj_artikal(&k->artikli[i])!=0)
105             return 0;
106
107     return 1;
108 }
109
110 /*
111 Funkcija ucitaj_niz_korpi ucitava podatke
112 za niz od n potrosackih korpi. Funkcija
113 vraca 1 ako je ucitavanje uspesno i 0 ako
114 nije. Ucitavanje je neuspesno ukoliko ne uspe
115 ucitavanje jedne od korpi.
```

```
116 */
118 int ucitaj_niz_korpi(KORPA korpe[], int n)
119 {
120     int i,j;
121     for(i=0; i<n; i++)
122         if(ucitaj_korpu(&korpe[i])==0)
123             return 0;
124
125     return 1;
126 }
127
128 /*
129 Funkcija stampaj_racun ispisuje na
130 standardni izlaz racun za datu korpu
131 tako sto za svaki artikal ispise
132 naziv, cenu i kolicinu i na kraju
133 ukupnu cenu za kupljene artikle.
134 */
135
136 void stampaj_racun(const KORPA* k)
137 {
138     int i,j;
139     for(i=0; i<k->br_art; i++)
140         printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
141             kolicina, k->artikli[i].cena);
142     printf("-----\n");
143     printf("\tukupno: %.2f\n", izracunaj_racun(k));
144 }
145
146 /*
147 Funkcija stampaj_racune_za_korpe
148 ispisuje na standardni izlaz racune
149 za svaku korpu u nizu potrosackih
150 korpi
151 */
152
153 void stampaj_racune_za_korpe(KORPA korpe[], int n)
154 {
155     int i;
156     for (i=0; i<n; i++)
157     {
158         printf("\nKorpa %d:\n", i);
159         stampaj_racun(&korpe[i]);
160     }
161 }
162
163 /*
164 Funkcija prosek racuna prosečnu cenu
165 potrosacke korpe za dati niz potrosackih
```

```
korpi
168 */
float prosek(KORPA korpe[], int n)
170 {
    int i;
172    float p;

    for(i=0; i<n; i++)
        p+=izracunaj_racun(&korpe[i]);
176
    return p/n;
178 }

180 int main()
{
    int n;
182    KORPA korpe[MAXPOT];

184
    printf("Unesi broj potrosackih korpi:");
186    scanf("%d", &n);

188    if(n<0 || n>MAXPOT)
    {
190        printf("Nekorektan unos broja potrosackih korpi: %d\n", n);
        return -1;
192    }

194    if (ucitaj_niz_korpi(korpe, n)==0)
        return -1;

196
    stampaj_racune_za_korpe(korpe, n);
198    printf("Prosecna cena potrosacke korpe: %.2f\n", prosek(korpe, n))
        ;

200    return 0;
}
```

### Rešenje 2.29

```
1 /*
    Struktura IZRAZ opisuje numericki izraz nad celim brojevima koji
    se sastoji
3    od dva celobrojna operanda, numericke operacije nad celim
    brojevima i
    vrednosti izraza:

5
    typedef struct izraz
7    {
    char o;
9    int x;
    int y;
```



```

11     } IZRAZ;

13     a) Napisati funkciju koja ispituje da li je dati izraz korektno
14        zadat i vraća 1 ako jeste a 0 u suprotnom. Podrazumevamo da je
15        izraz korektno zadat ako operacija odgovara +,-,* ili / i u
16        slucaju
17        deljenja drugi operand je razlicit od 0.

18     b) Napisati funkciju koja za dati izraz odredjuje vrednost izraza.

19     c) Napisati funkciju koja ucitava dati izraz. Funkcija
20        treba da ucita sa standardnog ulaza operaciju i dva
21        operanda u polja o, x i y strukture IZRAZ. Funkcija vraća
22        1 ako je ucitavanje bilo uspesno, tj. ako je izraz bio
23        korektno zadat ili 0 u suprotnom.

24     d) Napisati funkciju koja stampa dati izraz infiksno, u obliku
25        x o y = vr. Na primer, za izraz + 4 17 ispis treba
26        da bude 4+17=21

27

28     e) Napisati glavni program koji ucitava prirodan broj n<1000 a
29        zatim n izraza
30        u notaciji
31        + 4 17
32        - 8 -16
33        Program treba da ispise maksimalnu vrednost medju unetim izrazima
34        i da ispise one
35        izraze cija je vrednost manja od polovine maksimalne vrednosti.

36

37

38 */

41 #include <stdio.h>
42 #define MAX 1000

43 typedef struct izraz
44 {
45     char o;
46     int x;
47     int y;
48 } IZRAZ;

49

50 /*

51 Funkcija korektan_izraz vraća 1 ako je izraz korektan a 0
52 u suprotnom. Izraz je korektan ukoliko se sastoji od
53 aritmetickih operacija +,-,* ili /, i ukoliko je u slucaju
54 operacije deljenja drugi operand razlicit od nule.

55 */
56 int korektan_izraz(const IZRAZ* izraz)

```

```
{
61     if(izraz->o!='+' && izraz->o!='-' && izraz->o!='*' && izraz->o!='/'
        ')
        {
63             printf("Nedozvoljena operacija!\n");
            return 0;
65         }
        if(izraz->o=='/' && izraz->y==0)
67         {
            printf("Deljenje nulom!\n");
69             return 0;
        }
71     return 1;
}

73
75 /*
76  Promenljiva izraz ce se promeniti u funkciji
77  vrednost tako sto ce njenom neinicijalizovanom
78  polju vr biti dodeljena vrednost izraza. Zbog
79  toga ovu promenljivu funkciji prosledjujemo
80  po adresi, preko pokazivaca
81 */
82
83 int vrednost(const IZRAZ* izraz)
84 {
85     int v;
86
87     switch (izraz->o)
88     {
89         case '+':
90             v=izraz->x+izraz->y;
91             break;
92         case '-':
93             v=izraz->x-izraz->y;
94             break;
95         case '*':
96             v=izraz->x*izraz->y;
97             break;
98         case '/':
99             v=izraz->x/izraz->y;
100             break;
101     }
102     return v;
103 }
104
105 /*
106  Promenljiva izraz ce se promeniti u funkciji
107  ucitaj_izraz tako sto ce njenim neinicijalizovanim
108  poljima o,x,y biti dodeljene vrednosti ucitane
109  sa standardnog ulaza. Zbog toga ovu promenljivu
110  funkciji prosledjujemo po adresi, preko pokazivaca.
```

```
111     S obzirom da učitavanje karaktera nije prvo
113     učitavanje koje se obavlja u programu, funkcijom
115     getchar() "pokupimo" karakter kojim razdvajamo
        unos karaktera od prethodnog unosa (najcesce blanko
        znak)
117
118     */
119
120     int ucitaj_izraz(IZRAZ* izraz)
121     {
122         getchar();
123         scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
124         if (!korektan_izraz(izraz))
125             return 0;
126         return 1;
127     }
128
129     void stampaj_izraz(const IZRAZ* izraz)
130     {
131         printf("%d %c %d = %d\n", izraz->x, izraz->o, izraz->y, vrednost(
            izraz));
132     }
133
134     int max_vr(IZRAZ izrazi[], int n)
135     {
136         int i;
137         int max;
138         /* Trazimo maksimalnu vrednost izraza */
139         max=vrednost(&izrazi[0]);
140
141         /* U petlji... */
142         for(i=1; i<n; i++)
143             /* Ako je ona veca od maksimalne: */
144             if(vrednost(&izrazi[i])>max)
145                 /* Azuriramo max: */
146                 max=vrednost(&izrazi[i]);
147         return max;
148     }
149
150     int main()
151     {
152         int n;
153         IZRAZ izrazi[MAX];
154         int max;
155         int i;
156
157         /* Ucitavamo broj izraza: */
158         scanf("%d", &n);
159         if(n<0 || n>MAX)
160         {
```

```
163     printf("Nekorektna vrednost broja n!\n");
164     return -1;
165 }
166
167 /* U petlji ucitavamo jedan po jedan izraz: */
168 for(i=0; i<n; i++)
169     if(ucitaj_izraz(&izrazi[i])==0)
170     {
171         printf("Nekorektan unos\n");
172         return -1;
173     }
174
175 printf("Svi izrazi:\n");
176 for(i=0; i<n; i++)
177     stampaj_izraz(&izrazi[i]);
178
179 max = max_vr(izrazi, n);
180 printf("Maksimalna vrednost izraza:%d\n", max);
181
182 printf("Izrazi cija je vrednost manja od polovine maksimalne
183     vrednosti:\n");
184
185 for(i=0; i<n; i++)
186     if(vrednost(&izrazi[i])<max/2)/* Ako je vrednost tekuceg izraza
187         manja od polovine maksimalne, ispisujemo ga. */
188         stampaj_izraz(&izrazi[i]);
189
189 return 0;
190 }
```

# 3

## Datoteke

**Zadatak 3.1** Tekst

[Rešenje 3.1]

**Zadatak 3.2** Tekst

[Rešenje 3.2]

**Zadatak 3.3** Tekst

[Rešenje 3.3]

**Zadatak 3.4** Tekst

[Rešenje 3.4]

**Zadatak 3.5** Tekst

[Rešenje 3.5]

**Zadatak 3.6** Tekst

[Rešenje 3.6]

(a) 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
```

### 3 Datoteke

---

- (b) Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*.

#### Primer 1

```
|| ULAZ.TXT
|| Volim programiranje.
|| IZLAZ.TXT
|| Vipgmae
```

- (c) 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 obradjuvane:
|| Petlje
|| Funkcije
|| Nizovi
|| Strukture
||
|| IZLAZ:
|| Teme koje su obradjuvane:
|| Funkcije
|| Strukture
```

#### Primer 2

```
|| POKRETANJE: ./a.out test.txt
|| IZLAZ:
|| Greska: Pogresan broj argumenata!
```

- (d) Napisati program koji prebrojava koliko se linija datoteke *ulaz.txt* završava niskom  $s$  koja se učitava sa standardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera, kao i da dužina niske  $s$  neće biti veća od 20 karaktera.

#### Primer 1

```
|| ULAZ.TXT
|| abcde abcde
|| abcde aab
|| abcde abcde abcde
|| abcde abcde Aab
|| abcde abcde ab
|| abcde abcde abcde abcde
||
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: ab
|| Broj linija: 3
```

#### Primer 2

```
|| ULAZ.TXT
|| abcde abcde
|| abcde
|| abcde abcde AB
||
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku s: ab
|| Broj linija: 0
```

- (e) 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
```

- (f) 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
```

- (g) 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: Nedoovoljen broj tacaka!
```

- (h) 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
```

- (i) Napisati program koji linije koji se učitavaju sa standardnog ulaza sve do kraja ulaza prepisuje u datoteku *izlaz.txt* i to, ako je prilikom pokretanja zadata opcija *-v* ili *-V* samo one linije koje počinju velikim slovom, ako je zadata opcija *-m* ili *-M* samo one linije koje počinju malim slovom, a ako je opcija izostavljena sve linije. Pretpostaviti da linije neće biti duže od 80 karaktera.

#### Primer 1

```
POKRETANJE: ./a.out -m
INTERAKCIJA SA PROGRAMOM:
Unesite recenice:
programiranje u C-u je zanimljivo
Volim programiranje!
Kada porastem bicu programer!
u slobodno vreme programiram

IZLAZ.TXT
programiranje u C-u je zanimljivo
u slobodno vreme programiram
```

#### Primer 2

```
POKRETANJE: ./a.out -V
INTERAKCIJA SA PROGRAMOM:
Unesite recenice:
programiranje u C-u je zanimljivo
Volim programiranje!
Kada porastem bicu programer!
u slobodno vreme programiram

IZLAZ.TXT
Volim programiranje!
Kada porastem bicu programer!
```

#### Primer 3

```
POKRETANJE: ./a.out -k
INTERAKCIJA SA PROGRAMOM:
Greska: Pogresno pokretanje programa!
```

## 3.1 Rešenja

### Rešenje 3.1

```
1 /*
```



```
3      Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
      datoteku izlaz.txt karakter po karakter.
4
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 int main()
11 {
12     int c;
13     FILE *ulaz, *izlaz;
14
15     /*
16      Promenljive ulaz i izlaz predstavljaju
17      pokazivace na ugradjenu strukturu FILE.
18      Unutar ove strukture nalaze se polja neophodna
19      za rad sa datotekama.
20
21      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     }
```

```
53     izlaz= fopen("izlaz.txt", "w");
54     if(izlaz==NULL)
55     {
56         fprintf(stderr,"error fopen(): Neuspelo otvaranje datoteke
57         izlaz.txt za citanje.\n");
58         exit(EXIT_FAILURE);
59     }
60
61     /*
62     Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
63     Povratna vrednost ove funkcije je ascii kod unetog
64     karaktera.
65
66     Funkcija fputc ispisuje karakter c u datoteku izlaz.
67     */
68
69     while((c=fgetc(ulaz))!=EOF)
70         fputc(c,izlaz);
71
72     /*
73     Nakon zavrsetka rada sa datotekama, neophodno ih je
74     zatvoriti pomocu ugradjene funkcije fclose.
75     */
76     fclose(ulaz);
77     fclose(izlaz);
78     return 0;
79 }
```

#### Rešenje 3.2

```
2  /*
3  Napisati program koji u datoteci cije se ime navodi kao prvi
4  argument komandne linije odredjuje liniju maksimalne duzine i
5  ispisuje je na standarni izlaz. Ukoliko ima vise takvih linija,
6  ispisati onu koja je leksikografski prva. Mozemo pretpostaviti
7  da datoteka ne sadrzi linije duze od 80 karaktera.
8  */
9
10 #include <stdio.h>
11 #include <stdlib.h>
12 #include <string.h>
13 #define MAX_LEN 81
14
15 int main(int argc, char* argv[])
16 {
17     char linija[MAX_LEN];
18     char max_linija[MAX_LEN];
19     int duzina;
20     int max_duzina;
21
22     FILE *ulaz, *izlaz;
```

```
22  /*
23     Proveravamo da li poziv programa ima dovoljan broj argumenata.
24  */
25  if(argc!=2)
26  {
27      fprintf(stderr, "Greska: program se pokrece sa: %s
28         ime_ulazne_datoteke\n", argv[0]);
29      exit(EXIT_FAILURE);
30  }
31
32  ulaz=fopen(argv[1], "r");
33  if(ulaz==NULL)
34  {
35      fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke %s
36         za citanje.\n", argv[1]);
37      exit(EXIT_FAILURE);
38  }
39
40  /*
41     Funkcija fgets učitava jednu liniju teksta maksimalne duzine
42     MAX_LEN
43     iz datoteke ulaz u string linija. Ukoliko učitavanje ne uspe (
44     na primer,
45     zato sto smo dosli do kraja datoteke), povratna vrednost ove
46     funkcije
47     bice prazan pokazivac (NULL).
48  */
49
50  max_duzina=0;
51  while(fgets(linija, MAX_LEN, ulaz)!=NULL)
52  {
53      duzina = strlen(linija);
54      /*
55         Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u
56         petlju.
57         Ovu promenljivu menjamo kada je duzina učitana linije
58         veca od max_duzina ili kada su jednake, ali je učitana
59         linija
60         leksikografski ispred trenutne linije sa maksimalnom duzinom
61         .
62
63         Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii
64         kodova prva dva
65         razlicita karaktera stringova s1 i s2 na istim indeksima,
66         ukoliko oni
67         postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga
68         osetljiva
69         na mala i velika slova (npr 'D' je leksikografski ispred 'p
70         ').
71     */
72  }
```

```
62     if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija,
max_linija)<0))
    {
64         strcpy(max_linija, linija);
        max_duzina=duzina;
66     }
    }
68
69     /*
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.
    */
76     fputs(max_linija, stdout);
78     fclose(ulaz);
    return 0;
80 }
```

#### Rešenje 3.3

```
/*
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
13  /*
    Funkcija ucitaj_i_prebroj ucitava brojeve
14  iz datoteke na koju pokazuje f i prebrojava
    koliko je medju njima k-tocifrenih brojeva
16  */
    int ucitaj_i_prebroj(FILE* f, int k)
18  {
    int n;
20  int x;
    int i;
```

```
22     int br;

24     /* U datoteci je prvo naveden ukupan broj brojeva. */
    fscanf(f, "%d", &n);

26     /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
       */
28     if(n<=0)
    {
30         fprintf(stderr, "Greska: broj n mora biti prirodan\n");
        exit(EXIT_FAILURE);
32     }

34     br=0;
    for(i=0; i<n; i++)
36     {
        fscanf(f, "%d", &x);
38         if(broj_cifara(x)==k)
            br++;
40     }

42     return br;
}

44 int broj_cifara(int x)
46 {
    int br_c;

48     br_c=0;

50     /*
52     Do while petlja je pogodnija od petlji sa preduslovom
        jer tacno racuna broj cifara i za broj 0.
54     */
    do
56     {
        br_c++;
58         x/=10;
    } while(x);

60     return br_c;
62 }

64 int main(int argc, char* argv[])
66 {
    int n;
    int k;
68     FILE* f;
    int br;

70     if(argc!=3)
72     {
```

```
74     fprintf(stderr, "Greska: program se pokrece sa: %s\n", argv[0]);
76     exit(EXIT_FAILURE);
78 }
80 f=fopen(argv[1], "r");
82 if(f==NULL)
84 {
86     fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s\n", argv[1]);
88     exit(EXIT_FAILURE);
90 }
92 /* Argumenti komandne linije su stringovi. Da bismo konvertovali
94    string
96    u ceo broj koristimo ugradjenu funkciju atoi. */
98 k = atoi(argv[2]);
100 if (k<=0)
102 {
104     fprintf(stderr, "Greska: broj k mora biti prirodan\n");
106     exit(EXIT_FAILURE);
108 }
110 printf("Broj %d-cifrenih brojeva u datoteci: %d\n", k,
112        ucitaj_i_prebroj(f,k));
114 fclose(f);
116 return 0;
118 }
```

#### Rešenje 3.4

```
1  /*
3   U datoteci cije se ime navodi kao prvi argument komandne
5   linije navedena je rec r i niz linija. Napisati
7   program koji u datoteku cije se ime navodi kao
9   drugi argument komandne linije upisuje sve linije
11  u kojima se rec r pojavljuje bar n puta, gde je
13  n prirodan broj koji se unosi sa standardnog ulaza. Ispis
15  treba da bude u formatu broj_pojavljivanja: linija.
17  */
19  #include <stdio.h>
21  #include <stdlib.h>
23  #define MAXL 81
25  #define MAXR 31
27  /*
29  Funkcija broj_pojavljivanja broji koliko
```

```

    se puta pojavio string t u stringu s
19 */
int broj_pojavljivanja(char s[], char t[])
21 {
    int br;
23     int i,j;
    /*
25     i - indeks karaktera u s
        j - indeks karaktera u t
27     br - brojac koliko se puta t javlja u s
    */
29     br=0;
    for(i=0;s[i];i++)
31     {
        for(j=0;t[j];j++)
33         if(s[i+j]!=t[j]) /* Ako naidjemo na razlicite karaktere, */
            break;        /* prekidamo petlju. */
35         /*
            Do prekida petlje moze doci ili zbog toga sto su pronadjeni
37         razliciti karakteri i usledio je break ili zbog toga sto
            je prestao da vazi uslov petlje, odnosno karakter t[j] je
39         jednak '\0'. Ako vazi drugi slucaj, to znaci da se string
            t nalazi u stringu s pocev od indeksa i i potrebno je
            uvecati
41         brojac br.
        */
43         if (t[j]!='\0')
            br++;
45     }

47     return br;
}

49 int main(int argc, char* argv[])
{
51     char rec[MAXR];
    char linija[MAXL];
53     FILE* in, *out;
    int n;
55     int br;

57     if(argc!=3)
    {
59         fprintf(stderr, "Greska: program se pokrece sa: %s
            ime_ulazne_datoteke ime_izlazne_datoteke\n", argv[0]);
        exit(EXIT_FAILURE);
61     }

63     in= fopen(argv[1], "r");
    if(in==NULL)
65     {
        fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
            .\n", argv[1] );
    }
}

```

### 3 Datoteke

```
67     exit(EXIT_FAILURE);
68 }
69
70 out= fopen(argv[2], "w");
71 if(out==NULL)
72 {
73     fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
74     .\n", argv[2] );
75     exit(EXIT_FAILURE);
76 }
77
78 printf("Unesi n:");
79 scanf("%d", &n);
80
81 if(n<=0)
82 {
83     fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
84     exit(EXIT_FAILURE);
85 }
86
87 fscanf(in, "%s", rec);
88
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 3.5

```
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 reciprocnog razlomka svakog razlomka iz niza
      (npr. za 2/3 treba upisati 1.5)
      Mozemo pretpostaviti da se u datoteci sa podacima o razlomcima
      nalazi 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
24  */
25
26  #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
      na koju pokazuje f u niz. Dimenzija niza, na koju
41      pokazuje pokazivac dim, nije poznata. Prva vrednost
      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
      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);
          if (niz[i].im==0)
```

```
59         return 0;
60     }
61     return 1;
62 }
63
64 RAZLOMAK recipročni(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;
80     char c;
81     int i;
82     int j;
83     int xoption=0;
84     int yoption=0;
85     int dim;
86     RAZLOMAK razlomci[MAX];
87     RAZLOMAK r;
88
89     /*
90      Prilikom pokretanja programa se, pored naziva ulazne i izlazne
91      datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
92      obe opcije, sto znaci da je minimalni broj argumenata 3.
93
94      Moguci nacini pokretanja:
95      ./a.out ulaz.txt izlaz.txt -x
96      ./a.out ulaz.txt izlaz.txt -y
97      ./a.out ulaz.txt izlaz.txt -yx
98      ./a.out ulaz.txt izlaz.txt -xy
99
100     */
101
102     if(argc!=4)
103     {
104         fprintf(stderr, "Greska: program se pokrece sa: %s
105             ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n", argv[0]);
106         exit(EXIT_FAILURE);
107     }
108
109     in= fopen(argv[1], "r");
```

```
111     if(in==NULL)
112     {
113         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", argv[1] );
114         exit(EXIT_FAILURE);
115     }

116     out= fopen(argv[2], "w");
117     if(out==NULL)
118     {
119         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", argv[2] );
120         exit(EXIT_FAILURE);
121     }

122     /* Ispitujemo da li je treca opcija u trazenom formatu. Prvi
        karakter mora biti '-' */

123     if (argv[3][0] != '-')
124     {
125         fprintf(stderr, "Greska u zadavanju opcija: program se pokrece
        sa: %s ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n",
        argv[0]);
126         exit(EXIT_FAILURE);
127     }

128     /* Ostali karakteri mogu biti 'x' ili 'y'. U zavisnosti od date
        opcije, postavljamo
        vrednosti indikatorskih promenljivih xoption i yoption. */

129     for(j=1; argv[3][j]!='\0'; j++)
130     {
131         switch(argv[3][j])
132         {
133             case 'x': xoption=1;
134                     break;
135             case 'y': yoption=1;
136                     break;
137             default:
138                 fprintf(stderr, "Greska: nedozvoljeni karakter\n"
139                 );
140                 exit(EXIT_FAILURE);
141         }
142     }

143     if(ucitaj_razlomke(razlomci, &dim, in)==0)
144     {
145         fprintf(stderr, "Greska pri zadavanju razlomaka\n");
146         exit(EXIT_FAILURE);
147     }

148     /*
149     U zavisnosti od datih opcija, vrsimo upis reciprocnih
```

```
155         razlomaka u traženom formatu.

157     Funkcija fprintf se koristi na isti način kao
funkcija printf uz dodatni prvi argument koji
159     označava naziv datoteke u koju se vrši upis.
    */
161     for (i=0; i<dim;i++)
    {
163         /*
            Ukoliko je brojilac razlomka jednak nuli,
165             nema smisla tražiti njegovu recipročnu vrednost
        */
167         if (razlomci[i].br==0)
            continue;

169         r = reciprocni(&razlomci[i]);

171         if (xoption)
            fprintf(out, "%d/%d ", r.br, r.im);

173         if (yoption)
            fprintf(out, "%f ", vrednost(&r));

175         fprintf(out, "\n");
177     }
179 }

181 fclose(in);
fclose(out);
183
185 return 0;
}
```

#### Rešenje 3.6

```
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
    potom
4  i podaci za svaki automobil. Program treba da:
5  a) izračuna prosečnu cenu po marki kola
6  b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
    zadaje
7  kao argument komandne linije, da ispiše automobile u tom cenovnom
    rangi zajedno sa prosečnom cenom odgovarajuće marke
8
9  Mozemo pretpostaviti da se model i marka sastoje od jedne reči i
11 da svaka od njih sadrži najviše 30 karaktera kao i da se u datoteci
    nalaze podaci za najviše 100 automobila.
13
15  */
```

```

#include <stdio.h>
17 #include <stdlib.h>
#include <string.h>
19 #define MAX 31
#define MAXA 100

21 typedef struct automobil
23 {
    char marka[MAX];
25     char model[MAX];
    float cena;
27 } AUTOMOBIL;

29 /*
    Struktura INFO sadrzi naziv
31     marke automobila, prosek cena
    za tu marku i broj automobila
33     te marke
*/
35 typedef struct info
37 {
    char marka[MAX];
    float vrednost;
39     int n;
} INFO;

41 int ucitaj_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 sadrzi 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;
70     for(i=0;i<n;i++)
71         if(strcmp(p[i].marka,m)==0)
72             return i;
73
74     return -1;
75 }

77 /*
78  Funkcija informacije_o_markama za niz automobila a dimenzije n
79  racuna proseke cena automobila po markama i smesta ih u niz
80  p. Na dimenziju niza p pokazuje pokazivac pn.
81
82  Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
83  sumu cena automobila te marke (koju cemo smestiti u polje vrednost
84  strukture
85  INFO), i broj automobila te marke (koju cemo smestiti u polje
86  n strukture INFO) i da na kraju podelimo ove dve promenljive
87  i tako dobijemo prosechnu vrednost cene.
88
89  Za svaki automobil a[i] proveravamo da li se njegova marka vec
90  nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
91  vredost cene automobila a[i] i uvecavamo broj automobila sa
92  tom markom. U suprotnom, dodajemo novi element u niz p. Posto
93  ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
94  na koju pokazuje pokazivac *pn.
95 */
96 void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
97 {
98     int i,j;
99     int ind;
100    for(i=0;i<n;i++)
101    {
102        /* Proveravamo da li se marka automobila a[i] vec nalazi u
103        nizu p (niz proseka po markama) */
104        ind = sadrzi(p,*pn1,a[i].marka);
105        if(ind!=-1) /* Ako se ne nalazi, uvodimo novi element niza na
106        kraj, na poziciju *pn. */
107        {
108            strcpy(p[*pn1].marka, a[i].marka);
109            p[*pn1].vrednost = a[i].cena;
110            p[*pn1].n = 1;
111            (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
112            */
113        }
114        else /* Ako se nalazi, azuriramo polja strukture. */
115        {
116            p[ind].vrednost+=a[i].cena;
117            p[ind].n++;
118        }
119    }
120 }

```

```

117     /* Na osnovu sume cena i broja automobila racunamo prosečnu
118        vrednost. */
119     for(i=0;i<*pn1;i++)
120         p[i].vrednost = p[i].vrednost/p[i].n;
121 }
122
123 void stampaj_informacije(INFO p[], int n)
124 {
125     printf("Informacije o broju automobila i prosečnoj ceni po markama
126            :\n");
127     int i;
128     for(i=0;i<n;i++)
129         printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
130 }
131
132 /*
133 Funkcija stampa automobile cija je cena manja od maksimalne
134 cene koju je korisnik naveo u komandnoj liniji da je spreman
135 da plati, zajedno sa prosečnom cenom za tu marku automobila
136 */
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
138                        n1)
139 {
140     /*
141      S obzirom da je niz p formiran na osnovu niza a, marka svakog
142      automobila iz niza a se sigurno nalazi u nizu p. Zbog toga
143      nije neophodno proveravati da li je povratna vrednost funkcije
144      sadrži različita od -1.
145      */
146     int i;
147     printf("Kola u vašem cenovnom rangu:\n");
148     for(i=0;i<n;i++)
149         if(a[i].cena<g)
150             printf("%s %s %.2f\n", a[i].marka, a[i].model, p[sadrzi(p,n1,
151                             a[i].marka)].vrednost);
152 }
153
154 int main(int argc, char* argv[])
155 {
156     AUTOMOBIL kola[MAXA];
157     FILE* f;
158     char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
159                    ulaza. */
160     float granica; /* Maksimalna cena koju je korisnik spreman da
161                    plati.
162
163                        Zadaje se kao argument komandne linije.
164                    */
165     INFO infos[MAXA];
166     int dim_kola,dim_infos;
167     int i;

```

```
163     if(argc!=2)
165     {
166         fprintf(stderr,"Greska: program se pokrece sa: %s
167         gornja_granica_cene \n", argv[0]);
168         exit(EXIT_FAILURE);
169     }
170
171     /* Argumenti komandne linije su stringovi. Da bismo od stringa
172     dobili
173     realan broj, koristimo ugradjenu funkciju atof. */
174     granica = atof(argv[1]);
175
176     printf("Unesi naziv datoteke:");
177     scanf("%s", dat);
178
179     f=fopen(dat, "r");
180
181     if(f==NULL)
182     {
183         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
184         .\n", dat);
185         exit(EXIT_FAILURE);
186     }
187
188     if (ucitaj_podatke(f,kola,&dim_kola,MAXA)==0)
189     {
190         fprintf(stderr, "Greska pri učitavanju podataka\n");
191         exit(EXIT_FAILURE);
192     }
193
194     informacije_o_markama(kola, dim_kola, infos, &dim_infos);
195
196     stampaj_informacije(infos,dim_infos);
197
198     stampaj_kandidate(kola, dim_kola, granica, infos, dim_infos);
199
200     fclose(f);
201     return 0;
202 }
```



# 4

## Razni zadaci

### 4.1 Rešenja



## Dodatak A

# Ispitni rokovi

### A.1 Ispitni rokovi MNVRLA

#### A.1.1 Kvalifikacioni zadaci

#### A.1.2 Praktični deo ispita, jun ...

### A.2 Ispitni rokovi I smer

### A.3 Rešenja