

## 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>Uvodni zadaci</b>	<b>1</b>
1.1	Rešenja . . . . .	8
<b>2</b>	<b>Kontrola toka</b>	<b>19</b>
2.1	Naredbe grananja . . . . .	19
2.2	Rešenja . . . . .	28
2.3	Petlje . . . . .	52
2.4	Rešenja . . . . .	71
2.5	Funkcije . . . . .	88
2.6	Rešenja . . . . .	97
<b>3</b>	<b>Predstavljanje podataka</b>	<b>115</b>
3.1	Nizovi . . . . .	115
3.2	Rešenja . . . . .	127
3.3	Pokazivači . . . . .	154
3.4	Rešenja . . . . .	160
3.5	Niske . . . . .	173
3.6	Rešenja . . . . .	183
3.7	Višedimenzioni nizovi . . . . .	200
3.8	Rešenja . . . . .	207
3.9	Strukture . . . . .	207
3.10	Rešenja . . . . .	213
<b>4</b>	<b>Ulaz i izlaz programa</b>	<b>233</b>
4.1	Standardni tokovi . . . . .	233
4.2	Argumenti komandne linije . . . . .	233
4.3	Datoteke . . . . .	233
4.4	Rešenja . . . . .	247

<b>5</b>	<b>Razni zadaci</b>	<b>265</b>
5.1	Rešenja	265
<b>A</b>	<b>Ispitni zadaci</b>	<b>267</b>
A.1	Testovi/Kolokvijumi	267
A.1.1	Programiranje 1, i-smer, kolokvijum	267
A.2	Kvalifikacioni zadaci	271
A.3	Ispitni rokovi	271
A.3.1	Programiranje 1, i-smer, Završni ispit, januar, 23.01.2016.	271
A.3.2	Programiranje 1, i-smer, Završni ispit, februar, 11.02.2016.	276
A.3.3	1. Grupa, I smer, Programiranje 1 2015/2016, ispit, jun	278
A.3.4	Praktični deo ispita, jun ...	280
A.4	Rešenja	280

# Predgovor

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

*Autori*





# 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](#)]

## 1 Uvodni zadaci

---

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

## 1 Uvodni zadaci

---

### 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 prirodnom 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 celzijusima: 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]

**Zadatak 1.22** Sa standardnog ulaza se učitavaju dve realne promenljive. Razmeniti vrednosti promenljivima i nove vrednosti ispisati na standardni izlaz.

[Rešenje 1.22]

**Zadatak 1.23** Unose se koordinate suprotnih temena pravougaonika (gornje levo i donje desno teme). Pretpostaviti da su stranice pravougaonika paralelene koordinatnim osama. Odrediti obim i površinu pravougaonika.

[Rešenje 1.23]

**Zadatak 1.24** Date su dve celobrojene promenljive a i b. Promenljivoj a dodeliti njihovu sumu, a promenljivoj b njihovu razliku bez korišćenja pomoćne promenljive.

[Rešenje 1.24]

## 1 Uvodni zadaci

---

**Zadatak 1.25** Napisati program koji na standardni izlaz ispisuje sledeći tekst:

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Karakteri : % { * + = a  
|| Brojevi: 43, -56, 455
```

[Rešenje 1.25]

**Zadatak 1.26** Napisati program koji na mesto stotina i hiljada umeće cifre c1 i c2. Da li se može desiti da za neke ulazne podatke dodje do prekoračenja? Obrazložiti.

[Rešenje 1.26]

**Zadatak 1.27** Sa standardnog ulaza se unose dva cela broja. Na standardni izlaz ispisati maksimum ova dva broja.

[Rešenje 1.27]

**Zadatak 1.28** Data su 3 cela broja a, b, c. Dodeliti promenljivoj rez vrednost 1 ako:

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

U suprotnom promenljivoj dodeliti vrednost 0. Proveriti ispisom na standardni izlaz.

[Rešenje 1.28]

**Zadatak 1.29** Program treba da proveri da li se tačke  $A(x_1, y_1)$  i  $B(x_2, y_2)$  nalaze u istom kvadrantu. Na standardni izlaz ispisati odgovor DA ili NE.

[Rešenje 1.29]

**Zadatak 1.30** Program treba da proveri da li se tačke  $A(x_1, y_1)$ ,  $B(x_2, y_2)$  i  $C(x_3, y_3)$  nalaze na istoj pravci. Na standardni izlaz ispisati odgovor DA ili NE.

[Rešenje 1.30]

**Zadatak 1.31** Polje šahovske table se definiše parom prirodnih brojeva ne većih od 8: prvi se odnosi na red, drugi na kolonu. Ako su dati takvi parovi, napisati program koji proverava:

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

[Rešenje 1.31]

**Zadatak 1.32** Sa standardnog ulaza unose se dve promeljive  $x$  i  $y$ . Izračunati vrednost izraza:

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

Rezultat ispisati na standardni izlaz.

[Rešenje 1.32]

**Zadatak 1.33** Tekst

[Rešenje 1.33]

**Zadatak 1.34** Napisati program koji za unete brojeve a11, a12, a21, a22 tipa float izračunava i ispisuje na standardni izlaz determinantu matrice:

a11 a12  
a21 a22

Pri ispisu vrednosti se zaokružuju na 4 decimale.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 1.34]

### 1.1 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
   return 0;
23 }
```

#### 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()
```



```

{
17     int x;
    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 */

41

43     /* 4) ispis proizvoda */
    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
        moze sadrzati realan broj */
49                                                /* ukoliko
        je x=7, a y=2, tada ce nakon naredbe */
                                                /*
        rezultat=x/y; promenljiva rezultat imati vrednost 2 */
51                                                /* a ne
        2.5 */

53     printf("ostatak pri celobrojnem deljenju: %d %% %d=%d\n",x,y,x%y);
                                                /*
        operator % izracunava ostatak pri celobrojnem deljenju */
55                                                /* 7%2 ima
        vrednost 1 (jer je 7=3*2+1) */

```

## 1 Uvodni zadaci

---

```
57     % u naredbi printf pisemo %% */
    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
3   i na standardni izlaz ispisuje njegov obim i povrsinu
*/
5
7 #include <stdio.h>
   #include <math.h> /* biblioteka matematickih funkcija; za prevodjenje
   je neophodno ukljuciti opciju -lm
   npr. gcc primer.c -lm */
9 int main()
{
11     int r;
    float O;
```

```
13 float P;
14 printf("Unesi poluprecnik kruga:");
15 scanf("%d", &r);

17 O=2*r*M_PI; /* M_PI - konstanta pi koja se nalazi u biblioteci math
    .h */
18 P=r*r*M_PI;

19 printf("Obim: %f, površina: %f\n",O,P);

21 return 0;
23 }
```

### Rešenje 1.5

```
1 /*
2  Napisati program koji učitava trocifreni broj koji se
3  unosi sa standardnog ulaza i ispisuje njegove cifre na
4  standardni izlaz.
5  */
6  #include <stdio.h>
7  int main()
8  {
9      int x;
10     int cifra_jedinice;
11     int cifra_desetice;
12     int cifra_stotine;

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

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

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

20     /*
21      2. nacin, bez uvođenja dodatnih promenljivih cifra_jedinice,
22      cifra_desetice i cifra_stotine:
23      */

24     printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10, x/100)
25     ;
26     /*
27     return 0;
28     */
29 }
```

### Rešenje 1.6

## 1 Uvodni zadaci

---

```
1  /*
2  Napisati program koji ucitava trocifreni broj koji se
3  unosi sa standardnog ulaza i ispisuje broj dobijen obrtanjem
4  njegovih cifara.
5  */
6  #include <stdio.h>
7  int main()
8  {
9      int x;
10     int obrnuto_x;
11
12     int cifra_jedinice;
13     int cifra_desetice;
14     int cifra_stotine;
15
16     printf("Unesi trocifreni broj:");
17     scanf("%d", &x);
18
19     cifra_jedinice = x%10;
20     cifra_desetice = (x/10)%10;
21     cifra_stotine = x/100;
22
23     obrnuto_x = cifra_jedinice*100 + cifra_desetice*10 + cifra_stotine
24     ;
25
26     printf("Obrnuto x: %d\n", obrnuto_x);
27
28     return 0;
29 }
```

### Rešenje 1.7

```
1  /*
2  Napisati program koji za unetu duzinu stranice jednakostranicnog
3  trougla
4  ispisuje njegovu povrstinu.
5  */
6  #include <stdio.h>
7  #include <math.h>
8  int main()
9  {
10     unsigned int a;
11     float P;
12
13     printf("Unesi duzinu stranice jednakostranicnog trougla:");
14     scanf("%d",&a);
15
16     P = (a*a*sqrt(3))/4;
```

```
18     printf("Povrsina jednakostraniceg 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
4   raspolaganju su novcanice od 1000,100,50,10 i 1 dinar. Na primer,
   za unetu cenu 5178, program na standardni izlaz treba da ispise:
6   5178=5*1000+ 1*100 +1*50 +2*10 +8*1
*/
8
#include <stdio.h>
10
int main()
12 {
    int x;
14     printf("Unesi cenu:");
    scanf("%d", &x);
16
    printf("%d=%d*1000+ ", x,x/1000);
18     x=x%1000;
    printf("%d*100 +", x/100);
20     x=x%100;
    printf("%d*50 +",x/50);
22     x=x%50;
    printf("%d*10 +", x/10);
24     x=x%10;
    printf("%d*1\n", x);
26     return 0;
}
```

### Rešenje 1.9

```
1  /*
   Napisati program koji za tri cela broja koja se unose sa standardnog
   ulaza
3  ispisuje njihovu aritmeticku sredinu na standardni izlaz.
   */
5
#include<stdio.h>
7
int main()
9 {
    int a, b, c;
```

## 1 Uvodni zadaci

---

```
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 [1.22](#)

Rešenje [1.23](#)

Rešenje [1.24](#)

Rešenje [1.25](#)

Rešenje [1.26](#)

Rešenje [1.27](#)

Rešenje [1.28](#)

Rešenje [1.29](#)

Rešenje [1.30](#)

### Rešenje 1.31

### Rešenje 1.32

### Rešenje 1.33

```
1 #include <stdio.h>
2 #include <math.h>
3 #include <limits.h>
4
5 /* u zaglavlju limits.h
6 su definisane maksimalne i minimalne
7 vrednosti za svaki tip podataka
8 npr. INT_MAX konstanta je najveći ceo
9 broj koji može da se stavi
10 u promenljivu tipa int
11 zbog toga za poslednji test primer
12 ne dobijamo željeni broj
13 jer je došlo do prekoračenja
14 novibroj je veći od INT_MAX
15 */
16
17 /* test primeri:
18 broj: 140
19 c1: 2
20 c2: 3
21
22 novibroj: 13240
23 -----
24 broj: 526
25 c1: 7
26 c2: 4
27
28 novibroj: 54726
29 -----
30 broj: 25
31 c1: 9
32 c2: 5
33
34 novibroj: 5925
35 -----
36 test primer koji dovodi do prekoračenja, pa zbog toga
37 ne dobijamo željeni rezultat:
38
39 broj: 100000000
40 c1: 5
41 c2: 1
42
43 novibroj: neočekivan rezultat ---> PREKORACENJE
```



```
45 */
47 int main(){
48     int broj,c1,c2,z1,z2;
49     int novibroj;
50     int dostatak1, dostatak2 ;
51     printf("unesi broj: ");
52     scanf("%d", &broj);
53     printf("unesi c1: ");
54     scanf("%d", &c1);
55     printf("unesi c2: ");
56     scanf("%d", &c2);
57
58     /* najbolje odmah da se kastuje z1 jer se kasnije cesto
59     koristi u racunu pa da ne ponavljamo (int) */
60     // za stotine pozicija je 3 ---> z1 = (int)pow(10,3-1);
61     z1 = (int)pow(10,2);
62
63     dostatak1 = broj % z1;
64
65     /*
66     levi ostatak je u stvari ovaj deo --> broj / z1 * z1 * 10
67     inace taj deo moze da se racuna i kao --> (broj - broj % z1) * 10
68     */
69     novibroj = broj / z1 * z1 * 10 + z1 * c1 + dostatak1 ;
70
71     //sada u novibroj insertujemo cifru c2 na poziciju 4 - za hiljade
72
73     z2 = (int)pow(10,3);
74
75     dostatak2 = novibroj % z2;
76
77     /*
78     levi ostatak je u stvari ovaj deo --> broj / z2 * z2 * 10
79     inace taj deo moze da se racuna i kao --> (broj - broj % z2) * 10
80     */
81     novibroj = novibroj / z2 * z2 * 10 + z2 * c2 + dostatak2 ;
82
83     printf("Novi broj je: %d\n", novibroj);
84     printf("Maksimalna vrednost za int je: %d\n", INT_MAX);
85
86     return 0;
87 }
```

## Rešenje 1.34



## 2

# Kontrola toka

## 2.1 Naredbe grananja

**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](#)]

**Zadatak 2.8** Tekst

[Rešenje [2.8](#)]

**Zadatak 2.9** Tekst

[Rešenje [2.9](#)]

**Zadatak 2.10** Tekst

[Rešenje [2.10](#)]

**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** 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!
```

[Rešenje [3.108](#)]

**Zadatak 2.16** 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!
```

[Rešenje 3.108]

**Zadatak 2.17** 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.
```

[Rešenje 3.108]

**Zadatak 2.18** 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!
```

[Rešenje 3.108]

**Zadatak 2.19** 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
```

[Rešenje 3.108]

**Zadatak 2.20** 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: >  
|| *>*
```

[Rešenje 3.108]

**Zadatak 2.21** 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
```

[Rešenje 3.108]

**Zadatak 2.22** 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!
```

[Rešenje 3.108]

**Zadatak 2.23** 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!
```

[Rešenje 3.108]

**Zadatak 2.24** 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.
```

[Rešenje 3.108]

**Zadatak 2.25** 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]
```

[Rešenje 3.108]

**Zadatak 2.26** 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
```

[Rešenje 3.108]

**Zadatak 2.27** 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!
```

[Rešenje 3.108]

**Zadatak 2.28** 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 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!
```



[Rešenje 3.108]

**Zadatak 2.29** Napisati program koji za uneti datum u formatu *dan.mesec.godina.* proverava da li je korektan.

*Primer 1*

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

*Primer 2*

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

[Rešenje 3.108]

**Zadatak 2.30** Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum prethodnog dana.

*Primer 1*

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

*Primer 2*

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

[Rešenje 3.108]

**Zadatak 2.31** 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.
```

[Rešenje 3.108]

**Zadatak 2.32** Napisati program kojim se sabiraju samo pozitivne vrednosti promenljivih a, b, c.

[Rešenje 2.82]

**Zadatak 2.33** Sa standarnog ulaza unosi se jedan karakter. Ako je karakter malo slovo zameniti ga velikim slovom, ako je veliko slovo zameniti malim slovom, ako je cifra ispisati u pitanju je cifra. Ako je bilo koji drugi karakter onda ga ispisati na standardni izlaz.

[Rešenje 2.155]

## 2 Kontrola toka

---

**Zadatak 2.34** Sa standardnog ulaza se unosi četvorocifren ceo broj. Napisati program koji datom broju razmenjuje najmanju i najveću cifru. Dobijeni broj ispisati na izlaz. Ako broj nije četvorocifren ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 3842  
|| 3248
```

*Primer 2*

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

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 123  
|| -1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -45678  
|| -1
```

[Rešenje 2.84]

**Zadatak 2.35** Sa standardnog ulaza se unosi 5 karaktera. Ispisati na izlazu koliko se puta pojavilo veliko ili malo slovo a.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.85]

**Zadatak 2.36** Sa standardnog ulaza se unose 5 karaktera. Ispisati na izlazu koliko puta su se pojavile cifre.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

*Primer 4*

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

[Rešenje 2.116]

**Zadatak 2.37** 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 -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 384 123 245  
|| 384245
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 123 345 5  
|| -1
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 1242 234 324  
|| -1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 374 23 898  
|| -1
```

[Rešenje 2.37]

**Zadatak 2.38** Korisnik unosi 3 cela broja: (p), (q) i (r). Nakon toga unosi i dva karaktera, koji imaju sledeci smisao:

'k' -logička konjukcija

'd' -logička disjunkcija

'm' -relacija manje

'v' -relacija veće

Nakon toga se računa vrednost izraza (p) op1 (q) op2 (r) i ispisuje rezultat.

[Rešenje 2.38]

**Zadatak 2.39** Tekst

[Rešenje 2.39]

**Zadatak 2.40** Tekst

[Rešenje 2.40]

**Zadatak 2.41** Tekst

[Rešenje 2.41]

### Zadatak 2.42 Tekst

[Rešenje 2.42]

## 2.2 Rešenja

### Rešenje 2.1

```
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 2.2

```
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  */
6  #include <stdio.h>
7
8  int main()
9  {
10     int x;
11     float rx;
12
13     printf("Unesi jedan ceo broj:");
14     scanf("%d",&x);
15
16     /*
17      obratiti paznju:
18      x==0 - relacija jednakosti (da li je vrednost promenljive x
19      jednaka nuli)
20      x=0 - naredba dodele (promenljiva x dobija vrednost nula)
21     */
22
23     if (x==0)
24         printf("Nedozvoljeno deljenje nulom\n");
25     else
26     {
27         rx = 1.0/x;
28         printf("Recipročna vrednost unetog broja:%f\n",rx);
29     }
30
31     return 0;
32 }
```

### Rešenje 2.3

```
1  #include <stdio.h>
2  /*
3  Napisati program koji za uneti ceo broj x ispisuje da li je jednak
4  nuli,
5  manji od nule ili veci od nule.
6  */
7  int main()
8  {
9     int x;
10     printf("Unesi ceo broj:");
11     scanf("%d",&x);
12
13     /*
14      obratiti paznju:
15      x==0 - relacija jednakosti (da li je vrednost promenljive x
16      jednaka nuli)
17      x=0 - naredba dodele (promenljiva x dobija vrednost nula)
18     */
19     if (x==0)
```

## 2 Kontrola toka

---

```
    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 2.4

```
1  /*
   * Napisati program koji za godinu koja se unosi sa standardnog ulaza
   * na standardni izlaz
   * 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 2.5

```
1  /*
   * Napisati program koji za 2 cela broja uneta sa standardnog ulaza
   * ispisuje njihov minimum na standardni izlaz.
   */
3
5
   #include <stdio.h>
7
   int main()
   {
9       int a,b;
       int min1;
11      int min2;
       int min3;
13
       scanf("%d%d",&a,&b);
15
```

```

17  /* 1. nacin */
18  if (a<b)
19      min1=a;
20  else
21      min1=b;
22
23  printf("Minimum unetih brojeva (1.nacin) je %d\n",min1);
24
25  /* 2. nacin */
26  min2 = (a<b) ? a : b;
27  printf("Minimum unetih brojeva (2.nacin) je %d\n",min2);
28
29  /* 3. nacin */
30  min3=a;
31  if (b<a)
32      min3 = b;
33  printf("Minimum unetih brojeva (3.nacin) je %d\n",min3);
34
35  return 0;
36 }

```

## Rešenje 2.6

```

/*
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
19     scanf("%d%d%d",&a,&b,&c);
20
21     if (a<b)
22     {
23         if (a<c) /* poredak: a<b,a<c => a,b,c ili a,c,b */
24             min=a;
25         else /* poredak: a<b, a>=c => a<b, c<=a => c,a,b */
26             min=c;
27     }
28     else if (b<c)
29     {
30         min=b;
31     }
32     else
33     {
34         min=c;
35     }
36
37     cena_bez_popusta = min*a + min*b + min*c;
38     cena_sa_popustom = cena_bez_popusta - min;
39
40     printf("Minimum cena je %d\n",min);
41     printf("Ukupna cena bez popusta je %d\n",cena_bez_popusta);
42     printf("Ukupna cena sa popustom je %d\n",cena_sa_popustom);
43
44     return 0;
45 }

```

```
26     else          /* b<=a */
27         if (b<c) /* poredak: b<=a,b<c => b,a,c ili b,c,a */
28             min=b;
29         else      /* poredak: b<=a, c<=b => c,b,a */
30             min=c;

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

34     /* 2. nacin */
35     /* najpre odredimo minimum brojeva a,b*/
36     if (a<b)
37         min1=a;
38     else
39         min1=b;
40
41     if (c<min1)
42         min1=c;
43     printf("Minimum unetih brojeva (2.nacin) je %d\n",min1);
44
45     /* 3. nacin */
46     min2=a;
47     if(min2>b)
48         min2=b;
49     if(min2>c)
50         min2=c;
51
52     printf("Minimum unetih brojeva (3.nacin) je %d\n",min2);

54     cena_bez_popusta=a+b+c;
55     cena_sa_popustom = cena_bez_popusta - min2 + 1;

56
57     printf("Cena sa popustom: %.2f\n Cena bez popusta: %d\n Usteda:
58           %.2f\n", cena_sa_popustom, cena_bez_popusta, cena_bez_popusta-
59           cena_sa_popustom);
60
61     return 0;
62 }
```

### Rešenje 2.7

```
2  /*
3  Napisati program koji za koeficijente kvadratne jednacine
4  koji se unose sa standardnog ulaza na standardni izlaz
5  ispisuje koliko realnih resenja jednacina ima i ako ih ima, ispisuje
6  resenja jednacine
7  zaokruzena na dve decimalne.
8  */
9  #include <stdio.h>
10 #include <math.h>
11 int main()
12 {
```



```

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 /* proveravamo da li je kvadratna jednacina korektno zadata */
18 if (a==0)
19     if (b==0)
20         if(c==0) /* slucaj a==0 && b==0 && c==0 */
21             printf("Jednacina ima beskonacno mnogo resenja\n");
22         else /* slucaj a==0 && b==0 && c!=0 */
23             printf("Jednacina nema resenja\n");
24     else /* slucaj a==0 && b!=0 */
25     {
26         x1=-c/b;
27         printf("Jednacina ima jedinstveno realno resenje %.2f\n",x1)
28     ;
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 2.8

```

1 /*
2  Napisati program koji za karakter koji ucitava jedan karakter i :
3  - u slucaju da je uneta cifra, ispisuje nju i njen ascii kod
4  - u slucaju da je uneto malo slovo, ispisuje njega, njegov ascii kod,
5    odgovarajuće veliko slovo i njegov ascii kod

```

## 2 Kontrola toka

```

- 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>
10 int main()
11 {
12     char c;
13     printf("Unesi jedan karakter:");
14     scanf("%c", &c);
15
16     if (c>='0' && c<='9')
17         printf("cifra:%c ascii:%d\n",c,c);
18     else if (c>='A' && c<='Z')
19         printf("veliko slovo:%c ascii:%d odgovarajuće malo:%c, ascii:%d
20 \n",c,c,c-'A'+ 'a',c-'A'+ 'a'); /* Razlika izmedju ascii koda
21 svakog malog i odgovarajućeg velikog slova
22
23                                     je konstanta koja se može
24 sračunati izrazom 'a'-'A' (i iznosi 32) */
25     else if (c>='a' && c<='z')
26         printf("malo slovo:%c ascii:%d odgovarajuće veliko:%c, ascii:%d
27 \n",c,c,c-'a'+ 'A',c-'a'+ 'A');
28     else
29         printf("karakter:%c ascii:%d\n",c,c);
30
31     return 0;
32 }
```

### Rešenje 2.9

```

1 /*
2
3 Napisati program koji učitava tri cela broja i ispisuje zbir onih
  unetih brojeva
  koji su pozitivni.
4
5 */
6 #include<stdio.h>
7 int main()
8 {
9     int a,b,c;
10    int s;
11    printf("Unesi prvi ceo broj:");
12    scanf("%d",&a);
13    printf("Unesi drugi ceo broj:");
14    scanf("%d",&b);
15    printf("Unesi treci ceo broj:");
16    scanf("%d",&c);
17
18    s=0; /* inicijalizujemo promenljivu s na nulu */
19 }
```

```

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);
37  return 0;
}

```

### Rešenje 2.10

```

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);
26
27     /* 2. nacin */
28     y=x;

```

## 2 Kontrola toka

---

```
29     if (y<0)
        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 racunanje 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 2.11

```
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()
{
12     char c;
    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");
                    break;
36         default : printf("Niste uneli samoglasnik\n");
                    break;
    }

    return 0;
```

32 }

## Rešenje 2.12

```
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
46                 printf("jesen\n");
47             break;
```

```
47     case 10:
48     case 11:
49         printf("jesen\n");
50         break;
51     case 12:
52         if (d<22)
53             printf("jesen\n");
54         else
55             printf("zima\n");
56     }
57     return 0;
58 }
```

### Rešenje 2.13

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

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

11
12 ili ovako
13
14 Unesi jedan cetvorocifreni broj: 9952
15 Cifre su mu uredjene nerastuce.

16
17 ili ovako
18
19 Unesi jedan cetvorocifreni broj: 9572
20 Cifre su mu nisu uredjene.

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

24
25 */

26
27 #include <stdio.h>
28 #include <stdlib.h>

29
30
31 int main()
32 {
33     int x;
34     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:");
scanf("%d", &x);

41

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

45

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

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

57     if (c1000<=c100 && c100<=c10 && c10<=c1)
        printf("Cifre su uredjene neopadajuće \n");
59     else if (c1000>=c100 && c100>=c10 && c10>=c1)
        printf("Cifre su uredjene nerastuće \n");
61     else
        printf("Cifre nisu uredjene\n");
63 }
return 0;
65 }

```

### Rešenje 2.14

```

1  /*
   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
   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
7  unosi redom A,B,C,D, pri cemu ABCD opisuje pravougaonik cije su
   stranice AB,BC,CD i DA, a dijagonale AC i BD
   na primer, tacke (1,1),(2,1),(2,2),(1,2) cine pravougaonik cije
   su stranice paralelne koordinatnim osama i ciji je obim 4
9  a tacke (1,1),(2,2),(3,3),(4,4) ne cine pravougaonik
   h - proverava da li su unete tacke kolinearne i ukoliko jesu,
   ispisati jednacinu prave kojoj pripadaju

```

```

11     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
*/
23
#include<stdio.h>
25 #include<math.h>
int main()
27 {
    char c;
29     float x1,y1,x2,y2,x3,y3,x4,y4;
    float kab,kbc,kad;
31     float dab,dad;
    float delta, deltap, deltaq;
33     float 0;
    float k,n;
35
    printf("Unesi jedan karakter:");
37     scanf("%c",&c);

    printf("Unesi realne koordinate 4 tacke:");
39     scanf("%f%f%f%f%f%f%f",&x1,&y1,&x2,&y2,&x3,&y3,&x4,&y4);
41
    switch (c)
43     {
        case 'k':
45         if (y1==y2 && y3==y4 && x1==x4 && x2==x3)
            {
47                 dab = sqrt(pow(x1-x2,2)+pow(y1-y2,2)); // funkcija pow(x
,y) racuna vrednost stepene funkcije  $x^y$ 
                dad = sqrt(pow(x1-x4,2)+pow(y1-y4,2)); // x i y su
                realne vrednosti
49                 0 = 2*dab + 2*dad;
                printf("Obim pravougaonika je %f\n",0);
51             }
            else
53                 printf("Tacke ne cine pravougaonik sa stranicama koje su
                paralelne koordinatnim osama\n");

```



```

55     break;
56     case 'h':
57         if ((x1-x2)!=0) // ukoliko se tacke A(x1,y1) i B(x2,y2) ne
            nalaze na pravoj koja je paralelna x osi
            {
58             k = (y1-y2)/(x1-x2); //izracunamo k,n za pravu odredjenu
            tackama A(x1,y1) i B(x2,y2)
59             n = y1-k*x1;

60             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);
62             else
63                 printf("Tacke nisu kolinearne\n");
64             }
65             else // ukoliko se A i B nalaze na pravoj koja je paralelna
            x osi
66                 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
67             ",x1);
68             else
69                 printf("Tacke nisu kolinearne\n");
70             break;
71     case 'j':
72         delta = x1*y2-x2*y1;
73         deltap = x2*(y4-y3)-y2*(x4-x3);
74         deltaq = x1*(y4-y3)-y1*(x4-x3);
75         if (delta!=0)
76             printf("Sistem ima jedinstveno resenje %.2f, %.2f\n",
            deltap/delta, deltaq/delta);
77         else if (deltap==0 && deltaq==0)
78             printf("Sistem je neodredjen ili nema resenja.\n");
79         else
80             printf("Sistem nema resenja\n");
81         break;
82     default:
83         printf("Nekorektan unos\n");
84 }
85 return 0;
86 }

```

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [2.82](#)

```
1 #include <stdio.h>
3 int main()
4 {
5     int broj;
6     scanf("%d", &broj);
7
8     /* Uzimamo apsolutnu vrednost broja,
9        jer nas interesuju cifre, ne i znak.
```

```

11     Racunamo je tako sto broj negiramo
    ako je manji od nule, a ako nije
13     ne menjamo ga. */
    broj = (broj < 0) ? -broj : broj;

15
    /* Ako broj nije cetvorocifren,
    * završavamo program */
17    if (broj <= 999 || broj >= 10000)
19    {
        printf("-1");
21        /* return u main funkciji završava program,
        bilo koja vrednost koja se vrati a da
23        nije 0, predstavlja gresku. Uobicajeno je
        da to bude -1, ali videcemo da postoji i
25        "prenosiviji" nacin da ovo uradimo */
        return -1;
27    }

29    /* Izdvajamo cifre broja */
    int d = broj % 10;
31    int c = (broj / 10) % 10;
    int b = (broj / 100) % 10;
33    int a = broj / 1000;

35    /* Pretpostavljamo da je najveća cifra a */
    int max = a;

37
    /* Ako je b veća od a, onda je b maksimum */
39    if (b > max)
        max = b;

41    /* Ako je c veća od trenutnog maksimuma, bilo
    da je on a bilo da je on b, onda je c maksimum */
43    if (c > max)
        max = c;

45    /* Slicno za d */
    if (d > max)
47        max = d;

49    /* Stampamo rezultat */
    printf("%d\n", max);

51    return 0;
53 }

```

### Rešenje 2.155

### Rešenje 2.84

```

1 #include <stdio.h>

```

```
3 int main()
4 {
5     int broj;
6     scanf("%d", &broj);
7
8     // Da bismo lakse odredili da li je cetvorocifren
9     int absBroj = broj < 0 ? -broj : broj;
10    if ( absBroj <= 999 || absBroj >= 10000)
11    {
12        printf("-1");
13        return -1;
14    }
15
16    int a = absBroj % 10;
17    int b = (absBroj / 10) % 10;
18    int c = (absBroj / 100) % 10;
19    int d = absBroj / 1000;
20
21    int max = a, min = a;
22    // cuvamo i stepen da bismo lakse zamenili cifre
23    /* Ideja:
24       4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1,
25       ^ ^   odnosno oduzemo 100 i dodamo 900 */
26    int stepenMax = 1, stepenMin = 1;
27
28    if (b > max)
29    {
30        max = b;
31        stepenMax = 10;
32    }
33    if (b < min)
34    {
35        min = b;
36        stepenMin = 10;
37    }
38
39    if (c > max)
40    {
41        max = c;
42        stepenMax = 100;
43    }
44    if (c < min)
45    {
46        min = c;
47        stepenMin = 100;
48    }
49
50    if (d > max)
51    {
52        max = d;
53        stepenMax = 1000;
54    }
55 }
```

```

55     if (d < min)
56     {
57         min = d;
58         stepenMin = 1000;
59     }

61

62     int rez;
63     /* Ideja:
64        4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1,
65        ^ ^   odnosno oduzemo 100 i dodamo 900 */

66     if (broj > 0)
67         rez = broj - max*stepenMax + min*stepenMax
68                 - min*stepenMin + max*stepenMin;
69     else
70         rez = broj + max*stepenMax - min*stepenMax
71                 + min*stepenMin - max*stepenMin;
72

73     printf("%d\n", rez);
74     return 0;
75 }
76
77

```

### Rešenje 2.85

```

1  #include <stdio.h>
2  #include <ctype.h>
3
4  int main()
5  {
6      int br_a = 0;
7      if (tolower(getchar()) == 'a')
8          br_a++;
9      if (tolower(getchar()) == 'a')
10         br_a++;
11     if (tolower(getchar()) == 'a')
12         br_a++;
13     if (tolower(getchar()) == 'a')
14         br_a++;
15     if (tolower(getchar()) == 'a')
16         br_a++;
17
18     printf("%d\n", br_a);
19
20     return 0;
21 }

```

### Rešenje 2.116

```
1 #include <stdio.h>
  #include <ctype.h>
3
4 int main()
5 {
6     int br_cif = 0;
7     if (isdigit(getchar()))
8         br_cif++;
9     if (isdigit(getchar()))
10        br_cif++;
11    if (isdigit(getchar()))
12        br_cif++;
13    if (isdigit(getchar()))
14        br_cif++;
15    if (isdigit(getchar()))
16        br_cif++;
17
18    printf("%d\n", br_cif);
19
20    return 0;
21 }
```

### Rešenje 2.37

```
1 #include <stdio.h>
3 int main()
4 {
5     int br1, br2, br3;
6     scanf("%d%d%d", &br1, &br2, &br3);
7
8     if (br1 > 999 || br1 < 100 || br2 > 999 || br2 < 100
9         || br3 > 999 || br3 < 100)
10    {
11        printf("-1");
12        return -1;
13    }
14
15    int max1 = br1;
16    if (br2 > max1)
17        max1 = br2;
18    if (br3 > max1)
19        max1 = br3;
20
21    /* Ako je br1 vec najveci, onda pretragu
22       za sledecim najvecim krecemo od br2 */
23    int max2 = br1 != max1 ? br1 : br2;
24    if (br1 > max2 && br1 != max1)
25        max2 = br1;
26    if (br2 > max2 && br2 != max1)
```

```

27     max2 = br2;
    if (br3 > max2 && br3 != max1)
29     max2 = br3;

31     int rez = max1*1000 + max2;
    printf("%d\n",rez);
33
    return 0;
35
}
```

### Rešenje 2.38

### Rešenje 2.39

```

#include <stdio.h>
2 #include <ctype.h> // !!!

4 // Upotreba funkcija isalpha, isdigit, toupper, tolower

6 // isalpha( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
  je karakter slovo (malo ili veliko), inace 0
// isdigit( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
  je karakter cifra, inace 0
8 // isupper( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
  je karakter veliko slovo, inace 0
// islower( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
  je karakter malo slovo, inace 0
10 // toupper( karakter ) - ukoliko je karakter malo slovo, funkcija
  vraća odgovarajuće veliko slovo,
  //                               inace vraća isti karakter
12 // tolower( karakter ) - ukoliko je karakter veliko slovo, funkcija
  vraća odgovarajuće malo slovo,
  //                               inace vraća isti karakter

14 int main()
16 {
    char c;
18     char veliko_slovo;
    char malo_slovo;

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

24     if(isalpha(c))
    {
26         printf("Karakter %c je slovo\n",c);

28     if(isupper(c))
        printf("Veliko slovo\n");
    }
```

```
30  else
    printf("Malo slovo\n");
32
    veliko_slovo = toupper(c); // malo -> veliko slovo
34    malo_slovo = tolower(c);  // veliko -> malo slovo
36
    printf("Veliko slovo: %c, malo slovo: %c\n", veliko_slovo,
    malo_slovo);
38
    }
    else if(isdigit(c))
40        printf("Karakter %c je cifra\n",c);
    else
42        printf("Karakter %c je znak\n",c);
44
46    printf("=====Bez koriscenja funkcija=====\\n");
48
    // Isti rezultat bez koriscenja ugradjenih funkcija
50    if((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z'))
    {
52        printf("Karakter %c je slovo\n",c);
54    if(c >= 'A' && c <= 'Z')
        printf("Veliko slovo\n");
56    else
        printf("Malo slovo\n");
58
        if(c >= 'a' && c <= 'z')
60    {
        veliko_slovo = c - ('a' - 'A');
62        malo_slovo = c;
        }
        else if(c >= 'A' && c <= 'Z')
64    {
        malo_slovo = c + ('a' - 'A');
66        veliko_slovo = c;
68    }
70
        printf("Veliko slovo: %c, malo slovo: %c\n", veliko_slovo,
        malo_slovo);
72    }
    else if(c >= '0' && c <= '9')
74        printf("Karakter %c je cifra\n",c);
    else
76        printf("Karakter %c je znak\n",c);
78
    return 0;
```



80 }

## Rešenje 2.40

```
#include <stdio.h>

2 // Za uneti redni broj dana u nedelji ispisati njegov naziv
4
6 int main()
{
8     int broj_dana;

    printf("Unesite broj dana: ");
10     scanf("%d",&broj_dana);

12     switch(broj_dana)
    {
14         case 1: printf("Dan je ponedeljak\n");
                    break; // Obavezan izlazak iz case-a!
16         case 2: printf("Dan je utorak\n");
                    break; // Obavezan izlazak iz case-a!
18         case 3: printf("Dan je sreda\n");
                    break; // Obavezan izlazak iz case-a!
20         case 4: printf("Dan je cetvrtak\n");
                    break; // Obavezan izlazak iz case-a!
22         case 5: printf("Dan je petak\n");
                    break; // Obavezan izlazak iz case-a!
24         case 6: printf("Dan je subota\n");
                    break; // Obavezan izlazak iz case-a!
26         case 7: printf("Dan je nedelja\n");
                    break; // Obavezan izlazak iz case-a!
28         default: printf("Lose unet broj!\n"); // Ako ni jedna provera
                    ne prolazi
    }
30
    return 0;
32 }
```

## Rešenje 2.41

```
#include <stdio.h>
2 #include <stdlib.h> // Potrebno za exit funkciju

4 // Za unetu godinu i mesec ispisuje se naziv meseca i koliko dana ima
    u tom mesecu te godine

6 int main()
{
8     int godina;
```

```
    int mesec;

10    int prestupna;

12

14    printf("Unesite godinu: ");
    scanf("%d",&godina);

16

18    if(godina < 0)
    {
        printf("Lose uneta godina!\n");
        exit(EXIT_FAILURE);
    }

22

    if((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
        // Provera da li je godina prestupna,
        prestupna = 1;
        // bitno za mesec februar
    else
        prestupna = 0;

26

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

30

    switch(mesec)
    {
32        case 1: printf("Januar, 31 dan\n");
                break;
34        case 2:
                if(prestupna)
                    printf("Februar, 29 dana\n");
                else
                    printf("Februar, 28 dana\n");
                break;
36
38        case 3: printf("Mart, 31 dan\n");
                break;
40        case 4: printf("April, 30 dana\n");
                break;
42        case 5: printf("Maj, 31 dan\n");
                break;
44        case 6: printf("Jun, 30 dana\n");
                break;
46        case 7: printf("Jul, 31 dan\n");
                break;
48        case 8: printf("Avgust, 31 dan\n");
                break;
50        case 9: printf("Septembar, 30 dana\n");
                break;
52        case 10: printf("Oktobar, 31 dan\n");
                break;
54        case 11: printf("Novembar, 30 dana\n");
```

```
        break;
60     case 12: printf("Decembar, 31 dan\n");
        break;
62     default: printf("Lose unet redni broj meseca!\n");
    }
64
    return 0;
66 }
```

### Rešenje 2.42

```
#include <stdio.h>
2
// Za uneti datum određuje ispisuje se naziv godisnjeg doba kome
datum pripada
4
int main()
6 {
    int godina;
    int mesec;
    int dan;
10
    printf("Unesite datum (DD MM GGGG): ");
12    scanf("%d%d%d", &dan, &mesec, &godina);

    if(dan < 0 || godina < 0)
14        printf("Lose unet datum!\n");

    switch(mesec) // Dodati provere za redni broj dana!
16    {
        case 1: printf("Zima\n");
20        break;

        case 2: printf("Zima\n");
22        break;

        case 3:
24
            if(dan < 21)
26                printf("Zima\n");
            else
28                printf("Prolece\n");
            break;
30

        case 4: printf("Prolece\n");
32        break;

        case 5: printf("Prolece\n");
34        break;

        case 6:
36
            if(dan < 21)
38
```

```
40         printf("Prolece\n");
41     else
42         printf("Leto\n");
43     break;
44
45     case 7: printf("Leto\n");
46         break;
47
48     case 8: printf("Leto\n");
49         break;
50
51     case 9:
52         if(dan < 23)
53             printf("Leto\n");
54         else
55             printf("Jesen\n");
56         break;
57
58     case 10: printf("Jesen\n");
59         break;
60
61     case 11: printf("Jesen\n");
62         break;
63
64     case 12:
65         if(dan < 22)
66             printf("Jesen\n");
67         else
68             printf("Zima\n");
69         break;
70
71     default: printf("Lose unet redni broj meseca!\n");
72 }
73
74 return 0;
75 }
```

## 2.3 Petlje

**Zadatak 2.43** Tekst

[Rešenje [A.28](#)]

**Zadatak 2.44** Tekst

[Rešenje [2.44](#)]

**Zadatak 2.45** Tekst

[Rešenje [2.45](#)]

**Zadatak 2.46** Tekst

[Rešenje [2.46](#)]

**Zadatak 2.47** Tekst

[Rešenje [2.47](#)]

**Zadatak 2.48** Tekst

[Rešenje [2.48](#)]

**Zadatak 2.49** Tekst

[Rešenje [2.49](#)]

**Zadatak 2.50** Tekst

[Rešenje [2.50](#)]

**Zadatak 2.51** Tekst

[Rešenje [2.51](#)]

**Zadatak 2.52** Tekst

[Rešenje [2.52](#)]

**Zadatak 2.53** Tekst

[Rešenje [2.53](#)]

**Zadatak 2.54** Tekst

[Rešenje [2.54](#)]

**Zadatak 2.55** Tekst

[Rešenje [2.55](#)]

**Zadatak 2.56** Tekst

[Rešenje [2.56](#)]

**Zadatak 2.57** 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.

## 2 Kontrola toka

---

### 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
```

[Rešenje 2.82]

**Zadatak 2.58** 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
```

[Rešenje 2.82]

**Zadatak 2.59** 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
```

[Rešenje 2.82]

**Zadatak 2.60** 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!
```

[Rešenje 2.82]

**Zadatak 2.61** 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
```

[Rešenje 2.82]

**Zadatak 2.62** 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
```

[Rešenje 2.82]

**Zadatak 2.63** 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.

## 2 Kontrola toka

---

### 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
```

[Rešenje [2.82](#)]

**Zadatak 2.64** Sa standradnog 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
```

[Rešenje [2.82](#)]

**Zadatak 2.65** Sa standradnog 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
```

[Rešenje [2.82](#)]

**Zadatak 2.66** 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

```

[Rešenje 2.82]

**Zadatak 2.67** 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

```

[Rešenje 2.82]

**Zadatak 2.68** 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

```

[Rešenje 2.82]

**Zadatak 2.69** 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

```

[Rešenje 2.82]

**Zadatak 2.70** 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.

## 2 Kontrola toka

---

### 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
```

[Rešenje 2.82]

**Zadatak 2.71** 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!
```

[Rešenje 2.82]

**Zadatak 2.72** 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
```

[Rešenje 2.82]

**Zadatak 2.73** 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

```

[Rešenje 2.82]

**Zadatak 2.74** 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 2*

```

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

```

[Rešenje 2.82]

**Zadatak 2.75** 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

```

[Rešenje 2.82]

**Zadatak 2.76** 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.
```

[Rešenje 2.82]

**Zadatak 2.77** 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
```

[Rešenje 2.82]

**Zadatak 2.78** 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
```

[Rešenje 2.82]

**Zadatak 2.79** 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
**
**

```

[Rešenje 2.82]

**Zadatak 2.80** Napisati program koji za uneti ceo broj  $n$  i karakter  $c$  iscrtava rub jednakokrako 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: +
+
++
+ +
+ +
++++

```

[Rešenje 2.82]

**Zadatak 2.81** 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
* *
*
* *

```

[Rešenje 2.82]

**Zadatak 2.82** 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
*
*
*
*
*****
*
*
*
*
*
```

[Rešenje 2.82]

**Zadatak 2.83** Napisati program koji određuje N-ti član Fibonačijevog niza.

[Rešenje 2.155]

**Zadatak 2.84** Napisati program koji broj N transformiše tako što mu uklanja nule sa desne strane. Napr. 12000 se transformiše u 12.

[Rešenje 2.84]

**Zadatak 2.85** Napisati program koji proverava da li je dati prirodan broj N palindrom.

[Rešenje 2.85]

**Zadatak 2.86** Sa standarnog ulaza unosi se ceo broj n, a potom n realnih brojeva. Odrediti koliko puta je prilikom unosa došlo do promene znaka.

[Rešenje 2.116]

**Zadatak 2.87** Napisati program koji za realno x i prirodan broj n izračunava:

a)  $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \dots + n \cdot x^n$

b)  $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \dots + \frac{1}{x^n}$

c)  $S = (1 + \cos(x)) \cdot (1 + \cos(x^2)) \cdot \dots \cdot (1 + \cos(x^n))$  (čuvanje međurezultata)

d)  $S = 1 - 2 + 3 - 4 + 5 - \dots (-1)^n \cdot n$

[Rešenje 2.116]

**Zadatak 2.88** Za učitani broj  $n$  napisati program koji računa:

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

[Rešenje 2.116]

**Zadatak 2.89** Napisati program koji za zadato  $N$  ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.90**

[Rešenje 2.116]

**Zadatak 2.91** Napisati program koji za zadato  $N$  ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.92** Napisati program koji za zadato  $N$  ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.93** Napisati program koji za zadato N ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.94** Napisati program koji za zadato N ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.95** Napisati program koji za zadato N ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.96** Napisati program koji za zadato N ispisuje:

*Primer 1*

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



[Rešenje 2.116]

**Zadatak 2.97** Napisati program koji za zadato N ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.98** Napisati program koji za zadato N ispisuje:

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.99** Sa standardnog ulaza unosi se neoznačen broj N. Napisati program koji za uneto N iscrtava kvadrat dimenzije N koji na glavnoj dijagonali ima zvezdice.

*Primer 1*

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

[Rešenje 2.116]

**Zadatak 2.100** Sa standardnog ulaza unose se neoznačeni celi brojevi M i N. Napisati program koji za učitane brojeve M i N ispisuje jedan do drugog N kvadrata čija je svaka strana sastavljena od M zvezdica. Zvezdice su međusobno razdvoje

## 2 Kontrola toka

prazninom.

### Primer 1

```

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

```

## Primer 2

```

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

```

[Rešenje 2.116]

**Zadatak 2.101** Sa standardnog ulaza unosi se ceo pozitivan broj  $n$ . Pretpostavlja se da je unos ispravan. Napisati program koji stampa zvezdice i minuseve sledeceg oblika za:

### Primer 1

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

## Primer 2

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

[Rešenje 2.116]

**Zadatak 2.102** Unosi se broj  $N$  ( $N \geq 2$ ). Napisati program koji na standardni izlaz ispisuje sledeću sliku:

### Primer 1

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

  *
 * *
*   *
* * * *
*       *
*         *
* * * * *

```

[Rešenje 2.116]

**Zadatak 2.103** Napisati program koji ispisuje vrednost funkcije  $\cos(x)$  u 10 ravnomerno razmaknutih tačaka intervala  $[a, b]$  ( $a$  i  $b$  su vrednosti tipa `double`, za koje važi  $a < b$  i učitavaju se sa tastature). Pri ispisu vrednosti se zaokružuju na 4 decimale. Za neispravan unos, program ispisuje broj -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 1 10
|| 0.5403 -0.4161 -0.9900 -0.6536 0.2837 0.9602
|| 0.7539 -0.1455 -0.9111 -0.8391
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 0 28.274
|| 1.0000 -1.0000 1.0000 -1.0000 1.0000 -1.0000
|| 1.0000 -1.0000 1.0000 -1.0000
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 1 -3
|| -1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve a i b: 0 1
|| 1.0000 0.9938 0.9754 0.9450 0.9028 0.8496 0.7859
|| 0.7125 0.6303 0.5403
```

[Rešenje 2.116]

**Zadatak 2.104** 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. Za neispravan unos, program ispisuje broj -1.

*Primer 1*

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

*Primer 2*

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

*Primer 7*

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

*Primer 4*

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

[Rešenje 2.116]

**Zadatak 2.105** Napisati program koji za uneto  $n$  ispisuje „trougao” sačinjen od „koordinata” svojih tačaka.

*Primer 1*

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

*Primer 2*

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

### Primer 3

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

### Primer 4

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

[Rešenje 2.116]

**Zadatak 2.106** Napisati C program koji na standardni izlaz ispisuje odgovor da li je uneti prirodan broj deljiv sumom svojih cifara.

[Rešenje 2.116]

**Zadatak 2.107** Napisati C program koji učitava sa standardnog ulaza cele brojeve dok ih je manje od 100 ili dok ne naiđe na nulu. Program treba da ispiše na standardni izlaz broj sa minimalnom poslednjom cifrom. Ako ih ima više neka ispiše bilo koji (ne koristiti nizove).

[Rešenje 2.116]

**Zadatak 2.108** Napisati program koji sledeću sumu računa sa minimalnim brojem operacija

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

Celobrojne vrednosti brojeva  $n$  i  $x$  se učitavaju sa standardnog ulaza.

[Rešenje 2.116]

**Zadatak 2.109** Sa standardnog ulaza unosi se broj. Izbaciti sve one cifre iz broja koje su jednake zbiru svojih suseda.

[Rešenje 2.116]

**Zadatak 2.110**  $A_0$  papir ima površinu  $1m^2$  i odnos stranica  $1 : \sqrt{2}$ .  $A_1$  papir dobija se podelom papira  $A_0$  po dužoj ivici.  $A_2$  papir dobija se podelom  $A_1$  papira po dužoj ivici itd. Napisati program koji za uneto  $k$  ispisuje dimenzije papira  $A_k$  u milimetrima.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 4
|| 297 210
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3
|| 297 420
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| 74 105
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 9
|| 37 52
```

[Rešenje 2.116]

**Zadatak 2.111** Sa standardnog ulaza unosi se ceo pozitivan broj  $n$  veći od 0. Napisati program koji računa sledeću vrednost:

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

U slučaju greške ispisati -1.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 0
|| -1
```

*Primer 4*

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

[Rešenje 2.116]

**Zadatak 2.112** 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 n: 22
|| 33
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 1230
|| 1331
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 0
|| 1
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 23456
|| 33557
```

[Rešenje 2.116]

**Zadatak 2.113** Sa standardnog ulaza unosi se ceo pozitivan broj  $N$ , a potom  $N$  celih brojeva. Naći sumu brojeva koji su deljivi sa 5, a nisu deljivi sa 7. U slučaju greške ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5 2 35 5 -175 -20
|| -15
```

*Primer 2*

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

*Primer 3*

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

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 6 2205 -1904 2 7 -540 5
|| -535
```

[Rešenje 2.116]

**Zadatak 2.114** Sa standardnog ulaza unosi se ceo pozitivan neparan broj  $n$ . Napisati program koji za uneto  $n$  izračunava:

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

U slučaju greške ispisati -1.

*Primer 1*

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

*Primer 2*

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

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 20
|| -1
```

*Primer 4*

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

[Rešenje 2.116]

**Zadatak 2.115** Sa standardnog ulaza unose se realni brojevi  $x$  i  $a$  i ceo pozitivan broj  $n$  veći od 0. Napisati program koji za učitane vrednosti  $x$ ,  $a$  i  $n$  izračunava:

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

U slučaju greške ispisati -1.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 3.2 0.2 5
|| 367940960.000000
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 2 1 3
|| 101.000000
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 2.6 0.3 3
|| 76.164085
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5.4 7 -2
|| -1
```

[Rešenje 2.116]

**Zadatak 2.116** Napisati program koji za argument komandne linije  $n$  ispisuje sve brojeve od 1 do  $n$ , zatim svaki drugi broj od 1 do  $n$ , zatim svaki treći broj od 1 do  $n$  itd., završavajući sa svakim  $n$ -tim (tj. samo sa 1). U slučaju greške ispisati -1.

#### Primer 1

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

#### Primer 2

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

#### Primer 3

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

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -23
```

[Rešenje 2.116]

## 2.4 Rešenja

### Rešenje A.28

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

## 2 Kontrola toka

---

```
3  */
5  #include<stdio.h>
7  int main()
8  {
9      int i=0;          /* promenljiva i kontrolise koliko puta ce se petlja
10         izvrsiti */
11     while (i<10) /* pre ulaska u telo petlje proverava se da li je */
12     {
13         /* ispunjen uslov petlje */
14         printf("We love C programming\n");
15         i++; /* operator ++ uvecava i promenljivu za 1
16             ili i+=1;
17             ukoliko ne bismo menjali vrednost promenljive i doslo
18             bi
19             do beskonacne petlje!
20         */
21     }
22     /*
23     brojanje u while petlji smo mogli realizovati i preko uslova:
24
25     i=1;
26     while(i<=10)
27     {
28         ...
29     }
30
31     ili
32
33     i=2;
34     while(i<=11)
35     {
36         ...
37     }
38
39     ili
40
41     i=3;
42     while(i<13)
43     {
44         ...
45     }
46
47     Brojanje pocet od 0 uz koriscenje stroge nejednakosti
48     je u duhu programskog jezika C i zato cemo ovaj nacin
49     brojanja najcesce koristiti
50     */
51     return 0;
52 }
```



## Rešenje 2.44

```
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 2.45

```
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");
}
```

## 2 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 2.46

```
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 sadrzati 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 2.47

```

/*
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()
9 {
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 */

```

## 2 Kontrola toka

```
24     p=p*x;
25 }
26
27 printf("Proizvod unetih brojeva je %d\n",p);
28
29 return 0;
30 }
```

### Rešenje 2.48

```
/*
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 2.49

```
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 2.50

```

/*
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
       */

```

```
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 2.51

```
/* 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 2.52

```

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 2.53

```

/*
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 */
18     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 */
30     printf("for: ");
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 2.54

```
/*
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 2.55

```

/*
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 */

```

```
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 2.56

```
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)
```

```
37     if (i==x/i) /* u slucaju kada je delilac koren broja, npr 4
    za 16, ispisujemo ga jednom */
    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 [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.82](#)

Rešenje [2.155](#)

Rešenje [2.84](#)

Rešenje [2.85](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

Rešenje [2.116](#)

## 2.5 Funkcije

Zadatak 2.117    Tekst

[Rešenje [2.117](#)]

Zadatak 2.118    Tekst

[Rešenje [2.118](#)]

Zadatak 2.119    Tekst

[Rešenje [2.119](#)]

Zadatak 2.120    Tekst

[Rešenje [2.120](#)]

Zadatak 2.121    Tekst

[Rešenje [2.121](#)]



**Zadatak 2.122** Tekst

[Rešenje [2.122](#)]

**Zadatak 2.123** Tekst

[Rešenje [2.123](#)]

**Zadatak 2.124** Tekst

[Rešenje [2.124](#)]

**Zadatak 2.125** Tekst

[Rešenje [2.125](#)]

**Zadatak 2.126** Tekst

[Rešenje [2.126](#)]

**Zadatak 2.127** Tekst

[Rešenje [2.127](#)]

**Zadatak 2.128** Tekst

[Rešenje [2.128](#)]

**Zadatak 2.129** Tekst

[Rešenje [2.129](#)]

**Zadatak 2.130** Tekst

[Rešenje [2.130](#)]

**Zadatak 2.131** Napisati funkciju  $\text{int min}(\text{int } x, \text{int } y, \text{int } z)$  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
```

[Rešenje [2.155](#)]

## 2 Kontrola toka

---

**Zadatak 2.132** 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
```

[Rešenje 2.155]

**Zadatak 2.133** 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
```

[Rešenje 2.155]

**Zadatak 2.134** 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!
```

[Rešenje 2.155]

**Zadatak 2.135** Napisati funkciju *void grafikon\_h(int a, int b, int c, int d)* koja vrši horizontalno prikazivanje zadatah vrednosti. Napisati program koji učitava četiri pozitivna cela broja i prikazuje rezultat poziva funkcije. U slučaju

pogrešnog unosa, ispisati poruku o grešci.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 4 1 7 5
****
*
*****
****
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 8 -2 5 4
Greska: pogresan unos!
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 5 2 2 10
*****
**
**
*****
```

[Rešenje 2.155]

**Zadatak 2.136** 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
*
* *
* *
****
****
```

[Rešenje 2.155]

**Zadatak 2.137** Napisati funkciju `int prestupna(int godina)` koja za zadanu godinu proverava da li je prestupna. Funkcija treba da vrati 1 ako je godina

## 2 Kontrola toka

---

prestupna ili 0 ako nije. Napisati program koji učitava dva cela broja  $g1$  i  $g2$  i ispisuje sve godine iz intervala  $[g1, g2]$  koje su prestupne.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2001 2010  
|| Prestupne godine su: 2004 2008
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2005 2015  
|| Prestupne godine su: 2008 2012
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2010 2001  
|| Greska: pogresan unos!
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dve godine: 2001 2002  
|| Nema prestupnih godina u ovom intervalu!
```

[Rešenje [2.155](#)]

**Zadatak 2.138** Napisati funkciju `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu.

[Rešenje [2.155](#)]

**Zadatak 2.139** Napisati funkciju `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan.

[Rešenje [2.155](#)]

**Zadatak 2.140** Napiati funkciju `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum određuje datum sledećeg dana.

[Rešenje [2.155](#)]

**Zadatak 2.141** Napisati funkciju `int broj_dana1(int dan, int mesec, int godina)` koja određuje broj dana od početka godine do datog datuma.

[Rešenje [2.155](#)]

**Zadatak 2.142** Napisati funkciju `int broj_dana2(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine.

[Rešenje 2.155]

**Zadatak 2.143** Napisati funkciju `int broj_dana3(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma.

[Rešenje 2.155]

**Zadatak 2.144** 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!
```

[Rešenje 2.155]

**Zadatak 2.145** 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
```

[Rešenje 2.155]

**Zadatak 2.146** Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose

## 2 Kontrola toka

---

sa standardnog 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
```

[Rešenje 2.155]

**Zadatak 2.147** 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
```

[Rešenje 2.155]

**Zadatak 2.148** 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!
```

[Rešenje 2.155]

**Zadatak 2.149** Napisati funkciju *int faktorijel(int n)* koja računa faktoriyel broja *n*. Napisati i program koji učitava dva cela broja *x* i *y* ( $0 \leq x, y \leq 12$ ) i ispisuje vrednost zbira  $x! + y!$ .

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: 4 5
|| 144
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: 18 -5
|| Greska: pogresan unos!
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: 6 0
|| 721
```

[Rešenje 2.155]

**Zadatak 2.150** 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!
```

[Rešenje 2.155]

**Zadatak 2.151** Broj je Armstrongov ako je jednak sumi nekog 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.

## 2 Kontrola toka

---

- c) 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!
```

[Rešenje 2.155]

**Zadatak 2.152** Napisati funkciju *int par\_nepar(int n)* koja ispituje da li su cifre datog celog broja naizmenično parne i neparne. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i testira rad funkcije.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2749  
|| Broj ispunjava uslov!
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -963  
|| Broj ispunjava uslov!
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 27449  
|| Broj ne ispunjava uslov!
```

[Rešenje 2.155]

**Zadatak 2.153** 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
```



[Rešenje 2.155]

**Zadatak 2.154** 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!
```

[Rešenje 2.155]

**Zadatak 2.155** 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
```

[Rešenje 2.155]

## 2.6 Rešenja

Rešenje 2.117

```
#include <stdio.h>
2
int kvadrat(int x)
```

## 2 Kontrola toka

```
4 {
6     /* promenljive u listi argumenata funkcije, kao i one
       deklarisanе u samoj funkciji, lokalne su za tu funkciju
       sto znaci da se promenljive x i y neće "videti" nigde izvan
       funkcije kvadrat (ni u funkciji main ni u funkciji kub)
       */
10
12     int y;
14     y = x*x;
16     return y;
18 }
20
22 int kub(int a)
24 {
26     /*
       u listi argumenata funkcije mozemo, a ne moramo, imati
       promenljivu
       istog naziva kao promenljiva koja je deklarisanа u main
       funkciji
       (u ovom slucaju promenljiva a); ova promenljiva se razlikuje
       od promenljive a deklarisanе u main funkciji i vidljiva je
       samo unutar funkcije kub
       */
28     return a*a*a;
30 }
32
34 int main()
36 {
38     int a, kv, kb;
39     printf("Unesi ceo broj:");
40     scanf("%d", &a);
41
42     kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
       funkcije kvadrat */
43     kb = kub(a);      /* promenljivoj kb dodeljujemo povratnu vrednost
       funkcije kub */
44
45     printf("Kvadrat broja %d je %d, a njegov kub je %d\n", a, kv, kb);
46     return 0;
47 }
```

### Rešenje 2.118

```
1 /*
2  Napisati program koji za uneti realan broj x i ceo broj n ispisuje
3  vrednost stepena x^n. Unosenje promenljivih, racunanje stepena i
4  ispis promenljivih realizovati u posebnim funkcijama.
5  */
6
7 #include <stdio.h>
8 #include <stdlib.h>
```

```
9 float stepen(float a, int b)
11 {
12     float s=1;
13     int i;
14
15     for(i=0;i<abs(b);i++)
16         s=s*a;
17
18     return b>0 ? s : 1/s; /* ukoliko je izlozilac b negativan,
19                            izracunamo a^|b| i vracamo reciprocnu vrednost
20                            izracunatog stepena */
21 }
22
23 int main()
24 {
25     int n;
26     float x;
27     float s;
28
29     printf("Unesi jedan realan i jedan ceo broj:");
30     scanf("%f%d",&x,&n);
31
32     s = stepen(x,n);
33
34     printf("%f^%d=%f\n",x,n,s);
35
36     return 0;
37 }
38
39 }
```

### Rešenje 2.119

```
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     }
```

## 2 Kontrola toka

```
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("najveci zajednicki delilac za %d i %d je %d\n", a,b,nzd);
35
36     return 0;
37 }
```

### Rešenje 2.120

```
/*
2  Napisati funkciju koja za dato n vraca zbir reciprocnih vrednosti
   brojeva od 1 do n.
   Napisati program koji omogucava korisniku da unese prirodan broj n, a
   potom ispisuje zbir reciprocnih
4  vrednosti brojeva od 1 do n koristeći funkciju float zbir_reciprocnih
   (int n). Rezultat zaokruziti
   na dve decimalne.
6  */
7
8  #include <stdio.h>
9
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 }
```

```

24 printf("Zbir reciprocnih vrednosti brojeva od 1 do %d je %.2f\n", n
    , zbir_reciprocnih(n));
/* povratna vrednost funkcije zbir_reciprocnih je float; funkciju
   mozemo pozvati u okviru
26 naredbe printf i umesto specifikatora %.2f bice ispisana
   povratna vrednost funkcije
   zbir_reciprocnih zaokruzena na dve decimale */
28 return 0;
}

```

### Rešenje 2.121

```

1  /*
   Napisati funkciju koja racuna aritmeticku sredinu cifara datog celog
   broja.
3  Napisati potom glavni program koji omogucava korisniku da unese ceo
   broj
   i racuna aritmeticku sredinu njegovih cifara primenom napisane
   funkcije. Ispisati
5  izracunatu vrednost zaokruzenu na dve decimale.
   */
7
9  #include<stdio.h>
   #include<stdlib.h>

11 float aritmeticka_sredina(int x)
   {
13     int zbir_cifara=0;
     int broj_cifara=0;
15     char cifra;

17     if (x==0)      /* u slucaju da je uneta 0 */
         return 0; /* aritmeticka sredina cifara iznosi 0 i tu vrednost
            vracamo */
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,

```

## 2 Kontrola toka

```

                                neophodno je da bar
jednu od njih konvertujemo u realnu                                kako bismo izbegli
35 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 2.122

```

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

11  void ispis(float x, float y, int n) /* funkcija nema povratnu
   vrednost; zbog toga je povratni tip void */
   {
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;
        float t;
25     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)
32  {
33      printf("Nekorektan unos\n");
34      return -1;
35  }
36  if (b<a) /* u slucaju da je desni kraj intervala manji od levog */
37  {      /* zamenimo im mesta */
38      t=a;
39      a=b;
40      b=t;
41  }
42
43
44
45  ispis(a,b,n);
46
47  return 0;
48  }

```

### Rešenje 2.123

```

/*
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
8  #include<stdio.h>
9  #include<stdlib.h>
10
11 int broj_ncifara(int x)
12 {
13     int s=0;
14     char cifra;
15     x = abs(x);
16
17     while(x)
18     {
19         cifra = x%10;
20         s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
21                        neparna,
22                        a 0 kada je cifra parna */
23         x/=10;
24     }
25
26     return s;
27 }

```

## 2 Kontrola toka

---

```
26 | int main()
28 | {
    |     int x;
30 |     do
    |     {
32 |         scanf("%d",&x);
        printf("Broj neparnih cifara u zapisu broja %d: %d\n", x,
        broj_ncifara(x));
34 |     } while(x!=0);
36 |     return 0;
    | }
```

### Rešenje 2.124

```
1 | /*
    | Napisati funkciju koja ispituje da li je dati ceo broj prost.
    | Funkcija treba
3 | da vrati 1 ako je broj prost i 0 u suprotnom. Napisati potom glavni
    | program
    | koji za uneti ceo broj n ispisuje prvih n prostih brojeva.
5 | */
7 | #include <stdio.h>
    | #include <math.h>
9 |
11 | int prost (int x) /* 1-broj je prost, 0-broj nije prost */
    | {
    |     int i;
13 |
    |     if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
15 |         return 1;
17 |
    |     if (x%2==0) /* parni brojevi nisu prosti */
    |         return 0;
19 |
    |     for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
21 |         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()
    | {
```



```

35  int n;
scanf("%d",&n);
37  int i,j;

i=1; /* kandidat za prost broj */
39  j=0; /* brojac prostih brojeva */
while(j<n)
41  {
    if (prost(i))          /* ako je broj prost */
43  {
        printf("%d\n", i); /* stampamo ga i */
45        j++;             /* uvecavamo brojac prostih brojeva */
    }
47    i++; /* bilo da je i prost ili ne, uvecavamo ga za 1 i
        nastavljam sa sledecom iteracijom */
    }
49
51  return 0;
}

```

### Rešenje 2.125

```

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

11 int sadrzi(int x, int c)
   {
13     char cifra;
        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:");

```

## 2 Kontrola toka

```
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 2.126

```
/*
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
8 da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

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
16 A za uneti broj -284:
    Sve cifre broja su parne.
    Broj sadrži različite cifre
18
19 */
20 #include <stdio.h>
21 #include <stdlib.h>
22
23 int sve_parne_cifre(int x) /* funkcija vraća 1 ako su sve cifre broja
    parne i 0 u suprotnom */
24 {
25     char d;
26     x=abs(x); /* uzimamo apsolutnu vrednost broja za slučaj da je
        broj negativan */
27     while (x>0)
28     {
29         d=x%10; /* izdvajamo cifru broja */
30
31         if (d%2==1) /* u slučaju da je neparna, to znači da nisu sve
            cifre broja parne */
32             return 0; /* vraćamo 0 */
33
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
                   vratamo 1
40                   */
42 }

44 int sve_cifre_jednake(int x) /* funkcija vraca 1 ako su sve cifre
    broja jednake i 0 u suprotnom */
{
46     char d;
    char prva_cifra;
48     x=abs(x);
    prva_cifra = x%10; /* izdvajamo prvu cifru broja */
50     x/=10;           /* broj delimo sa 10 jer smo prvu cifru vec
    izdvojili */

52     while(x)
    {
54         d = x%10;

56         if (d!=prva_cifra)
            return 0;

58         x/=10;
60     }

62     return 1;
    }
64 main()
{
66     int x;
    int d;

68     printf("unesi ceo broj:");
70     scanf("%d", &x);

72     if (sve_parne_cifre(x))
        printf("Sve cifre broja su parne\n");
74     else
        printf("Broj sadrzi bar jednu neparnu cifru\n");

76     if (sve_cifre_jednake(x))
        printf("Sve cifre broja su jednake\n");
78     else
        printf("Broj sadrzi razlicite cifre \n");
80 }
82 }
```

### Rešenje 2.127

```
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
8  #include <stdio.h>
10
12 int je_stepen(unsigned x, unsigned n) /* funkcija vraća izlozilac
   stepena ukoliko broj x jeste neki stepen broja n */
14 {
16     int i=1;
18     int s=n;
20     while(s<x)
22     {
24         s=s*n;
26         i++;
28     }
30     if (s==x)
32         return i;
34     return -1;
36 }
38
40 int main()
42 {
44     unsigned x;
46     unsigned n;
48     int st;
50
52     scanf("%u%u",&x,&n);
54
56     st = je_stepen(x,n);
58
60     if (st!=-1)
62         printf("%u=%u^%d\n",x,n,st);
64     else
66         printf("%u nije stepen broja %u\n",x,n);
68
70     return 0;
72 }
```

### Rešenje 2.128

---

```
/*
2
   Napisati funkciju
4
   double e_na_x(double x, double eps)
6
   koja racuna vrednost e^x kao parcijalnu sumu reda
8   suma(x^n/n!), gde indeks n ide od
   od 0 do beskonacno, pri cemu se sumiranje vrsi dok
10  je razlika sabiraka u redu po apsolutnoj vrednosti
   manja od eps. Napisati potom program koji omogucuje
12  korisniku da unese jedan realan broj x i ispisuje
   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
       prvi sabirak (kome odgovara n=0) iznosi 1; zbog toga promenljive
36     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 2.129

```
/*
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
  zavrшава 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.

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 #include<stdio.h>

20 int zbir_cifara(int x)
22 {
    int s=0;
24 char cifra;
    while(x)
26 {
        cifra = x%10;
28 s+=cifra;
        x/=10;
    }
```

```

30     }
    return s;
32 }

34 int srecan(int x)
{
36     int s; /* promenljiva s sadrzi sumu cifara */

38     do
    {
40         s=zbir_cifara(x);
        x=s; /* kada izracunamo sumu cifara, dodeljujemo je promenljivoj
            x jer iz te promenljive izdvajamo cifre u funkciji zbir_cifara
            */
42     } while(x>=10);

44     return (x==1);

46 }

48 int main()
{
50     unsigned n;
    int i;
52     printf("Unesi jedan neoznaceni broj:");
    scanf("%u",&n);

54     for(i=1;i<=n;i++)
56         if (srecan(i))
            printf("%d je srecan\n", i);

58     return 0;
60 }

```

### Rešenje 2.130

```

/*
2  . a) Napisati funkciju

4     int konverzija (int c)

6  koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.

8  b) Napisati program koji omogućava korisniku da unese niz karaktera
   sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
10 Na primer, za uneti tekst "Kolokvijum iz Progi je 1.12." program
   treba da ispise "kOLOVKIJUM IZ pROGI JE 1.12."

12 */
14 #include <stdio.h>

```

```
16 int konverzija(int c)
17 {
18     /* ključna rec return vraća povratnu vrednost funkcije (ako je ima)
19     */
20     /* i završava izvršavanje funkcije */
21
22     if (c>='A' && c<='Z')
23         return c+'a'-'A';
24
25     if (c>='a' && c<='z')
26         return c-'a'+'A';
27
28     return c;
29 }
30
31 int main()
32 {
33     int c;
34
35     while((c=getchar())!=EOF) /* korisnik unosi karakter po karakter
36         do konstante EOF */
37         putchar(konverzija(c)); /* funkcija putchar ispisuje jedan
38                                 karakter na standardni izlaz */
39
40     return 0;
41 }
```

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)



Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)

Rešenje [2.155](#)



# 3

## Predstavljanje podataka

### 3.1 Nizovi

**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](#)]

**Zadatak 3.7** Tekst

[Rešenje [3.7](#)]

#### Zadatak 3.8 Tekst

[Rešenje 3.8]

#### Zadatak 3.9 Tekst

[Rešenje 3.9]

#### Zadatak 3.10 Tekst

[Rešenje 3.10]

**Zadatak 3.11** 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
```

[Rešenje 3.37]

**Zadatak 3.12** 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

```

[Rešenje 3.37]

**Zadatak 3.13** 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!

```

[Rešenje 3.37]

**Zadatak 3.14** 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

```

### 3 Predstavljanje podataka

---

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: U  
|| Unesite karakter: 4  
|| Unesite karakter: a  
|| Unesite karakter: u  
|| Unesite karakter: *  
|| u a 4 U
```

[Rešenje 3.37]

**Zadatak 3.15** 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.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 251 125  
|| Brojevi se zapisuju istim ciframa!
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 8898 9988  
|| Brojevi se ne zapisuju istim ciframa!
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: -7391 1397  
|| Brojevi se zapisuju istim ciframa!
```

[Rešenje 3.37]

**Zadatak 3.16** 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!
```

[Rešenje 3.37]

**Zadatak 3.17** 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:
```

[Rešenje 3.37]

**Zadatak 3.18** Sa standardnog ulaza se, redom, učitavaju dimenzija i elementi dva niza  $a$  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
```

[Rešenje 3.37]

### 3 Predstavljanje podataka

---

**Zadatak 3.19** 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
```

[Rešenje 3.37]

**Zadatak 3.20** 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

```

[Rešenje 3.37]

**Zadatak 3.21** 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

```

[Rešenje 3.37]

**Zadatak 3.22** 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

```

### 3 Predstavljanje podataka

---

[Rešenje 3.37]

**Zadatak 3.23** 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
```

[Rešenje 3.37]

**Zadatak 3.24** Napisati funkciju *int zbir(int a[], int n, int i, int j)* koja računa zbir elemenata niza celih brojeva *a* dužine *n* od pozicije *i* do pozicije *j*. Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 5
Unesite elemente niza: 11 5 6 48 8
Unesite vrednosti za i i j: 0 2
Zbir je: 22
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 3
Unesite elemente niza: -2 8 1
Unesite vrednosti za i i j: 8 12
Greska: nekorektne vrednosti granica!
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 7
Unesite elemente niza: -2 5 9 11 6 -3 -4
Unesite vrednosti za i i j: 2 5
Zbir: 23
```

[Rešenje 3.37]

**Zadatak 3.25** Napisati funkciju *float zbir\_pozitivnih(float a[], int n, int k)* koja izračunava zbir prvih  $k$  pozitivnih elemenata realnog niza  $a$  dužine  $n$ . Napisati i program koji testira rad funkcije. Pretpostaviti da dužina niza neće biti veća od 100.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 8
|| Unesite elemente niza:
|| 2.34 1 -12.7 5.2 -8 -6.2 7 14.2
|| Unesite vrednost za k: 3
|| Zbir je: 8.54
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 3
|| Unesite elemente niza:
|| -6.598 -8.14 -15
|| Unesite vrednost za k: 4
|| Zbir je: 0.00
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 7
|| Unesite elemente niza:
|| -35.11 5.29 -1.98 12.1 12.2 -3.33 -4.17
|| Unesite vrednost za k: 15
|| Zbir: 29.59
```

[Rešenje 3.37]

**Zadatak 3.26** 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
```

[Rešenje 3.37]

**Zadatak 3.27 Filip-Janicic?** Napisati funkciju (i program koji je testira) koja:

- (a) proverava da li dati niz sadrži dati broj;
- (b) pronalazi indeks prve pozicije na kojoj se u nizu nalazi dati broj (-1 ako niz ne sadrži broj).
- (c) pronalazi indeks poslednje pozicije na kojoj se u nizu nalazi dati broj (-1 ako niz ne sadrži broj).
- (d) izračunava zbir svih elemenata datog niza brojeva;
- (e) izračunava prosek (aritmetičku sredinu) svih elemenata datog niza brojeva;
- (f) izračunava najmanji element datog elemenata niza brojeva;
- (g) određuje poziciju najvećeg elementa u nizu brojeva (u slučaju više pojavljivanja najvećeg elementa, vratiti najmanju poziciju);
- (h) proverava da li je dati niz brojeva uređen neopadajuće

[Rešenje 3.37]

**Zadatak 3.28 Filip-Janicic?** Napisati funkciju (i program koji je testira) koja:

- (a) izbacuje poslednji element niza;
- (b) izbacuje prvi element niza (napisati varijantu u kojoj je bitno očuvanje redosleda elemenata i varijantu u kojoj nije bitno očuvanje redosleda);
- (c) izbacuje element sa date pozicije k ;
- (d) ubacuje element na kraj niza;
- (e) ubacuje element na početak niza;
- (f) ubacuje dati element x na datu poziciju k ;
- (g) izbacuje sva pojavljivanja datog elementa x iz niza.

Napomena: funkcija kao argument prima niz i broj njegovih trenutno popunjenih elemenata, a vraća broj popunjenih elemenata nakon izvođenja zahtevane operacije.

[Rešenje 3.37]

**Zadatak 3.29 Filip-Janicic?** Napisati funkciju (i program koji je testira) koja:

- (a) određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva;
- (b) određuje dužinu najvećeg neopadajućeg podniza datog niza celih brojeva;
- (c) određuje da li se jedan niz javlja kao podniz uzastopnih elemenata drugog;
- (d) određuje da li se jedan niza javlja kao podniz elemenata drugog (elementi ne moraju da budu uzastopni, ali se redosled pojavljivanja poštuje);
- (e) obrće dati niz brojeva;
- (f) rotira sve elemente datog niza brojeva za k pozicija ulevo;
- (g) rotira sve elemente datog niza brojeva za k pozicija udesno;
- (h) izbacuje višestruka pojavljivanja elemenata iz datog niza brojeva (napisati varijantu u kojoj se zadržava prvo pojavljivanje i varijantu u kojoj se zadržava poslednje pojavljivanje).
- (i) spaja dva niza brojeva koji su sortirani neopadajući u treći niz brojeva koji je sortiran neopadajući.

[Rešenje 3.37]

**Zadatak 3.30** Napisati funkciju `int f3(int a[], int n, int b[], int m)` i ispituje da li prvi sadrži bar dva broja koji se pojavljuju u drugom nizu. Povratna vrednost je dakle, 0, ili 1. Testirati pozivom u main-u. Maksimalna dužina niza je 100 elemenata.

[Rešenje 3.37]

**Zadatak 3.31** Napisati C funkciju koja u proslećenom nizu eliminiše sve brojeve koji nisu deljivi svojim indeksom (vrednost na indeksu 0 zadržati, jer nije dozvoljeno deljenje sa 0). Niz reorganizovati, tako da nema *rupa* koje su nastale eliminacijom elemenata. Kao rezultat funkcije vratiti novu dimenziju niza.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza: 10
Unesite elemente niza:
4 2 1 6 7 8 10 2 16 3
4 2 6 16
```

[Rešenje 3.37]

**Zadatak 3.32** Implementirati funkciju `int min_max(int a[], int n)` koja prihvata celobrojni niz, pronalazi indekse najmanjeg i najvećeg elementa tog niza koristeći samo jedan prolaz (jednu petlju), a zatim kao povratnu vrednost vraća manji od ta dva indeksa.

Program testirati pozivom funkcije iz main programa i ispisom rezultata na standardni izlaz, pri čemu korisnik sa standardnog ulaza unosi niz dužine 10 elemenata.

[Rešenje 3.37]

**Zadatak 3.33** Napisati funkciju `void brojanje(int a[], int brojac[], int N)` čiji su argumenti `a` i `brojac` celobrojni nizovi dimenzije `N`. Vrednosti elemenata niza `a` su između 0 i `N - 1`. Funkcija izračunava elemente niza `brojac` tako da je `brojac[i]` jednak broju pojavljivanja broja `i` u nizu `a`. Program testirati pozivom funkcije iz main programa - korsnik učitava broj `N` i potom niz `a` dužine `N`, potom poziva funkciju i potom na standardnom izlazu izpisuje dobijeni niz.

[Rešenje 3.37]

**Zadatak 3.34** Napisati funkciju `int ind(int a[],int n)` koja kao povratnu vrednost ima indeks onog elementa niza koji je po vrednosti najbliži srednjoj vrednosti onih elemenata niza brojeva koji su deljivi sa 3.

Program testirati pozivom funkcije iz main programa i ispisom rezultata na standardni izlaz, pri čemu korisnik sa standardnog ulaza unosi broj `n`, a zatim niz od `n` celih brojeva (maksimalna dimenzija niza je 100 elemenata).

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 5
|| Unesite elemente niza:
|| 1 2 3 4 5
|| 2
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj elemenata niza: 5
|| Unesite elemente niza:
|| 3 6 2 4 7
|| 3
```

[Rešenje 3.37]

**Zadatak 3.35** Sa standardnog ulaza se unosi jedna linija teksta. Napisati program koji prikazuje koliko puta se javilo svako od slova engleskog alfabeta (ne praviti razliku između velikih i malih slova).

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| haHJJkL
|| a:1 b:0 c:0 d:0 e:0 f:0 g:0 h:2 i:0 j:2 k:1 l:1 m:0 n:0 o:0 p:0 q:0 r:0 s:0t:0 u:0 v:0 w:0 x:0 y:0 z:0
```

## Primer 2

INTERAKCIJA SA PROGRAMOM:

DanaS j3 \_j\_utRo laBU78d

a:3 b:1 c:0 d:2 e:0 f:0 g:0 h:2 i:0 j:2 k:0 l:1 m:0 n:1 o:1 p:0 q:0 r:1 s:1t:1 u:2 v:0 w:0 x:0 y:0 z:0

## Primer 3

INTERAKCIJA SA PROGRAMOM:

Sao PaoLo 1998 \_JuZna Amerika90

a:5 b:0 c:0 d:2 e:1 f:0 g:0 h:0 i:1 j:1 k:1 l:1 m:1 n:1 o:3 p:1 q:0 r:1 s:1t:0 u:1 v:0 w:0 x:0 y:0 z:0

## Primer 4

INTERAKCIJA SA PROGRAMOM:

Iaxx kk 3yyy 4qqqq

a:0 b:0 c:0 d:0 e:0 f:0 g:0 h:0 i:1 j:0 k:2 l:0 m:0 n:0 o:0 p:0 q:4 r:0 s:0t:0 u:0 v:0 w:0 x:3 y:3 z:0

[Rešenje 3.37]

**Zadatak 3.36** Napisati program koji sa standardnog ulaza učitava 50 celih brojeva i razdvaja ih na parne i neparne tako što parne brojeve upisuje na početak niza, a neparne na kraj niza. Ispisati niz dobijen na taj način. Nije dozvoljeno koristiti dodatne nizove.

[Rešenje 3.37]

**Zadatak 3.37**

- Napisati funkciju `void brojanje(int a[], int brojac[], int N)` čiji su argumenti `a` i `brojac` celobrojni nizovi dimenzije  $N$ . Vrednosti elementa niza `a` su između 0 i  $N - 1$ . Funkcija izračunava elemente niza `brojac` tako da je  $i$ -ti element `brojac[i]` jednak broju pojavljivanja broja  $i$  u nizu `a`.
- Za celobrojni niz `a` dimenzije  $N$  kažemo da je *permutacija* ako sadrži sve brojeve  $i$ :  $0 \leq i \leq N$ . Sastaviti funkciju `int DaLiJePermutacija(int a[], int N)` koja vraća 1 ako je niz `a` permutacija, a 0 inače. (koristiti funkciju `brojanje`).

[Rešenje 3.37]

## 3.2 Rešenja

## Rešenje 3.1

### 3 Predstavljanje podataka

---

```
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
12  /*
13  Pretprocesorskom direktivom define uvode se simbolicka imena (u ovom
   slucaju
15  MAX) kojima se pridružuje 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];
       int n;
       int i;
       int s;
35
37  printf("Unesi dimenziju niza:");
       scanf("%d", &n);
39
       if (n<1 || n>100)
```



```

41 {
42     printf("Neispravan unos\n");
43     return -1;
44 }
45
46 /*
47     prvi element niza ima indeks 0, a poslednji n-1,
48     gde je n broj elemenata niza; elementima niza pristupamo
49     preko indeksa; na primer, ako niz a ima 5 elemenata, mozemo
50     im pristupiti pomocu
51     a[0], a[1], a[2], a[3], a[4]
52
53 */
54
55 for (i=0; i<n; i++)
56 {
57     printf("a[%d]=",i);
58     scanf("%d", &a[i]);
59 }
60
61 for (i=0; i<n; i++)
62 {
63     printf("b[%d]=",i);
64     scanf("%d", &b[i]);
65 }
66
67 s=0;
68
69 for (i=0; i<n; i++)
70     s = s + a[i]*b[i];
71
72 printf("Skalarni proizvod: %d\n",s);
73 return 0;
74 }

```

### Rešenje 3.2

```

1 /*
2     Napisati program koji ucitava broj elemenata niza (n<=100),
3     zatim ucitava elemente niza i ispisuje:
4     a) elemente niza koji se nalaze na parnim indeksima
5     b) parne elemente niza
6
7 */
8
9 #include <stdio.h>
10 #define MAX 100
11
12 int main()
13 {
14     int a[MAX];

```

### 3 Predstavljanje podataka

```
16  int n;  
    int i;  
  
18  
    printf("Unesi dimenziju niza:");  
20  scanf("%d", &n);  
  
22  if (n<1 || n>MAX)  
  {  
24    printf("Nekorektan unos\n");  
    return -1;  
26  }  
  
28  
  
30  for (i=0; i<n; i++)  
  {  
32    printf("a[%d]=",i);  
    scanf("%d", &a[i]); /* učitavamo jedan po jedan element niza */  
34  }  
  
36  printf("Elementi sa parnim indeksima:\n");  
  for (i=0; i<n; i+=2)  
38    printf("a[%d]=%d\n",i,a[i]);  
  
40  printf("Parni elementi:\n");  
  for (i=0; i<n; i++)  
42    if (a[i]%2==0)  
      printf("a[%d]=%d\n",i,a[i]);  
44  
46  return 0;  
}
```

#### Rešenje 3.3

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

```
15      U zapisu broja -252, cifra 5 se pojaviljuje 1 puta
16
17  */
18
19  #include<stdio.h>
20  #include<stdlib.h>
21  #define MAX 100
22
23  int main()
24  {
25      int x;
26      int brojaci[10];
27      char cifra;
28      int original;
29      int i;
30
31      printf("Unesi jedan ceo broj:");
32      scanf("%d",&x);
33
34      /*
35       svaki element niza brojaci predstavlja
36       brojac za jednu cifru:
37       brojac[0] sadrzi broj nula
38       brojac[1] sadrzi broj jedinica
39       ...
40       brojac[9] sadrzi broj devetki
41
42       brojaci se inicijalizuju na vrednost 0
43  */
44
45      for(i=0;i<10;i++)
46          brojaci[i]=0;
47
48      /*
49       vrednost promenljive x ce biti unistena
50       u while petlji jer je u svakom koraku delimo
51       sa 10; njenu vrednost cuvamo u promenljivoj
52       original kako bismo mogli da je iskoristimo
53       na kraju prilikom ispisa
54  */
55      original = x;
56
57      /*
58       Uzimamo apsolutnu vrednost broja za slucaj
59       da je uneti broj negativan
60  */
61      x=abs(x);
62
63      /* Izdvajanje cifara broja */
64      do
65      {
66          cifra = x%10;
```

### 3 Predstavljanje podataka

```
        brojaci[cifra]++; /* Uvecavamo brojac odgovarajuće cifre */
67     x/=10;
    } while(x);

69     /* Ispis brojaca koji su razliciti od nule */
71     for(i=0;i<10;i++)
        if(brojaci[i])
73         printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n",
            original, i, brojaci[i]);

75     return 0;
77 }
```

#### Rešenje 3.4

```
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

11 int main()
12 {
13     /* Za svaku dekadnu cifru definisemo jedan brojac (tj. imamo niz
       od 10 brojaca): brojaci[0] broji koliko se puta pojavio karakter
       '0', brojaci[1] broji koliko se puta pojavio karakter '1' i tako
15     dalje. Svi brojac se inicijalizuju nulama.
       */
17     int cifre[10];
18     int mala[26];
19     int velika[26];

21     int c, i;

23     for(i=0;i<10;i++)
        cifre[i]=0;

25     for(i=0;i<26;i++)
26     {
27         mala[i]=0;
28         velika[i]=0;
29     }

31     while((c = getchar()) != EOF)
```

```

33 {
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
        upravo
        vrednost 0, 1, ..., 9 za karaktere '0', '1', ..., '9' respektivno,
45     a to su upravo indeksi u nizu brojac (jer niz ima 10 elemenata,
        pa su indeksi od 0 do 9). Time postizemo da brojac[0] broji
47     karaktere '0', itd. Isto vazi i za brojace za mala i velika slova.
        */
49 }
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,
            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,
            mala[i]);
61
62 for(i = 0; i < 26; i++)
63     if (velika[i]!=0)
64         printf("Karakter %c se pojavljuje %d puta\n", 'A' + i,
            velika[i]);
65
66 return 0;
67 }

```

### Rešenje 3.5

```

/*
2  Napisati program koji ucitava 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

```

### 3 Predstavljanje podataka

---

```
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");
39     for(i=0;i<n;i++)
40     {
41         printf("b[%d]=",i);
42         scanf("%d", &b[i]);
43     }
44
45     /*
46      Koristimo dva indeksa:
47      1. i, sa kojim pristupamo
48         elementima niza a i b, i koji uvecavamo za 1
         nakon svake iteracije,
49      2. j, sa kojim pristupamo
50         elementima niza c; s obzirom da u svakoj
51         iteraciji dodeljujemo vrednost za dva
52         elementa niza c (c[j] i c[j+1]), indeks
53         j uvecavamo za 2 nakon svake iteracije
54     */
55     for(i=0,j=0;i<n;i++,j+=2)
56     {
```

```

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 3.6

```

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>
#define MAX 100
13
15 int main()
17 {
    int a[MAX];
19
    int n;
    int i,j;
    char poslednja_cifra;
21     int novo_n;

23     printf("Unesi dimenziju niza:");
    scanf("%d", &n);
25

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

33     printf("\nUnesi elemente niza a:\n");
    for(i=0;i<n;i++)
35     {
        printf("a[%d]=",i);
37         scanf("%d", &a[i]);
    }
}

```

### 3 Predstavljanje podataka

```
39 }
41 /*
43  Dodatni indeks j se uvecava u slucaju da element na indeksu
45  i treba da ostane u nizu, tj da je deljiv svojim
47  indeksom i; u suprotnom, j se nece uvecati i
49  element i ce u narednoj iteraciji biti zamenjen elementom koji
51  jeste deljiv svojim indeksom
53  */
55  for(i=0,j=0;i<n;i++)
57  {
59      poslednja_cifra = a[i]%10;
61
63      /*
65          zbog lenjog izracunavanja, ako je prvi uslov
67          u disjunkciji tacan, drugi se nece ispitivati
69          (jer ce tada disjunkcija biti tacna bez obzira
71          da li je drugi uslov tacan ili ne)
73          */
75      if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
77      {
79          a[j]=a[i];
81          j++;
83      }
85  }
87  /*
89      Izbacivanjem elemenata dimenzija niza se menja, odnosno
91      smanjuje se za broj izbacenih elemenata
93      */
95  novo_n=j;
97
99  printf("Nakon izmena:\n");
101  for(i=0;i<novo_n;i++)
103      printf("a [%d]=%d\n",i,a[i]);
105
107  return 0;
109 }
```

#### Rešenje 3.7

```
2  /*
3  a) Napisati funkciju koja ucitava sadrzaj niza.
4  b) Napisati funkciju koja stampa sadrzaj niza.
5  c) Napisati funkciju koja racuna sumu elemenata niza.
6  d) Napisati funkciju koja racuna prosechnu vrednost elemenata niza.
7  e) Napisati funkciju koja izracunava minimum elemenata niza.
8  f) Napisati funkciju koja izracunava poziciju maksimalnog elementa
9  u nizu.
10 g) Napisati program koji testira prethodne funkcije.
```



```
10 */
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) */
55 int minimum (int a[],int n)
56 {
57     int m;
58     int i;
59     m = a[0];
60 }
```

```
62  /*
63     minimum inicijalizujemo na prvi element niza (a[0])
64     u svakom koraku poredimo vrednost minimuma
65     sa jednim elementom niza, iduci redom; s obzirom
66     da je minimum inicijalizovan na a[0], nema potrebe
67     porediti a[0] sa a[0] i zbog toga indeksiranje kreće
68     od 1
69 */
70
71     for(i=1;i<n;i++)
72         if (m>a[i])
73             m = a[i];
74
75     return m;
76 }
77
78 /* f) */
79 int max_pozicija (int a[],int n)
80 {
81     int m;
82     int m_poz;
83     int i;
84     m = a[0];
85     m_poz=0;
86
87     for(i=1;i<n;i++)
88         if (m<a[i])
89         {
90             m = a[i];
91             m_poz=i;
92         }
93     }
94
95     return m_poz;
96 }
97
98
99
100 int main()
101 {
102     int a[MAX];
103     int n;
104     printf("Unesi dimenziju niza:");
105     scanf("%d",&n);
106
107     if (n<1 || n>MAX)
108     {
109         printf("Nekorektan unos\n");
110         return -1;
111     }
112 }
```

```

114     ucitaj(a,n);
115     printf("Ucitani niz:");
116     stampaj(a,n);

118     printf("Suma elemenata niza: %d\n", suma(a,n));
119     printf("Prosečna 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 3.8

```

/*
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      neopadajuće.
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 */
25 #include <stdio.h>
26 #define MAX 100

/* a) */
void ucitaj(int a[], int n)
{
    int i;
    for(i=0;i<n;i++)
    {
        printf("Unesi element na poziciji %d:",i);
        scanf("%d",&a[i]);
    }
}

```

### 3 Predstavljanje podataka

---

```
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
41 /* c) */
42 int sadrzi(int a[], int n, int m)
43 {
44     int i;
45     /*
46      poredimo jedan po jedan element niza a sa datim m; ukoliko
47      ustanovimo jednakost, to znaci da niz sadrzi element jednak
48      m i vracamo 1
49     */
50     for(i=0;i<n;i++)
51         if (a[i]==m)
52             return 1;
53
54     /*
55      ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
56      ispunjen,
57      to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
58      vrati 0
59     */
60     return 0;
61 }
62
63 /* d) */
64 int prvo_pojavljivanje(int a[], int n, int m)
65 {
66     int i;
67     /*
68      poredimo jedan po jedan element niza a sa datim m; ukoliko
69      ustanovimo jednakost, vracamo indeks elementa niza a koji
70      je jednak sa m
71     */
72     for(i=0;i<n;i++)
73         if (a[i]==m)
74             return i;
75
76     /*
77      ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
78      ispunjen,
79      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     int i,j;

100
    /*
102     uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
104     uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
106     .
108     i tako redom dok je prva pozicija manja od druge
    */
110     for(i=0, j=n-1; i<j; i++, j--)
        if(a[i]!=a[j])
            return 0;

112     return 1;
}

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

120
    /*
122     Funkcija neopadajuci proverava da li je dati niz sortiran
        neopadajuće i vraća
        1 ako jeste, a 0 u suprotnom

124     Sortiranost proveravamo na sledeci 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]
```

### 3 Predstavljanje podataka

---

```
126     proveravamo
    da li vazí da je drugi član para manji od prvog. Ako naídjemo na
    par za koji
    to ne vazí, 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 znači
    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;
135
136     return 1;
137 }
138
139 /* 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
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++;          /* uvećavamo dužinu 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 dužine
160              */
161             if(i==n-2)
162             {
163                 j++;
164                 if(j>max_dužina)
165                     max_dužina=j;
166             }
167         }
168         else
169         {
170             /*
171              izašli smo iz konstantnog intervala
172
173              ukoliko smo imali bar dva elementa u konstantnom
```

```
intervalu,
172     vrednost promenljive j ce biti 1, a duzina tog intervala
    je 2;
    zbog toga je neophodno takve (pozitivne) j uvecati za 1;
174
    sa druge strane, ako su a[i] i a[i+1] razliciti,
176     duzina tog intervala je 0
    */
178
    if (j>0)
180         j++;

    /* azuriramo maksimalnu duzinu uspona */
    if(j>max_duzina)
184         max_duzina=j;
    /*
186         duzina uspona se postavlja na nulu
        kako bi mogli da je iskoristimo
188         za naredni uspon
    */
190     j=0;

192 }

194 }

196     return max_duzina;
198 }

200 int main()
202 {
    int a[MAX];
204     int n;
    int m;
206     int i;

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

210     if (n<1 || n>MAX)
    {
212         printf("Nekorektan unos\n");
214         return -1;
    }

216     ucitaj(a,n);
218     printf("Ucitani niz:");
    stampaj(a,n);

220     printf("Unesi jedan ceo broj:");
```

### 3 Predstavljanje podataka

```
222     scanf("%d",&m);
224
226     if(sadrzi(a,n,m))
227         printf("Niz sadrzi element cija je vrednost %d\n", m);
228     else
229         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
230
231     i = prvo_pojavljivanje(a,n,m);
232     if(i!=-1)
233         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
234         prvog pojavljivanja u nizu je %d\n", m,i);
235     else
236         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
237
238     i = poslednje_pojavljivanje(a,n,m);
239     if(i!=-1)
240         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
241         poslednjeg pojavljivanja u nizu je %d\n", m,i);
242     else
243         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
244
245     if(palindrom(a,n))
246         printf("Elementi niza cine palindrom\n");
247     else
248         printf("Elementi niza ne cine palindrom\n");
249
250     if(neopadajuci(a,n))
251         printf("Niz je sortiran neopadajuće\n");
252     else
253         printf("Niz nije sortiran neopadajuće\n");
254
255     printf("Duzina najduzeg konstantnog intervala: %d\n",
256           najduza_konstanta(a,n));
257
258     return 0;
259 }
```

### Rešenje 3.9

```
1  /*
2
3  a) Napisati funkciju koja učitava sadržaj niza.
4  b) Napisati funkciju koja stampa sadržaj niza.
5  c) Napisati funkciju koja proverava da li niz sadrzi neku vrednost
   m.
6  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
21     /* a) */
      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     }
31
33     /* b) */
      void stampaj(int a[], int n)
      {
          int i;
          for(i=0; i<n; i++)
35              printf("%d\t", a[i]);
          printf("\n");
37     }
39
41     /* c) */
      int sadrzi(int a[], int n, int m)
43     {
          int i;
          /*
45             poredimo jedan po jedan element niza a sa datim m; ukoliko
              ustanovimo jednakost, to znaci da niz sadrzi element jednak
              m i vracamo 1
47             */
          for(i=0; i<n; i++)
49              if (a[i]==m)
51                  return 1;
      }

```

### 3 Predstavljanje podataka

---

```
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, vraćamo 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     /*
        krećemo 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) */
int palindrom(int a[], int n)
97 {
99     int i, j;
```

```
101  /*
102     uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
103     uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
104     .
105     .
106     i tako redom dok je prva pozicija manja od druge
107  */
108  for(i=0,j=n-1;i<j;i++,j--)
109      if(a[i]!=a[j])
110          return 0;
111
112  return 1;
113 }
114
115 /* g) */
116 int neopadajuci(int a[], int n)
117 {
118     int i;
119
120     /*
121     Funkcija neopadajuci proverava da li je dati niz sortiran
122     neopadajuće i vraća
123     1 ako jeste, a 0 u suprotnom
124
125     Sortiranost proveravamo na sledeci nacin: za svaki par susednih
126     elemenata
127     a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
128     proveravamo
129     da li vazi da je drugi clan para manji od prvog. Ako naidjemo na
130     par za koji
131     to ne vazi, niz nije sortiran i funkcija vraća 0. Ukoliko se
132     petlja završi
133     a da pritom uslov a[i]<a[i-1] nije nijednom bio ispunjen, to znaci
134     da je
135     niz sortiran i funkcija vraća 1
136
137     */
138     for(i=1; i<n; i++)
139         if (a[i]<a[i-1])
140             return 0;
141
142     return 1;
143 }
144
145 /* h) */
146 int najduza_konstanta(int a[], int n)
147 {
148     int i; /* indeks niza */
149     int j; /* duzina intervala */
150     int duzina;
151     int max_duzina=0;
```

```
147 for(i=0,j=0;i<n-1;i++)
149 {
151     if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
153     {
155         j++;          /* uvecavamo duzinu konstantnog intervala */
157         /*
158         ako se niz zavrшава konstantnim intervalom (nalazimo se u
159         poslednjoj
160         iteraciji petlje i tada je i==n-2), ispitujemo da li je
161         taj konstantni
162         interval maksimalne duzine
163         */
164         if(i==n-2)
165         {
166             j++;
167             if(j>max_duzina)
168                 max_duzina=j;
169         }
170     }
171     else
172     {
173         /*
174         izašli smo iz konstantnog intervala
175
176         ukoliko smo imali bar dva elementa u konstantnom
177         intervalu,
178         vrednost promenljive j ce biti 1, a duzina tog intervala
179         je 2;
180         zbog toga je neophodno takve (pozitivne) j uvecati za 1;
181
182         sa druge strane, ako su a[i] i a[i+1] razliciti,
183         duzina tog intervala je 0
184         */
185         if (j>0)
186             j++;
187
188         /* azuriramo maksimalnu duzinu uspona */
189         if(j>max_duzina)
190             max_duzina=j;
191         /*
192         duzina uspona se postavlja na nulu
193         kako bi mogli da je iskoristimo
194         za naredni uspon
195         */
196         j=0;
197     }
198 }
```

```
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     učitaj(a,n);
218     printf("Učitani niz:");
219     stampaj(a,n);
220
221     printf("Unesi jedan ceo broj:");
222     scanf("%d",&m);
223
224     if(sadrzi(a,n,m))
225         printf("Niz sadrzi element cija je vrednost %d\n", m);
226     else
227         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
228
229     i = prvo_pojavljivanje(a,n,m);
230     if(i!=-1)
231         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog prvog pojavljivanja u nizu je %d\n", m,i);
232     else
233         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
234
235
236     i = poslednje_pojavljivanje(a,n,m);
237     if(i!=-1)
238         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog poslednjeg pojavljivanja u nizu je %d\n", m,i);
239     else
240         printf("Niz ne sadrzi element cija je vrednost %d\n", m);
241
242     if(palindrom(a,n))
243         printf("Elementi niza cine palindrom\n");
```

### 3 Predstavljanje podataka

```
245     else
246         printf("Elementi niza ne cine palindrom\n");
247
248     if(neopadajuci(a,n))
249         printf("Niz je sortiran neopadajuce\n");
250     else
251         printf("Niz nije sortiran neopadajuce\n");
252
253     printf("Duzina najduzeg konstantnog intervala: %d\n",
254           najduza_konstanta(a,n));
255
256     return 0;
257 }
```

#### Rešenje 3.10

```
1  /*
2   a) Napisati funkciju koja sve vrednosti niza uvecava za vrednost m.
3   b) Napisati funkciju koja obrce 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)
```

```
{
35     int i;
36     for(i=0; i<n; i++)
37         a[i] += m;
38 }
39
40
41 void obrni(int a[], int n)
42 {
43     int t;
44     int i, j;
45     /*
46      Niz obrnemo tako sto razmenimo vrednosti elemenata na pozicijama
47      0 i n-1,
48      zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
49      druge
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
71 void rotirajk(int a[], int n, int k)
72 {
73     int i;
74     /*
75      k puta rotiramo niz za jednu poziciju
76      ulevo
77     */
78     for(i=0; i<k; i++)
79         rotiraj1(a, n);
80 }
81
82 int main()
83 {
```

### 3 Predstavljanje podataka

---

```

    int a[MAX];
85  int n;
    int i;
87  int k;
    int m;

89
    printf("Unesi dimenziju niza:");
91  scanf("%d",&n);

93  if (n<1 || n>MAX)
    {
95      printf("Nekorektan unos\n");
        return -1;
97  }

99  učitaj(a,n);

101  printf("Unesi jedan ceo broj:");
    scanf("%d", &m);
103
    uvecaj(a,n,m);
105  printf("Elementi niza nakon uvecanja za %d:\n",m);
    stampaj(a,n);
107
    obrni(a,n);
109  printf("Elementi niza nakon obrtanja:\n");
    stampaj(a,n);
111
    printf("Unesi jedan pozitivan ceo broj:");
113  scanf("%d",&k);

115  if (k<=0)
    {
117      printf("Nekorektan unos\n");
        return -1;
119  }

121  rotiraj1(a,n);
    printf("Elementi niza nakon rotiranja za 1 mesto ulevo:\n");
123  stampaj(a,n);

125  rotirajk(a,n,k);
    printf("Elementi niza nakon rotiranja za %d mesto ulevo:\n",k);
127  stampaj(a,n);

129  return 0;
}
```

#### Rešenje 3.37



Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje 3.37

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

Rešenje [3.37](#)

### 3.3 Pokazivači

Zadatak 3.38    Tekst

[Rešenje [3.56](#)]

Zadatak 3.39    Tekst

[Rešenje [3.39](#)]

Zadatak 3.40    Tekst

[Rešenje [3.40](#)]

Zadatak 3.41    Tekst

[Rešenje [3.41](#)]

Zadatak 3.42    Tekst

[Rešenje [3.42](#)]

Zadatak 3.43    Tekst

[Rešenje [3.43](#)]

**Zadatak 3.44** 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
```

[Rešenje 3.52]

**Zadatak 3.45** Napisati program koji ispisuje argumente komandne linije koji počinju slovom *z*.

*Primer 1*

```
|| POKRETANJE: ./a.out zima jabuka zvezda Zrak
|| INTERAKCIJA SA PROGRAMOM:
|| zima zvezda
```

*Primer 2*

```
|| POKRETANJE: ./a.out bundeva pomorandza
|| INTERAKCIJA SA PROGRAMOM:
```

*Primer 3*

```
|| POKRETANJE: ./a.out sanke zapad zujanje
|| INTERAKCIJA SA PROGRAMOM:
|| zapad zujanje
```

*Primer 4*

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

[Rešenje 3.52]

**Zadatak 3.46** Napisati program koji ispisuje broj argumenata komandne linije koji sadrže slovo *z*.

### 3 Predstavljanje podataka

---

#### 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
```

[Rešenje 3.52]

**Zadatak 3.47** Napisati program koji na osnovu broja  $n$  koji se zadaje kao argument komandne linije ispisuje cele brojeve iz intervala  $[-n, n]$ .

#### Primer 1

```
|| POKRETANJE: ./a.out 2
|| INTERAKCIJA SA PROGRAMOM:
|| -2 -1 0 1 2
```

#### Primer 2

```
|| POKRETANJE: ./a.out 4
|| INTERAKCIJA SA PROGRAMOM:
|| -4 -3 -2 -1 0 1 2 3 4
```

#### Primer 3

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

#### Primer 4

```
|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| Greska: nedostaje argument komandne linije!
```

[Rešenje 3.52]

**Zadatak 3.48** Napisati program koji proverava da li se među zadatim argumentima komandne linije nalaze barem dva ista.

#### Primer 1

```
|| POKRETANJE: ./a.out pec zima deda mraz pec
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima ima istih.
```

#### Primer 2

```
|| POKRETANJE: ./a.out xyz abc abc abc efgh
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima ima istih.
```

## Primer 3

```

|| POKRETANJE: ./a.out 11 15 abc 888
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima nema istih.

```

## Primer 4

```

|| POKRETANJE: ./a.out
|| INTERAKCIJA SA PROGRAMOM:
|| Medju argumentima nema istih.

```

[Rešenje 3.52]

**Zadatak 3.49** 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

```

[Rešenje 3.52]

**Zadatak 3.50** 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

```

### 3 Predstavljanje podataka

---

#### Primer 2

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

[Rešenje 3.52]

**Zadatak 3.51** 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
```

[Rešenje 3.52]

**Zadatak 3.52** 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

```

[Rešenje 3.52]

**Zadatak 3.53** Tekst

[Rešenje 3.56]

**Zadatak 3.54** Ako su celi brojevi  $a$  i  $b$  argumenti komandne linije napraviti niz  $A[0] = a$ ,  $A[1] = a+1$ ,  $A[2] = a+2$ , ...,  $A[b-a] = b$  i ispisati ga. Pretpostaviti da je maksimalna dužina niza 200 elemenata. Proveriti da li  $a < b$  i  $b - a < 200$  i ako ovi uslovi nisu ispunjeni ispisati poruku da je došlo do greške. U slučaju da je dato manje ili više argumenata komandne linije ispisati poruku o grešci.

*Primer 1*

```

|| POKRETANJE: ./a.out 34
|| INTERAKCIJA SA PROGRAMOM:
|| greska

```

*Primer 2*

```

|| POKRETANJE: ./a.out 12 20
|| INTERAKCIJA SA PROGRAMOM:
|| 12 13 14 15 16 17 18 19 20

```

*Primer 3*

```

|| POKRETANJE: ./a.out 30 8
|| INTERAKCIJA SA PROGRAMOM:
|| greska

```

*Primer 4*

```

|| POKRETANJE: ./a.out -4 -1
|| INTERAKCIJA SA PROGRAMOM:
|| -4 -3 -2 -1

```

[Rešenje 3.56]

**Zadatak 3.55** Uobičajena praksa na UNIX sistemima je da se argumenti komandne linije dele na opcije i argumente u užem smislu. Opcije počinju znakom '-' nakon čega obično sledi jedan ili više karaktera koji označavaju koja je opcija u pitanju. Ovim se najčešće upravlja funkcionisanjem programa i neke mogućnosti se uključuju ili isključuju. Argumenti načšće predstavljaju opisne informacije poput na primer imena datoteka. Napisati program koji ispisuje sve opcije koje su navedene u komandnoj liniji.

#### Primer 1

```
POKRETANJE: ./a.out -abc input.txt -d -Fg output
INTERAKCIJA SA PROGRAMOM:
a b c d F g
```

#### Primer 2

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

#### Primer 3

```
POKRETANJE: ./a.out ulaz.txt
INTERAKCIJA SA PROGRAMOM:
```

[Rešenje 3.56]

**Zadatak 3.56** Parametri komandne linije su  $n$ ,  $a$ ,  $b$  ( $a < b$ ). Treba popuniti prvih  $n$  elemenata niza  $A$  celim slučajnim brojevima koji su između  $a$  i  $b$ . Ištampati niz  $A$  na standardni izlaz. Maksimalan broj elemenata niza  $A$  je 200. Ukoliko nisu zadati svi argumenti komandne linije ili ne zadovoljavaju potrebna svojstva ispisati poruku o grešci.

[Rešenje 3.56]

## 3.4 Rešenja

### Rešenje 3.56

```
1  /*
2  Napisati funkciju uredi koja uredjuje svoja dva
   celobrojna argumenta tako da se u prvom nalazi manji
4  a u drugom veci. Napisati potom glavni program koji
   ucitava dva cela broja i uredjuje njihove vrednosti
6  primenom napisane funkcije. Na primer, ako su ucitane
   promenljive x=5 i y=2, njihove vrednosti nakon
8  primene funkcije uredi treba da budu x=2 i y=5.
   */
10 #include <stdio.h>
12
13 /*
14 Argumenti funkcije uredi_pogresno, promenljive a i b,
   predstavljaju lokalne promenljive za ovu funkciju
16 i prestaju da postoje po zavrsetku funkcije. Zbog toga
   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.
   */
```



```
20 void uredi_pogresno(int a, int b)
21 {
22     int t;
23
24     if (a>b)
25     {
26         t = a;
27         a = b;
28         b = t;
29     }
30     printf("uredi_pogresno :: a=%d, b=%d\n", a, b);
31     printf("uredi_pogresno :: &a=%p, &b=%p\n", &a, &b);
32 }
33
34 /*
35  Argumenti funkcije uredi_tacno, promenljive pa i pb,
36  takodje su lokalne promenljive za ovu funkciju i
37  prestaju da postoje kada se funkcija završi.
38  Njima prosledjujemo adrese promenljivih a i b koje zelimo
39  da razmenimo u slucaju da je a>b.
40
41  Promenljivoj a pristupamo preko pokazivacke promenljive
42  pa sa *pa i slicno, promenljivoj pb pristupamo sa *pb.
43
44  Vrednosti promenljivih *pa i *pb razmenjujemo kao
45  i vrednosti bilo koje dve celobrojne promenljive.
46
47  */
48 void uredi_tacno(int * pa, int * pb)
49 {
50     int t;
51     if (*pa>*pb)
52     {
53         t = *pa;
54         *pa = *pb;
55         *pb = t;
56     }
57     printf("uredi_tacno :: *pa=%d, *pb=%d\n ", *pa, *pb);
58     printf("uredi_tacno :: pa=%p, pb=%p\n ", pa, pb);
59 }
60
61 int main()
62 {
63     int a,b;
64
65     printf("Unesi dve celobrojne promenljive:");
66     scanf("%d%d",&a,&b);
67
68     printf("main :: a=%d, b=%d\n", a,b);
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);
```

### 3 Predstavljanje podataka

---

```
72  /*
74      Funkcija uredi_tacno kao argument ima dve pokazivacke
      promenljive
      (int*,int*). Zbog toga joj je u pozivu funkcije neophodno
      proslediti
76      adrese promenljivih koje zelimo da uredimo rastuce, &a i &b.
      */
78
      uredi_tacno(&a, &b);
80      printf("main :: nakon uredi_tacno, a=%d, b=%d\n", a, b);
82
      return 0;
}
```

#### Rešenje 3.39

```
1  /*
   Napisati funkciju koja za boju datu u rgb formatu
   racuna cmy format po formulama:
3      C = 1 - ( R / 255 )
   M = 1 - ( G / 255 )
5      Y = 1 - ( B / 255 )
7
   Napisati program koji ucitava boju u rgb formatu,
   primenjuje odgovarajucu funkciju i ispisuje boju u cmy formatu.
9
11 */
13 #include <stdio.h>
   #include <math.h>
15
   void rgb_to_cmy(float* a, float* b, float* c)
17 {
   /* Zagrade su neophodne jer aritmeticke operacije
   imaju veci prioritet od operatora dereferenciranja (*).
   */
21     *a=1-(*a)/255;
     *b=1-(*b)/255;
23     *c=1-(*c)/255;
25
   /*
   Pomocu return ne mozemo vratiti vise od jedne vrednosti.
27
   Ceste greske:
29     return a,b,c;           return vraca samo jednu vrednost
     return a; return b; return c; return ce vratiti samo a
31
   Zato je neophodno da promenljive ciju vrednost
33     zelimo da promenimo prenesemo preko pokazivaca.
   */
}
```

```

35 }
37
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
66     printf("Nakon konverzije: %.2f,%.2f,%.2f\n", a,b,c);
67
68     return 0;
69 }

```

### Rešenje 3.40

```

1 /*
2  Napisati funkciju koja za dve prave date svojim koeficijentima
3  pravca i slobodnim clanovima odredjuje njihovu tacku preseka.
4  Funkcija treba da vrati 1 ako se prave seku i 0 ako nemaju
5  tacku preseka (ako su paralelne). Napisati glavni program
6  koji ucitava podatke o pravama, poziva napisanu funkciju i
7  ispisuje odgovarajucu poruku.
8  */
9
10 #include<stdio.h>
11
12 /*

```

### 3 Predstavljanje podataka

```
14      Funkcija presek treba da izracuna tri vrednosti:
15      1. indikator da li su koeficijenti pravca jednaki ili ne
16      2. prvu koordinatu preseccne tacke (ukoliko prave nisu paralelne)
17      3. drugu koordinatu preseccne tacke (ukoliko prave nisu paralelne)
18
19      Indikator funkcija vraca kao povratnu vrednost, preko kljucne reci
20      return.
21
22      Koordinate preseccne tacke (ako postoji) funkcija vraca preko
23      liste argumenata, zbog cega promenljive kojima ce koordinate
24      biti dodeljene prenosimo preko pokazivaca (promenljive px i py)
25
26      Promenljive koje sadrze podatke o pravama (k1,n1,k2,n2) se ne
27      menjaju u funkciji i zbog toga ih ne moramo prenositi preko
28      pokazivaca.
29  */
30  int presek(float k1, float n1, float k2, float n2, float* px, float*
31      py)
32  {
33      if (k1==k2)
34          return 0;
35
36      *px = -(n1-n2)/(k1-k2);
37      *py = k1*(px)+n1;
38      return 1;
39  }
40
41  int main()
42  {
43      float k1,k2,n1,n2;
44      float x,y;
45
46      printf("Unesi k i n za prvu pravu:");
47      scanf("%f%f",&k1,&n1);
48
49      printf("Unesi k i n za drugu pravu:");
50      scanf("%f%f",&k2,&n2);
51
52      if(presek(k1,n1,k2,n2,&x,&y))
53          printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
54      else
55          printf("Prave su paralelne\n");
56
57      return 0;
58  }
```

#### Rešenje 3.41

```
1  /*
```

```

    Napisati program koji ispisuje broj navedenih argumenata komandne
    linije,
3   a zatim i same argumenate 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
27      return 0;
    }

```

### Rešenje 3.42

```

1   /*
    Napisati funkciju koja za dva data stringa str i
3   accept odredjuje koliko se uzastopnih karaktera stringa str
    nalazi u stringu accept pocev od pocetka niza str. Napisati
5   potom program koji testira napisanu funkciju za dva stringa
    koji se unose kao argumenti komandne linije. Primeri upotrebe:
7
    1:
9   ./a.out aladin bal
    3
11
    2:
13  ./a.out aladin lad
    4
15
    3:
17  ./a.out Aladin ala
    0
19
21  */

```

### 3 Predstavljanje podataka

---

```
23 #include <stdio.h>
24 #include <string.h>
25
26 /*
27 Funkcija strspn(str,accept) je ugradjena funkcija koja vraca broj
28 karaktera
29 stringa str koji se nalaze u stringu accept, pocev od pocetka
30 stringa str.
31
32 Funkcija strspn se nalazi u zaglavlju string.h.
33
34 Funkcija strspn_klon je jedna implementacija funkcije strspn.
35
36 U zadacima cemo uvek koristiti ugradjenu funkciju strspn osim ako
37 u tekstu zadatka
38 nije naglaseno da se ona ne sme koristiti. Funkcija strspn_klon
39 sluzi da pokaze na koji
40 nacin radi ugradjena funkcija strspn.
41
42 Ugradjena funkcija strspn poziva se na isti nacin kao funkcija
43 strspn_klon:
44 strspn(s1,s2)
45
46 */
47
48 int strspn_klon(char str[], char accept[])
49 {
50     int br=0;
51     int i;
52
53     for(i=0; str[i];i++)
54         if(strchr(accept, str[i])!=NULL)
55             br++;
56     else /* ako pronadjemo karakter u stringu str koji nije */
57         break; /* u stringu accept, prekidamo petlju */
58
59     return br;
60 }
61
62 int main(int argc, char* argv[])
63 {
64     int br;
65
66     if(argc<3)
67     {
68         printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
69         arg2\n");
70         return -1;
71     }
72
73     br = strspn_klon(argv[1],argv[2]);
```

```

printf("Broj karaktera stringa %s koji se nalaze u stringu %s,
    pocev od pocetka stringa %s: %d\n", argv[1], argv[2], argv[1], br);
69 return 0;
}

```

### Rešenje 3.43

```

/*
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 1:
    ./a.out abcd 2
20 cdef

22 2:
    ./a.out abcd 2 -p D
24 cdef

26 3:
    ./a.out abcd 2 -p L
28 yzab

30 4:
    ./a.out abcd -3 -p L
32 Nekorektan unos

34 5:
    ./a.out abcd 3 -p X
36 Nekorektan unos

```

```
38 6:
39 ./a.out ab12cd 2 -p D
40 cd12ef
41
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;
67     if (c=='L')
68         znak = -1;
69
70     for(i=0; s[i];i++)
71         if(isalpha(s[i]))
72         {
73             /*
74              Promenljiva t predstavlja sifrirani karakter s[i].
75              Ako je promenljiva t izvan opsega malih ili velikih slova
76              ,
77              dodajemo joj ili oduzimamo ukupan broj slova u abecedi
78              (26),
79              u zavisnosti od pravca sifriranja, kako bismo omogucili
80              kruzno sifriranje.
81              */
82             t = s[i]+znak*k;
83             if(((islower(s[i]) && (t<'a' || t>'z')) || (isupper(s[i]) &&
84                (t<'A' || t>'Z'))))
85                 s[i]=t-znak*26;
86             else
87                 s[i]=t;
```



```
86     }
87 }
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 */

```

### 3 Predstavljanje podataka

```
138 { /* korektnost unosa ispitana, a unos je korektan
    jedino za argc==3 ili argc==5 */
139     /*
    Ispitujemo korektnost pretposlednjeg argumenta koji mora da
    bude u formatu "-p".
140     Ovaj argument je string argv[3]. Njegovom prvom karakteru (
    koji treba
    da bude '-') pristupamo sa argv[3][0] a drugom sa argv
    [3][1].
142     */
    if (argv[3][0] != '-')
144     {
        printf("Nekorektan unos: pri zadavanju opcija prvi karakter
        mora biti '-' \n");
146         return -1;
    }

148     if (argv[3][1] != 'p')
150     {
        printf("Nekorektan unos: nedozvoljena opcija\n");
152         return -1;
    }

154     /*
156     Nakon argumenta -p sledi argument koji zadaje vrednost ove
    opcije. To je
    poslednji argument kome pristupamo sa argv[4]. Ovaj argument
    treba
158     da sadrzi samo jedan karakter - 'L' ili 'D' i njemu
    pristupamo sa
    argv[4][0].
160     */
    if(argv[4][0] == 'L' || argv[4][0] == 'D')
162        pravic=argv[4][0];
    else
164    {
        printf("Nekorektan unos: pravic moze biti L ili D\n");
166        return -1;
    }

168 }

170 strcpy(rec, argv[1]);
    sifruj(rec, pravic, k);

172     printf("Sifrovana rec: %s\n", rec);

174     return 0;
176 }
```

#### Rešenje 3.52

Rešenje 3.52

Rešenje 3.52

Rešenje 3.52

Rešenje 3.52

Rešenje 3.52

Rešenje 3.52

Rešenje 3.52

Rešenje 3.52

Rešenje 3.56

```
1  #include <stdio.h>
2
3  void suma(int a, int b, int *s);
4
5
6  int main()
7  {
8      int a,b,s;
9
10     scanf("%d%d",&a,&b);
11
12     suma(a,b,&s);
13
14     printf("suma: %d\n",s);
15
16     return 0;
17 }
18
19 void suma(int a, int b, int *s)
20 {
21     *s = a + b;
22 }
```

Rešenje 3.56

```
1 #include <stdio.h>

3 void suma(int a, int b, int *s);

5

7 int main()
8 {
9     int a,b,s;
10
11     scanf("%d%d",&a,&b);
12
13     suma(a,b,&s);
14
15     printf("suma: %d\n",s);
16
17     return 0;
18 }

19 void suma(int a, int b, int *s)
20 {
21     *s = a + b;
22 }
```

#### Rešenje 3.56

```
1 #include <stdio.h>

3 void suma(int a, int b, int *s);

5

7 int main()
8 {
9     int a,b,s;
10
11     scanf("%d%d",&a,&b);
12
13     suma(a,b,&s);
14
15     printf("suma: %d\n",s);
16
17     return 0;
18 }

19 void suma(int a, int b, int *s)
20 {
21     *s = a + b;
22 }
```

## Rešenje 3.56

```
1  #include <stdio.h>
2
3  void suma(int a, int b, int *s);
4
5
6  int main()
7  {
8      int a,b,s;
9
10     scanf("%d%d",&a,&b);
11
12     suma(a,b,&s);
13
14     printf("suma: %d\n",s);
15
16     return 0;
17 }
18
19 void suma(int a, int b, int *s)
20 {
21     *s = a + b;
22 }
```

## 3.5 Niske

Zadatak 3.57 Tekst

[\[Rešenje 3.57\]](#)

Zadatak 3.58 Tekst

[\[Rešenje 3.58\]](#)

Zadatak 3.59 Tekst

[\[Rešenje 3.59\]](#)

Zadatak 3.60 Tekst

[\[Rešenje 3.60\]](#)

Zadatak 3.61 Tekst

[\[Rešenje 3.61\]](#)

Zadatak 3.62 Tekst

[Rešenje [3.62](#)]

#### Zadatak 3.63 Tekst

[Rešenje [3.63](#)]

#### Zadatak 3.64 Tekst

[Rešenje [3.64](#)]

#### Zadatak 3.65

- 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.
- Napisati funkciju *int samoglasnik\_na\_kraju(char s[])* koja proverava da li se niska *s* završava samoglasnikom (koristiti funkciju iz tačke a)).
- 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!
```

[Rešenje [3.91](#)]

**Zadatak 3.66** 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

```

[Rešenje 3.91]

**Zadatak 3.67** 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

```

[Rešenje 3.91]

**Zadatak 3.68** 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: 0xErA9
|| Nekorektan heksadekadni broj!

```

[Rešenje 3.91]

### 3 Predstavljanje podataka

---

**Zadatak 3.69** Napisati funkciju *int heksa\_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
```

[Rešenje 3.91]

**Zadatak 3.70** 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!
```

[Rešenje 3.91]

**Zadatak 3.71** 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*

```

[Rešenje 3.91]

**Zadatak 3.72** 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: aaioo124
|| Unesite nisku s: aeioo
|| 6

```

*Primer 3*

```

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

```

[Rešenje 3.91]

**Zadatak 3.73** 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!

```

[Rešenje 3.91]

**Zadatak 3.74**

- Napisati funkciju

### 3 Predstavljanje podataka

---

```
int prepis(char a[] [21], int na, char b[] [21])
```

koja iz niza reči **a** dužine **na** prepisuje u niz **b** reči koje su zapisane samo malim ili samo velikim slovima. Informaciju o dužini niza **b** (broj reči koje zadovoljavaju prethodni uslov) vratiti kao povratnu vrednost funkcije.

- Napisati program koji sa standardnog ulaza učitava prvo broj reči (strogo veći od nule, manji od 50), a zatim i same reči razdvojene blanko znakom (smatrati da reči koje se unose sa ulaza neće biti duže od 20 karaktera - ovaj uslov ne proveravati). Za slučaj kada je broj reči izvan traženog opsega ispisati -1 i prekinuti izvršavanje programa. Korišćenjem prethodno definisane funkcije **prepis**, odrediti sve reči koje su zapisane samo malim ili samo velikim slovima. Rezultat ispisati na standardni izlaz. Napomena: Ukoliko se pri rešavanju zadatka ne bude koristila funkcija **prepis**, zadatak neće biti pregledan i nosiće nula poena.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 3 abc ABC aBc  
|| abc ABC
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 2 mmB RGa
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| -3  
|| -1
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 4 2abc AVF$ abc AV4  
|| abc
```

[Rešenje 3.91]

**Zadatak 3.75** Napisati funkciju `void min_razlika(char s[], char s1[], char s2[])` koja u datotoj nisci **s** pronalazi dve reči koje imaju minimalnu razliku između svojih samoglasnika. ( Reč je niz karaktera između dve praznine; razmak između samoglasnika reči **dan**as i **jut**ro je 2, a razmak između **sut**rk i **mno**zenje je 5). Testirati pozivom u **main**-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.76** Napisati funkciju `int pp(char s[], char t[])` koja određuje poziciju poslednjeg karaktera niske **s** sadržanog u okviru niske **t**, zanemarujući pri tom razliku između velikih i malih slova, ili -1 ako takvog karaktera nema. Testirati pozivom u **main**-u. Maksimalna dužina niske je 20 karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| a4BA3Bc A3b
|| 5
```

[Rešenje 3.91]

**Zadatak 3.77** Napisati funkciju `int f1(char s[])` koja prihvata tu nisku i proverava da li niska sadrži veliko slovo. Funkcija vraća 1 ako sadrži veliko slovo, inače 0. Testirati pozivom u main-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.78** Napisati funkciju `void ukloniSlova(char s[])` koja iz niske `s` uklanja sva mala i velika slova. Testirati pozivom u main-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.79** Napisati funkciju `unsigned btoi(char* s, unsigned char b)` koja određuje vrednost zapisa datog neoznačenog broja `s` u datoj osnovi `b`. Napisati funkciju `void itob(unsigned n, unsigned char b, char* s)` koja datu vrednost `n` zapisuje u datoj osnovi `b` i smešta rezultat u nisku `s`. Napisati zatim program koji čita liniju po liniju sa standardnog ulaza i obrađuje ih sve dok ne naiđe na praznu liniju. Svaka linija sadrži jedan dekadni, oktalni ili heksadekadni broj (zapisan kako se zapisuju konstante u programskom jeziku C). Program za svaki uneti broj ispisuje njegov binarni zapis. Pretpostaviti da će svi uneti brojevi biti u opsegu tipa `unsigned`.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 0x49 0x1ABC
|| 1001001 11010101111100
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 012 435 0x64FE
|| 1010 110110011 110010011111110
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 123 0777
|| 1111011 111111111
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| 981
|| 1111010101
```

[Rešenje 3.91]

**Zadatak 3.80** Implementirati funkciju `int str_str(char s[], char t[])` koja proverava da li niska `s` sadrži nisku `t`. Zatim napisati program koji sa

### 3 Predstavljanje podataka

---

standardnog ulaza učitava pet redova (svaki red ima najviše 100 karaktera) i koji ispisuje sve redne brojeve linija koje sadrže nisku **program** (linije se numerišu od broja 1). Ukoliko ne postoji red sa niskom **program** ispisati -1.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
novi red*program
c prog. jezik
c? programskih jezik
Programski odbor
<b>program</b>
1 3 5
```

[Rešenje 3.91]

**Zadatak 3.81** Napisati funkciju `void sifrat(char* rec, char* kljuc)` koja šifruje `rec` na sledeći način: za svako slovo reči `rec` i odgovarajuće slovo ključa određuje koliki je (alfabetski) razmak između njih i označimo taj broj sa `k`. Potom to slovo `reci` zamenjuje `k`-tim slovom alfabeta. Podrazumeva se da je ključ duži od reči.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
bac
dfge
bed
```

[Rešenje 3.91]

**Zadatak 3.82** Napisati funkciju `void obrni(char rec[], int k)` koja rotira reč za `k` mesta ulevo.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
sveska
2
eskasv
```

[Rešenje 3.91]

**Zadatak 3.83** Napisati sledeće funkcije:

```
int poredjenje(char* s1, char* s2);
// vraca 1 ako su s1 i s2 iste niske, 0 u suprotnom

void uVelikaSlova(char* s);
// pretvara sva slova niske s u velika, ostale znakove ne menja
```

Napisati program koji sa standardnog ulaza učitava dve reči (dužine najviše 20 znakova) i, koristeći ove dve funkcije, ispisuje da li su one jednake ako se sva slova pretvore u velika slova.

#### Primer 1

INTERAKCIJA SA PROGRAMOM:

```
isPit2010
IsPit2010
jesu jednake
```

[Rešenje 3.91]

**Zadatak 3.84** Napisati program kojim se sadržaj unetog stirnga šifrira tako što se svako slovo zamenjuje sledećim ASCII slovom, a znakovi 'z' i 'Z' zamenjuju redom sa 'a' i 'A'. Uneta reč nije duža od 20 karaktera.

[Rešenje 3.91]

**Zadatak 3.85** Napisati funkciju `void sifruj(char rec[], char sifra[])` koja na osnovu date reči formira šifru koja se dobija tako što se svako slovo u reči zameni sa naredna tri slova koja su mu susedna u abecedi. Na primer, reč "tamo" treba da bude zamenjena sa "uvwbcnoppqr" a reč "zec" sa "abcfghdef". Napisati program koji šifrue unetu reč sa standardnog ulaza i štampa dobijeni rezultat na standardni izlaz. Za reč pretpostaviti da nije duža od 20 karaktera. Unos reči ostvariti koristeći specifikator "

[Rešenje 3.91]

**Zadatak 3.86** Sa ulaza se unosi reč koja nije duža od 20 znakova. Napisati program koji formira i štampa rezultujuću reč koja se dobija tako što se uneta reč kopira 4 puta pri čemu se između svakog kopiranja umeće crtica. Na primer ako je uneta reč **ana**,formirana reč treba da bude **ana-ana-ana-ana**. Zadatak uraditi:

- (a) pisanjem odgovarajuće funkcije koja vrši nadovezivanje reči,
- (b) koristeći postojeću funkciju iz biblioteke `string.h` (`strcat`).

Napomena: voditi računa da se za rezultujuću reč odvoji odgovarajuća količina memorije.

[Rešenje 3.91]

**Zadatak 3.87** Sa ulaza se unosi reč koja nije duža od 20 znakova. Napisati funkciju koja svako pojavljivanje znaka koji se zadaje kao prvi argument funkcije

### 3 Predstavljanje podataka

---

udvaja a svako po- javljivanje znaka koji se zadaje kao drugi argument funkcije izbacuje. Napisati program koji poziva ovu funkciju za reč unetu sa standardnog ulaza i za znakove koji se takoę zadaju sa standardnog ulaza. Na primer, ako se unese reč **ana** i znakovi **a** i **n**, tada funkcija treba da izmeni reč tako da ona postane **aaaa**, ako se unese reč **abrakadabra** i znakovi **a** i **b**, tada funkcija treba reč da izmeni tako da ona postane **aaraakaadkkraa**.

Napomena: voditi računa da novonastala izmenjena reč može imati veći broj karaktera i u skladu sa tim rezervisati odgovarajuću količinu memorije. Dopusšteno je koristiti pomoćan niz.

[Rešenje 3.91]

**Zadatak 3.88** Napisati funkciju `void ukloni(char *s);` koja iz niske uklanja sva slova iza kojih neposredno sledi slovo koje je u abecedi nakon njih (veličina slova se zanemaruje). Testirati funkciju u programu koji učitava liniju teksta (najviše 100 karaktera).

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| zdRaVo suIma
|| zRVo vma
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| 12345AbcD
|| 12345D
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| JeD1aN D52Va.
|| JeD1N D52Va.
```

#### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| abcd efg
|| d g
```

[Rešenje 3.91]

### Zadatak 3.89

- Napisati C funkciju `int procitaj_recenicu(char *s, int max_len)`, koja sa standardnog ulaza čita rečenicu i smešta je u nisku `s`. Čitanje rečenice se zaustavlja ako se pročita simbol `.` ili je već učitano `max_len-1` karaktera. Funkcija treba da vrati broj pročitanih karaktera.
- Napisati C funkciju `void prebroj(char *s, int *broj_malih, int *broj_velikih)`, koja za zadatu nisku `s` računa broj malih i velikih slova koji se u njoj pojavljuju.
- Napisati glavni program koji sa standardnog ulaza čita rečenice i na standardni izlaz ispisuje onu kod koje je razlika broja malih i velikih slova najveća.

[Rešenje 3.91]

**Zadatak 3.90**

- a) Uvesti tip podataka **Sifra** kojim se opisuje način šifrovanja alfanumeričkih karaktera. Svaka šifra se opisuje celobrojnomo vrednoscu **b** koja određuje broj pozicija pomeranja, kao i karakterom 'L' ili 'D' koji određuje smer pomeranja (levo ili desno).
- b) Napisati funkciju `void sifruj(char rec[], Sifra s)` koja transformiše zadataku reč **rec** po šifri **s**. Reč se šifruje tako što se svako slovo zamenjuje slovom za **b** mesta levo ili desno od njega u abecedi, i to ciklično, a isto tako i za cifre.  
Npr: za **b=2**, i **smer='D'** : a se menja sa c, b sa d,..., x sa z, y sa a, z sa b, 1 sa 3, .. 8 sa 0, 9 sa 1
- c) Sa standardnog ulaza se zadaje način šifrovanja i to u obliku **2 D 5 L** (šifra može biti i duža). Potom se učitava **n** i **n** reči sa standardnog ulaza (maksimalna dužina reči je 20 karaktera). Ispisati reči na standardni izlaz nakon primenjenih svih zadatah načina šifrovanja.

[Rešenje 3.91]

**Zadatak 3.91** Implementirati funkciju `int strspn(char* s, char* t)` koja izračunava dužinu početnog dela niske **s** sastavljenog isključivo od karaktera sadržanih u niski **t**.

Napisati i program koji sa standardnog ulaza učitava dve niske (dužine najviše 100 karaktera, svaku u zasebnom redu) i ispisuje rezultat poziva funkcije **strspn** na standardni izlaz.

Na primer, za učitane podatke "734a.bf62", "0123456789") program ispisuje vrednost 3.

[Rešenje 3.91]

## 3.6 Rešenja

**Rešenje 3.57**

```

/*
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.
7  */
8
9  #include <stdio.h>
10 #include <ctype.h>
11
12 /*
13  Kada je niz argument funkcije, dodatni argument je obavezno
14  njegova dimenzija. Kod stringova to nije slucaj jer svaki string
15  ima isti poslednji element - terminirajucu nulu - i to je oznaka
16  kraja stringa.
17  */
18 void konvertuj(char s[])
19 {
20     int i;
21
22     for(i=0; s[i]!='\0'; i++)
23         if (s[i]>='a' && s[i]<='z')
24             s[i] = toupper(s[i]); /* toupper - konvertuje malo slovo u
25             odgovarajuce veliko */
26         else if (s[i]>='A' && s[i]<='Z')
27             s[i] = tolower(s[i]); /* tolower - konvertuje veliko slovo
28             u odgovarajuce malo */
29     /*
30      Funkcije toupper i tolower se nalaze u zaglavlju ctype.h.
31
32      Konverzija malog slova u veliko bez upotrebe funkcije toupper:
33      s[i] = s[i]-'a'+'A';
34      Konverzija velikog slova u malo bez upotrebe funkcije tolower:
35      s[i] = s[i]+'a'-'A';
36
37      */
38 }
39
40 int main()
41 {
42     /*
43      Poslednji karakter svakog stringa je terminirajuca
44      nula '\0', specijalni karakter ciji je ASCII kod 0.
45
46      Ukoliko pretpostavljamo da string sadrzi najvise 30
47      karaktera, neophodno je deklarirati niz od 31 karaktera,
48      pri cemu se dodatni izdvaja za terminirajucu nulu.
49
50      */
51     char s[31];
52     printf("Unesi string:");
53
54     /*
55      Za razliku od nizova koji se ucitavaju i stampaju
56      element po element, stringovi se mogu ucitati i
57      odstampati pomocu jedne scanf/printf naredbe koriscenjem
58      */
59 }
```



```

56     specifikatora %s.

58     Funkcija scanf učitava string do prvog pojavljivanja razmaka.
    */
60     scanf("%s", s);

62     konvertuj(s);

64     printf("Konvertovani string: %s\n", s);

66     return 0;

68 }

```

### Rešenje 3.58

```

/*
2   Napisati funkciju skрати koja uklanja beline sa
   kraja datog stringa.

4

   Napisati glavni program koji testira napisanu
6   funkciju na stringu "rep belina".

8 */

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

12

14 /*
   Funkcija koja racuna duzinu niza
   ne racunajuci '\0'.

16

   U biblioteci string.h definisan je veliki
18   broj funkcija za rad sa stringovima,
   ukljucujuci i funkciju strlen koja racunana
20   duzinu stringa.

22   Funkcija strlen_klon predstavlja jednu
   implementaciju funkcije strlen.

24

   U zadacima cemo uvek koristiti ugradjenu
26   funkciju strlen osim ako u tekstu zadatka
   nije naglaseno da se ona ne sme koristiti.
28   Funkcija strlen_klon služi da pokaze na koji
   nacin radi ugradjena funkcija strlen.

30

   Ugradjena funkcija strlen poziva se na
32   isti nacin kao funkcija strlen_klon:
   strlen(s1)

34 */

```

### 3 Predstavljanje podataka

---

```
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'; /* DOdajemo terminirajucu nulu iza indeksa i (prvi
61         neblanko karakter gledano sdesna nalevo).*/
62
63     /*
64      * Ugradjena funkcija isspace nalazi se u biblioteci ctype.h i
65      * vraca 1 ako
66      * je dati karakter blanko znak a 0 u suprotnom.
67
68      * Unarni logicki operator ! oznacava negaciju.
69      */
70 }
71
72 int main()
73 {
74     /*
75      * Ukoliko string ne zelimo da učitavamo po pokretanju programa
76      * vec da ga unapred zadamo, to mozemo uraditi na sledeci nacin:
77      */
78     char s[]="rep belina";
79     /* U ovom slucaju nije neophodno navoditi dimenziju stringa vec
80      * ce ona biti automatski postavljena na broj karaktera u stringu +
81      * 1 za
82      * terminirajucu nulu. */
83
84     printf("Pre skracivanja: %s*\n", s);
```

```

84   skрати(s);
      printf("Posle skracivanja: %s*\n", s);
86
      return 0;
88 }

```

### Rešenje 3.59

```

/*
2   Napisati program koji ucitava 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.
*/
8
#include <stdio.h>
10 #include <string.h>
12
#define MAX 30
/*
14   Na stringove ne mozemo primeniti naredbu dodele.
   Ukoliko zelimo da jedan string "dodelimo" drugom,
16   mozemo koristiti ugradjenu funkciju strcpy(s,t)
   koja kopira karaktere stringa t
18   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   Karakteri stringa original se, jedan po jedan,
26   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 sluzi 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 */

```

### 3 Predstavljanje podataka

---

```
44 void strcpy_klon(char kopija[], char original[])
45 {
46     int i;
47     for(i=0; original[i]; i++)
48         kopija[i]=original[i];
49
50     kopija[i] = '\0';
51 }
52
53 int main()
54 {
55     char src[MAX+1];    /* src, skraceno od source (izvor, odnosno sta
56                          kopiramo) */
57     char dst[3*MAX+1];  /* dst, skraceno od destination (odrediste,
58                          odnosno gde kopiramo) */
59
60     /*
61      Vazno je izdvojiti dovoljno memorijskog prostora
62      za string dst: on treba da bude tri puta veci od
63      maksimalne duzine stringa src + jedan karakter za
64      terminirajucu nulu.
65     */
66
67     printf("Unesi jedan string:");
68     scanf("%s", src);
69
70     strcpy_klon(dst,src);
71
72     /*
73      Funkcija strcat(s,t) nadovezuje karaktere stringa
74      t na kraj stringa s i novi string terminira
75      karakterom '\0' .
76
77      Funkcija strcat se nalazi u biblioteci string.h.
78     */
79     strcat(dst,src);
80     strcat(dst,src);
81
82     printf("Kada nadovezemo string %s triput: %s\n",src,dst);
83
84     return 0;
85 }
```

#### Rešenje 3.60

```
1  /*
2     Napisati funkciju int ucitaj_liniju(char s[], int n)
3     koja ucitava liniju maksimalne duzine n u string s
4     i vraca duzinu učitane linije. Linija moze da sadrzi
5     blanko znakove ali ne moze da sadrzi \n ili EOF.
```

```
7      Napisati potom glavni program koji ucitava linije
      do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko
9      ima vise linija maksimalne duzine, ispisati prvu. Mozemo
      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
25     string karakter po karakter.

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

29
      U standardnoj biblioteci stdio.h postoji definisana
31     funkcija char *gets(char *s) koja ucitava karaktere
      dok se ne pojavi novi red ili EOF. Ova funkcija
33     dopusta unosenje vise karaktera nego sto string
      s sadrzi, sto moze dovesti do neocekivanog ponasanja
35     programa.

37     Pored funkcije gets, koja vrsi ucitavanje sa standardnog
      ulaza, u standardnoj biblioteci stdio.h postoji
39     i ugradjena funkcija fgets koja vrsi ucitavanje iz
      datoteke. Nju cemo koristiti za nekoliko casova
41     kada budemo radili datoteke. Prototim funkcije fgets je
      ovakav:

43
      char *fgets(char *s, int size, FILE *stream);

45
      Argumenti funkcije fgets su:
47     s - string u koji vrsimo ucitavanje
      size - maksimalna duzina unetog stringa
49     stream - datoteka iz koje vrsimo ucitavanje

51
      Funkcija fgets, za razliku od funkcije gets, ne dopusta
      unos vise karaktera od date vrednosti size. Zbog toga
53     je ona sigurnija nego funkcija gets. Funkciju fgets
      mozemo koristiti i za unos sa standardnog ulaza
55     ukoliko kao treci argument navedemo stdin.

57 */
      int ucitaj_liniju(char s[], int n)
```

### 3 Predstavljanje podataka

---

```
59 {
61     int i=0;
63     int c;
65     while((c=getchar())!='\n' && i<n-2 && c!=EOF)
67     {
69         s[i] = c;
71         i++;
73     }
75     /* Ucitavamo najvise n-2 karaktera jer na kraju dodajemo jos
77     dva: '\n' i '\0' */
79     s[i]='\n';
81     s[i+1]='\0';
83     return i;
85 }
87 int main()
89 {
91     char linija[MAX];
93     char najduza_linija[MAX];
95     int max_duzina=0;
97     int duzina;
99     /*
101     Petlja se zavrшава ukoliko je promenljiva duzina
103     jednaka nuli, sto cemo postici zadavanjem linije koja ne sadrzi
105     nijedan karakter osim EOF.
107     */
109     while ((duzina=ucitaj_liniju(linija, MAX))>0)
111     {
113         /*
115         Proveravamo da li je uneta linija duza od trenutnog
117         maksimuma i azuriramo promenljive max_duzina i najduza_linija
119         */
121         if (max_duzina<duzina)
123         {
125             max_duzina = duzina;
127             strcpy(najduza_linija,linija);
129         }
131     }
133     printf("Najduza linija: %s duzine: %d\n", najduza_linija,
135           max_duzina);
137     return 0;
139 }
```

## Rešenje 3.61

```
2  /*
   Napisati program koji pretvara nisku u ceo broj.
   Npr. za ulaz "-1238" se generise rezultat -1238
4  Pogledati funkcije atoi i atof koje postoje u biblioteci stdlib.h
   */
6
8  #include <stdio.h>
   #include <ctype.h>
   #define MAX 10
10 /*
   String b se sastoji od karaktera koji
12  cine jedan ceo broj, onim redom kojim
   se karakteri pojavljuju u zapisu broja.
14
   Ako je prvi karakter stringa b '-',
16  to znaci da je broj negativan i
   funkcija znak_broja vraca -1
18
   U suprotnom, broj je pozitivan i
20  funkcija znak_broja vraca 1
22 */
24 int znak_broja(char b[])
   {
26     if(b[0]=='-')
         return -1;
28     return 1;
   }
30
32 /*
   Funkcija formiraj_broj na osnovu
34  karaktera koji cine broj iz stringa
   b vraca ceo broj koji odgovara
36  zapisu datom u stringu b.
38
   Ako su cifre broja a,b,c i d, tada
   broj mozemo kreirati kao:
40  a*10^3 + b*10^2 + c*10^1 + d*10^0
42
   Medjutim, efikasnije je koristiti
   Hornerovu semu:
44
   10*(10*(10*(10*0 + a)+b)+c)+d
46
   */
48 int formiraj_broj(char b[])
50 {
```

```
int i;
52 int n=0;
int znak = znak_broja(b);
54
/*
56 Ako je broj negativan, cifre u nizu b
pocinju od indeksa 1
58 */
i=0;
60 if(znak== -1)
62     i=1;
64
/*
Funkcija isdigit proverava da li je broj
66 cifra. Nalazi se u biblioteci ctype.h
68
Proveravamo da li je karakter u zapisu
broja cifra kako bismo se osigurali
70 od nekorektnog unosa, npr ako korisnik
unesse -123abc. Ovaj unos je moguc jer
72 se vrsi sa scanf("%s",broj), gde unosimo
karaktere do prvog blanko znaka
74
Ako naidjemo na karakter koji nije cifra,
76 prekidamo petlju
78
*/
for(; b[i]!='\0'; i++)
80     if(isdigit(b[i]))
        n = n*10 + b[i] - '0';
82     else
        break;
84
/* Formirani broj mnozimo znakom: */
86
n*=znak;
88 return n;
90 }
92 int main()
{
94     char broj[MAX];
int n;
96
/* Ucitavamo broj: */
98 scanf("%s", broj);
100
/* Ispisujemo rezultat: */
printf("Broj zapisan kao int: %d\n", formiraj_broj(broj));
102
```



```

104     return 0;
    }

```

### Rešenje 3.62

```

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

6  #include <stdio.h>
   #include <string.h>
8  #define MAX 10
   /*

10     Funkcija transformisi_negativan vraca
12     1 ako je broj negativan i 0 u suprotnom, a
   uz to, ako broj jeste negativan, funkcija
14     treba da ga konvertuje u njegovu apsolutnu
   vrednost. S obzirom da funkcija treba da vrati dve
16     vrednosti, to realizujemo na sledeci nacin:
   1. indikator da li je broj negativan
18     ce vratiti kao povratnu vrednost
   2. apsolutnu vrednost broja ce vratiti
20     preko liste argumenata, zbog cega broj
   prenosimo preko pokazivaca
22
   */
24 int transformisi_negativan(int* pn)
   {
26     if(*pn<0)
       {
28         *pn = -(*pn);
         return 1;
30     }
       return 0;
32 }

34 int formiraj_niz_cifara(int n, char b[], int neg)
   {
36     int i=0;
       char cifra;

38     do
40     {
         cifra = n%10;

42         /* Promenljiva b predstavlja string.
         Da bismo na neku poziciju u stringu
         upisali karakter koji odgovara nekoj
46         cifri, npr '2', neophodno je da

```

### 3 Predstavljanje podataka

---

```

48         odgovarajucoj poziciji dodelimo vrednost
        ASCII koda te cifre, konkretno za '2'
        ASCII kod je '0'+2.

50
        Greska bi bila navesti b[i]=2
52         jer 2 nije ASCII kod koji odgovara karakteru
        '2'.
54     */
    b[i]=cifra+'0';

56
    n/=10;
58    i++;
} while(n);

60
/* Ako je broj negativan, dodajemo znak minus: */
62 if(neg)
{
64     b[i]='-';
    i++;
66 }

68 /* Svaki string se zavrшава terminirajucom nulom: */
b[i]='\0';
70 }

72 void obrni(char s[])
{
74
    char t;
76    int i,j;
    /*
78     Karaktere stringa obrcemo tako sto razmenimo karaktere na
    pozicijama 0 i n-1,
    zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
    druge
80     */

82    for(i=0,j=strlen(s)-1;i<j;i++, j--)
    {
84        t = s[i];
        s[i] = s[j];
86        s[j] = t;
    }

88 }

90 void broj_u_niz_cifara(int n, char broj[])
92 {
    int negativan;

94
    /* Odredjujemo znak broja: */
96    negativan=transformisi_negativan(&n);
```

```

98  /* Izdvajamo cifre broja i smestamo ih u niz: */
    formiraj_niz_cifara(n, broj, negativan);
100
    /* S obzirom da cifre izdvajamo sa kraja broja, u nizu ce biti u
       obrnutom redosledu.
       Na primer, za broj 234 niz ce sadrzati cifre 4 3 2. */
102    obrni(broj);
104 }

106 int main()
    {
108     int n;
        char broj[MAX];
110     int negativan;

112     /* Ucitavamo broj: */
        scanf("%d", &n);
114
        /* Kreiramo broj na osnovu niza cifara: */
116     broj_u_niz_cifara(n, broj);

118     /* Ispisujemo rezultat: */
        printf("Broj zapisan kao string: %s\n", broj);
120
        return 0;
122 }

```

### Rešenje 3.63

```

/*
2  Napisati program koji ucitava 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
#include <stdio.h>
10 #include <string.h>

12 /*
    Funkcija strcmp(s,t) je ugradjena funkcija koja utvrđuje da
    li su stringovi
14    s i t jednaki. Ukoliko jesu, vraca 0, a u suprotnom vraca
    razliku
    ASCII kodova prva dva razlicita karaktera na istim pozicijama

```

### 3 Predstavljanje podataka

---

```
16      (npr strcmp("aa","ab") ce vratiti -1 a strcmp("ab","aa") 1).
18      Funkcija strcmp se nalazi u zaglavlju string.h.
20      Funkcija strcmp_klon je jedna implementacija funkcije strcmp.
22      U zadacima cemo uvek koristiti ugradjenu funkciju strcmp osim
      ako u tekstu zadatka
      nije naglaseno da se ona ne sme koristiti. Funkcija
24      strcmp_klon sluzi da pokaze na koji
      nacin radi ugradjena funkcija strcmp.
26      Ugradjena funkcija strcmp poziva se na isti nacin kao funkcija
      strcmp_klon:
      strcmp(s1,s2)
28      gde poredimo stringove s1 i s2.
30  */
32  int strcmp_klon(char s1[], char s2[])
33  {
34      int i;
35      for(i=0; s1[i]==s2[i];i++)
36          if (s1[i]=='\0')
37              return 0;
38      return s1[i] - s2[i];
39  }
40
41  int main()
42  {
43      char s1[21];
44      char s2[21];
45      char* p;
46
47      printf("Unesi dva stringa:");
48      scanf("%s%s",s1,s2);
49
50      /*
51      Funkcija strstr(s,t) je ugradjena funkcija koja utvrdjuje da
52      li je string t
53      podstring stringa s i ako jeste, vraca pokazivac (char*) na
54      karakter
55      stringa s odakle pocinje prvo pojavljivanje stringa t, a NULL
56      u suprotnom.
57
58      NULL je pokazivac koji ne pokazuje ni na sta, odnosno ne
59      sadrzi adresu
60      nijedne promenljive.
61
62      Podsetimo se veze nizova(a time i stringova) i pokazivaca:
63      ako je string deklarisan sa s1[21], tada je njegov naziv s1
```

```

62     ekvivalentan adresi prvog karaktera stringa:
        s1 <=> &s1[0]
        i nadalje redom:
64     s1+1 <=> &s1[1]
        ...
66     u opstem slucaju:
        s1+i <=> &s1[i]
68
        To znaci da se indeks elementa na koji pokazuje s1+i moze
70     dobiti tako sto od s1+i oduzmemo pokazivac na pocetak niza:
        s1+i-s1 <=> i. Ovako od pokazivaca na karakter u stringu
72     dobijamo njegov indeks u stringu.

74     */

76     p = strstr(s1,s2);

78     if (strcmp_klon(s1,s2)==0)
        printf("Uneti stringovi su jednaki\n");
80     else if (p!=NULL)
        printf("%s jeste podstring od %s pocev od pozicije : %d\n", s2,
            s1, p-s1);
82     else
        printf("%s NIJE podstring od %s\n", s2,s1);

84     return 0;
86 }

```

### Rešenje 3.64

```

/*
2     Napisati program koji za uneti string s i karakter c utvrdjuje
        da li se c pojavljuje u stringu s i ukoliko se pojavljuje,
4     ispisuje indeks prvog pojavljivanja a u suprotnom ispisuje
        odgovarajucu poruku. Mozemo pretpostaviti da string ima najviše
6     20 karaktera.
    */

8
#include <stdio.h>
10 include <string.h>

12 int main()
{
14     char s[21];
        char c;
16     char* p;

18     printf("Unesi karakter:");
        c=getchar();
20     printf("Unesi string:");
        scanf("%s", s);

```

```
22
23  /*
24     Da smo ucitavali obrnutim redom (prvo string pa karakter)
25     to bismo realizovali na sledeci nacin:
26     printf("Unesi string:");
27     scanf("%s",s);
28     getchar();
29     printf("Unesi karakter:");
30     c=getchar();

31
32     Dodatni getchar() bi sluzio da "pokupi" karakter kojim
33     razdvajamo unos stringa i karaktera (razmak, novi red ili
34     slicno).

35
36  */
37
38  /*
39     Funkcija strchr(s,c) je ugradjena funkcija koja vraca pokazivac
40     na prvi karakter u stringu s koji je jednak karakteru c, ako
41     takav
42     postoji, a NULL u suprotnom.

43     Indeks od pokazivaca dobijamo na isti nacin kao u prethodnom
44     zadatku
45     sa strstr.
46  */
47
48  p = strchr(s,c);
49  if(p!=NULL)
50      printf("%c se pojavljuje u %s na poziciji %d\n", c, s, p-s);
51  else
52      printf("%c se NE pojavljuje u %s\n",c, s);
53
54  return 0;
55 }
```

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

Rešenje 3.91

## 3.7 Višedimenzioni nizovi

### Zadatak 3.92

a) Napisati funkciju

```
int reflektivna(int a[][MAX], int n)
```

kojom se za relaciju zadatom matricom  $a$  (matruca je kvadratna) ispitije da li je reflektivna.

b) Napisati funkciju

```
int simetricna(int a[][MAX], int n)
```

kojom se za relaciju zadatom matricom  $a$  (matruca je kvadratna) ispitije da li je simetricna.

c) Napisati funkciju

```
int tranzitivna(int a[][MAX], int n)
```

kojom se za relaciju zadatom matricom  $a$  (matruca je kvadratna) ispitije da li je tranzitivna.

Dva elementa  $i$  i  $j$  ( $i @ j$ ) su u relaciji akko  $a[i][j] = 1$

Relacija je reflektivna ako za svako  $i$  važi:  $i @ i = 1$

Relacija je simetricna ako za svako  $i$  i  $j$  važi:  $i @ j = 1 \Rightarrow j @ i = 1$

Relacija je tranzitivna ako za svako  $i, j$  i  $k$  važi:  $i @ j = 1$  i  $j @ k = 1 \Rightarrow i @ k = 1$

Funkcija postavlja na 1 odgovarajuci indikator.

b) Sa standardnog ulaza prvo se unose dimenzija kvadratne matrice  $n$ , a nakon toga elementi matrice. Učitati matricu, i ispitati da li je relacija koju predstavlja relacija ekvivalencije (reflektivna, simetrična i tranzitivna).

[Rešenje 3.108]

**Zadatak 3.93** Napisati funkciju `float sumD(float a[][max], int n)` koja određuje sumu elemenata iznad glavne dijagonale. Potom napisati funkciju `float sumd(float a[][max], int n)` koja određuje sumu elemenata ispod



glavne dijagonale. Funkciju testirati pozivom u main-u. Matrica je maksimalne dimenzije 50x50. Matrica je kvadratna.

[Rešenje 3.108]

**Zadatak 3.94** Napisati funkciju  
`void transponovana(float a[][max], int m, int n, float b[][max])` koja  
 određuje transponovanu matricu matricu. Pozivom u main-u testirati funkciju.  
 Matrica je maksimalne dimenzije 50x50.

[Rešenje 3.108]

**Zadatak 3.95** Napisati funkciju  
`void mnozenje(int a[][max], int n, int m, int b[][max], int k,  
 int t, int c[][max])`  
 koja računa proizvod dve matrice. Pozivom u main-u testirati funkciju. Matrica  
 je maksimalne dimenzije 50x50. Testirati da li su podaci korektno uneti i testirati  
 da li je moguće matrice množiti.

[Rešenje 3.108]

**Zadatak 3.96** Napisati funkciju u kojoj se razmenjuju elementi k-te i t-te  
 vrste matrice (k i t su argumenti funkcije). Funkciju testirati pozivom u main-u i  
 ispisom novodobijene matrice na standardni izlaz. Sa standardnog ulaza učitavaju  
 se dimenzije matrice, a potom i elementi matrice i brojevi k i t. Maksimalna  
 dimenzija matrice je 50x50. Funkciju testirati pozivom u main-u.

[Rešenje 3.108]

**Zadatak 3.97** Sa standardnog ulaza unose se celi pozitivni brojevi m i n koji  
 označavaju broj vrsta i broj kolona matrice. Potom se unose elementi matrice.  
 Nakon unosa elemenata matrice, unose se još dva broja p i k ( $p \leq m$ ,  $k \leq n$ ). Na  
 standardi izlaz ispisati sume svih podmatrica (dimenzije  $p \times k$ ) unete matrice. U  
 slučaju greške ispisati -1.

**Napomena 1:** Ne razmatrati slučaj negativnih brojeva.

**Napomena 2:** Nije bitan redosled kojim se ispisuju sume.

#### Primer 1

INTERAKCIJA SA PROGRAMOM:

```
3 4
1 2 3 4
5 6 7 8
9 10 11 12
3 3
54 63
```

#### Primer 2

INTERAKCIJA SA PROGRAMOM:

```
3 4
1 2 3 4
5 6 7 8
9 10 11 12
2 3
24 30 48 54
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
3 2
1 2
3 4
5 6
7 8
-1
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
5 3
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
2 2
7 5 20 19 18 23 4 8
```

[Rešenje 3.108]

**Zadatak 3.98** Sa standardnog ulaza zadata je dimenzija kvadratne matrice  $n$  ( $0 < n \leq 50$ ), a zatim i vrednosti pojedinačnih elemenata. Ukoliko je  $n$  izvan ovog opsega ispisati  $-1$  i prekinuti izvršavanje programa. Napisati program koji:

- Učitava matricu i ispisuje je na izlaz. U slučaju greške ispisati  $-1$  i prekinuti izvršavanje programa.
- Ispituje da li su elementi matrice po kolonama, vrstama i dijagonalama (glavnoj i sporednoj) sortirani strogo rastuće. Za svaki od ovih slučajeva redom ispisati  $1$  ako jesu i  $0$  ako nisu sortirani - videti primere.

#### Primer 1

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

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
2
6 9
4 10
6 9
4 10
0 1 0
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
4
5 5 7 9
6 10 11 13
8 12 14 15
13 15 16 20
5 5 7 9
6 10 11 13
8 12 14 15
13 15 16 20
1 0 1
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
1
5
5
1 1 1
```

[Rešenje 3.108]

**Zadatak 3.99** Sa standardnog ulaza se unosi broj  $n$  ( $0 < n \leq 10$ ), a potom i elementi kvadratne matrice dimenzije  $n \times n$ . Elementi matrice su celi brojevi. Proveriti da li važi da su zbrovi elemenata kolona matrice uređjeni u strogo rastućem poretku. **Napomena 1:** Ukoliko program uvek ispisuje **da** ili uvek ispisuje **ne** smatraće se netačnim i poeni se ne mogu osvojiti.

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
4
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
ne
```

*Primer 2*

INTERAKCIJA SA PROGRAMOM:

```
3
1 2 3
4 5 6
7 8 9
da
```

*Primer 3*

INTERAKCIJA SA PROGRAMOM:

```
3
2 -2 1
1 2 2
2 1 -2
ne
```

*Primer 4*

INTERAKCIJA SA PROGRAMOM:

```
5
-1 0 2 0 20
0 0 0 10 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
da
```

[Rešenje 3.108]

**Zadatak 3.100** Sa standardnog ulaza unosi se broj  $n$  ( $0 < n \leq 200$ ), a potom i elementi kvadratne matrice dimenzije  $n \times n$ . Elementi matrice su celi brojevi. Proveriti da li je uneta matrica ortonormirana i na standardni izlaz ispisati **da** ako jeste ili **ne** ako nije ortonormirana. Matrica je ortonormirana ako je skalarni proizvod svakog para različitih vrsta jednak 0, a skalarni proizvod vrste sa samom sobom 1. U slučaju greške ispisati -1.

**Napomena 1:** Skalarni proizvod vektora  $a = (a_1, a_2, \dots, a_n)$  i  $b = (b_1, b_2, \dots, b_n)$  je  $a_1 \cdot b_1 + a_2 \cdot b_2 + \dots + a_n \cdot b_n$ .

**Napomena 2:** Ukoliko program uvek ispisuje **da** ili uvek ispisuje **ne** smatraće se netačnim i poeni se ne mogu osvojiti.

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
4
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
da
```

*Primer 2*

INTERAKCIJA SA PROGRAMOM:

```
3
1 2 3
4 5 6
7 8 9
ne
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
```

```
3
2 -2 1
1 2 2
2 1 -2
ne
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
```

```
5
-1 0 2 0 20
0 0 0 10 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
da
```

[Rešenje 3.108]

**Zadatak 3.101** Napisati funkciju koja kao argumente prima kvadratnu matricu celih brojeva i njenu dimenziju, a vraća 1 ako je matrica donja trougaona, odnosno 0 ako nije. Pretpostavka je da je maksimalna dimenzija matrice 100. Matrica je donja trougaona ako se u gornjem trouglu (iznad glavne dijagonale, ne uključujući je) nalaze sve nule.

[Rešenje 3.108]

**Zadatak 3.102** Napisati program koji sa standardnog ulaza unosi prvo dimenziju matrice ( $n < 10$ ) pa zatim elemente matrice i izračunava sumu elemenata iznad sporedne dijagonale matrice.

[Rešenje 3.108]

**Zadatak 3.103** Za datu kvadratnu matricu kažemo da je *magični kvadrat* ako je suma elemenata u svakoj koloni i svakoj vrsti jednaka. Napisati program koji sa standardnog ulaza učitava prirodni broj  $n$  ( $n < 10$ ) i zatim elemente kvadratne matrice, proverava da li je ona *magični kvadrat* i ispisuje odgovarajuću poruku na standardni izlaz.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
```

```
4
1 5 3 1
2 1 2 5
3 2 2 3
4 2 3 1
da
```

[Rešenje 3.108]

**Zadatak 3.104** Napisati program koji sa standardnog ulaza učitava prvo dimenzije matrice ( $n$  i  $m$ ) a zatim redom i elemente matrice (ne postoje pretpostavke o dimenziji matrice). Nakon toga na standardni izlaz, zapisati indekse ( $i$

i  $j$ ) onih elemenata matrice koji su jednaki zbiru svih svojih susednih elemenata (pod susednim elementima podrazumevamo okolnih 8 polja matrice ako postoje).

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
4 5
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
1 1
1 3
3 2
3 4
```

[Rešenje 3.108]

**Zadatak 3.105** Sa standarnog ulaza se zadaje prvo dimenziju kvadratne matrice  $n$  ( $n < 100$ ), a zatim elemente matrice. Nakon toga, na standardni izlaz ispisati redni broj kolone koja ima najveći zbir elemenata.

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
3
1 2 3
7 3 4
5 3 1
0
```

[Rešenje 3.108]

**Zadatak 3.106** Napisati funkciju koja treba da ispiše elemente matrice u grupama koje su paralelne sa sporednom dijagonalom matrice. Može se pretpostaviti da matrica nije dimenzije veće od  $100 \times 100$ .

*Primer 1*

INTERAKCIJA SA PROGRAMOM:

```
3
1 2 3
4 5 6
7 8 9
1
2 4
3 5 7
6 8
9
```

[Rešenje 3.108]

### 3 Predstavljanje podataka

**Zadatak 3.107** Sa standardnog ulaza učitava se broj  $n$ , a zatim i kvadratna matrica koja sadrži brojeve tipa `double` dimenzije  $n \times n$ . Napisati program koji izračunava i ispisuje razliku (na dve decimale) između zbira elemenata gornjeg trougla i zbira elemenata donjeg trougla matrice – gornji trougao čine svi elementi iznad sporedne dijagonale (ne računajući dijagonalu), a donji trougao čine svi elementi ispod sporedne dijagonale (računajući dijagonalu). U slučaju greške u datoteku upisati **GRESKA**.

#### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| 3
|| 2 3.2 4
|| 7 8.8 1
|| 2.3 1 1
|| -2.10
```

#### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| 4
|| 2.3 1 12 8
|| 4 -8.2 7 14.5
|| 1 -2.5 9 11
|| 3 4.3 -5.7 2
|| 49.4
```

#### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| -4
|| GRESKA
```

[Rešenje 3.108]

**Zadatak 3.108** Kao argumenti komandne linije zadate su dimenzije matrice  $A$  ( $m$  i  $n$ ). Element matrice se naziva sedlo ako je istovremeno najmanji u svojoj vrsti, a najveći u svojoj koloni. Ispisati indekse i vrednosti onih elemenata matrice koji su sedlo. Pretpostaviti da je maksimalna dimenzija matrice  $50 \times 50$ . Ukoliko nisu zadati svi potrebni argumenti komandne linije ispisati poruku da je došlo do greške. Ukoliko su dimenzije van opsega ispisati poruku o grešci.

#### Primer 1

```
|| POKRETANJE: ./a.out 2 3
|| INTERAKCIJA SA PROGRAMOM:
|| 1 2 3
|| 0 5 6
|| 0 0 1
```

#### Primer 2

```
|| POKRETANJE: ./a.out 3 3
|| INTERAKCIJA SA PROGRAMOM:
|| 10 3 20
|| 15 5 100
|| 30 -1 200
|| 1 1 5
```

#### Primer 3

```
|| POKRETANJE: ./a.out 3
|| INTERAKCIJA SA PROGRAMOM:
|| greska
```

#### Primer 4

```
|| POKRETANJE: ./a.out 200 3
|| INTERAKCIJA SA PROGRAMOM:
|| greska
```

[Rešenje 3.108]

## 3.8 Rešenja

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

## 3.9 Strukture

**Zadatak 3.109** Tekst

[Rešenje 3.109]

**Zadatak 3.110** Tekst

[Rešenje 3.110]

**Zadatak 3.111** Tekst

[Rešenje 3.111]

**Zadatak 3.112** Tekst

[Rešenje 3.112]

**Zadatak 3.113** Tekst

[Rešenje 3.113]

**Zadatak 3.114** 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
```

[Rešenje 3.125]

**Zadatak 3.115** 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

```

[Rešenje 3.125]

**Zadatak 3.116** 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

```

[Rešenje 3.125]

**Zadatak 3.117** 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 proveri njegovu želju. Ako ima više studenata sa istim imenom i prezimenom ispisati sve želje. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

### 3 Predstavljanje podataka

---

#### 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
```

[Rešenje 3.125]

**Zadatak 3.118** Definisati strukturu *Grad* u kojoj se nalazi ime grada (niska dužine 20 karaktera) i prosečna temperatura u toku decembra (realan broj). Napisati program koji učitava imena  $n$  ( $0 < n < 50$ ) gradova i njihove prosečne temperature, a zatim ispisuje one gradove koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni. *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:
```

[Rešenje 3.125]

**Zadatak 3.119** 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
```

[Rešenje 3.125]

**Zadatak 3.120** Napisati funkcije koje izračunavaju zbir, razliku i proizvod dva razlomka, `razlomak zbir(razlomak a, razlomak b)` itd. Unosi se broj  $n$  a potom i  $n$  razlomaka sa standardnog ulaza (najviše 100). Ispisati njihov zbir, razliku i proizvod na standardni izlaz.

[Rešenje 3.125]

**Zadatak 3.121** Napraviti strukturu *VOCE* koja sadrži ime (ne duže od 20 karaktera) i cenu (tipa `float`). Sa standardnog ulaza unosi se broj voćki (ne vići od 200), a potom uneti niz voća i pozvati funkciju koja izračunava prosečnu cenu voća. Potom ispisati imena onih voćki čija je cena veća od prosečne.

[Rešenje 3.125]

**Zadatak 3.122** Sa standardnog ulaza učitava se  $n$  ( $0 < n \leq 200$ ), a potom i spisak (dužine  $n$ ) engleskih reči i njihov prevod na srpski jezik. Potom se učitava jedna reč sa standardnog ulaza. Na standardni izlaz ispisati odgovarajući prevod date reči ili podatak o tome da se reč ne nalazi na spisku.

```
apple jabuka
pineapple ananas
orange narandza
```

### 3 Predstavljanje podataka

---

```
pear kruska
grape grozdje
```

i reč *orange* program treba da ispiše *narandza* a za reč *cherry* program treba da ispiše poruku *Rec se ne nalazi u rečniku*. U programu se mogu koristiti funkcije iz zaglavlja *string.h*.

[Rešenje 3.125]

**Zadatak 3.123** Napisati program koji sa standardnog ulaza čitava najpre broj artikala (ceo broj manji od 20) a zatim podatke o artiklima. Artikli su voćke koje imaju po dva podatka: naziv voćke i cenu (naziv voćke je karakterska niska dužine do 20 karaktera). Program potom traži od korisnika da unese neku cenu i štampa na standardni izlaz sve voćke koje imaju zadatu cenu.

Primer rada programa:

```
4
jabuka 30
kruska 40
ananas 60
limun 40
```

```
Unesite cenu: 40
Voce te cene je: kruska limun
```

[Rešenje 3.125]

**Zadatak 3.124** Definirati strukturu koja opisuje dete atributima ime deteta (ne veće od 20 karaktera) , pol deteta (m ili z) i ocena. Ocenu je svako dete dalo radu obdaništa. Maksimalan broj dece je 100. Napisati program koji:

a) Sa standardnog ulaza se unosi  $n$ , a potom podaci o  $n$  dece. Koristiti strukturu:

```
typedef struct
{
    char ime[20];
    char pol;
    int ocena;
} DETE;
```

b) ispisati na standardni izlaz statistiku: koliko ima dečaka, a koliko devojčica i prosečnu ocenu. Potom ispisuje imena dece brojnijeg pola.

[Rešenje 3.125]

**Zadatak 3.125**

- Definisati tip podataka **TACKA** pogodan za predstavljanje tačke Dekartovske ravni (čije su  $x$  i  $y$  koordinate podaci tipa **double**).
- Definisati funkciju **double rastojanje(TACKA a, TACKA b)** koja izračunava rastojanje između dve tačke.
- Definisati funkciju **unsigned ucitaj\_poligon(TACKA\* tacke, unsigned n)** koja učitava  $n$  puta po dve vrednosti tipa **double** (koje predstavljaju koordinate temena poligona) i upisuje ih u zadati niz tačaka. Funkcija vraća broj uspešno učitanih tačaka.
- Definisati funkciju **double obim(TACKA\* poligon, unsigned n)** koja izračunava obim poligona sa  $n$  tačaka u zadatom nizu (napomena: ne zaboraviti stranicu koja spaja poslednje i prvo teme).
- Definisati funkciju **double maksimalna\_stranica(TACKA\* poligon, unsigned n)** koja izračunava dužinu najduže stranice poligona sa  $n$  tačaka u zadatom nizu (napomena: ne zaboraviti stranicu koja spaja poslednje i prvo teme).
- Definisati funkciju **main** u kojoj se sa standardnog ulaza učitava celobrojna nenegativna vrednost  $N$  ( $0 < N \leq 100$ ).

Inače, poziva se funkcija **ucitaj\_poligon**. Ukoliko je uspešno učitano  $m$  tačka ( $N$  ne mora da bude jednako  $m$ ), onda se poziva funkcija **obim** za  $m$  učitanih tačaka i ispisuje njen rezultat na standardni izlaz (ukoliko ova funkcija nije implementirana — ispisati na standardni izlaz simbol `?`). Posle toga se poziva funkcija **maksimalna\_stranica** za  $m$  učitanih tačaka i ispisuje njen rezultat na standardni izlaz (ukoliko ova funkcija nije implementirana — ispisati na standardni izlaz simbol `?`).

[Rešenje 3.125]

**3.10 Rešenja****Rešenje 3.109**

```

1  /*
3  Data je struktura koja opisuje koordinate
   tacke u ravni:

```

```
5      typedef struct point
7      {
9          float x;
10         float y;
11     } POINT;

12     U glavnom programu date su dve tacke: tacka
13     A sa fiksnim koordinatama (1,2) i tacka B
14     cije koordinate zadaje korisnik. Napisati
15     funkcije:
16     a) za racunanje rastojanja izmedju dve date tacke
17     b) za odredjivanje tacke koja se nalazi na
18         sredini duzi odredjene dvema datim tackama
19
20     Testirati napisane funkcije u glavnom programu.
21
22     */
23
24     #include <stdio.h>
25     #include <math.h>
26
27     typedef struct point
28     {
29         float x;
30         float y;
31     } POINT;
32
33     /*
34     Poljima strukture pristupamo pomocu
35     operatora .
36
37     Ako je promenljiva a tipa POINT,
38     njenim koordinatama pristupamo
39     pomocu a.x i a.y
40     */
41
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()
```

```

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 3.110

```

1  /*
2     Data je struktura
3     typedef struct Student
4     {
5         char ime[MAX];
6         char prezime[MAX];
7         char smer;
8         float prosek;
9     } STUDENT;
10
11 I   Napisati funkciju koja ucitava sa standardnog ulaza podatke o
12     studentu. Mozemo pretpostaviti da
13     ime i prezime studenta ne sadrže više od 30 karaktera.
14 II  Napisati funkciju koja ispisuje podatke o studentu na standardni
15     izlaz.
16 III Ucitati niz od n studenata i :
17     a) ispisati imena i prezimena onih koji su na smeru R
18     b) ispisati podatke za studenta sa najvećim prosekom; ako ima
19     više takvih studenata, ispisati
20     1) sve njih
21     2) prvog

```

```
19      3) poslednjeg
20  */
21
22  #include <stdio.h>
23  #define MAXST 100
24  #define MAX 31
25
26  typedef struct Student
27  {
28      char ime[MAX];
29      char prezime[MAX];
30      char smer;
31      float prosek;
32  } STUDENT;
33
34  /*
35   I
36
37   Ako je dat pokazivac na strukturnu promenljivu s,
38   poljima ove strukture pristupamo sa
39   (*s).ime, (*s).prezime, itd.
40
41   Zgrade su neophodne zbog prioriteta operatora:
42   operator * ima veci prioritet nego opeator . .
43
44   Operator -> pruza skraceni zapis za prethodno
45   navedeni pristup poljima:
46   s->ime je skraceno za (*s).ime
47   s->prezime je skraceno za (*s).prezime
48   itd.
49
50  */
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.
```



```

69     Prilikom poziva funkcije, za svaki argument funkcije kreira se
       promenljiva
71     koja predstavlja lokalnu kopiju argumenta i koja prestaje da
       postoji po završetku
       funkcije. S obzirom da se strukture sastoje od više polja,
       zauzimaju
73     više memorije nego nestrukturane promenljive. Zbog toga je za
       njihovo kopiranje
       potrebno više vremena i više memorijskih resursa nego za kopiranje
       nestrukturnih
75     promenljivih.

77     Da bismo učinili program efikasnijim, korisno je da strukturu uvek
       kao
       argument funkcije prenosimo po adresi (preko pokazivaca), bez
       obzira
79     da li će se struktura u toj funkciji menjati ili ne. Pokazivac na
       strukturu
       zauzima manje memorije nego sama struktura pa je izrada njegove
       kopije
81     brza a kopija pokazivaca uzima manji memorijski prostor nego kopija
       strukture.

83     Kada prenosimo strukturnu promenljivu u funkciju po adresi (preko
       pokazivaca), tada
85     imamo mogućnost da je u funkciji menjamo. Ukoliko želimo da
       onemogućimo promenu,
       uz argument dodajemo ključnu rec const. Ako pokušamo da promenimo
       argument
87     funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
       s->smer='X');,
       kompajler će prijaviti grešku. Na ovaj način obezbeđujemo da
       promenljiva
89     koju smo preneli po adresi ne da bismo je promenili već radi
       povećanja efikasnosti programa,
       ne bude, čak ni slučajno, izmenjena u funkciji.

91     */

93     void ispisi(const STUDENT* s)
95     {
97         printf("%s %s, %c, %.2f\n", s->ime, s->prezime, s->smer, s->prosek);
99     }

101     float najveći_prosek(STUDENT studenti[], int n)
103     {
104         float m;
105         int i;
106         /* Pretpostavimo da student sa indeksom 0 ima
           maksimalni prosek. */

```

### 3 Predstavljanje podataka

---

```
107     m = studenti[0].prosek;
    for(i=1;i<n;i++)
        if(m<studenti[i].prosek) /* Ako student sa indeksom i ima veci
            prosek od maksimalnog, */
109         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;
    int i;
119     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.
                    */
127                 s = studenti[i];
                    break;
129             }
    return s;
131 }

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:
```

```
153 - strukturu kao povratnu vrednost
    - prenos strukture preko pokazivaca u funkciju, s obzirom da ce se
      promeniti u funkciji

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

157 */
159 int main()
160 {
161     STUDENT studenti[MAXST];
162     int n;
163     int i;
164     float max_prosek;
165     STUDENT student_sa_max_prosekom;
166     int indeks;
167
168     /* printf("Unesi broj studenata:"); */
169     scanf("%d", &n);
170
171     if (n<0 || n>MAXST)
172     {
173         printf("Nekorektan unos\n");
174         return -1;
175     }
176
177     /* printf("Unesi podatke o studentima:"); */
178     for(i=0;i<n;i++)
179     {
180         /* printf("%d. student:\n", i); */
181         ucitaj(&studenti[i]);
182     }
183
184     printf("Studenti sa R smerom:\n");
185     for(i=0;i<n;i++)
186         if(studenti[i].smer == 'R')
187             ispisi(&studenti[i]);
188     printf("-----\n");
189
190     /* b)1)
191
192     Stampamo podatke o svim studentima sa
193     maksimalnim prosekom.
194     */
195
196     max_prosek = najveći_prosek(studenti, n);
197     printf("Svi studenti koji imaju maksimalni prosek:");
198     for(i=0;i<n;i++)
199         if(studenti[i].prosek==max_prosek)
```

### 3 Predstavljanje podataka

---

```
203         ispisi(&studenti[i]);
204
205     /* b)2) */
206     student_sa_max_prosekom = prvi_student_sa_najvecim_prosekom(
207         studenti,n,max_prosek);
208
209     printf("Prvi student u nizu sa najvecim prosekom: ");
210     ispisi(&student_sa_max_prosekom);
211
212     /* b)3) */
213     poslednji_student_sa_najvecim_prosekom(studenti,n,max_prosek,&
214         student_sa_max_prosekom);
215
216     printf("Poslednji student u nizu sa najvecim prosekom: ");
217     ispisi(&student_sa_max_prosekom);
218
219     return 0;
220 }
```

#### Rešenje 3.111

```
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
9      {
10         char s[21];
11         int duzina;
12     }REC;
13
14 Na primer, ako su unesene sledece reci:
15 Danas imamo ispit iz programiranja1.
16 Nadam se da nece biti tesko!
17 onda ispis izgleda ovako:
18
19     Danas
20     imamo
21     ispit
22     iz
23     programiranja1.
24     Nadam
25     se
26     da
27     nece
28     biti
29     tesko!
30
31 Program realizovati kroz sledece funkcije:
32 a) Funkciju za ucitavanje jedne reci u strukturu REC.
```

```

29 b) Funkciju za učitavanje niza struktura koja vraća dimenziju niza
30 c) Funkciju koja određuje maksimalnu dužinu reči u datom nizu
31 d) Funkciju koja ispisuje reči u traženom formatu

33 Mozemo pretpostaviti da nijedna rec ne sadrži više od 30 karaktera i
    da neće biti
    uneto više od 1000 reči.

35 */

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

43 typedef struct rec
44 {
45     char s[MAX];
46     int duzina;
47 }REC;

49 void ucitaj_rec(REC* rec)
50 {
51     scanf("%s", rec->s);
52     rec->duzina = strlen(rec->s);
53 }

55 /*
56  U funkciji ucitaj_niz_reci argument n označava broj
57  elemenata niza reči, koji će biti poznat tek po
58  završetku funkcije. Ova promenljiva će dobiti svoju
59  vrednost u funkciji i zbog toga mora biti prenesena
60  preko pokazivaca.

61 */

63 void ucitaj_niz_reci(REC reci[], int* pn, int granica)
64 {
65     int i=0;
66     do
67     {
68         ucitaj_rec(&reci[i]);
69         i++;
70     }
71     while(reci[i-1].duzina>0 && (i-1)<granica);

72     /*
73      S obzirom da se promenljiva i učitava
74      pre ispitivanja uslova, uslov ispitujemo
75      za rec sa indeksom i-1

76 */

```

```
81     *pn = i-1;
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 }
109
110 /*
111  * Da bismo realizovali ispis u trazenom formatu, pre
112  * svake reci ispisujemo onoliko razmaka koliko iznosi
113  * razlika maksimalne duzine i duzine date reci.
114  */
115
116 void ispis(REC reci[], int n, int max_d)
117 {
118     int i, j;
119
120     for(i=0; i<n; i++)
121     {
122         for(j=0; j<max_d-reci[i].duzina; j++)
123             printf(" ");
124         printf("%s\n", reci[i].s);
125     }
126 }
127
128 int main(int argc, char* argv[])
129 {
130     REC reci[MAXRECI];
```

```

131     int najveca_duzina;
132     int n;

133     ucitaj_niz_reci(reci, &n, MAXRECI);
134     najveca_duzina = max_duzina(reci,n);
135     ispis(reci, n, najveca_duzina);

137     return 0;
}

```

### Rešenje 3.112

```

/*
2   Napisati program koji izracunava prosečnu cenu jedne potrosacke
3   korpe. Potrosacka korpa se sastoji od broja kupljenih artikala i
4   niza kupljenih artikala. Svaki artikal odredjen je svojim nazivom,
5   kolicinom i cenom. Program treba da ucita broj potrosaca n (
6   najvise 100),
7   zatim podatke za n potrosackih korpi i da na osnovu učitanih
8   podataka
9   izracuna prosečnu cenu potrosacke korpe. Učitavanje se vrši sa
10  standardnog
11  ulaza pri čemu se prvo zadaje broj artikala, a zatim za svaki
12  artikal naziv,
13  kolicina i cena. Mozemo pretpostaviti da nijedan
14  potrosac neće kupiti više od 20 artikala, kao i da naziv svakog
15  artikla
16  sadrži maksimalno 30 karaktera.
17
18 */
19
20 #include <stdio.h>
21 #define MAXART 20
22 #define MAXPOT 100
23 #define MAXNAZIV 31
24
25 typedef struct artikal
26 {
27     char naziv[MAXNAZIV];
28     int kolicina;
29     float cena;
30 } ARTIKAL;
31
32 typedef struct korpa
33 {
34     int br_art;
35     ARTIKAL artikli[MAXART];
36 } KORPA;
37
38 /*
39 Funkcija ucitaj_artikal ucitava podatke za jedan

```

### 3 Predstavljanje podataka

---

```
36     artikal i vraca 1 ako je učitavanje bilo uspesno
37     a 0 u suprotnom. Učitavanje je neuspesno ukoliko
38     kolicina nekog artikla ili njegova cena nisu pozitivni
39     brojevi.
40
41     S obzirom da funkcija učitaj_artikal treba da vrati
42     dve vrednosti (ucitanu strukturu i indikator uspesnosti),
43     strukturu ARTIKAL prenosimo preko pokazivaca a
44     indikator uspesnosti vracamo kao povratnu vrednost.
45
46     */
47
48     int učitaj_artikal(ARTIKAL* a)
49     {
50         scanf("%s", a->naziv);
51         scanf("%d", &a->kolicina);
52
53         if (a->kolicina<=0)
54         {
55             printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina);
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     */
76     float izracunaj_racun(const KORPA* k)
77     {
78         int i;
79         float racun=0;
80         for(i=0;i<k->br_art;i++)
81             racun+=k->artikli[i].kolicina * k->artikli[i].cena;
82         return racun;
83     }
84
85     /*
86     Pri učitavanju korpe, zadaje se broj artikala a zatim
```



```
86     podaci za svaki artikal.

88     Funkcija ucitaj_korpu vraca 1 ako je učitavanje uspesno
89     i 0 u suprotnom. Do neuspesnog učitavanja može doći
90     ako broj artikala u korpi nije pozitivan ili ako dodje
91     do neuspesnog učitavanja nekog artikla.
92 */

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     return 1;
107 }

110 /*
111 Funkcija ucitaj_niz_korpi učitava podatke
112 za niz od n potrosackih korpi. Funkcija
113 vraca 1 ako je učitavanje uspesno i 0 ako
114 nije. Učitavanje je neuspesno ukoliko ne uspe
115 učitavanje 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     return 1;
125 }

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 */

136 void stampaj_racun(const KORPA* k)
```

### 3 Predstavljanje podataka

---

```
138 {
139     int i,j;
140     for(i=0;i<k->br_art;i++)
141         printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
            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 prosecnu cenu
165  potrosacke korpe za dati niz potrosackih
166  korpi
167 */
168 float prosek(KORPA korpe[], int n)
169 {
170     int i;
171     float p;
172
173     for(i=0;i<n;i++)
174         p+=izracunaj_racun(&korpe[i]);
175
176     return p/n;
177 }
178
179 int main()
180 {
181     int n;
182     KORPA korpe[MAXPOT];
183
184     printf("Unesi broj potrosackih korpi:");
185     scanf("%d", &n);
186
187     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 3.113

```

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
    zadat i vraća 1 ako jeste a 0 u suprotnom. Podrazumevamo da je
15  izraz korektno zadat ako operacija odgovara +,-,* ili / i u
    slucaju
    deljenja drugi operand je razlicit od 0.

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

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

25  d) Napisati funkciju koja stampa dati izraz infiksno, u obliku
    x o y = vr. Na primer, za izraz + 4 17 ispis treba
27  da bude 4+17=21
29

```

### 3 Predstavljanje podataka

---

```
31      e) Napisati glavni program koji ucitava prirodan broj n<1000 a
      zatim n izraza
      u notaciji
33      + 4 17
      - 8 -16
35      Program treba da ispise maksimalnu vrednost medju unetim izrazima
      i da ispise one
      izraze cija je vrednost manja od polovine maksimalne vrednosti.
37
39
41      */
43      #include <stdio.h>
45      #define MAX 1000
47      typedef struct izraz
48      {
49          char o;
50          int x;
51          int y;
52      } IZRAZ;
53
54      /*
55      Funkcija korektan_izraz vraca 1 ako je izraz korektan a 0
56      u suprotnom. Izraz je korektan ukoliko se sastoji od
57      aritmetickih operacija +,-,* ili /, i ukoliko je u slucaju
58      operacije deljenja drugi operand razlicit od nule.
59      */
60      int korektan_izraz(const IZRAZ* izraz)
61      {
62          if(izraz->o!='+' && izraz->o!='-' && izraz->o!='*' && izraz->o!='/'
63              ')')
64          {
65              printf("Nedozvoljena operacija!\n");
66              return 0;
67          }
68          if(izraz->o=='/' && izraz->y==0)
69          {
70              printf("Deljenje nulom!\n");
71              return 0;
72          }
73          return 1;
74      }
75
76      /*
77      Promenljiva izraz ce se promeniti u funkciji
78      vrednost tako sto ce njenom neinicijalizovanom
79      polju vr biti dodeljena vrednost izraza. Zbog
80      toga ovu promenljivu funkciji prosledjujemo
81      po adresi, preko pokazivaca
```

```
81  */
82  int vrednost(const IZRAZ* izraz)
83  {
84      int v;
85
86      switch (izraz->o)
87      {
88          case '+':
89              v=izraz->x+izraz->y;
90              break;
91          case '-':
92              v=izraz->x-izraz->y;
93              break;
94          case '*':
95              v=izraz->x*izraz->y;
96              break;
97          case '/':
98              v=izraz->x/izraz->y;
99              break;
100     }
101     return v;
102 }
103
104 /*
105  Promenljiva izraz ce se promeniti u funkciji
106  ucitaj_izraz tako sto ce njenim neinicijalizovanim
107  poljima o,x,y biti dodeljene vrednosti ucitane
108  sa standardnog ulaza. Zbog toga ovu promenljivu
109  funkciji prosledjujemo po adresi, preko pokazivaca.
110
111  S obzirom da učitavanje karaktera nije prvo
112  učitavanje koje se obavlja u programu, funkcijom
113  getchar() "pokupimo" karakter kojim razdvajamo
114  unos karaktera od prethodnog unosa (najcesce blanko
115  znak)
116
117  */
118
119  int ucitaj_izraz(IZRAZ* izraz)
120  {
121      getchar();
122      scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
123      if (!korektan_izraz(izraz))
124          return 0;
125      return 1;
126  }
127
128
129  void stampaj_izraz(const IZRAZ* izraz)
130  {
131
```

### 3 Predstavljanje podataka

---

```
    printf("%d %c %d = %d\n", izraz->x, izraz->o, izraz->y, vrednost(
        izraz));
133 }

135 int max_vr(IZRAZ izrazi[], int n)
136 {
137     int i;
138     int max;
139     /* Trazimo maksimalnu vrednost izraza */
140     max=vrednost(&izrazi[0]);
141
142     /* U petlji... */
143     for(i=1; i<n; i++)
144     /* Ako je ona veca od maksimalne: */
145         if(vrednost(&izrazi[i])>max)
146             /* Azuriramo max: */
147             max=vrednost(&izrazi[i]);
148     return max;
149 }

151 int main()
152 {
153     int n;
154     IZRAZ izrazi[MAX];
155     int max;
156     int i;
157
158     /* Ucitavamo broj izraza: */
159     scanf("%d", &n);
160     if(n<0 || n>MAX)
161     {
162         printf("Nekorektna vrednost broja n!\n");
163         return -1;
164     }
165
166     /* U petlji ucitavamo jedan po jedan izraz: */
167     for(i=0; i<n; i++)
168         if(ucitaj_izraz(&izrazi[i])==0)
169         {
170             printf("Nekorektan unos\n");
171             return -1;
172         }
173
174     printf("Svi izrazi:\n");
175     for(i=0; i<n; i++)
176         stampaj_izraz(&izrazi[i]);
177
178     max = max_vr(izrazi, n);
179
180     printf("Maksimalna vrednost izraza:%d\n", max);
181 }
```

```
183     printf("Izrazi cija je vrednost manja od polovine maksimalne  
        vrednosti:\n");  
  
185     for(i=0; i<n; i++)  
        if(vrednost(&izrazi[i])<max/2)/* Ako je vrednost tekuceg izraza  
187         manja od polovine maksimalne, ispisujemo ga. */  
            stampaj_izraz(&izrazi[i]);  
  
189     return 0;  
}
```

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125

Rešenje 3.125





# 4

## Ulaz i izlaz programa

### 4.1 Standardni tokovi

### 4.2 Argumenti komandne linije

### 4.3 Datoteke

**Zadatak 4.1** Tekst

[Rešenje [4.1](#)]

**Zadatak 4.2** Tekst

[Rešenje [4.2](#)]

**Zadatak 4.3** Tekst

[Rešenje [4.3](#)]

**Zadatak 4.4** Tekst

[Rešenje [4.4](#)]

**Zadatak 4.5** Tekst

[Rešenje [4.5](#)]

**Zadatak 4.6** Tekst

[Rešenje [4.6](#)]

**Zadatak 4.7** Napisati program koji prebrojava mala slova u datoteci *test.txt*.

*Primer 1*

```
TEST.TXT
Abcd EFGH+ijKLMN

IZLAZ:
Broj malih slova je: 5
```

*Primer 2*

```
TEST.TXT
PrograMiranje

IZLAZ:
Broj malih slova je: 11
```

[Rešenje 4.33]

**Zadatak 4.8** Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*.

*Primer 1*

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

[Rešenje 4.33]

**Zadatak 4.9** Kao argumenti komandne linije se zadaju ime datoteke i ceo broj  $k$ . Napisati program koji na standardni izlaz ispisuje sve linije zadate datoteke čija je dužina veća od  $k$ . Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera.

*Primer 1*

```
POKRETANJE: ./a.out test.txt 7
TEST.TXT
Teme koje su obradjuivane:
Petlje
Funkcije
Nizovi
Strukture

IZLAZ:
Teme koje su obradjuivane:
Funkcije
Strukture
```

*Primer 2*

```
POKRETANJE: ./a.out test.txt

IZLAZ:
Greska: Pogresan broj argumenata!
```

[Rešenje 4.33]

**Zadatak 4.10** Napisati program koji prebrojava koliko se linija datoteke *ulaz.txt* završava niskom  $s$  koja se učitava sa standardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća od 80 karaktera, kao i da dužina niske  $s$

neće biti veća od 20 karaktera.

#### Primer 1

```

ULAZ.TXT
abcde abcde
abcde aab
abcde abcde abcde
abcde abcde Aab
abcde abcde ab
abcde abcde abcde abcde

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 3

```

#### Primer 2

```

ULAZ.TXT
abcde abcde
abcde
abcde abcde AB

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: ab
Broj linija: 0

```

[Rešenje 4.33]

**Zadatak 4.11** Napisati program koji pronalazi maksimum brojeva zapisanih u datoteci *brojevi.txt*.

#### Primer 1

```

BROJEVI.TXT
2.36 -16.11 5.96 8.88
-265.31 54.96 38.4

IZLAZ:
Najveci broj je: 54.96

```

[Rešenje 4.33]

**Zadatak 4.12** U datoteci *studenti.txt* se nalaze informacije o studentima: prvo broj studenata, a zatim u pojedinačnim linijama korisničko ime i pet poslednjih ocena koje je student dobio. Napisati program koji pronalazi studenta koji je ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da broj studenata neće biti veći od 100.

#### Primer 1

```

STUDENTI.TXT
mr15239 10 9 9 8 10
m114005 8 8 9 8 10
m115112 9 8 8 7 10
mr15007 10 10 10 10 10
mn13208 7 7 9 6 10

IZLAZ:
korisnicko ime: mr15007, prosek ocena: 10.00

```

[Rešenje 4.33]

**Zadatak 4.13** U datoteci *tacke.txt* se nalazi prvo broj tačaka, a zatim u pojedinačnim linijama  $x$  i  $y$  koordinate tačke. Napisati program koji u datoteku *rastojanja.txt* upisuje rastojanje svake od pročitanih tačaka od koordinatnog početka, a na standardni izlaz koordinate tačke koja je najudaljenija. Koristiti strukturu *Tacka* sa poljima  $x$  i  $y$ , kao i funkciju kojom se računa rastojanje. Pretpostaviti da broj tačaka u datoteci neće biti veći od 50.

### Primer 1

```
TACKE.TXT
4
11 -2
3 5
8 -8
0 4

RASTOJANJA.TXT
11.18
5.29
11.31
4.00

IZLAZ:
Najudaljenija je tačka: 8 -8
```

### Primer 1

```
TACKE.TXT
-2
0 0
9 -8

IZLAZ:
Greska: Nedozvoljen broj tacaka!
```

[Rešenje 4.33]

**Zadatak 4.14** Napisati program koji za reč  $s$  maksimalne dužine 20 karaktera koja se zadaje sa standardnog ulaza u datoteku *rotacije.txt* upisuje sve rotacije reči  $s$ .

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite rec: abcde

ROTACIJE.TXT
abcde
bcdea
cdeab
deabc
eabcd
```

[Rešenje 4.33]

**Zadatak 4.15** Napisati program koji linije koji se učitavaju sa standardnog ulaza sve do kraja ulaza prepisuje u datoteku *izlaz.txt* i to, ako je prilikom pokretanja zadata opcija  $-v$  ili  $-V$  samo one linije koje počinju velikim slovom, ako je zadata opcija  $-m$  ili  $-M$  samo one linije koje počinju malim slovom, a ako je opcija izostavljena sve linije. Pretpostaviti da linije neće biti duže od 80 karak-

tera.

#### Primer 1

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

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

#### Primer 2

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

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

#### Primer 3

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

[Rešenje 4.33]

**Zadatak 4.16** Sa standardnog ulaza učitavaju se imena dve tekstualne datoteke i jedan karakter. Napisati program koji prepisuje datoteku čije se ime navodi kao prvo u datoteku čije ime se navodi kao drugo. Ukoliko je učitani karakter u program prilikom prepisivanja treba da zamenjuje sva mala slova velikim, a ukoliko je učitani karakter 1 sva velika slova se zamenjuju malim. U slučaju greške ispisati -1. Greška može biti neuspešno otvaranje datoteke ili pogrešno zadati karakter. Maksimalna dužina naziva datoteke je 20 karaktera.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
ulaz.txt izlaz.txt u
ULAZ.TXT
danas je lep dan
i Ja zelim
da postanem programer
IZLAZ.TXT
DANAS JE LEP DAN
I JA ZELIM
DA POSTANEM PROGRAMER
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
prva.dat druga.dat l
PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120
DRUGA.DAT
cena soka je 30
cena vina je 150
cena limunade je 200
cena sendvica je 120
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
  primer.c prazna.txt V
PRIMER.C
#include <stdio.h>
int main()
{
}
PRAZNA.TXT

IZLAZ:
-1
```

[Rešenje 4.33]

**Zadatak 4.17** Sastaviti program koji sa standardnog ulaza prima ime datoteke koju treba otvoriti. Ispisati (na standardnom izlazu) koja cifra (među svim ciframa koje se pojavljuju u datoteci) ima najveći broj pojavljivanja. U slučaju greške pri otvaranju datoteke ispisati -1. Ukoliko nema cifara u datoteci ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  ulaz.txt
ULAZ.TXT
  danas je lep dan
  i Ja zelim
  da postanem programer
IZLAZ:
-1
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  prva.dat druga.dat l
PRVA.DAT
  Cena soka je 30
  Cena vina je 150
  Cena limunade je 200
  Cena sendvica je 120
IZLAZ:
0
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
  primer.c
PRIMER.C
#include <stdio.h>
int main()
{
}
PRAZNA.TXT

IZLAZ:
-1
```

[Rešenje 4.33]

**Zadatak 4.18** Prvi red datoteke `matrice.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice A. Svaki

sledeći red sadrži po jednu vrstu matrice. Napisati program koji pronalazi sve elemente matrice A koji su jednaki zbiru svih svojih susednih elemenata i štampa ih u obliku

(broj vrste, broj kolone, vrednost elementa).

U slučaju greške prilikom otvaranja datoteke ispisati -1. Pretpostaviti da je sadržaj datoteke ispravan.

#### Primer 1

```
MATRICE.TXT
1 2 3 4
7 2 15 -3
-1 3 1 3
IZLAZ:
(1, 0, 7)
(1, 2, 15)
```

[Rešenje 4.33]

**Zadatak 4.19** Napisati program koji za dve datoteke čija su imena data kao prvi i drugi na standardnom ulazu, radi sledeće: za cifru u prvoj datoteci, u drugu datoteku se upisuje 0, za slovo se upisuje 1, a za sve ostale karaktere se upisuje 2. Maksimalna dužina naziva datoteka je 20 karaktera.

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
prva.dat druga.dat
PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120
DRUGA.DAT
111121111121120021111211111211200021111211111111211200021111211111112112000
```

[Rešenje 4.33]

**Zadatak 4.20** Ako je data tekstualna datoteka `plain.txt` napraviti tekstualnu datoteku `sifra.txt` tako što se svako slovo zamenjuje svojim prethodnikom (ciklično) suprotne velicine 'b' sa 'A', 'B' sa 'a', 'a' sa 'Z', 'A' sa 'z', itd. Podrazumevati da se na sistemu koristi tabela karaktera ASCII.

[Rešenje 4.33]

**Zadatak 4.21** Sa standardnog ulaza se učitava ime tekstualne datoteke i prirodan broj k. Podrazumeva se da zadata datoteka sadrži samo slova i beline i

## 4 Ulaz i izlaz programa

---

da je svaka reč iz datoteke dužine najviše 100. Program treba da učitava reči iz datoteke, da svaku reč rotira za  $k$  mesta i da tako dobijenu reč upiše u datoteku čije je ime `rotirano.txt`. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.33]

**Zadatak 4.22** Napisati program koji u datoteku `izlaz.txt` prepisuje sve reči iz datoteke `ulaz.txt` čiji je zbir ascii kodova slova strogo veći od 1000. Reči su odvojene prazninama i nisu duže od 200 karaktera.

### Primer 1

```
ULAZ.TXT
Sa standardnog ulaza unosi se neoznaceni
ceo broj. Formirati novi broj koji se dobija
izbacivanjem svake druge cifre iz polaznog
broja.
IZLAZ.TXT
standardnog izbacivanjem
```

### Primer 2

```
ULAZ.TXT
i sada jedan kratak primer
p1: 1234567890
p2: ABCDEFGHIJ
p3: abcdefghij
IZLAZ.TXT
abcdefghij
```

### Primer 3

```
ULAZ.TXT
konstruisanje test-primer sa
i dugackim recima kao prestolonaslednik
brojevima1234567890
IZLAZ.TXT
konstruisanje test-primer sa
prestolonaslednik
brojevima1234567890
```

### Primer 4

```
ULAZ.TXT
ima jos dugackih reci: predskazanje,
potom
nelogicnosti, zanemarivati, odugovlaciti, a ima
i i malih reci koje su kratke
predosecaj
IZLAZ.TXT
predskazanje, nelogicnosti,
zanemarivati, odugovlaciti,
predosecaj
```

[Rešenje 4.33]

**Zadatak 4.23** U datoteci `razno.txt` nalazi se tekst. U datoteku `palindromi.txt` prepisati sve reči iz datoteke `razno.txt` koje su palindromi. Reč je palindrom ako se čita isto sa leve i desne strane. Za reč smatramo niz karaktera koji se nalazi između belina i koji nije duži od 200 karaktera. Dozvoljeno je korišćenje specifikatora za čitanje reči. Maksimalan broj reči nije poznat. U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

### Primer 1

```
RAZNO.TXT
Ana i melem su primeri palindroma.
PALINDROMI.TXT:
Ana i melem
```

### Primer 2

```
RAZNO.TXT
jabuka neven pomorandza kuk
Oko kapAk pero radar caj
PALINDROMI.TXT:
neven kuk\datoteka{Oko kapAk radar}
```



[Rešenje 4.33]

**Zadatak 4.24** U datoteci čije se ime navodi na standardnom ulazu programa nalazi se broj  $n$ , a zatim i  $n$  reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

(a) ispisuje ga [3],

(b) iz njega uklanja sve duplikate i u datoteku `rez.txt` ispisuje transformisani niz [4]

U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT
  12 jha14 hahaha deda mraz deda
  mraz deda deda jase konj konj konj
IZLAZ:
  jha14 hahaha deda mraz deda mraz deda
  deda jase konj konj konj
REZ.TXT:
  jha14 hahaha deda mraz jase konj
```

*Primer 2*

```
INTERAKCIJA SA PROGRAMOM:
  dat2.txt
DAT2.TXT
  14
  so secer supa so ljuto secer kiselo slatko
  ljuto
  paprika, ljuta paprika, ljuto dete
IZLAZ:
  so secer supa so ljuto secer kiselo slatko
  ljuto paprika, ljuta paprika, ljuto dete
REZ.TXT:
  so secer supa ljuto kiselo slatko
  paprika, ljuta dete
```

[Rešenje 4.33]

**Zadatak 4.25** U datoteci čije se ime navodi na standardnom ulazu programa nalazi se broj  $n$ , a zatim i  $n$  reči (dužine najviše 50 karaktera). Napisati program koji učitava ovaj niz i

(a) ispisuje ga, [3]

(b) u datoteku `rez.txt` upisuje sve reči koje sadrže prvu reč i podvlaku. [4]

## 4 Ulaz i izlaz programa

U slučaju greške ispisati -1. Maksimalna dužina naziva datoteka je 20 karaktera.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
  dat1.txt
DAT1.TXT
  7 rec Opet _rec Reci rec_enica
  Dva recica_
IZLAZ:
  rec Opet _rec Reci rec_enica
  Dva recica_
REZ.TXT:
  _rec rec_enica recica_
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
  dat2.txt
DAT2.TXT
  11 Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
  suncanica.
IZLAZ:
  Sunce sija iznad grada
  Sunce_Moje Jedan Dva Su_nce Sve Sunce123_123
  suncanica.
REZ.TXT:
  Sunce_Moje Sunce123_123
```

[Rešenje 4.33]

**Zadatak 4.26** Imena dve datoteke se zadaje na standarnom ulazu. U prvoj datoteci navedena je rec `r` i niz linija. Napisati program koji u drugu datoteku upisuje sve linije u kojima se rec `r` pojavljuje bar `n` puta, gde je `n` prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu `broj_pojavljivanja: linija`. Linije brojati počevši od 1. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.33]

**Zadatak 4.27** Napisati program koji poredi dva fajla i ispisuje redni broj linija u kojima se fajlovi razlikuju. Imena fajlova se zadaju kao argumenti komandne linije. U slučaju neuspešnog otvaranja datoteka ispisati poruku o grešci. Pretpostaviti da je maksimalna dužina reda u datoteci 200 karaktera. Ukoliko nisu zadati potrebni argumenti komadne linije ispisati poruku o grešci. Linije brojati počevši od 1.

### Primer 1

```
POKRETANJE: ./a.out ulaz.txt izlaz.txt
ULAZ.TXT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ.TXT:
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke iste
IZLAZ:
```

### Primer 2

```
POKRETANJE: ./a.out primer1.dat primer2.dat
PRIMER1.DAT
  danas vezbamo
  analizu
  ovo je primer kad
  su datoteke razlicite
PRIEMR2.DAT
  danas vezbamo
  programiranje
  ovo je primer kad su
  datoteke razlicite
IZLAZ:
  2 3 4
```

*Primer 3*

```

POKRETANJE: ./a.out prva.dat
IZLAZ:
greska

```

*Primer 2*

```

POKRETANJE: ./a.out prva.dat druga.dat
PRVA.DAT
ovo je primer
kada su
datoteke
razlicite duzine
DRUGA.DAT
kada su
programiranje
datoteke
razlicite
duzine
i kada treba ispisati broj
tih redova
IZLAZ:
1 4 5 6 7

```

[Rešenje 4.33]

**Zadatak 4.28** Definirati strukturu

```

typedef struct{
    unsigned int a, b;
    char ime[5];
}_pravougaonik;

```

kojom se opisuje pravougaonik dužinama svojih stranica i imenom. Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava pravougaonike (nepoznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine među pravougaonicima koji nisu kvadrati. U slučaju unosa nekorektnih dužina stranica pravougaonika ili neko-rektne vrednosti broja *n*, ispisati -1 i odmah prekinuti izvršavanje programa. Maksimalan broj pravougaonika je 200.

*Primer 1*

```

POKRETANJE: ./a.out pravougaonici.dat
PRAVOUGAONICI.DAT
2 4 p1
3 3 p2
1 6 p3
IZLAZ:
p2 8

```

*Primer 2*

```

POKRETANJE: ./a.out dva.dat
DVA.DAT
5 2 pm
4 7 pv
IZLAZ:
28

```

### Primer 3

```
|| POKRETANJE: ./a.out tri.dat
|| TRI.DAT
|| 5 5 m
|| 3 3 s
|| 8 8 xl
|| IZLAZ:
|| m s xl
```

### Primer 4

```
|| POKRETANJE: ./a.out primerx.dat
|| PRIMERX.DAT
|| 9 7 p
|| IZLAZ:
|| 63
```

### Primer 5

```
|| POKRETANJE: ./a.out prazna.dat
|| PRAZNA.DAT
|| IZLAZ:
```

[Rešenje 4.33]

**Zadatak 4.29** Ime datoteke dato je kao argument komandne linije. U datoteci se nalaze otvorene i zatvorene zagrade i još nekakav tekst. Proveriti da li su zagrade pravilno uparene. Npr. `ab( cd) ..` odgovor je `jesu`, a `..)ba()` odgovor je `nisu`. Ukoliko nisu zadati svi argumenti komandne linije ispisati poruku o grešci.

### Primer 1

```
|| POKRETANJE: ./a.out
|| zagrade.txt
|| ZAGRADE.TXT
|| ab( cd) ..
|| ((3+4)*5+1)*9
|| IZLAZ:
|| jesu
```

### Primer 2

```
|| POKRETANJE: ./a.out
|| primer2.dat
|| PRIMER2.DAT
|| (7+8
|| nisu(
|| uparene
|| IZLAZ:
|| nisu
```

### Primer 3

```
|| POKRETANJE: ./a.out
|| primer3.dat
|| PRIMER3.DAT
|| )) 7 + 6 ((
|| IZLAZ:
|| nisu
```

### Primer 4

```
|| POKRETANJE: ./a.out
|| IZLAZ:
|| greska
```

[Rešenje 4.33]

**Zadatak 4.30** Napraviti strukturu `STUDENT` koja sadrži:

- ime (u polju se čuva ime i prezime studenta, napr. "Marko Markovic", maksimalna dužina polja je 100 karaktera),
- oc (sadrži najviše 10 ocena studenta)

- `br_ocena` (ukupan broj ocena za studenata)
- `pr_oc` (prosečna ocena)

U datoteci se nalaze podaci o studentima. Za svakog studenta unosi se ime i prezime razdvojeno razmakom (uputstvo: može se korisiti `strcat` da spoji ime i prezime koji se mogu pročitati sa specifikatorom `%s`), a potom ocene koje se završavaju sa 0. Pronaći studenta koji ima najveći prosek i ispisati sve njegove podatke (prosek ispisati na 2 decimale). Maksimalan broj studenta je 100. Ime datoteke se zadaje kao argument komandne linije.

*Primer 1*

```
|| POKRETANJE: ./a.out
   student.txt
|| STUDENTI.TXT
   Marko Markovic 5 6 7 8 9 0
   Jelena Jankovic 10 10 10 0
   Filip Viskovic 10 9 8 7 6 0
   Jana Peric 10 10 9 9 8 8 7 7
   0
|| IZLAZ:
   Jelena Jankovic 10 10 10 0
   10.00
```

*Primer 2*

```
|| POKRETANJE: ./a.out
|| IZLAZ:
   greska
```

[Rešenje 4.33]

**Zadatak 4.31**

- Napisati C funkciju `int unesiSkup(char *s, FILE* f)` kojom se unosi skup elemenata iz datoteke `F`. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiđe na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspesno učitani.
- Napisati funkciju `void prebroj(char *s, int *br_slova, int *br_cifara)` kojom se određuje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.
- Napisati glavni program gde se unose podaci o skupu elemenata. Ime datoteke se zadaje kao argument komandne linije. Na standardni izlaz ispisati informacije o broju slova i cifara (koristiti funkcije pod a) i b)).

### Primer 1

```
POKRETANJE: ./a.out skup.txt
SKUP.TXT
  abc56ighj9012hjFGHH
IZLAZ:
  broj slova: 13
  broj cifara: 6
```

### Primer 2

```
POKRETANJE: ./a.out skup2.txt
SKUP2.TXT
  ovdeimamo$dolar
IZLAZ:
  broj slova: 9
  broj cifara: 0
```

### Primer 3

```
POKRETANJE: ./a.out skup3.txt
SKUP3.TXT
  broj3
  broj5
IZLAZ:
  broj slova: 4
  broj cifara: 1
```

### Primer 4

```
POKRETANJE: ./a.out
IZLAZ:
  greska
```

[Rešenje 4.33]

### Zadatak 4.32 Definirati strukturu

```
typedef struct{
    int x;
    int y;
    int z;
} vektor;
```

kojom se opisuje trodimenzioni vektor. U datoteci **vektori.txt** nalazi se nepoznati broj vektora (maksimalno ih može biti 200). Učitati ih u niz i ispisuje na standardnom izlazu koordinate vektora sa najvećom dužinom. Dužina vektora se izračunava po formuli:

$$|v| = \sqrt{x^2 + y^2 + z^2}$$

U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

### Primer 1

```
VEKTORI.TXT
  2
  4 -1 7
  3 1 2
IZLAZ:
  4 -1 7
```

### Primer 2

```
VEKTORI.TXT
  67
IZLAZ:
  -1
```

### Primer 3

```
VEKTORI.TXT
  3
  0 0 0
  0 1 0
  1 0 0
IZLAZ:
  0 1 0
```

*Primer 4*

```

VEKTORI.TXT
4
3 0 1
4 5 2
1 0 0
2 -1 2
IZLAZ:
4 5 2

```

[Rešenje 4.33]

**Zadatak 4.33** Prvi red datoteke `ulaz.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice  $A$ . Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji nalazi i štampa sve četvorke oblika  $(A(i,j), A(i+1,j), A(i,j+1), A(i+1,j+1))$  u kojima su svi elementi međusobno različiti.

[Rešenje 4.33]

## 4.4 Rešenja

## Rešenje 4.1

```

1  /*
   Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
3  datoteku izlaz.txt karakter po karakter.
   */
5
7  #include <stdio.h>
   #include <stdlib.h>
9
11 int main()
   {
13     int c;
       FILE *ulaz, *izlaz;
15
17     /*
       Promenljive ulaz i izlaz predstavljaju
       pokazivace na ugradjenu strukturu FILE.
       Unutar ove strukture nalaze se polja neophodna
       za rad sa datotekama.
19
       Kada zelimo da radimo sa nekom datotekom,
       moramo je prvo otvoriti. Ugradjena funkcija
21

```

```
23     fopen(dat, mode) otvara datoteku sa nazivom
24     dat. Datoteka moze biti otvorena za citanje,
25     pisanje ili nadovezivanje, sto odredjuje
26     argument mode koji moze imati vrednost "r" (read),
27     "w"(write) ili "a"(append).
28 */
29
30     ulaz=fopen("ulaz.txt", "r");
31
32     /*
33     Do neuspesnog otvaranja datoteke moze doci
34     ukoliko ne postoji datoteka sa datim nazivom
35     ili je putanja do datoteke pogresna. U tom
36     slucaju, funkcija fopen vraca pokazivac na NULL
37     i tada treba prijaviti gresku. Datoteka stderr
38     predstavlja standardnu datoteku u koju se upisuju
39     greske. Stderr je podrazumevano postavljen
40     na standardni izlaz.
41
42     Ugradjena funkcija exit prouzrokuje zavrsetak programa.
43     Argument ove funkcije je jedna od konstanti definisanih
44     u biblioteci stdlib.h koje pokazuju da li se program
45     zavrrio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).
46
47     */
48     if(ulaz==NULL)
49     {
50         fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke ulaz
51         .txt za citanje.\n");
52         exit(EXIT_FAILURE);
53     }
54
55     izlaz= fopen("izlaz.txt", "w");
56     if(izlaz==NULL)
57     {
58         fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke
59         izlaz.txt za citanje.\n");
60         exit(EXIT_FAILURE);
61     }
62
63     /*
64     Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
65     Povratna vrednost ove funkcije je ascii kod unetog
66     karaktera.
67
68     Funkcija fputc ispisuje karakter c u datoteku izlaz.
69
70     */
71     while((c=fgetc(ulaz))!=EOF)
72         fputc(c, izlaz);
73
74     /*
```



```

73     Nakon zavrsetka rada sa datotekama, neophodno ih je
       zatvoriti pomocu ugradjene funkcije fclose.
75     */
       fclose(ulaz);
       fclose(izlaz);
77     return 0;
}

```

## Rešenje 4.2

```

/*
2   Napisati program koji u datoteci cije se ime navodi kao prvi
   argument komandne linije odredjuje liniju maksimalne duzine i
4   ispisuje je na standarni izlaz. Ukoliko ima vise takvih linija,
   ispisati onu koja je leksikografski prva. Mozemo pretpostaviti
6   da datoteka ne sadrzi linije duze od 80 karaktera.
   */
8   #include <stdio.h>
   #include <stdlib.h>
10  #include <string.h>
   #define MAX_LEN 81
12
   int main(int argc, char* argv[])
14  {
       char linija[MAX_LEN];
       char max_linija[MAX_LEN];
16       int duzina;
       int max_duzina;
18
       FILE *ulaz, *izlaz;
20
       /*
22        Proveravamo da li poziv programa ima dovoljan broj argumenata.
       */
24       if(argc!=2)
       {
26           fprintf(stderr, "Greska: program se pokrece sa: %s
           ime_ulazne_datoteke\n", argv[0]);
           exit(EXIT_FAILURE);
28       }
30
       ulaz=fopen(argv[1], "r");
32       if(ulaz==NULL)
       {
34           fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke %s
           za citanje.\n", argv[1]);
           exit(EXIT_FAILURE);
36       }
38       /*

```

```

    Funkcija fgets ucitava jednu liniju teksta maksimalne duzine
    MAX_LEN
40    iz datoteke ulaz u string linija. Ukoliko učitavanje ne uspe (
    na primer,
    zato sto smo dosli do kraja datoteke), povratna vrednost ove
    funkcije
42    bice prazan pokazivac (NULL).
    */
44
    max_duzina=0;
46    while(fgets(linija, MAX_LEN, ulaz)!=NULL)
    {
48        duzina = strlen(linija);
        /*
50        Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u
        petlju.
        Ovu promenljivu menjamo kada je duzina učitana linije
52        veca od max_duzina ili kada su jednake, ali je učitana
        linija
        leksikografski ispred trenutne linije sa maksimalnom duzinom
        .
54
        Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii
        kodova prva dva
56        razlicita karaktera stringova s1 i s2 na istim indeksima,
        ukoliko oni
        postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga
        osetljiva
58        na mala i velika slova (npr 'D' je leksikografski ispred 'p
        ').
        */
60
        if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija,
        max_linija)<0))
        {
64            strcpy(max_linija, linija);
            max_duzina=duzina;
66        }
    }
68
    /*
70    Funkcija fputs ispisuje string koji je njen prvi argument u
    datoteku
    koja je njen drugi argument. Sve funkcije za učitavanje iz
    datoteka i
72    upis u datoteke (fgetc, fputc, fgets, fputs, ...) mozemo
    koristiti
    i kada radimo sa standardnim ulazom i standardnim izlazom. Kao
    nazive
74    datoteka tada navodimo stdin i stdout.
    */

```

```

76     fputs(max_linija, stdout);
77
78     fclose(ulaz);
79     return 0;
80 }

```

### Rešenje 4.3

```

/*
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>
9 #include <stdlib.h>
10 #include <math.h>

12 /*
   Funkcija ucitaj_i_prebroj ucitava brojeve
14   iz datoteke na koju pokazuje f i prebrojava
   koliko je medju njima k-tocifrenih brojeva
16 */
17 int ucitaj_i_prebroj(FILE* f, int k)
18 {
19     int n;
20     int x;
21     int i;
22     int br;

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

26     /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
       */
27     if(n <= 0)
28     {
29         fprintf(stderr, "Greska: broj n mora biti prirodan\n");
30         exit(EXIT_FAILURE);
31     }

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

```

```
40     }
42     return br;
43 }
44
45 int broj_cifara(int x)
46 {
47     int br_c;
48
49     br_c=0;
50
51     /*
52      Do while petlja je pogodnija od petlji sa preduslovom
53      jer tacno racuna broj cifara i za broj 0.
54     */
55     do
56     {
57         br_c++;
58         x/=10;
59     } while(x);
60
61     return br_c;
62 }
63
64 int main(int argc, char* argv[])
65 {
66     int n;
67     int k;
68     FILE* f;
69     int br;
70
71     if(argc!=3)
72     {
73         fprintf(stderr, "Greska: program se pokrece sa: %s
74         naziv_datoteke k \n", argv[0]);
75         exit(EXIT_FAILURE);
76     }
77
78     f=fopen(argv[1], "r");
79
80     if(f==NULL)
81     {
82         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
83         \n", argv[1]);
84         exit(EXIT_FAILURE);
85     }
86
87     /* Argumenti komandne linije su stringovi. Da bismo konvertovali
88     string
89     u ceo broj koristimo ugradjenu funkciju atoi. */
90     k = atoi(argv[2]);
```

```

    if (k<=0)
90  {
        fprintf(stderr, "Greska: broj k mora biti prirodan\n");
92      exit(EXIT_FAILURE);
    }

94      printf("Broj %d-cifrenih brojeva u datoteci: %d\n", k,
        ucitaj_i_prebroj(f,k));

96      fclose(f);
98      return 0;
}

```

### Rešenje 4.4

```

1  /*
   U datoteci cije se ime navodi kao prvi argument komandne
3  linije navedena je rec r i niz linija. Napisati
   program koji u datoteku cije se ime navodi kao
5  drugi argument komandne linije upisuje sve linije
   u kojima se rec r pojavljuje bar n puta, gde je
7  n prirodan broj koji se unosi sa standardnog ulaza. Ispis
   treba da bude u formatu broj_pojavljivanja: linija.
9  */

11 #include <stdio.h>
   #include <stdlib.h>
13 #define MAXL 81
   #define MAXR 31

15 /*
17  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
39         je prestao da vazi uslov petlje, odnosno karakter t[j] je
        jednak '\0'. Ako vazi drugi slucaj, to znaci da se string
        t nalazi u stringu s pocev od indeksa i i potrebno je
        uvecati
41         brojac br.
        */
43         if (t[j]!='\0')
            br++;
45     }

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

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

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

69     out= fopen(argv[2], "w");
    if(out==NULL)
    {
71         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", argv[2] );
        exit(EXIT_FAILURE);
    }

73     printf("Unesi n:");
    scanf("%d", &n);

75
77     if(n<=0)
    {
79         fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
        exit(EXIT_FAILURE);
    }
83 }
```

```

85     fscanf(in, "%s", rec);
87
88     while(fgets(linija, MAXL, in) != NULL)
89     {
90         br = broj_pojavljivanja(linija, rec);
91         if (br >= n)
92             fprintf(out, "%d: %s\n", br, linija);
93     }
94     fclose(in);
95     fclose(out);
96     return 0;
97 }

```

### Rešenje 4.5

```

1  /* Program se pokrece tako sto se navedu nazivi dve datoteke(ulazna i
   izlazna) i opcije.
   U datoteci cije se ime navodi kao prvi argument komandne linije
   nalaze se podaci o razlomcima:
3  u prvom redu se nalazi broj razlomaka, a u svakom sledecem redu
   brojilac i imenilac jednog razlomka.
   Potrebno je kreirati strukturu koja opisuje razlomak i ucitati niz
   razlomaka
5  iz datoteke, a potom:
   a) ukoliko je navedena opcija x, upisati u datoteku cije je ime
   drugi argument komandne linije
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 najvise 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 #include <stdio.h>

```

```
27 #include <stdlib.h>
   #include <ctype.h>
29
   #define MAX 100
31
   typedef struct razlomak
33 {
       int br;
35       int im;
   } RAZLOMAK;
37
   /*
39   Funkcija ucitaj_razlomke ucitava 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   ucitavamo u promenljivu dim.
       Funkcija fscanf se koristi isto kao i funkcija scanf
45   uz dodatni prvi argument koji predstavlja naziv
47   datoteke iz koje se vrsi ucitavanje.
49   */
   int ucitaj_razlomke(RAZLOMAK niz[], int* dim, FILE* f)
51 {
       int i;
53
       fscanf(f, "%d", dim);
55       for (i=0; i<*dim; i++)
       {
           fscanf(f, "%d %d", &niz[i].br, &niz[i].im);
57           if (niz[i].im==0)
59               return 0;
       }
61       return 1;
   }
63
   RAZLOMAK reciprocni(RAZLOMAK* r)
65 {
       RAZLOMAK rec;
67       rec.im = r->br;
       rec.br = r->im;
69       return rec;
   }
71
   float vrednost(RAZLOMAK* r)
73 {
       return 1.0*r->br/r->im;
75 }

77 int main(int argc, char* argv[])
   {
```



```
79 FILE *in, *out;
   char c;
81 int i;
   int j;
83 int xoption=0;
   int yoption=0;
85 int dim;
   RAZLOMAK razlomci[MAX];
87 RAZLOMAK r;

89 /*
   Prilikom pokretanja programa se, pored naziva ulazne i izlazne
91 datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
   obe opcije, sto znaci da je minimalni broj argumenata 3.

93
   Moguci nacini pokretanja:
95     ./a.out ulaz.txt izlaz.txt -x
     ./a.out ulaz.txt izlaz.txt -y
97     ./a.out ulaz.txt izlaz.txt -yx
     ./a.out ulaz.txt izlaz.txt -xy
99
101 */

103 if(argc!=4)
   {
       fprintf(stderr, "Greska: program se pokrece sa: %s
105     ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n", argv[0]);
       exit(EXIT_FAILURE);
   }

107

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

115

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

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

125 if (argv[3][0] != '-')
   {
```

```

127     fprintf(stderr, "Greska u zadavanju opcija: program se pokrece
sa: %s ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n",
argv[0]);
    exit(EXIT_FAILURE);
129 }

131 /* Ostali karakteri mogu biti 'x' ili 'y'. U zavisnosti od date
opcije, postavljamo
vrednosti indikatorskih promenljivih xoption i yoption. */
133
135 for(j=1;argv[3][j]!='\0';j++)
    switch(argv[3][j])
    {
137         case 'x': xoption=1;
                    break;
139         case 'y': yoption=1;
                    break;
141         default:
                    fprintf(stderr, "Greska: nedozvoljeni karakter\n"
);
                    exit(EXIT_FAILURE);
    }
143
145
147 if(ucitaj_razlomke(razlomci, &dim, in)==0)
{
149     fprintf(stderr, "Greska pri zadavanju razlomaka\n");
    exit(EXIT_FAILURE);
151 }

153 /*
154     U zavisnosti od datih opcija, vrsimo upis reciprocnih
155     razlomaka u trazenom formatu.

157     Funkcija fprintf se koristi na isti nacin kao
158     funkcija printf uz dodatni prvi argument koji
159     oznacava naziv datoteke u koju se vrši upis.
160 */
161 for (i=0; i<dim;i++)
{
163     /*
164         Ukoliko je brojilac razlomka jednak nuli,
165         nema smisla traziti njegovu reciprocnu vrednost
166     */
167     if (razlomci[i].br==0)
        continue;

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

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

```

```

175     if (yoption)
176         fprintf(out, "%f ", vrednost(&r));
177
178     fprintf(out, "\n");
179 }
180
181 fclose(in);
182 fclose(out);
183
184 return 0;
185 }

```

### Rešenje 4.6

```

1  /*
2   Za svaki automobil poznati su marka, model i cena. Iz datoteke cije
3   se ime zadaje sa standardnog ulaza ucitava se broj automobila a
4   potom
5   i podaci za svaki automobil. Program treba da:
6   a) izracuna prosečnu cenu po marki kola
7   b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
8   zadaje
9   kao argument komandne linije, da ispise automobile u tom cenovnom
10  rang u zajednu sa prosečnom cenom odgovarajuće marke
11
12  Mozemo pretpostaviti da se model i marka sastoje od jedne reci i
13  da svaka od njih sadrži najviše 30 karaktera kao i da se u datoteci
14  nalaze podaci za najviše 100 automobila.
15
16  */
17
18  #include <stdio.h>
19  #include <stdlib.h>
20  #include <string.h>
21  #define MAX 31
22  #define MAXA 100
23
24  typedef struct automobil
25  {
26      char marka[MAX];
27      char model[MAX];
28      float cena;
29  } AUTOMOBIL;
30
31  /*
32   Struktura INFO sadrži naziv
33   marke automobila, prosek cena
34   za tu marku i broj automobila
35   te marke
36  */
37  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;
    for(i=0; i < n; i++)
71         if(strcmp(p[i].marka, m) == 0)
            return i;

73     return -1;
75 }

77
/*
79     Funkcija informacije_o_markama za niz automobila a dimenzije n
    racuna proseke cena automobila po markama i smesta ih u niz
    p. Na dimenziju niza p pokazuje pokazivac pn.

81
    Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
83     sumu cena automobila te marke (koju cemo smestiti u polje vrednost
    strukture
    INFO), i broj automobila te marke (koju cemo smestiti u polje
85     n strukture INFO) i da na kraju podelimo ove dve promenljive
    i tako dobijemo prosečnu vrednost cene.
```

```

87      Za svaki automobil a[i] proveravamo da li se njegova marka vec
89      nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
91      vredost cene automobila a[i] i uvecavamo broj automobila sa
93      tom markom. U suprotnom, dodajemo novi element u niz p. Posto
95      ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
97      na koju pokazuje pokazivac *pn.
99  */
101 void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
103 {
105     int i,j;
107     int ind;
109     for(i=0;i<n;i++)
111     {
113         /* Proveravamo da li se marka automobila a[i] vec nalazi u
115         nizu p (niz proseka po markama) */
117         ind = sadrzi(p,*pn1,a[i].marka);
119         if(ind==-1) /* Ako se ne nalazi, uvodimo novi element niza na
121         kraj, na poziciju *pn. */
123         {
125             strcpy(p[*pn1].marka, a[i].marka);
127             p[*pn1].vrednost = a[i].cena;
129             p[*pn1].n = 1;
131             (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
133             */
135         }
137         else /* Ako se nalazi, azuriramo polja strukture. */
139         {
141             p[ind].vrednost+=a[i].cena;
143             p[ind].n++;
145         }
147     }
149     /* Na osnovu sume cena i broja automobila racunamo prosechnu
151     vrednost. */
153     for(i=0;i<*pn1;i++)
155         p[i].vrednost = p[i].vrednost/p[i].n;
157 }
159
161 void stampaj_informacije(INFO p[], int n)
163 {
165     printf("Informacije o broju automobila i prosečnoj ceni po markama
167     :\n");
169     int i;
171     for(i=0;i<n;i++)
173         printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
175 }
177
179 /*
181 Funkcija stampa automobile cija je cena manja od maksimalne
183 cene koju je korisnik naveo u komandnoj liniji da je spreman

```

```
135     da plati, zajedno sa prosecom cenom za tu marku automobila
136     */
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
    n1)
138 {
139     /*
140     S obzirom da je niz p formiran na osnovu niza a, marka svakog
141     automobila iz niza a se sigurno nalazi u nizu p. Zbog toga
142     nije neophodno proveravati da li je povratna vrednost funkcije
143     sadrzi razlicita od -1.
144     */
145     int i;
146     printf("Kola u vasem cenovnom rangu:\n");
147     for(i=0;i<n;i++)
148         if(a[i].cena<g)
149             printf("%s %s %.2f\n", a[i].marka, a[i].model, p[sadrzi(p,n1
    ,a[i].marka)].vrednost);
150 }
151
152 int main(int argc, char* argv[])
153 {
154     AUTOMOBIL kola[MAXA];
155     FILE* f;
156     char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
    ulaza. */
157     float granica; /* Maksimalna cena koju je korisnik spreman da
    plati.
158                     Zadaje se kao argument komandne linije.
159                     */
160     INFO infos[MAXA];
161     int dim_kola,dim_infos;
162     int i;
163
164     if(argc!=2)
165     {
166         fprintf(stderr,"Greska: program se pokrece sa: %s
    gornja_granica_cene \n", argv[0]);
167         exit(EXIT_FAILURE);
168     }
169
170     /* Argumenti komandne linije su stringovi. Da bismo od stringa
    dobili
171     realan broj, koristimo ugradjenu funkciju atof. */
172     granica = atof(argv[1]);
173
174     printf("Unesi naziv datoteke:");
175     scanf("%s", dat);
176
177     f=fopen(dat, "r");
178
179     if(f==NULL)
180     {
```

```
181     fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s\n", dat);
182     exit(EXIT_FAILURE);
183 }
184
185 if (ucitaj_podatke(f,kola,&dim_kola,MAXA)==0)
186 {
187     fprintf(stderr, "Greska pri učitavanju podataka\n");
188     exit(EXIT_FAILURE);
189 }
190
191 informacije_o_markama(kola, dim_kola, infos, &dim_infos);
192
193 stampaj_informacije(infos,dim_infos);
194
195 stampaj_kandidate(kola, dim_kola, granica, infos, dim_infos);
196
197 fclose(f);
198 return 0;
199 }
```

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)

Rešenje [4.33](#)



# 5

## Razni zadaci

### 5.1 Rešenja



# Dodatak A

## Ispitni zadaci

### A.1 Testovi/Kolokvijumi

#### A.1.1 Programiranje 1, i-smer, kolokvijum

##### Grupa I

**Zadatak A.1** Napisati URM program koji izračunava funkciju:

$$f(x, y) = \begin{cases} 2x - y & 2x \geq y \\ 3y & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.2** Sa standardnog ulaza unose se jedan karakter (**p** ili **n**) i dva pozitivna trocifrena broja. Na osnovu vrednosti unetog karaktera izračunati i ispisati na standardni izlaz:

**p** - zbir cifara na parnim pozicijama unetih brojeva

**n** - zbir cifara na neparnim pozicijama unetih brojeva

Cifre se broje sa desne strane, tako da cifri jedinice odgovara pozicija 1.

U slučaju greške ( ukoliko karakter nije p ili n ili nisu uneti pozitivni trocifreni brojevi ) ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| p 235 645
|| 8
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| n 567 101
|| 14
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| A 432 543
|| -1
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| p 102 1234
|| -1
```

[Rešenje A.28]

**Zadatak A.3** Sa standardnog ulaza učitava se pozitivan ceo broj  $i$  i ceo broj  $i$  ( $1 \leq i$ ). Na standardni izlaz ispisati broj koji se dobija kada se ukloni  $i$ -ta cifra broja. Cifre se broje sa desne strane, tako da cifri jedinice odgovara pozicija 1. Neispravan ulaz je kada se unose negativan broj ili negativna vrednost ili nula za  $i$  i u tom slučaju na standardni izlaz ispisati -1. Ukoliko broj nema  $i$ -tu cifru broj ostaje nepromenjen.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| 35243 2
|| 3523
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| -14423 1
|| -1
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| 1234 5
|| 1234
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| 523156 6
|| 23156
```

[Rešenje A.28]

## Grupa II

**Zadatak A.4** Napisati URM program koji izračunava funkciju:

$$f(x, y, z) = 4x + 2y + 3z$$

[Rešenje A.28]

**Zadatak A.5** Korisnik unosi 7 karaktera koji predstavljaju indeks studenta koji je oblika OOGGBBB. OO je oznaka smeru i može biti mi, ma, mr, ms, mm, mv. GG je oznaka godine upisa. BBB je oznaka broja koji može biti jednocifren, trocifren ili dvocifren sa vodećim nulama. Na osnovu ovih podataka na standardni

izlaz ispisati ime smeru kome student pripada i indeks u obliku broj/godina. U slučaju greške ( ukoliko OO kao oznaka smeru nije ispravna ili ostali karakteri nisu brojevi ) ispisati -1. Nazivi smerova su: mi - informatika, ma - astronomija, mr - racunarstvo i informatika, ms - statistika, mm - teorijska matematika, mp - primenjena matematika

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| mi11275  
|| informatika 275/2011
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| mm98005  
|| teorijska matematika 5/1998
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| mo23112  
|| -1
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| ms12001  
|| statistika 1/2012
```

[Rešenje A.28]

**Zadatak A.6** Državna lutrija došla je na ideju o novoj igri na sreću. Ova igra na sreću igra se tako što se izvuče jedan broj od 1000 do 9999, Nagrada koja se dobija ako ste pogodili izvučen broj je proizvod njegovih parnih cifara i samog broja. Vaš zadatak je da na osnovu izučenog broja izračunate nagradu koja se dobija. Kao ulaz sigurno ćete dobiti ispravan broj. Ako broj nema parnih cifara, nagrada je sam taj broj. Na standardni izlaz ispišite nagradu.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 1321  
|| 2642
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 3284  
|| 210176
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 1111  
|| 1111
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 2222  
|| 35552
```

### Primer 5

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 6031  
|| 0
```

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| 4321  
|| 34568
```

[Rešenje A.28]

## Grupa III

**Zadatak A.7** Napisati URM program koji izračunava funkciju:

$$f(x, y, z) = \begin{cases} 2 \cdot x + 2 \cdot y & x \leq z \\ z + 3 & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.8** Napisati C program koji sa standardnog ulaza učitava 4 velika slova abecede i nenegativan ceo broj k. Program na standardni izlaz ispisuje 4 karaktera koji se dobijaju cikličkim pomeranjem (u okviru karakterske tabele) unetih karaktera za k mesta unapred. Na primer, karakter A pomeren za 4 mesta unapred postaje E dok karakter Z pomeren za 3 mesta unapred postaje C. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ako neki od unetih karaktera ne predstavlja veliko slovo abecede ili ako je broj k negativan, pretpostaviti da se na ulazu uvek zadaje tačno četiri karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| BABA 3
|| EDED
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| DEDA 26
|| DEDA
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| ZABC 53
|| ABCD
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| PERA -2
|| -1
```

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| abcd
|| -1
```

[Rešenje [A.28](#)]

**Zadatak A.9** Napisati C program koji sa standardnog ulaza učitava dva četvorocifrena, pozitivna, cela broja i proverava da li je broj koji se dobija učešljanjem unetih brojeva palindrom. Ako uneti brojevi imaju cifre a1 a2 a3 a4 i b1 b2 b3 b4 tada su cifre učešljanog broja a1 b1 a2 b2 a3 b3 a4 b4. Broj je palindrom ako se čita isto sa obe strane. Ukoliko je broj palindrom ispisati na standardni izlaz 1, ukoliko nije tada ispisati 0, a u slučaju neispravnog ulaza ispisati -1, neispravnim ulazom smatraju se negativni brojevi i brojevi sa brojem cifara manjim ili većim od 4.

Primer 1:

1234 5678

Primer 2:

1342 2431

Primer 3:

1234 4321

Primer 4:

-1234 1234

## A.2 Kvalifikacioni zadaci

### A.3.1 Programiranje 1, i–smer, Završni ispit, januar, 23.01.2016.

$$f(x) = \begin{cases} 2(x-1) & x \geq 1 \\ 0 & \text{inače} \end{cases}$$

## Grupa I

$$a_n = \begin{cases} 1 & n = 1 \\ 3 & n = 2 \\ 2a_{n-1} + 3a_{n-2} + 4 & n \geq 3 \end{cases}$$

Primer 1:	Primer 2:	Primer 3:	Primer 4:
-123	1	4	10
-1	1	39	29523

**Zadatak A.12** (7 poena) Napisati funkciju

271

koja proverava koji karakter se najviše puta pojavio u niski *s*. Taj karakter smešta u promenljivu *c*, a broj pojavljivanja karaktera u promenljivu *br*. Sa standardnog ulaza unosi se linija teksta (može sadržati beline). Testirati rad funkcije *f3* programom koji sa standardnog ulaza učitava nisku i na standardni izlaz ispisati koji karakter se najviše puta pojavio u okviru nje, kao i broj pojavljivanja datog karaktera. Ukoliko postoji više karaktera čiji broj pojavljivanja odgovara maksimalnom broju, ispisati onaj sa najmanjim kodom u ASCII tabeli. Pretpostaviti da se na sistemu koristi ASCII tabela.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
abrakadabra	cvrcak	jorgovan99	s@rm@ ponek@d v@zno
a 5	c 2	9 2	@ 4

[Rešenje [A.28](#)]

**Zadatak A.13** (7 poena) Igra "Minesweeper" sastoji se od pravougaone table izdvojene na polja koja mogu biti bezbedna ili su na njima rasporedjene mine. Sa standardnog ulaza učitavaju se brojevi *n* i *m* koji označavaju dimenzije table. Nako toga unosi se broj *k* kojim se navodi koliko mina se nalazi na tabli i *k* pozicija (*i*, *j*) koja označavaju pozicije na tabli na kojima se nalaze mine (i-ti red, j-ta kolona). Korisnik zatim unosi koordinate *l* i *m* za koje se na standardni izlaz ispisuje broj koliko se mina nalazi na poljima susednim tom polju. Proveravaju se susedna polja u svih 8 pravaca. Ukoliko je polje koje se proverava baš mina ispisati na standardni izlaz **MINA**. Maksimalna dimenzija table je 100x100. Ukoliko je neka od koordinata izvan dimenzija table ili su dimenzije table izvan dozvoljenih granica na standardni izlaz ispisati -1.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz: Izlaz:	Ulaz: Izlaz:	Ulaz Izlaz:	Ulaz: Izlaz:
4 4 2	4 4 MINA	2 3 -1	101 10 -1
3	2	1	1
0 1	0 1	-1 0	45 67
1 2	1 2	2 2	30 31
2 3	2 3		
2 2	2 3		

[Rešenje [A.28](#)]

**Zadatak A.14** (7 poena) Služba gradskog prevoza želi da u svakom trenutku ima evidenciju o opterećenju svojih linija. Na linijama saobraćaju autobusi, trolejbusi i tramvaji. Maksimalni kapacitet autobusa je 25, trolejbusa 20 a tramvaja 30 putnika. Broj linije je pozitivan ceo broj manji od 1000.



- a) (1 poen) Definirati strukturu kojom se opisuje vozilo. Svako vozilo zadato je svojim tipom (autobus, trolejbus, tramvaj), linijom na kojom saobraća i brojem putnika koji se u vozilu nalaze.
- b) (6 poena) Sa standardnog ulaza se učitava broj  $n$  ( $0 \leq n \leq 1000$ ),  **$n$  vozila i broj linije**. Za zadati broj linije na standardni izlaz ispisati ukupan broj slobodnih mesta na toj liniji. Koristiti strukturu definisanu pod a). Neispravnim ulazom smatraju se negativan broj putnika, broj putnika veći od dozvoljenog kapaciteta za navedeni tip vozila, tip vozila sa nazivom različitim od navedena tri ili negativan broj linije. U tim slučajevima na standardni izlaz ispisati -1.

Primer 1:

Ulaz:  
4  
autobus 27 18  
trolejbus 28 15  
tramvaj 7 29  
autobus 27 24  
27

Izlaz:  
8

Primer 2:

Ulaz:  
3  
AutobuS 65 23  
Kombi 1 10  
minibus 6 21  
6

Izlaz:  
-1

Primer 3:

Ulaz  
3  
tramvaj 7 29  
tramvaj 3 15  
tramvaj 12 12  
14

Izlaz:  
0

-----  
Primer 4:

Ulaz:  
2  
autobus 26 20  
tramvaj 9 32

Izlaz:  
-1

Primer 5:

Ulaz:  
500  
-1

[Rešenje [A.28](#)]

## Grupa II

**Zadatak A.15** (4 poena) Napisati C program koji za uneti niz celobrojnog tipa i neparne dužine  $n$  ispisuje po  $k$  elemenata levo i desno od sredine niza (ne uključujući sredinu). Prvo se unosi  $n$ , zatim niz od  $n$  elemenata, a na kraju i  $k$ .

Neispravnim ulazom se smatra niz parne ili negativne dužine, kao i  $k$  koje je negativno ili veće od polovine dužine niza. U slučaju neispravnog ulaza ispisati -1 na standardni izlaz.

Smatrati da je maksimalna veličina niza 100 elemenata.

Primer 1:

Ulaz:  
5  
1 2 3 4 5

Primer 2:

Ulaz:  
9  
9 8 7 6 5 4 3 2 1

Primer 3:

Ulaz:  
6  
1 2 3 4 5 6

Primer 4:

Ulaz:  
3  
1 2 3

Primer 5:

Ulaz:  
5  
10 9 8 7 6

2	1	5	10	-6
Izlaz:	Izlaz:	Izlaz:	Izlaz:	Izlaz:
1 2 3 4	6 4	-1	-1	-1

[Rešenje [A.28](#)]

**Zadatak A.16** (7 poena) Barkod kodira broj proizvoda dodajući mu kontrolnu cifru. Kontrolna cifra izračunava se kao poslednja cifra zbira jedinica u zapisu svake cifre broja proizvoda. Npr. broj 86012 kodira se kao 1000 0110 0000 0001 0010 a kontrolna cifra je  $(1 + 1 + 1 + 1 + 1) \bmod 10 = 5$ .

Napisati funkciju

```
void kontrolna(char broj_proizvoda[], int *kont)
```

koja izračunava kontrolnu cifru broja proizvoda, koji se zadaje kao niska, i smešta ga u promenljivu kont. Niska može sadržati beline i druge karaktere, ali ih pri izračunavanju kontrolne cifre treba ignorisati, samo cifre uzeti u obzir.

Napisati program koji sa standardnog ulaza učitava liniju teksta kojom je predstavljen broj proizvoda i testira funkciju kontrolna. Na standardni izlaz ispisati izračunatu kontrolnu cifru. Maksimalna dužina niske je 100 karaktera.

Na sistemu se koristi ASCII tabela. Ukoliko ne postoji ni jedna cifra u bar-kodu, onda je kontrolna cifra 0.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz:	Ulaz:	Ulaz:	Ulaz:
86012	001-223-4	555 555-555	AB-- 123 --BA
Izlaz:	Izlaz:	Izlaz:	Izlaz:
5	6	8	4

[Rešenje [A.28](#)]

**Zadatak A.17** (7 poena) Napisati program koji ispisuje proseku zbiru svih kolona matrice čiji su elementi tipa `double`.

Prvo se unosi broj redova matrice  $n$ , zatim broj kolona matrice  $m$ , i onda  $n$  redova sa po  $m$  elemenata.

Maksimalna veličina matrice je  $100 \times 100$ . Ukoliko je ulaz neispravan (za vrednosti  $m$  i  $n$ ) prekinuti rad programa i ispisati -1.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz:	Ulaz:	Ulaz:	Ulaz:
4 4	3 2	2 4	3 3
0.2 0.4 0.7 1.3	1.23 4.56	0.1 0.2 0.3 0.4	1 0 0
1.5 1.7 2.2 2.5	0 1	10.98 7.65 4.32 1	0 1 0

6.3 -1.2 4.4 5.6	7.89 1	Izlaz:	0 0 1
1.6 2.3 2.8 3.5	Izlaz:	6.2375	Izlaz:
Izlaz:	7.8400		1.000
8.9500			

[Rešenje A.28]

**Zadatak A.18** (7 poena) Profesor na jednom predmetu je uveo pravilo da njegov predmet položio svako ko na ispitu osvoji broj poena koji je veći ili jednak od proseka poena umanjenog za 10.

- a) (1 poen) Definirati strukturu kojom se opisuje svaki student sa indeksom (indeks-u-obliku-alas-naloga) i brojem poena koji je osvojio (ceo broj od 0 do 100).
- b) (6 poena) Na ulazu ćete dobiti  $n$  ( $0 \leq n \leq 300$ ), broj studenata koji su polagali predmet, i onda  $n$  redova oblika

indeks-u-obliku-alas-naloga broj-poena-na-ispitu

Ispisati na standardni izlaz indekse svih studenata koji su položili ovaj predmet. Koristiti strukturu definisanu pod a).

Smatrati da je indeks pravilno zapisan. U slučaju loše vrednosti za  $n$  ili loše vrednosti za broj poena ispisati -1.

Primer 1:	Primer 2:	Primer 3:	Primer 4:	Primer 5:
Ulaz:	Ulaz:	Ulaz:	Ulaz:	Ulaz:
4	4	6	4	3
mi12123 80	mr12345 91	mi00001 20	mi11110 100	mi05900 98
mi15512 70	ml54321 80	mi00002 32	mi11111 99	mi13034 120
mi15555 99	mv36925 29	mi00003 96	mi11112 98	mi11234 34
mi13333 40	mi14725 55	mi00004 52	mi11113 87	Izlaz:
Izlaz:	Izlaz:	mi00005 41	Izlaz:	-1
mi12123	mr12345	mi00006 15	mi11110	
mi15512	ml54321	Izlaz:	mi11111	
mi15555	mi14725	mi00003	mi11112	
		mi00004	mi11113	
		mi00005		

[Rešenje A.28]

### A.3.2 Programiranje 1, i–smer, Završni ispit, februar, 11.02.2016.

**Zadatak A.19** Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} 2(x - y) & x \geq y \\ 0 & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.20** Sa standardnog ulaza se unose celi, nenegativni brojevi sve dok se ne unese nula. Na standardni izlaz ispisati kvadrat razlike najvećeg i najmanjeg od unetih brojeva. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ukoliko je unet negativan broj ili ukoliko nije unet ni jedan broj osim nule.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
1 2 3 4 5 0	1 2 3 -4 5 0	1 1 1 1 0	0
16	-1	0	-1

[Rešenje [A.28](#)]

**Zadatak A.21 a)** Napisati funkciju

```
void mutacije(char s1[], char s2[], int *br)
```

koja za navedene niske **s1** i **s2** iste dužine proverava na koliko mesta se karakteri niski razlikuju i rezultat upisuje u promenljivu **br**. Pri poređenju ignorisati beline.

**b)** Napisati program koji sa standardnog ulaza učitava dve DNK sekvence (niske karaktera A, T, C ili G) iste dužine i testira funkciju **mutacije** ispisujući vrednost promenljive **br** na standardni izlaz. Maksimalna dužina niski je 100 karaktera. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ukoliko neka od niski sadrži karakter koji ne pripada skupu {A, T, C, G} i nije belina ili je jedna niska duža od druge.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz:	Ulaz:	Ulaz:	Ulaz:
AGTC CGCT AGT	ATCG ATCG ATCG	AGTTGTTGT ATGX	AGGGATGGATGAG
AGTCC GC TAGT	ACCG ATGC ATCA	TTGTATGGA GGAT	TTGATGACGT
Izlaz:	Izlaz:	Izlaz:	Izlaz:
0	3	-1	-1

[Rešenje A.28]

**Zadatak A.22** Krtice su organizovano napale baštu šargarepa. Farmer je napravio pravougaonu mapu bašte dimenzija  $n \times m$ , gde je znakom **X** označio polje na kome se nalazi krtičnjak, dok je netaknuta polja označio znakom **-**. Kako je bašta velika, farmer želi da bez mnogo muke izračuna broj krtičnjaka u proizvoljnom pravougaonom delu svoje bašte. Sa standardnog ulaza unose se dimenzije mape **n** i **m**, zatim mapa bašte sa oznakama krtičnjaka i netaknutih polja. Nakon toga farmer zadaje koordinate (**i1**, **j1**) i (**i2**, **j2**) koje označavaju gornji levi i donji desni ugao pravouganika za koji farmer pita koliko krtičnjaka je obuhvaćeno na mapi tim pravouganikom. Na standardni izlaz ispisati broj krtičnjaka u zadatom pravouganiku. Maksimalna dimenzija mape je 100 x 100. U slučaju neispravnih koordinata uglova pravouganika, neispravnih dimenzija mape ili oznaka na tabli van skupa { **X**, **-** } na standardni izlaz ispisati -1.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz:	Ulaz:	Ulaz:	Ulaz:
4 4	4 4	4 4	4 4
- - X -	- - X K	- X - X	- X - X
X - - -	- X - -	X - X -	X - X -
- X - X	X - - -	- X - X	- X - X
X - X -	X X - X	X - X -	X - X -
0 1	1 2	3 4	0 0
2 2	3 4	1 2	3 3
Izlaz:	Izlaz:	Izlaz:	Izlaz:
2	-1	-1	8

[Rešenje A.28]

**Zadatak A.23** Vlasnik pekare želi da utvrdi koliko je isplativa prodaja njegovog najskupljeg peciva.

**a)** Definisati strukturu **Pecivo** koja sadrži podatke o imenu peciva (najviše 50 karaktera) i ceni peciva (realan broj tipa double).

**b)** Sa standardnog ulaza se unosi broj **n** a zatim mesečni obračun sa **n** prodatih komada peciva, pri čemu je naziv peciva u jednom redu a cena u narednom. Na standardni izlaz ispisati ukupnu zaradu od prodaje najskupljeg peciva zaokruženu na dva decimalna mesta. U slučaju negativne cene peciva ili u slučaju da je **n** manje ili jednako nuli ispisati -1. Pretpostaviti da će samo jedna vrsta peciva imati maksimalnu cenu.

Primer 1:	Primer 2:	Primer 3:	Primer 4:
Ulaz:	Ulaz:	Ulaz:	Ulaz:
5	3	-1	5
burek sa mesom	mafin		kroasan sa dzemom
100.50	-50.03		49.99
buhtla sa cokoladom	krofna		kroasan sa dzemom
50.00	56.00		49.99
burek sa mesom	krofna		kroasan sa dzemom
100.50	56.00		49.99
rol virsla			kroasan sa dzemom
75.00			49.99
kroasan sa kremom			kroasan sa dzemom
60.00			49.99
Izlaz:	Izlaz	Izlaz:	Izlaz:
201.00	-1	-1	249.95

[Rešenje [A.28](#)]

### A.3.3 1. Grupa, I smer, Programiranje 1 2015/2016, ispit, jun

**Zadatak A.24** Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} x - y + 2 & x + 2 \geq y \\ 0 & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.25** Napisati program koji sa standardnog ulaza učitava prvo pozitivan ceo broj  $n$  ( $0 < n \leq 99$ ), a zatim  $n$  celih brojeva i izračunava zbir parnih. Izračunati zbir ispisati na standardni izlaz. U slučaju greške (za  $n \leq 0$  ili  $n \geq 100$ ) na standardni izlaz ispisati -1.

Ulaz	5 1 2 3 4 5	5 -1 -2 -3 -4 -5	3 10 -10 10	-3 1 2 3
Izlaz	6	-6	10	-1

[Rešenje [A.28](#)]

**Zadatak A.26** Napisati funkciju `void f(char s[], char c, int *prva, int* poslednja)` koja u datoj nisci  $s$  pronalazi indekse prvog i poslednjeg pojavljivanja datog karaktera  $c$  i dobijene vrednosti redom smešta u promenljive  $prva$  i

*poslednja*. Ukoliko se karakter ne pojavljuje u nisci, obe vrednosti postaviti na -1.

Potom napisati program koji sa standardnog ulaza učitava karaktersku nisku (dužine ne veće od 150 karaktera) i jedan karakter i nakon toga poziva funkciju *f*, a potom na standardni izlaz ispisuje indekse prvog i poslednjeg pojavljivanja datog karaktera u datoj nisci. Pretpostaviti da je ulaz u ispravnom formatu.

Ulaz	ucionica i	ucionica u	ucionica o	ucionica p
Izlaz	2 5	0 0	3 3	-1 -1

[Rešenje A.28]

**Zadatak A.27** Sa standardnog ulaza se zadaje dimenzija kvadratne matrice  $n$  ( $0 < n \leq 99$ ), a zatim elementi matrice koji su celi brojevi. Na standardni izlaz ispisati redni broj vrste koja ima najveći zbir elemenata. U slučaju greške (za  $n \leq 0$  ili  $n \geq 100$ ) na standardni izlaz ispisati -1.

Ulaz	3				
1 2 3					
4 5 6					
7 8 9	5				
1 0 0 0 0					
1 1 0 0 0					
1 1 1 0 0					
1 1 1 1 0					
1 1 1 1 1	5				
1 0 -1 0 -1					
-1 0 -1 5 0					
1 -1 -1 0 1					
1 0 -3 0 -1					
0 -1 0 -1 0	-3				
1 2 3					
4 5 6					
7 8 9					
Izlaz	2	4	1	-1	

[Rešenje A.28]

**Zadatak A.28** Definisati strukturu *Tacka* za predstavljanje tačaka u ravni sa koordinatama tipa *double*. Sa standardnog ulaza se učitava broj  $n$  ( $1 < n \leq 99$ ), zatim niz od  $n$  tačaka tako sto se unosi prvo  $x$ , pa  $y$  koordinata za svaku tačku. Za zadate tačke ispisati na standardni izlaz dužinu najduže duži koja se može obrazovati od neke dve tačke iz učitano g niza. Rezultat ispisati na dve decimalne. Dužina duži između tačaka  $a(x1; y1)$  i  $b(x2; y2)$  se računa po formuli

$$\sqrt{(x1 - x2)^2 + (y1 - y2)^2}$$

U slučaju greške (za  $n \leq 1$  ili  $n \geq 100$ ) na standardni izlaz ispisati -1.

Ulaz	2				
3 4					
0 0	3				
0.5 0.3					
-5 3					
3 4	4				
1.2 1.2					
1.2 1.2					
1.2 1.2					
1.2 1.2	1				
0 0					
Izlaz	5.00	8.06	0.00	-1	

[Rešenje [A.28](#)]

### A.3.4 Praktični deo ispita, jun ...

## A.4 Rešenja

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)



Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)

Rešenje [A.28](#)