PROGRAMIRANJE 1

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

PROGRAMIRANJE 1
Zbirka zadataka sa rešenjima

Beograd
2016.

Autori:
dr Milena Vujošević Janičić, docent na Matematičkom fakultetu u Beogradu
dr Jovana Kovačević, docent na Matematičkom fakultetu u Beogradu
Danijela Simić, asistent na Matematičkom fakultetu u Beogradu
Anđelka Zečević, asistent na Matematičkom fakultetu u Beogradu

PROGRAMIRANJE 1
Zbirka zadataka sa rešenjima

Sadržaj

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 Napisati program koji na standardni izlaz ispisuje tekst `Zdravo svete!`. Resenje se ne uklapa sa tekstom zadatka. Ono sto su altrenativni ispisi u resenju moze da ostane pod komentarima, ali da se postavi kao dva razlicita resenja, ali uvek resenje mora da se poklapa sa test primerom. U okviru prvog resenja potrebno je i kratko uputstvo kako da se prevede i pokrene program. Takodje, komentar uz include i uz int main i uz return - to posle naravno necemo komentarisati.

[Rešenje 1.1]

Zadatak 1.2 Napisati program koji poziva korisnika da unese dva cela broja sa standardnog ulaza, a zatim ispisuje:

(a) unete vrednosti

(b) njihov zbir Ovo bi mogao da bude zadatak sa prodavnicom i kupovinom dva artikla, tako da bih takav neki zadatak dodala.

(c) njihovu razliku Ovo je u sustini isti zadatak kao zadatak sa prodavnicom i kusurom, samo je taj malko komplikovaniji. Ostavila bih oba, ali bih ovaj drugi priblizila po redosledu ovom zadatku.

(d) njihov proizvod Ovo je u sustini isti zadatak kao zadatak sa jabukama, ali bih ovaj drugi priblizila po redosledu ovom zadatku.

Ovde je problem sto oni jos nisu ucili if, a deljenje je zbog toga nezgodno. Ja bih izbacila poslednje dve stavke i uradila samo a,b,c i d
U resenju prokomentarisati deklaracije int x i int y: Deklariše se promenljiva

celobrojnog tipa u kojoj će biti čuvana vrednost prvog broja koji se unosi sa ulaza ili tako nekako, pa slično za y i rezultat. Na ovom mestu bih stavila punu recenicu koja pominje reci i deklaracija i celobrojni tip i slicno, a u ostalim resenjima samo komentare sta promenljiva predstavlja, ukoliko joj ime nije dovoljno deskriptivno. Na mestu gde je prvi put deklaracija float ili double promenljive, tu bih isto pomenula tip, da pomenemo jednostruku i dvostruku tacnost.

[Rešenje 1.2]

Zadatak 1.3 Napisati program koji sa standardnog ulaza učitava realnu vrednost izraženu u inčima, konvertuje tu vrednost u centimetre i ispisuje je na standardni izlaz zaokruženu na dve decimale. UPUTSTVO: *Jedan inč ima* 2.54 *centimetra.*

[Rešenje 1.3]

Zadatak 1.4 Napisati program koji sa standardnog ulaza učitava dužinu poluprečnika kruga i na standardni izlaz ispisuje njegov obim i površinu.

[Rešenje 1.4]

Zadatak 1.5 Napisati program koji za uneti pozitivan broj na standardni izlaz ispisuje njegove cifre jedinica, desetica i stotina. Preformulisala sam da broj nije trocifren jer nemamo if da proverimo da li je ulaz ispravan. Sa ovakvom formualcijom program moze da radi ispravno i ako broj nije trocifren. Ispraviti u resenju tipove, tj da brojevi budu unsigned kako bi uvek radio ispravno tj kako ne bi dobijao negativne cifre. Ispraviti komentar u printf-u koji pominje cele brojeve.

[Rešenje 1.5]

Zadatak 1.6 Napisati program koji učitava trocifreni broj sa standardnog ulaza i ispisuje broj dobijen obrtanjem njegovih cifara.

[Rešenje 1.6] Ovaj zadatak mozda prebaciti u if? Kako proveriti da li je uneti broj stvarno trocifren? Ili nekako formulisati zadatak tako da nije neophodno da broj bude trocifren. Npr, ispisuje broj koji se dobije kada cifra jedinica i cifra stotina zamene mesta. Sve zadatke koji se bave radom sa ciframa broja grupisati na jedno mesto.

Zadatak 1.7 Ovaj zadatak je radjen na praktikumu (jednakostranicni trougao), dole je naveden. Da li da ostane dole ili da se prebaci ovde? Mislim da redosled treba da bude po smislu, a ne po vezbe/praktikum podeli. Zato, ako mu je mesto ovde, onda ga tu treba i prebaciti. Zadatke poredjati na pocetku i

po tipovima, a posle mogu i da se mesaju: dakle da prvo ide par zadataka koji rade sa celobrojnim vrednostima, a onda zadaci koji rade sa realnim vrednostima. Redosled zadataka je vrlo osetljiva stvar i to bih ostavila za kasniju fazu: za sada samo okvirno kako nam se ucini da treba, a onda posle ce biti jos tumbanja. Zato nije vazno da se brojevi zadataka i resenja poklapaju u ovom trenutku, kada napravimo finalno rasporedjivanje onda cemo da menjamo te brojeve. Za rasporedjivanje zadataka vazna su nam i resenja.

[Rešenje 1.7]

Zadatak 1.8 Napisati program koji za unetu cenu proizvoda ispisuje najmanji broj novčanica koje je potrebno izdvojiti prilikom plaćanja proizvoda. Na raspolaganju su novčanice od 1000, 100, 50, 10 i 1 dinar.

[Rešenje 1.8]

Zadatak 1.9 Napisati program koji za tri cela broja koja se unose sa standardnog ulaza ispisuje njihovu artimetičku sredinu na standardni izlaz.

[Rešenje 1.9]

Zadatak 1.10 Ovaj zadatak je radjen na praktikumu (razmena vrednosti), dole je naveden. Da li da ostane dole ili da se prebaci ovde?

[Rešenje 1.10]

Zadatak 1.11 Napisati program za uneti ceo broj ispisuje njegov kvadrat i kub.

Primer 1

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

[Rešenje **??**]

Zadatak 1.12 Napisati program koji za unete dužine stranica 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 Sve geometrijske zadatke bih grupisala da budu susedni. Malo me zbunjuje sto su resenja za kvadrat i pravougaonik sa int, a za ove druge sa float, kao da stranica pravougaonika ne moze da bude realan broj? Mozda bi svi ti goemetrijski trebalo da rade sa realnim brojevima? Tako bi bili konzistentni... Napisati program koji za unetu dužinu stranice jednakostraničnog 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 pomaže moleru da izračuna površinu zidova prostorije koju treba da okreči. Za unete dimenzije sobe u metrima (dužinu, širinu i visinu), program treba da ispiše površinu zidova za krečenje pod pretpostavkom da na vrata i prozore otpada oko 20%. Omogućiti i da na osnovu unosene cene usluge po kvadratnom metru program izračuna ukupnu cenu krečenja.

Primer 1

```
Interakcija sa programom:
 Unesite dimenzije sobe: 4 4 3
 Unesite cenu po kvadratnom metru: 500
 Moler treba da okreci 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 ukupnu vrednost date količine jabuka.

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 koji treba da vrati kupcu. Za unetu cenu artikla, količinu artikla i iznos koji je kupac dao, program treba da ispiše vrednost kusura.

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 za uneti prirodni četvorocifreni broj:

(a) izračunava proizvod cifara

(b) izračunava razliku sume krajnjih i srednjih cifara

(c) izračunava sumu kvadrata cifara

(d) izračunava broj koji se dobija ispisom cifara u obrnutom poretku

(e) izračunava broj koji se dobija zamenom cifre jedinice i cifre stotine

Ponovo imamo problem sa time sto je broj cetvorocifren a nemamo if proveru?

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 ispisuje broj koji se dobija izbacivanjem cifre desetica u unetom prirodnom broju.

Primer 1

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

[Rešenje 1.18]

Zadatak 1.19 Napisati program koji za uneti prirodan broj x i unete vrednosti c i p ispisuje broj koji se dobija ubacivanjem cifre c u broj x na poziciji p. 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.* Izmenila bih da numeracija cifara pocinje od 0, jer se to uklapa sa tezinskim faktorom i nekako je logicnije.

Primer 1

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

[Rešenje 1.19]

Zadatak 1.20 Razbiti ovaj zadatak na tri zadatka i staviti da idu zajedno uz zadatak sa incima. Napisati program koji:

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

Primer 1

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

[Rešenje 1.20]

Zadatak 1.21 Napisati program koji za uneta vremena poletanja i sletanja aviona ispisuje dužinu trajanja leta. Možemo pretpostaviti da su poletanje i sletanje u istom danu. Ili da ovaj zadatak prebacimo u if, pa da moze da se proveri da li su vremena ispravno uneta, ili da ovde dodamo pretpostavku da su vremena ispravno zadata? Problem je sto u okviru grananja imamo slican zadatak. Mozda bi mogli da stavimo da ima dva ista zadatka, jedan sa pretpostavkom da su vremena ispravna, a drugi sa odgovarajucim ifovima koji to i proveravaju? Ne znam kako je najbolje, ali mi se cini da bi ta dva zadatka, zbog slicnosti, trebala da budu zajedno, tj jedan za drugim. Takodje, u jednom se zadaju sekunde, u drugom ne, mislim da bi format unosa trebao da bude isti.

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 standarnog ulaza se učitavaju vrednosti dve realne promenljive. Napisati program koji razmenjuje njihove vrednosti i ispisuje ih na standardni izlaz.

[Rešenje 1.22]

Zadatak 1.23 Napisati program koji za unete koordinate suprotnih temena pravougaonika (gornje levo i donje desno teme) ispisuje njegov obim i površinu. Pretpostaviti da su stranice pravougaonika paralelene koordinatnim osama.

[Rešenje 1.23]

Zadatak 1.24 Date su dve celobrojene promenljive a i b. Napisati program koji promenljivoj a dodeljuje njihovu sumu, a promenljivoj b njihovu razliku. NAPOMENA: *Ne koristiti pomoćne promenljive.*

[Rešenje 1.24]

Zadatak 1.25 Milena: Meni je ovaj zadatak potpuno nejasan, a kako nema resenje ne umem da ga formulisem kako treba ?!? Danijela: ideja u zadatku je ispis specijalnih karaktera i brojeva. Naime, desava se da kada radimo zvezdice, studenti pisu nesto ovako čhar c = '*'; printf("%c", c);"i slicno za znak + -, a nesto slicno pisu i kad trazimo da ispisu 0. Iako se to prica na predavanjima i vezbama oni pored brojnih informacija ne obrate paznju i 90% njih pravi slicne greske. Po meni, treba im na pocetku zadati zadatak tako da budu prinudjeni da sami moraju da razmisle i urade ono sto se trazi (i potom dati ispravno resenje). Cinjenica je da je ovo vise teorijski zadatak, ali dok oni pocnu da uce teoriju (sto je prvi test) nama na vezbama ova znanja uveliko trebaju.

Milena: Ok, razumem za ove specijalne karaktere, ali mi nedostaje neki adekvatan tekst koji bi ovom zadatku dao smisao. Koji je smisao za ispis ovih konstatni? Ako nije problem, pokusaj da preformulises zadatak da nekako ima neki smisao, tj da bude potpun i bez test primera. Npr, napisati program koji

na standardni izlaz ispisuje sve specijalne karaktere kao i brojeve od 0 do 9 (ili koje vec brojeve treba, mozda je dovoljan i samo jedan)...

Napisati program koji na standarni izlaz ispisuje sledeći tekst:

Primer 1

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

[Rešenje 1.25]

Zadatak 1.26 Sa standardnog unosa se unosi prirodan broj n i cifre c_1 i c_2. Napisati program ispisuje broj dobijen umetanjem cifara $c1$ i $c2$ na mesta stotina i hiljada broja n. NAPOMENA: *Za neke ulazne podatke može se dobiti neočekivan rezultat zbog prekoračenja, što ilustruje test primer broj xx.* Meni ovo deluje kao tekst zadatka cije je resenje dato kod 1.33, mozda gresim.

[Rešenje 1.26]

Zadatak 1.27 Sve zadatke sa operatorom ? grupisati na kraju ovog poglavlja. Napisati program koji za uneta dva cela broja ispisuje njihov maksimum.

[Rešenje 1.27]

Zadatak 1.28 Data su tri cela broja a, b, c. Napisati program koji dodeljuje promenljivoj rez vrednost 1 ako važi jedan od sledećih uslova: Milena: Deluje mi da ce resenje sa ? biti ruzno i da je ovo vise zadatak za if. Mozda od ovoga napraviti tri zadatka?

Danijela: Nije bas lepo, ali volim ovaj zadatak jer u jednom izrazu moraju dva puta da koriste ? sto je malo komplikovano i zahteva razmisljanje o sintaksi. Naravno, nije mnogo bitno, moze se pomeriti i u if ili potpuno odbaci.

Milena: Vazno je da resenja koja dajemo budu takva da imaju smisla, tj da ne ilustruju sintaksne mogucnosti jezika vec da su takva da bismo zadatak na taj nacin resavali i kada sve znamo, a ne da bismo na taj nacin resavali zadatak samo zato sto za bolje ne znamo. U tom smislu mi je zadatak sporan. U stvari, da li je ovo jedan zadatak ili su ovo tri zadatka? Meni je sve ovo sporno ako je u pitanju jedan zadatak, tj da treba sva tri uslova da budu ispunjena (da su razliciti parni brojevi pozitivni i ne veci od 100)? Ako su ovo tri zadatka, onda to ima smisla, ali treba od toga napraviti tri zadatka!

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 *rezultat* dodeliti vrednost 0. Ispisati vrednost promenljive *rezultat* na standardni izlaz.

[Rešenje 1.28]

Zadatak 1.29 Ne bih stampala da ne jer to bez if-a ne moze lepo da se uradi, tj printf u okviru operatora ? je mnogo ruzan stil programiranja. Zato bih i ovo formulisala kao štampanje vrednosti odgovarajuće promenljive koja ima vrednost 0 ili 1. Isto i u sledećem zadatku. Ili prebaciti zadatke u if. Napisati program koji ispituje da li se tačke $A(x_1, y_1)$ i $B(x_2, y_2)$ nalaze u istom kvadrantu i ispisuje odgovor `DA` ili `NE`.

[Rešenje 1.29]

Zadatak 1.30 Ne bih stampala da ne jer to bez if-a ne moze lepo da se uradi, tj printf u okviru operatora ? je mnogo ruzan stil programiranja. Zato bih i ovo formulisala kao štampanje vrednosti odgovarajuće promenljive koja ima vrednost 0 ili 1. Ili prebaciti zadatke u if. Napisati program koji ispituje da li se tačke $A(x_1, y_1)$, $B(x_2, y_2)$ i $C(x_3, y_3)$ nalaze na istoj pravoj i ispisujei odgovor `DA` ili `NE`.

[Rešenje 1.30]

Zadatak 1.31 Ovo bih mozda ostavila za if — potrebno je da provere i da li su koordiante ispravno unesene 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 Ovo mi je ok da bude reseno sa ?. Napisati program koji za unete vrednosti promenljivih `x` i `y` ispisuje vrednost sledećeg izraza:

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

.

[Rešenje 1.32]

Zadatak 1.33 Ovaj zadatak je uredu ukoliko mu se definise odgovarajuci tekst, a kao jedan test primer postavi ilustracija prekoracenja.) Ovo je ilustrativni zadatak kakav Milena ne voli da dolazi na vezbe tako da sumnjam da ce se odrzati ovde :)

[Rešenje 1.33]

Zadatak 1.34 Napisati program koji za unete realne vrednsoti a_{11}, a_{12}, a_{21}, a_{22} ispisuje vrednost determinante matrice:

```
a11 a12
a21 a22
```

Umesto verbatim staviti odgovarajući format za prikaz matrice. Pri ispisu vrednost zaokružiti na 4 decimale.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 1 2 3 4
 -2.0000
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: -1 0 0 1
 -1.0000
```

Primer 3

```
Interakcija sa programom:
 Unesite brojeve: 1.5 -2 3 4.5
 12.7500
```

Primer 4

```
Interakcija sa programom:
 Unesite brojeve: 0.01 0.01 0.5 7
 0.0650
```

[Rešenje 1.34]

1.1 Rešenja

Rešenje 1.1

```
1 /*
  Napisati program koji na standardni izlaz ispisuje tekst "Zdravo
      svete!"
3 */
  #include<stdio.h>
```

```c

int main()
{
  /* printf - funkcija pomocu koje se vrsi ispis */
  /* oznaka \n : prelazak u novi red */
  printf("Zdravo svete!\n");

  /* naredne dve naredbe ispisace reci Zdravo i svete u istom redu*/
  printf("Zdravo ");
  printf("svete \n");

  /* naredne dve naredbe ispisace reci Zdravo i svete u posebnim
    redovima */
  /* jer se u prvoj printf naredbi na kraju oznakom \n prelazi u novi
     red */

  printf("Zdravo \n");
  printf("svete \n");

  return 0;
}
```

Rešenje 1.2

```c
/*
Napisati program koji poziva korisnika da unese dva cela broja sa
    standardnog ulaza,
a zatim ispisuje:
1) unete vrednosti
2) njihov zbir
3) njihovu razliku
4) njihov proizvod
5) ceo deo pri deljenju jednog broja drugim brojem
6) ostatak pri deljenju jednog broja drugim brojem

*/

#include<stdio.h>

int main()
{
   int x;
   int y;
   int rezultat;

   printf("Unesi vrednost celobrojne promenljive x:");
   scanf("%d", &x); /* "%d" - specifikator tipa koji treba uneti (%d
    za int)
                       &x   - adresa promenljive x
                    */

```

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

    /* 1) ispis unetih vrednosti */
    printf("x=%d, y=%d\n", x,y); /* umesto prvog %d bice ispisana
     vrednost promenljive x */
                                 /* umesto drugog %d bice ispisana
     vrednost promenljive y */

    /* 2) ispis zbira */
    rezultat = x+y; /* dodelimo vrednost promenljivoj rezultat */
    printf("Zbir je %d\n", rezultat);

    /* 3) ispis razlike */
    printf("Razlika je %d\n",x-y); /* mozemo ispisivati direktno
     vrednost izraza x-y i bez */
                                   /* njegovog dodeljivanja posebnoj
     promenljivoj */


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

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

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

Rešenje 1.3

```
/*
   Napisati program koji sa standardnog ulaza ucitava realnu vrednost
     izrazenu
```

```
   u incima, konvertuje tu vrednost u centimetre i ispisuje je na
    standardni izlaz
   zaokruzenu na dve decimale.
*/
#include <stdio.h>

int main()
{
  float in;
  float cm;

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

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

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

  return 0;
}
```

Rešenje 1.4

```
/*
   Napisati program koji sa standardnog ulaza ucitava duzinu
    poluprecnika kruga
   i na standardni izlaz ispisuje njegov obim i povrsinu
*/

#include <stdio.h>
#include <math.h> /* biblioteka matematickih funkcija; za prevodjenje
     je neophodno ukljuciti opciju -lm
                     npr. gcc primer.c -lm */
int main()
{
  int r;
  float O;
  float P;
  printf("Unesi poluprecnik kruga:");
  scanf("%d", &r);

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

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

  return 0;
}
```

Rešenje 1.5

```c
/*
Napisati program koji ucitava trocifreni broj koji se
unosi sa standardnog ulaza i ispisuje njegove cifre na
standardni izlaz.
*/
#include <stdio.h>
int main()
{
   int x;
   int cifra_jedinice;
   int cifra_desetice;
   int cifra_stotine;

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

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

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

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

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

Rešenje 1.6

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

```
11
    int cifra_jedinice;
13  int cifra_desetice;
    int cifra_stotine;
15
    printf("Unesi trocifreni broj:");
17  scanf("%d", &x);

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

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

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

27  return 0;
}
```

Rešenje 1.7

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

6 #include <stdio.h>
  #include <math.h>
8 int main()
  {
10    unsigned int a;
      float P;
12
      printf("Unesi duzinu stranice jednakostranicnog trougla:");
14    scanf("%d",&a);

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

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

Rešenje 1.8

```
/*
```

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

#include <stdio.h>

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

   printf("%d=%d*1000+ ", x,x/1000);
   x=x%1000;
   printf("%d*100 +", x/100);
   x=x%100;
   printf("%d*50 +",x/50);
   x=x%50;
   printf("%d*10 +", x/10);
   x=x%10;
   printf("%d*1\n", x);
   return 0;
}
```

Rešenje 1.9

```c
/*
Napisati program koji za tri cela broja koja se unose sa standardnog
    ulaza
ispisuje njihovu aritmeticku sredinu na standardni izlaz.
*/

#include<stdio.h>

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

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

  as=(a+b+c)/3.0; /* da bismo dobili kolicnik, jedan argument mora da
     bude realan broj */

  /*
  moguce je i:
```

```
  as=1.0*(a+b+c)/3;
  ili
  as=((float)(a+b+c))/3;
  */

  printf("Aritmeticka sredina unetih brojeva je %f\n", as);
  return 0;
}
```

Rešenje 1.10

```
/*
   Napisati program koji poziva korisnika da unese dve celobrojne
    vrednosti,
   smesta ih u promenljive x i y, zamenjuje vrednosti tih
    promenljivih i
   stampa ih na standardni izlaz.
*/
#include<stdio.h>
int main()
{
   int x,y;
   int t;
   printf("Unesi dve celobrojne vrednosti:");
   scanf("%d%d",&x,&y);
   printf("x=%d, y=%d\n",x,y);
   t=x;  /* promenljiva t dobija vrednost promenljive x */
   x=y;  /* promenljiva x dobija vrednost promenljive y */
   y=t;  /* promenljiva y dobija vrednost promenljive t */
   printf("nakon zamene, x=%d, y=%d\n",x,y);
   return 0;
}
```

Rešenje 1.11

```
/* Napisati program koji omogucava korisniku da unese ceo broj, a
    zatim
ispisuje njegov kvadrat i kub */

#include <stdio.h>

int main(){
  int n;

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

  /* Ispisujemo trazene vrednosti */
```

```
  printf("Kvadrat: %d\n", n*n);
  printf("Kub: %d\n", n*n*n);

  /* Zavrsavamo sa programom */
  return 0;
}
```

Rešenje 1.12

```
/* Napisati program koji za unete stranice pravougaonika ispisuje
    njegov obim i
povrsinu */

#include <stdio.h>

int main(){
  int a, b;
  int obim, povrsina;

  /* Ucitavamo potrebne podatke */
  printf("Unesite duzine stranica pravougaonika: ");
  scanf("%d %d", &a, &b);

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

  /* Povrsina */
  povrsina=a*b;

  /* Ispisujemo trazene vrednosti */
  printf("Obim: %d\n", obim);
  printf("Povrsina: %d\n", povrsina);

  /* Zavrsavamo sa programom */
  return 0;
}
```

Rešenje 1.13

```
/* Napisati program koji za unete duzine stranica trougla, ispisuje
    njegov obim
i povrsinu */

/* Napomena: prilikom prevodjenja programa treba navesti opciju -lm
 * na primer, gcc trougao.c -lm
 */

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

```c

int main(){
  float a, b, c;
  float obim, s, povrsina;

  /* Ucitavamo potrebne podatke */
  printf("Unesite duzine stranica trougla: ");
  scanf("%f %f %f", &a, &b, &c);

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

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

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

  return 0;
}
```

Rešenje 1.14

```c
/* Napisati program koji za unete dimenzije sobe u metrima (duzinu, sirinu i
visinu) ispisuje koju povrsinu treba da okreci moler. Uracunati da na vrata i
prozore otpada oko 20%. Omoguciti i unos cene usluge po kvadratnom metru i
izracunati zaradu koju ostvaruje moler.
 */

#include <stdio.h>

int main(){
  int duzina, sirina, visina;
  int cena;
  float povrsina_za_krecenje;
  float ukupna_cena;

  /* Ucitavamo duzinu, sirinu i visinu sobe */
  printf("Unesite dimenzije sobe: ");
  scanf("%d %d %d", &duzina, &sirina, &visina);

  /* Ucitavamo cenu krecenja */
  printf("Unesite cenu po kvadratnom metru: ");
  scanf("%d", &cena);
```

```
23    /* Povrsina za krecenje odgovara povrsini kvadra - bez poda jer se
        on ne
kreci */
25    povrsina_za_krecenje=0.8*(duzina*sirina+2*duzina*visina+2*sirina*
        visina);
      ukupna_cena=povrsina_za_krecenje*cena;
27
      /* Ispisujemo trazene podatke */
29    printf("Moler treba da okreci %.2f kvadratna metra\n",
        povrsina_za_krecenje);
      printf("Cena krecenja je %.2f\n", ukupna_cena);
31
      /* Zavrsavamo sa programom */
33    return 0;
}
```

Rešenje 1.15

```
/* Napisati program koji za unetu kolicinu jabuka u kilogramima i
     unetu cenu
2 po kilogramu ispisuje ukupan iznos koji treba platiti */

4 #include <stdio.h>

6 int main(){
    int kolicina;
8   int cena;

10  /* Ucitavamo potrebne podatke */
    printf("Unesite kolicinu jabuka (u kg): ");
12  scanf("%d", &kolicina);

14  printf("Unesite cenu (u dinarima): ");
    scanf("%d", &cena);
16
    /* Ispisujemo trazeni iznos */
18  printf("Molimo platite %d dinara.\n", kolicina*cena);

20  /* Zavrsavamo sa programom */
    return 0;
22
}
```

Rešenje 1.16

```
1 /* Napisati program koji pomaze kasirki da obracuna kusur tako sto od
       nje trazi
da unese cenu artikla, kolicinu artikla i iznos koji je dobila od
     kupca. */
```

```c

#include <stdio.h>

int main(){
  int cena;
  int kolicina;
  int iznos;
  int kusur;

  /* Ucitavamo potrebne podatke */
  printf("Unesite redom cenu, kolicinu i iznos: ");
  scanf("%d %d %d", &cena, &kolicina, &iznos);

  /* Izracunavamo kusur */
  kusur=iznos - kolicina*cena;

  /* I ispisujemo trazenu vrednost */
  printf("Kusur je %d dinara.\n", kusur);

  return 0;
}
```

Rešenje 1.17

```c
/* Napisati program koji prirodnom cetvorocifrenom broju koji se
    unosi sa
standardnog ulaza:

a) izracunava proizvod cifara
b) izracunava razliku sume krajnjih i srednjih cifara
c) izracunava sumu kvadrata cifara
d) odredjuje broj koji se dobija ispisom cifara u obrnutom poretku
e) odredjuje broj koji se dobija zamenom cifre jedinice i cifre
    stotine
*/

#include <stdio.h>

int main(){

  int n;
  int j, d, s, h;
  int proizvod_cifara, razlika_cifara, suma_kvadrata, broj_obrnuto,
    broj_zamena;

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

  /* Izdvajamo cifre broja i to redom: j -jedinice, d - desetice, s -
     stotine i
```

```
h - hiljade */
  j=n%10;
  d=(n/10)%10;
  s=(n/100)%10;
  h=n/1000;

  /* Izracunavamo proizvod cifara */
  proizvod_cifara=j*d*s*h;
  printf("Proizvod cifara: %d\n", proizvod_cifara);

  /* Izracunavamo razliku sume krajnjih i srednjih cifara */
  razlika_cifara=(h+j)-(s+d);
  printf("Razlika sume krajnjih i srednjih: %d\n", razlika_cifara);

  /* Izracunavamo sumu kvadrata cifara */
  suma_kvadrata=j*j+d*d+s*s+h*h;
  printf("Suma kvadrata cifara: %d\n", suma_kvadrata);

  /* Odredjujemo broj zapisan istim ciframa ali u obrnutom redosledu
    */
  broj_obrnuto= j*1000+d*100+s*10+h;
  printf("Broj u obrnutom poretku: %d\n", broj_obrnuto);


  /* Odredjujemo broj u kojem su cifra jedinica i cifra stotina
    zamenile mesta
*/
  broj_zamena=h*1000+j*100+d*10+s;
  printf("Broj sa zamenjenom cifrom jedinica i stotina: %d\n",
    broj_zamena);


  return 0;
}
```

Rešenje 1.18

Rešenje 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 pocinje od 1.

Uputstvo: koristiti funkciju pow iz math.h biblioteke za racunanje
    stepena
broja.  Na primer, pow(10, 2)=100.000.
S obzirom da funkcija pow vraca double vrednosti, pre upotrebe je
    potrebno
```

```
konvertovati (kastovati) ove vrednosti u int tip.

Prilikom prevodjenja programa koristiti -lm opciju.
*/

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

int main(){
  int x, c, p;
  int levo, desno;
  int novo_x;

  /* Ucitavamo potrebne vrednosti */
  printf("Unesite redom x, c i p: ");
  scanf("%d %d %d", &x, &c, &p);

  /* Odredjujemo deo broja koji se nalazi desno od pozicije p */
  desno=x%(int)pow(10, p-1);

  /* Odredjujemo deo broja koji se nalazi levo od pozicije p */
  levo=x/(int)pow(10, p-1);

  /* Odredjujemo novi broj */
  novo_x=levo*(int)pow(10, p) +c*(int)pow(10, p-1) + desno;

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

  /* Zavrsavamo sa programom */
  return 0;

}
```

Rešenje 1.20

```
/* Napisati program koji:
1) unetu duzinu u miljama konvertuje u kilometre (1 mi = 1.609344 km)
2) unetu tezinu u funtama konvertuje u kilograme ( 1 lb = 0.45359237
    kg)
3) unetu temperaturu u celzijusima konvertuje u farenhajte (F =9 * C
    /5 + 32)
*/

 #include <stdio.h>


int main(){
  float milje;
  float kilometri;

```

```
  printf("Unesite duzinu u miljama: ");
  scanf("%f", &milje);
  kilometri=milje*1.609344;
  printf("Vrednost duzine u kilometrima je: %f\n", kilometri);


  float funte;
  float kilogrami;

  printf("Unesite tezinu u funtama: ");
  scanf("%f", &funte);
  kilogrami=funte*0.45359237;
  printf("Vrednost tezine u kilogramima je: %f\n", kilogrami);

  float celzjusi;
  float farenhajti;

  printf("Unesite temperaturu u celzijusima: ");
  scanf("%f", &celzjusi);
  farenhajti=9*celzjusi/5+32;
  printf("Vrednost temperature u farenhajtima je: %f\n", farenhajti);

  return 0;
}
```

Rešenje 1.21

```
/* Napisati program koji ucitava sa standardnog ulaza vreme poletanja
     i vreme
sletanja aviona, a potom ispisuje duzinu trajanja leta. Mozemo
    pretpostaviti da
su poletanje i sletanje u istom danu. */

#include <stdio.h>

int main(){

  int poletanje, poletanje_sat, poletanje_minut, poletanje_sekund;
  int sletanje, sletanje_sat, sletanje_minut, sletanje_sekund;
  int duzina, duzina_sat, duzina_minut, duzina_sekund;

  printf("Unesite vreme poletanja: ");
  scanf("%d %d %d", &poletanje_sat, &poletanje_minut, &
    poletanje_sekund);


  printf("Unesite vreme sletanja: ");
  scanf("%d %d %d", &sletanje_sat, &sletanje_minut, &sletanje_sekund)
    ;

  /* Pretvoricemo i vreme poletanja i vreme sletanja u sekunde */
```

```
    poletanje=poletanje_sat*3600+poletanje_minut*60+poletanje_sekund;
22    sletanje=sletanje_sat*3600 + sletanje_minut*60 +sletanje_sekund;

24    /* I izracunati razliku u sekundama */
  duzina=sletanje-poletanje;
26
    /* Izdvajamo broj sati, broj minuta i broj sekundi */
28  duzina_sat=duzina/3600;
  duzina_minut=(duzina%3600)/60;
30  duzina_sekund=(duzina%3600)%60;

32
    /* I ispisujemo rezultat */
34  printf("Duzina trajanja leta je: %d h %d min %d sec\n", duzina_sat,
      duzina_minut, duzina_sekund);

36
  return 0;
38 }
```

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

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

/* u zaglavlju limits.h
su definisane maksimalne i minimalne
vrednosti za svaki tip podataka
npr. INT_MAX konstanta je najveci ceo
broj koji moze da se stavi
u promenljivu tipa int
zbog toga za poslednji test primer
ne dobijamo zeljeni broj
jer je doslo do prekoracenja
novibroj je veci od INT_MAX
*/

/* test primeri:
broj: 140
c1: 2
c2: 3

novibroj: 13240
----------------------
broj: 526
c1: 7
c2: 4

novibroj: 54726
----------------------
broj: 25
c1: 9
c2: 5

novibroj: 5925
----------------------
test primer koji dovodi do prekoracenja, pa zbog toga
ne dobijamo zeljeni rezultat:

broj: 100000000
c1: 5
c2: 1

novibroj: neocekivan rezultat ---> PREKORACENJE

*/

int main(){
int broj,c1,c2,z1,z2;
int novibroj;
int dostatak1, dostatak2 ;
```

```
51 printf("unesi broj: ");
   scanf("%d", &broj);
53 printf("unesi c1: ");
   scanf("%d", &c1);
55 printf("unesi c2: ");
   scanf("%d", &c2);
57
   /* najbolje odmah da se kastuje z1 jer se kasnije cesto
59 koristi u racunu pa da ne ponavljamo (int) */
   // za stotine pozicija je 3 ---> z1 = (int)pow(10,3-1);
61 z1 = (int)pow(10,2);

63 dostatak1 = broj % z1;

65 /*
    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
   */
69 novibroj = broj / z1 * z1 * 10 + z1 * c1 + dostatak1 ;

71 //sada u novibroj insertujemo cifru c2 na poziciju 4 - za hiljade

73 z2 = (int)pow(10,3);

75 dostatak2 = novibroj % z2;

77 /*
    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
   */
81 novibroj = novibroj / z2 * z2 * 10 + z2 * c2 + dostatak2 ;


83
   printf("Novi broj je: %d\n", novibroj);
85 printf("Maksimalna vrednost za int je: %d\n", INT_MAX);

87 return 0;
   }
```

Rešenje 1.34

2

Kontrola toka

2.1 Naredbe grananja

TODO Iz svih resenja pobrisati formulaciju zadatka.

TODO U resenjima gde imena promenljivih nisu deskriptivna treba dodati komentare prilikom deklaracija cemu sluze odgovarajuca imena promenljivih.

TODO Da li pominjati stadndardni ulaz/izlaz? Negde se pominju, negde ne, deluje mi da to opterecuje zadatke, ali bi u svakom slucaju to rebalo da je konzistentno.

Zadatak 2.1 Napisati program koji za dva cela broja uneta sa standardnog ulaza ispisuje njihov minimum na standardni izlaz.

[Rešenje 2.1]

Zadatak 2.2 Napisati program koji za dva cela broja uneta sa standardnog ulaza ispisuje njihov maksimum na standardni izlaz. Ovaj zadatak mozda da ide bez resenja?

[Rešenje 2.2]

Zadatak 2.3 Napisati program koji za godinu koja se unosi sa standardnog ulaza na standardni izlaz ispisuje da li je prestupna.

[Rešenje 2.3]

Zadatak 2.4 Napisati program koji za uneti ceo broj ispisuje njegovu recipročnu vrednost. TODO U resenje dodati komentar na temu implicitne konverzije kod deljenja

[Rešenje 2.4]

Zadatak 2.5 Napisati program koji za uneti ceo broj x ispisuje njegov znak, tj da li je broj jednak nuli, manji od nule ili veći od nule.

[Rešenje 2.5]

Zadatak 2.6 Napisati program koji za uneto vreme (broj sati iz intervala $[0, 24)$ i broj minuta iz intervala $[0, 60)$) ispisuje koliko je sati i minuta ostalo do ponoći. TODO Dodati u rešenje proveru ispravnosti unetog vremena, tj ako neko unese neispravno vreme.

[Rešenje 2.6]

Zadatak 2.7 Sa standardnog ulaza se unose cene tri artikla. Ukoliko se najjeftiniji artikal dobija za 1 dinar, napisati program koji izračunava ukupnu cenu, kao i koliko dinara se uštedi zahvaljujući popustu.

[Rešenje 2.7]

Zadatak 2.8 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 2.8]

Zadatak 2.9 Napisati program koji za koeficijente kvadratne jednačine, koji se unose sa standardnog ulaza, ispisuje na standardni izlaz koliko realnih rešenja jednačina ima i ako ih ima, ispisuje rešenja jednačine zaokružena na dve decimale.

[Rešenje 2.9]

Zadatak 2.10 Napisati program koji učitava tri cela broja i ispisuje zbir onih unetih brojeva koji su pozitivni.

[Rešenje 2.10]

Zadatak 2.11 Napisati program koji za realan broj unet sa standardnog ulaza ispisuje njegovu apsolutnu vrednost.

[Rešenje 2.11]

Zadatak 2.12 Napisati program koji za karakter unet sa standardnog ulaza ispisuje da li je samoglasnik.

[Rešenje 2.12]

Zadatak 2.13 Napisati program koji za uneti dan i mesec ispisuje godišnje doba kojem pripadaju. NAPOMENA: *Podrazumevati da je unos korektan.*

[Rešenje 2.13]

Zadatak 2.14 Napisati program koji za uneti četvorocifreni broj proverava da li su njegove cifre uređene rastuće, opadajuće ili nisu uređene i štampa odgovarajuću poruku na standardni izlaz. Voditi računa o nekorektnim unosima. Mislim da bi uvek trebalo da vode racuna o nekorektnim unosima, osim kada se stavi napomena da se podrazumeva da je unos korektan? Zato bi ovde ovo izbrisala?

[Rešenje 2.14]

*** Zadatak 2.15** Zadatke sa swich-om bih grupisala na kraj Sa standardnog ulaza unose se jedan karakter i 8 realnih brojeva koji predstavljaju koordinate četiri tačke: $A(x_1, y_1), B(x_2, y_2), C(x_3, y_3), D(x_4, y_4)$. Na osnovu unetog karaktera ispisuje se odgovarajuća poruka na standardni izlaz:

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

- ukoliko je uneti karakter h - proverava da li su unete tačke kolinearne i ukoliko jesu, ispisuje jednačinu prave kojoj pripadaju. Na primer, tačke $(1, 2), (2, 3), (3, 4), (4, 5)$ su kolinearne i pripadaju pravoj $y = x + 1$, tačke $(1, 1), (1, 2), (1, 3), (1, 4)$ su kolinearne i pripadaju pravoj $x = 1$, a tačke $(1, 1), (2, 1), (2, 2), (1, 2)$ nisu kolinearne.

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

[Rešenje 2.15]

Zadatak 2.16 Napisati program koji za uneti četvorocifreni ceo broj ispisuje njegovu najveću cifru.

Primer 1

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

Primer 2

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

[Rešenje 2.16]

Zadatak 2.17 Broj je Armstrongov ako je jednak zbiru kubova svojih cifara. Napisati program koji za dati trocifren broj proverava da li je Armstrongov.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 2.17]

Zadatak 2.18 U nizu 12345678910111213....9899 ispisani su redom brojevi od 1 do 99. Napisati program koji za uneti ceo broj k ($1 \geq k \geq 189$) ispisuje cifru koja se nalazi na k-toj poziciji datog niza.

Primer 1

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

Primer 2

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

[Rešenje 2.18]

Zadatak 2.19 Sa standardnog ulaza se unosi četvorocifreni pozitivan broj. Napisati program koji ispisuje proizvod parnih cifara datog broja. Izmeniti poruku u resenju!

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, broj nije cetvorocifren!
```

[Rešenje 2.19]

Zadatak 2.20 Sa standarnog ulaza se unosi pet karaktera. Napisati program koji u slučaju da je prvi karakter veliko ili malo slovo *a* ispisuje unete karaktere obrnutim redosledom, a u suprotnom ništa ne ispisuje. Mozda umesto a da bude o, kao skracenica od obrni? Inace, ovaj zadatak je poprilicno besmislen :-)

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 2.20]

Zadatak 2.21 Napisati program koji za karakter koji učitava:

- u slučaju da je uneta cifra, ispisuje nju i njen ASCII kod Ovo se ne razlikuje od poslednje stavke: dakle ili ovde treba nesto dodati sto ce ga razlikovati od poslednje stavke, npr da se ispise i broj cifre, tj da vide c-'0'
- u slučaju da je uneto malo slovo, ispisuje njega, njegov ASCII kod, odgovarajuće veliko slovo i njegov ASCII kod
- u slučaju da je uneto veliko slovo, ispisuje njega, njegov ASCII kod, odgovarajuće malo slovo i njegov ASCII kod
- u ostalim slučajevima, ispisuje uneti karakter i njegov ASCII kod

[Rešenje 2.21]

Zadatak 2.22 Ovaj zadatak je jako slican sa prethodnim, ne znam da li nam trebaju oba resena. Mozda jedan da bude za vezbe, resen, a drugi za praktikume, neresen? Sa standarnog ulaza se unosi karakter *c*. Napisati program koji:

a) ako je c malo slovo, zamenjuje ga odgovarajućim velikim i ispisuje na standardni izlaz

b) ako je c veliko slovo, zamenjuje ga odgovarajućim malim i ispisuje na standardni izlaz

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

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

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 2.22]

Zadatak 2.23 Napisati program koji za unetih pet karaktera ispisuje koliko je među njima malih slova.

Primer 1

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

Primer 2

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

[Rešenje 2.23]

Zadatak 2.24 Sa standardnog ulaza se unosi četvorocifren ceo broj. Napisati program koji ispisuje broj koji se dobija kada se unetom broju razmene najmanja i najveća cifra. Izmeniti poruku o gresci u resenju. Izabrati najbolje test primere.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite broj: 247
 Greska, broj nije cetvorocifren!
```

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 **??**]

Zadatak 2.25 Spajanjem cifara dva trocifrena broja dobija se šestocifren broj. Na primer, spajanjem brojeva 321 i 654 dobija se broj 321654. Sa standardnog ulaza se unose tri neoznačena trocifrena broja. Napisati program koji spaja dva od ta tri trocifrena broja tako da se dobije naveći mogući šestocifren broj. Dobijeni šestocifreni broj ispisati na standardni izlaz. Ako neki od unetih brojeva nije trocifren, smatrati da ulaz nije ispravn. Izmeniti poruku o gresci u resenju. Izabrati najbolje test primere.

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, ulaz nije ispravan!
```

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 2.25]

Zadatak 2.26 Napisati program za rad sa intervalima. Za dva intervala realne prave $[a1, b1]$ i $[a2, b2]$, program treba da odredi:

a) dužinu zajedničkog dela ta dva intervala

b) najveći interval sadržan u datim intervalima (presek),a ako on ne postoji dati odgovarajuću poruku. (?! zar ovo nije isto sto i a?) pod a je duzina a

ovde je interval, pogledati test primer

c) dužinu realne prave koju pokrivaju ta dva intervala

d) najmanji interval koji sadrži date intervale.

Primer 1

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

Primer 2

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

[Rešenje 2.26]

Zadatak 2.27 Data je funkcija $f(x) = 2 \cdot cos(x) - x^3$. Sa standarnog ulaza se unosi realan broj x i broj k koje može biti 1, 2 ili 3. Napisati program koji izračunava vrednost funkcije $F(k, x) = f(f(f(...f(x)))$ gde je funkcija f primenjena k-puta. U slučaju neispravnog ulaza, odštampati odgovarajuću poruku o grešci. dodati test primer za neispravan ulaz

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 2.27]

Zadatak 2.28 Napisati program koji za uneti redni broj dana u nedelji ispisuje ime odgovarajućeg dana. U slučaju pogrešnog unosa ispisati odgovarajuću poruku.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 2.28]

Zadatak 2.29 Sa standardnog ulaza se učitavaju dva cela broja i jedan od karaktera +, -, *, / ili % koji predstavlja računsku operaciju. Napisatiti program koji ispisuje vrednost izraza dobijenog primenom ove operacije na date argumente. U slučaju pogrešnog unosa ispisati odgovarajuću poruku. NAPOMENA: *Koristiti naredbu switch.* Nisam sigurna da je potrebno ovde traziti da koriste switch, dovoljno je da resenje to koristi...

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 2.29]

Zadatak 2.30 Napisati program koji za uneti datum u formatu *dan.mesec.godina.* proverava da li je korektan.

Primer 1

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

Primer 2

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

[Rešenje 2.30]

Zadatak 2.31 Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum prethodnog dana.

Primer 1

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

Primer 2

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

[Rešenje 2.31]

Zadatak 2.32 Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum narednog dana. Mislim da bi bilo sasvim u redu da ovaj zadatak bude bez resenja.

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 2.32]

Zadatak 2.33 Grupisati zadatke koji rade sa karakterima. Sa standardnog ulaza se unosi 5 karaktera. Napisati program koji ispisuje 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 2.33]

Zadatak 2.34 Sa standardnog ulaza se unose 5 karaktera. Napisati program koji ispisuje 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 2.107]

Zadatak 2.35 Korisnik unosi tri cela broja: P, Q i R. Nakon toga unosi i dva karaktera, $op1$ i $op2$. Ovi karakteri predstavljaju operacije nad unetim brojevima i imaju naredno značenje:

- karakter **k** predstavlja logičku konjukciju

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

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

[Rešenje 2.35]

Zadatak 2.36 Tekst

[Rešenje 2.36]

Zadatak 2.37 Tekst

[Rešenje 2.37]

Zadatak 2.38 Tekst

[Rešenje 2.38]

Zadatak 2.39 Tekst

[Rešenje 2.39]

2.2 Rešenja

Rešenje 2.1

```
/*
  Napisati program koji za 2 cela broja uneta sa standardnog ulaza
  ispisuje njihov minimum na standardni izlaz.
*/

#include <stdio.h>
int main()
{
   int a,b;
   int min1;
   int min2;
   int min3;
```

```
13
    scanf("%d%d",&a,&b);
15

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

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

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

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

35  return 0;
}
```

Rešenje 2.2

Rešenje 2.3

```
/*
2   Napisati program koji za godinu koja se unosi sa standardnog ulaza
      na standardni izlaz
    ispisuje da li je prestupna.
4 */

6 #include <stdio.h>

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

14      if ((x%4==0 && x%100!=0) || x%400==0)
            printf("Godina je prestupna\n");
16      else
            printf("Godina nije prestupna\n");
18      return 0;
}
```

Rešenje 2.4

```c
/*
Napisati program koji za uneti ceo broj ispisuje njegovu reciprocnu
    vrednost.
Ukoliko je uneti broj jednak nuli, ispisati poruku "Nedozvoljeno
    deljenje nulom".
*/

#include <stdio.h>

int main()
{
   int x;
   float rx;

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

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

   if (x==0)
      printf("Nedozvoljeno deljenje nulom\n");
   else
   {
      rx = 1.0/x;
      printf("Reciprocna vrednost unetog broja:%f\n",rx);
   }

   return 0;
}
```

Rešenje 2.5

```c
#include <stdio.h>
/*
Napisati program koji za uneti ceo broj x ispisuje da li je jednak
    nuli,
manji od nule ili veci od nule.
*/
int main()
{
  int x;
  printf("Unesi ceo broj:");
  scanf("%d",&x);
```

```
11
   /*
13     obratiti paznju:
       x==0 - relacija jednakosti (da li je vrednost promenljive x
      jednaka nuli)
15     x=0 - naredba dodele (promenljiva x dobija vrednost nula)
   */
17 if (x==0)
     printf("Broj je jednak nuli\n");
19 else if (x<0)
     printf("Broj je manji od nule\n");
21 else
     printf("Broj je veci od nule\n");
23
   return 0;
25 }
```

Rešenje 2.6

```
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 itervalu [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 2.7

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

#include <stdio.h>
int main()
{
   int a,b,c;
   int min;
   int min1;
   int min2;
   int cena_bez_popusta, cena_sa_popustom;

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


   if (a<b)
      if (a<c) /* poredak: a<b,a<c => a,b,c ili a,c,b */
         min=a;
      else     /* poredak: a<b, a>=c => a<b, c<=a => c,a,b */
         min=c;
   else        /* b<=a */
      if (b<c) /* poredak:  b<=a,b<c => b,a,c ili b,c,a */
         min=b;
      else     /* poredak: b<=a, c<=b => c,b,a */
         min=c;

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

   /* 2. nacin */
   /* najpre odredimo minimum brojeva a,b*/
   if (a<b)
      min1=a;
   else
      min1=b;

   if (c<min1)
      min1=c;
   printf("Minimum unetih brojeva (2.nacin) je %d\n",min1);

   /* 3. nacin */
   min2=a;
   if(min2>b)
      min2=b;
```

```
49    if(min2>c)
         min2=c;
51
      printf("Minimum unetih brojeva (3.nacin) je %d\n",min2);
53
      cena_bez_popusta=a+b+c;
55    cena_sa_popustom = cena_bez_popusta - min2 + 1;

57    printf("Cena sa popustom: %.2f\n Cena bez popusta: %d\n Usteda:
      %.2f\n", cena_sa_popustom, cena_bez_popusta, cena_bez_popusta-
      cena_sa_popustom);

59    return 0;
}
```

Rešenje 2.8

Rešenje 2.9

```
1  /*
   Napisati program koji za koeficijente kvadratne jednacine
3  koji se unose sa standardnog ulaza na standardni izlaz
   ispisuje koliko realnih resenja jednacina ima i ako ih ima, ispisuje
       resenja jednacine
5  zaokruzena na dve decimale.
   */
7  #include <stdio.h>
   #include <math.h>
9  int main()
   {
11    float a,b,c;
      float D;
13    float x1,x2;
      printf("Unesi koeficijente kvadratne jednacine:");
15    scanf("%f%f%f",&a,&b,&c);

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

```
    {
31
       D=b*b-4*a*c; /* funkcija sqrt nalazi se u biblioteci math.h (
     prevodjenje sa -lm opcijom) */
33     if (D<0)
         printf("Jednacina nema realnih resenja\n");
35     else if (D>0)
       {
37       x1 = (-b+sqrt(D))/(2*a);
         x2 = (-b-sqrt(D))/(2*a);
39       printf("Jednacina ima dva razlicita realna resenja %.2f i %.2
     f\n",x1,x2);
       }
41     else
       {
43       x1 = (-b)/(2*a);
         printf("Jednacina ima jedinstveno realno resenje %.2f\n",x1);
45     }
    }
47
    return 0;
49 }
```

Rešenje 2.10

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

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

21  if (a>0)
       s=s+a; /* naredba dodele: vrednost izraza a desne strane znaka
      jednakosti
23               dodeljujemo promenljivoj sa leve strane znaka
      jednakosti.
```

```
                Staru vrednost promenljive s saberemo sa vrednoscu
    promenljive a
                i dobijenu vrednost upisemo u promenljivu s */

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

  if (c>0)
     s+=c;

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

Rešenje 2.11

```
/*

Napisati program koji za realan broj unet sa standardnog ulaza
ispisuje njegovu apsolutnu vrednost.

*/

#include<stdio.h>
#include<math.h>
#include<stdlib.h>
int main()
{
   float x;
   float y;

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

   /* 1. nacin */
   if (x>0)
     y=x;
   else
     y=-x;
   printf("Apsolutna vrednost broja %f je %f\n",x,y);

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

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

   /* 3. nacin - pogresan!*/
```

```
    y=abs(x); /* funkcija abs vraca 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 2.12

```
2  /*
   Napisati program koji poziva korisnika da unese jedan karakter i
       ispisuje
4  da li je uneti karakter samoglasnik.
   */
6
   #include <stdio.h>
8
   int main()
10 {
     char c;
12   printf("Unesi jedan karakter:");
     scanf("%c", &c);
14   switch(c)
     {
16     case 'A' :
       case 'E' :
18     case 'I' :
       case 'O' :
20     case 'U' :
       case 'a' :
22     case 'e' :
       case 'i' :
24     case 'o' :
       case 'u' : printf("Uneli ste samoglasnik\n");
26            break;
       default : printf("Niste uneli samoglasnik\n");
28           break;
     }
30
     return 0;
32 }
```

Rešenje 2.13

```c
/*
Napisati program koji za uneti dan i mesec ispisuje godisnje doba kom
pripadaju. Mozemo podrazumevati da je unos korektan.
*/

#include <stdio.h>

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

  switch(m) /* argument u naredbi switch mora biti celobrojna
    promenljiva */
  {
     case 1: /* argument u naredbi case mora biti celobrojna
    konstanta */
     case 2: /* ispitujemo da li je m==2 */
        printf("zima\n");
        break;
     case 3:
        if (d<21)
          printf("zima\n");
        else
          printf("prolece\n");
        break;
     case 4:
     case 5:
        printf("prolece\n");
        break;
     case 6:
        if (d<21)
          printf("prolece");
        else
          printf("leto");
        break;
     case 7:
     case 8:
        printf("leto");
        break;
     case 9:
        if (d<23)
          printf("leto\n");
        else
          printf("jesen\n");
        break;
     case 10:
     case 11:
        printf("jesen\n");
        break;
```

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

Rešenje 2.14

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

9 Unesi jedan cetvorocifreni broj: -1357
  Cifre su mu uredjene neopadajuce.
11
  ili ovako
13
  Unesi jedan cetvorocifreni broj: 9952
15 Cifre su mu uredjene nerastuce.

17 ili ovako

19 Unesi jedan cetvorocifreni broj: 9572
  Cifre su mu nisu uredjene.
21
  Unesi jedan cetvorocifreni broj: 123
23 Uneti broj nije cetvorocifren.

25 */

27

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

31 int main()
  {
33   int x;
     char c1;    /* cifre su brojevi {0,1,2,3,4,5,6,7,8,9} */
35   char c10;
     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 neopadajuce \n");
59     else if (c1000>=c100 && c100>=c10 && c10>=c1)
          printf("Cifre su uredjene nerastuce \n");
61     else
          printf("Cifre nisu uredjene\n");
63 }
   return 0;
65 }
```

Rešenje 2.15

```
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 unetetog 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
x1 * p + x2 * q = x4 - x3
y1 * p + y2 * q = y4 - y3
    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
              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

*/

#include<stdio.h>
#include<math.h>
int main()
{
   char c;
   float x1,y1,x2,y2,x3,y3,x4,y4;
   float kab,kbc,kad;
   float dab,dad;
   float delta, deltap, deltaq;
   float O;
   float k,n;

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

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

   switch (c)
   {
       case 'k':
          if (y1==y2 && y3==y4 && x1==x4 && x2==x3)
          {
             dab = sqrt(pow(x1-x2,2)+pow(y1-y2,2)); // 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
             O = 2*dab + 2*dad;
             printf("Obim pravougaonika je %f\n",O);
          }
          else
             printf("Tacke ne cine pravougaonik sa stranicama koje su
     paralelne koordinatnim osama\n");
          break;
       case 'h':
          if ((x1-x2)!=0) // ukoliko se tacke A(x1,y1) i B(x2,y2) ne
    nalaze na pravoj koja je paralelna x osi
          {
```

```
            k = (y1-y2)/(x1-x2); //izracunamo k,n za pravu odredjenu
     tackama A(x1,y1) i B(x2,y2)
59          n = y1-k*x1;

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

Rešenje 2.16

```
/* Sa standardnog ulaza se unosi ceo cetvorocifren broj. Napisati
    program koji
ispisuje njegovu najvecu cifru na standardni izlaz. */

#include <stdio.h>

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

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

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

  /* Proveravamo da li se radi o cetvorocifrenom broju */
  if(n<1000 || n>9999){
    /* Ako broj nije cetvorocifren, prijavljujemo gresku */
    printf("Greska: Niste uneli cetvorocifren broj!\n");
  }
  else{

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

    /* Odredjujemo maksimalnu cifru */
    max=j;
    if(d>max)
      max=d;
    if(s>max)
      max=s;
    if(h>max)
      max=h;

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

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

  }

  return 0;
}
```

Rešenje 2.17

```
/* Napisati program koji za dati trocifren broj proverava da li je
Amstrongov. Broj je Amstrongov ako je jednak zbiru kubova svojih
    cifara.
*/
```

```c
#include <stdio.h>

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

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

  /* Proveravamo da li je broj trocifren */
  if(n<100 || n>999){
    /* Ako broj nije trocifren, prijavljujemo gresku */
    printf("Greska: Niste uneli trocifren broj!\n");
  }
  else{

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

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

  return 0;

}
```

Rešenje 2.18

```c
/* Za ceo broj k izmedu 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. */


#include <stdio.h>


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

```
11
   printf("Unesite k: ");
13 scanf("%d", &k);

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

23     /* Odredjujemo broj dvocifrenih brojeva koji se mogu zapisati
     pomocu
         k cifara */
25
       if(k%2!=0){
27       /* Ako je k neparan broj, zapisan je ceo broj dvocifrenih
     brojeva */
         /* 9 oduzimamo jer je 9 broj cifara potrebnih za zapis
     jednocifrenih
29        * brojeva */
         n=(k-9)/2;
31
         /* Broj o kojem se radi je */
33       broj=9+n;

35       /* Ujedno, za neparno k se trazi cifra jedinica izdvojenog
     broja */
         printf("Na %d-toj poziciji je broj %d.\n", k, broj%10);
37
       }
39     else{
         /* Ako je k paran broj, zapisan je ceo broj dvocifrenih
     brojeva i
41       zapoceto je sa zapisom sledeceg */
         /* 9 oduzimamo jer je 9 broj cifara potrebnih za zapis
     jednocifrenih
43        * brojeva */
         n=(k-9)/2 +1;
45
         /* Broj o kojem se radi je */
47       broj= 9 + n;

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

```
    }

  return 0;
}
```

Rešenje 2.19

```
/* Sa standardnog ulaza se unosi cetvorocifreni pozitivan broj.
    Napisati program
koji racuna i ispisuje proizvod parnih cifara datog broja. Ukoliko
    uneti broj
nije pozitivna cetvorocifrena vrednost ispisati poruku Greska!. */

#include <stdio.h>

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

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

  /* Proveravamo da li je unet cetvorocifreni broj */
  if(n<1000 || n>9999){
    /* Ako nije, prijavljujemo gresku */
    printf("Greska!\n");
  }
  else{

    /* Ako jeste: */

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

    /* Inicijalizujemo broj parnih cifara na 0 */
    broj_parnih=0;
    /* Postavljamo proizvod parnih cifara na 1 (neutral za mnozenje)
    */
    proizvod_parnih=1;

    /* Proveravamo da li je cifra jedinica parna */
    if(j%2==0){
      proizvod_parnih=proizvod_parnih*j;
      broj_parnih++;
    }
```

```
43      /* Proveravamo da li je cifra desetica parna */
        if(d%2==0){
45        proizvod_parnih=proizvod_parnih*d;
          broj_parnih++;
47      }

49      /* Proveravamo da li je cifra stotina parna */
        if(s%2==0){
51        proizvod_parnih=proizvod_parnih*s;
          broj_parnih++;
53      }

55      /* Proveravamo da li je cifra hiljada parna */
        if(h%2==0){
57        proizvod_parnih=proizvod_parnih*h;
          broj_parnih++;
59      }

61      /* Proveravamo da li u zapisu broja ima parnih cifara i
        ispisujemo
          rezultat */
63      if(broj_parnih==0){
          printf("Proizvod parnih cifara: 0\n");
65      }
        else{
67        printf("Proizvod parnih cifara: %d\n", proizvod_parnih);
        }
69
    }
71
    return 0;
73
}
```

Rešenje 2.20

```
1  /* Sa standarnog ulaza unosi se 5 karaktera. Proveriti da li je prvi
       karakter
   veliko ili malo slovo a. Ako jeste, ispisati karaktere obrnutim
3  redosledom, a ako nije, nista ne ispisivati. */

5  #include <stdio.h>

7  int main(){

9    char c1, c2, c3, c4, c5;

11   /* Citamo karaktere */
     /* Obratiti paznju na format ucitavanja */
13   printf("Unesite karaktere: ");
```

```
  scanf("%c %c %c %c %c", &c1, &c2, &c3, &c4, &c5);

  /* Proveravamo da li je prvi karakter malo ili veliko slova a */
  if(c1=='a' || c1=='A'){
    /* I ako jeste, ispusujemo karaktere u obrnutom redosledu */
    printf("%c %c %c %c %c\n", c5, c4, c3, c2, c1);
  }

  return 0;
}
```

Rešenje 2.21

```

/*
Napisati program koji za karakter koji ucitava jedan karakter i :
- u slucaju da je uneta cifra, ispisuje nju i njen ascii kod
- u slucaju da je uneto malo slovo, ispisuje njega, njegov ascii kod,
     odgovarajuce veliko slovo i njegov ascii kod
- u slucaju da je uneto veliko slovo, ispisuje njega, njegov ascii
    kod, odgovarajuce malo slovo i njegov ascii kod
- u ostalim slucajevima, ispisuje uneti karakter i njegov ascii kod
*/
#include <stdio.h>
int main()
{
   char c;
   printf("Unesi jedan karakter:");
   scanf("%c", &c);

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

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

   return 0;
}
```

Rešenje 2.22

```c
/* Sa standarnog ulaza unosi se jedan karakter. Ako je u pitanju malo slovo,
zameniti ga odgovaraju´im velikim slovom i ispisati na standardni izlaz. Ako je
u pitanju veliko slovo, zameniti ga odgovaraju´im 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 standarni izlaz
izmedu dveju zvezdica. */

#include <stdio.h>

int main(){

  char c;

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

  /* Proveravamo da li je karakter malo slovo */
  if(c>='a' && c<='z'){
    /* I ako jeste, ispusujemo odgovarajuce veliko slovo */
    printf("%c\n", c-'a'+'A');
  }
  else{
    /* Proveravamo da li je karakter veliko slovo */
    if(c>='A' && c<='Z'){
      /* I ako jeste, ispusujemo odgovarajuce malo slovo */
      printf("%c\n", c-'A'+'a');
    }
    else{
      /* Proveravamo da li je karakter cifra */
      if(c>='0' && c<='9'){
        /* I ako jeste, ispusujemo odgovarajucu poruku */
        printf("cifra\n");
      }
      else{
        /* Inace ispisujemo karakter izmedju dveju zvezdica */
        printf("*%c*\n",c);
      }
    }
  }

  return 0;
}
```

Rešenje 2.23

```c
/* Sa standardnog ulaza se unosi 5 karaktera. Ispisati na izlazu broj
     unetih
malih slova. */

#include <stdio.h>

int main(){

  char c1, c2, c3, c4, c5;
  int broj_malih_slova=0;


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

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

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

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

  /* Proveravamo da li je cetvrti karakter malo slovo */
  if(c4>='a' && c4<='z'){
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

  /* Proveravamo da li je peti karakter malo slovo */
  if(c5>='a' && c5<='z'){
    /* I ako jeste, uvecavamo broj malih slova */
    broj_malih_slova++;
  }

  /* Ispisujemo rezultat */
  printf("Broj malih slova: %d\n", broj_malih_slova);


  return 0;
```

```
}
```

Rešenje ??

```
#include <stdio.h>

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

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

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

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

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

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

```
    }

    if (d > max)
    {
        max = d;
        stepenMax = 1000;
    }
    if (d < min)
    {
        min = d;
        stepenMin = 1000;
    }


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

    if (broj > 0)
        rez = broj - max*stepenMax + min*stepenMax
                   - min*stepenMin + max*stepenMin;
    else
        rez = broj + max*stepenMax - min*stepenMax
                   + min*stepenMin - max*stepenMin;

    printf("%d\n",rez);
    return 0;

}
```

Rešenje 2.25

```
#include <stdio.h>

int main()
{
    int br1, br2, br3;
    scanf("%d%d%d", &br1, &br2, &br3);

    if (br1 > 999 || br1 < 100 || br2 > 999 || br2 < 100
            || br3 > 999 || br3 < 100)
    {
        printf("-1");
        return -1;
    }

    int max1 = br1;
    if (br2 > max1)
        max1 = br2;
    if (br3 > max1)
```

```
19          max1 = br3;

21      /* Ako je br1 vec najveci, onda pretragu
           za sledecim najvecim krecemo od br2 */
23      int max2 = br1 != max1 ? br1 : br2;
        if (br1 > max2 && br1 != max1)
25          max2 = br1;
        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 2.26

```
/* Napisati program koji za dva data intervala realne prave (a1, b1)
     i (a2, b2)
2 odreduje:
a) duzinu zajednickog dela ta dva intervala
4 b) najveci interval sadrzan u datim intervalima (presek), a ako on ne
      postoji
dati odgovaraju´u poruku.
6 c) duzinu realne prave koju pokrivaju ta dva intervala
d) najmanji interval koji sadrzi date intervale
8 */

10 #include <stdio.h>
#include <stdlib.h>
12
int main() {
14
  int a1, b1, a2, b2;
16  int a3, b3;
  int duzina_zajednickog_dela, zajednicka_duzina;
18  int x, y; // krajevi najmanjeg intervala koji pokriva oba intervala

20  printf("Unesite redom a1, b1, a2 i b2: ");
  scanf("%d%d%d%d", &a1, &b1, &a2, &b2);
22
  /* Presek intervala [a1, b1] i [a2, b2]
24   * racuna se kao:
   * [a3, b3] = [max{a1,a2}, min{b1, b2}] */
26
  a3 = a1 > a2 ? a1 : a2;
28  b3 = b1 < b2 ? b1 : b2;
```

```
  /* U ovom slucaju, presek je prazan */
  if(a3 >= b3) {

    duzina_zajednickog_dela = 0;
    zajednicka_duzina = abs(b1-a1) + abs(b2-a2);
  }
  else {

    duzina_zajednickog_dela = abs(b3-a3);
    zajednicka_duzina = abs(b2-a1);
  }

  /* Racunanje "pokrivaca" */
  x = a1 < a2 ? a1 : a2;
  y = b1 > b2 ? b1 : b2;

  printf("Duzina zajednickog dela: %d\n", duzina_zajednickog_dela);

  if(a3 >= b3)
    printf("Presek intervala: prazan\n");
  else
    printf("Presek intervala: [%d, %d]\n", a3, b3);

  printf("Zajednicka duzina intervala: %d\n", zajednicka_duzina);
  printf("Najmanji interval: [%d, %d]\n", x, y);

  return 0;
}
```

Rešenje 2.27

```
/* Data je funkcija f (x) = 2 * cos(x) - x*x*x . Sa standarnog ulaza
    se unosi
realan broj x i broj k koje moze biti 1, 2 ili 3. Napisati program
    koji
izracunava F (k, x) = f (f (f (...f (x))) gde je funkcija f
    primenjena k-puta.
*/

#include <stdio.h>
#include <math.h>
int main(){
  float x;
  int k;
  float F;

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

    /* Proveravamo vrednost za k */
```

```
  if(k<1 || k>3){
    printf("Greska: nedozvoljena vrednost za k!\n");
    return 0;
  }
  printf("F(%f,%d)=", x, k);

    /* Analiziramo moguce slucajeve */
  if(k==1){
    F=2*cos(x)-x*x*x;
  }
  else{
    if(k==2){
      x=2*cos(x)-x*x*x;
      F=2*cos(x)-x*x*x;
    }
    else{
        x=2*cos(x)-x*x*x;
        x=2*cos(x)-x*x*x;
        F=2*cos(x)-x*x*x;
      }

  }

  /* Ispisujemo rezultat */
  printf("%f\n", F);

  return 0;
}
```

Rešenje 2.28

```
/* Napisati program koji za uneti broj n (1  n  7) koji predstavlja
    redni broj
dana u nedelji ispisuje ime dana. U slucaju pogresnog unosa ispisati
    odgovaraju
poruku. */

#include <stdio.h>

int main(){

  int dan;

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

  switch(dan){
    case 1:
      printf("ponedeljak\n");
      break;
    case 2:
```

```
19        printf("utorak\n");
          break;
21      case 3:
          printf("sreda\n");
23        break;
        case 4:
25        printf("cetvrtak\n");
          break;
27      case 5:
          printf("petak\n");
29        break;
        case 6:
31        printf("subota\n");
          break;
33      case 7:
          printf("nedelja\n");
35        break;
        default:
37        printf("Greska: nedozvoljeni unos!\n");
    }
39
    return 0;
41  }
```

Rešenje 2.29

```
1  /* Sa standardnog ulaza se ucitavaju dva cela broja i jedan od
       karaktera +, -,
   *, / ili % koji predstavlja operaciju koju treba izvrsiti nad unetim
       brojevima.
3  Napisatiti program koji koriscenjem switch naredbe analizira o kom
       karakteru je
   rec i na standardni izlaz ispisuje rezultat. U slucaju pogresnog
       unosa ispisati
5  odgovaraju´u poruku. */

7  #include <stdio.h>

9  int main(){

11   char op;
     int x, y;
13
     printf("Unesite operator i dva cela broja: ");
15   scanf("%c %d %d", &op, &x, &y);

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

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

  return 0;
}
```

Rešenje 2.30

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

#include <stdio.h>

int main(){
    int dan, mesec, godina, dozvoljen_broj_dana;

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

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

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

    /* Ako je mesec korektan, proveravamo broj dana */
    switch(mesec){
      case 1:
```

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

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


    return 0;
}
```

Rešenje 2.31

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

#include <stdio.h>

int main(){
```

```
7       int dan, mesec, godina;
        int prethodni_dan, prethodni_mesec, prethodni_godina;
9
        /* Citamo datum */
11      printf("Unesite datum: ");
        scanf("%d.%d.%d", &dan, &mesec, &godina);
13
        /* Racunamo dan, mesec i godinu prethodnog dana */
15      prethodni_dan=dan-1;
        prethodni_mesec=mesec;
17      prethodni_godina=godina;

19      /* I po potrebi vrsimo korekcije */

21      /* Ako je u pitanju prvi u mesecu */
        if(prethodni_dan==0){
23          /* Treba korigovati mesec */
            prethodni_mesec=mesec-1;
25          /* Ako je u pitanju januar */
            if(prethodni_mesec==0){
27              /* Treba korigovati i godinu */
                prethodni_mesec=12;
29              prethodni_godina=godina-1;
            }
31
            /* Analiziramo redni broj meseca kako bi odredili tacan dan*/
33          switch(prethodni_mesec){
              case 1:
35            case 3:
              case 5:
37            case 7:
              case 8:
39            case 10:
              case 12:
41              prethodni_dan=31;
                break;
43            case 2:
                if((prethodni_godina%4==0 && prethodni_godina%100!=0) ||
45                      prethodni_godina%400==0)
                    prethodni_dan=29;
47              else
                    prethodni_dan=28;
49              break;
              case 4:
51            case 6:
              case 9:
53            case 11:
                prethodni_dan=30;
55          }
        }
57
        /* Ispisujemo datum koji smo izracunali */
```

```
59      printf("Prethodni datum: %d.%d.%d\n",
                prethodni_dan, prethodni_mesec,prethodni_godina);
61
        return 0;
63  }
```

Rešenje 2.32

Rešenje 2.33

```
1   #include <stdio.h>
    #include <ctype.h>
3
    int main()
5   {
        int br_a = 0;
7       if (tolower(getchar()) == 'a')
            br_a++;
9       if (tolower(getchar()) == 'a')
            br_a++;
11      if (tolower(getchar()) == 'a')
            br_a++;
13      if (tolower(getchar()) == 'a')
            br_a++;
15      if (tolower(getchar()) == 'a')
            br_a++;
17
        printf("%d\n", br_a);
19
        return 0;
21  }
```

Rešenje 2.107

```
1   #include <stdio.h>
    #include <ctype.h>
3
    int main()
5   {
        int br_cif = 0;
7       if (isdigit(getchar()))
            br_cif++;
9       if (isdigit(getchar()))
            br_cif++;
11      if (isdigit(getchar()))
            br_cif++;
13      if (isdigit(getchar()))
            br_cif++;
```

```
15      if (isdigit(getchar()))
            br_cif++;
17
        printf("%d\n", br_cif);
19
        return 0;
21  }
```

Rešenje 2.35

Rešenje 2.36

```
1   #include <stdio.h>
    #include <ctype.h> // !!!
3
    // Upotreba funkcija isalpha, isdigit, toupper, tolower
5
    // isalpha( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
        je karakter slovo (malo ili veliko), inace 0
7   // isdigit( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
        je karakter cifra, inace 0
    // isupper( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
        je karakter veliko slovo, inace 0
9   // islower( karakter ) - funkcija vraca vrednost razlicitu od 0 ako
        je karakter malo slovo, inace 0
    // toupper( karakter ) - ukoliko je karakter malo slovo, funkcija
        vraca odgovarajuce veliko slovo,
11  //                       inace vraca isti karakter
    // tolower( karakter ) - ukoliko je karakter veliko slovo, funkcija
        vraca odgovarajuce malo slovo,
13  //                       inace vraca isti karakter

15  int main()
    {
17      char c;
        char veliko_slovo;
19      char malo_slovo;

21      printf("Unesite karakter: ");
        scanf("%c",&c);
23
        if(isalpha(c))
25      {
            printf("Karakter %c je slovo\n",c);
27
      if(isupper(c))
29      printf("Veliko slovo\n");
      else
31      printf("Malo slovo\n");
```

```
33          veliko_slovo = toupper(c); // malo -> veliko slovo
            malo_slovo = tolower(c);   // veliko -> malo slovo
35
            printf("Veliko slovo: %c, malo slovo: %c\n", veliko_slovo,
        malo_slovo);
37
        }
39      else if(isdigit(c))
            printf("Karakter %c je cifra\n",c);
41      else
            printf("Karakter %c je znak\n",c);
43


45
        printf("============Bez koriscenja funkcija============\n");
47
        // Isti rezultat bez koriscenja ugradjenih funkcija
49
        if((c >= 'A' && c <= 'Z') || (c >= 'a' && c <= 'z'))
51      {
            printf("Karakter %c je slovo\n",c);
53
      if(c >= 'A' && c <= 'Z')
55      printf("Veliko slovo\n");
      else
57      printf("Malo slovo\n");

59          if(c >= 'a' && c <= 'z')
      {
61              veliko_slovo = c - ('a' - 'A');
          malo_slovo = c;
63    }
            else if(c >= 'A' && c <= 'Z')
65    {
                malo_slovo = c + ('a' - 'A');
67        veliko_slovo = c;
      }
69


71          printf("Veliko slovo: %c, malo slovo: %c\n", veliko_slovo,
        malo_slovo);
        }
73      else if(c >= '0' && c <= '9')
            printf("Karakter %c je cifra\n",c);
75      else
            printf("Karakter %c je znak\n",c);
77


79      return 0;
}
```

Rešenje 2.37

```c
#include <stdio.h>

// Za uneti redni broj dana u nedelji ispisati njegov naziv

int main()
{
    int broj_dana;

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

    switch(broj_dana)
    {
        case 1: printf("Dan je ponedeljak\n");
                break; // Obavezan izlazak iz case-a!
        case 2: printf("Dan je utorak\n");
                break; // Obavezan izlazak iz case-a!
        case 3: printf("Dan je sreda\n");
                break; // Obavezan izlazak iz case-a!
        case 4: printf("Dan je cetvrtak\n");
                break; // Obavezan izlazak iz case-a!
        case 5: printf("Dan je petak\n");
                break; // Obavezan izlazak iz case-a!
        case 6: printf("Dan je subota\n");
                break; // Obavezan izlazak iz case-a!
        case 7: printf("Dan je nedelja\n");
                break; // Obavezan izlazak iz case-a!
        default: printf("Lose unet broj!\n"); // Ako ni jedna provera
     ne prolazi
    }

    return 0;
}
```

Rešenje 2.38

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

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

int main()
{
    int godina;
    int mesec;

    int prestupna;
```

```
12
14      printf("Unesite godinu: ");
        scanf("%d",&godina);
16
        if(godina < 0)
18      {
            printf("Lose uneta godina!\n");
20          exit(EXIT_FAILURE);
        }
22
        if((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
        // Provera da li je godina prestupna,
24          prestupna = 1;
        // bitno za mesec februar
        else
26          prestupna = 0;

28      printf("Unesite redni broj meseca: ");
        scanf("%d",&mesec);
30
        switch(mesec)
32      {
            case 1: printf("Januar, 31 dan\n");
34                  break;
            case 2:
36                  if(prestupna)
                        printf("Februar, 29 dana\n");
38                  else
                        printf("Februar, 28 dana\n");
40                  break;

42          case 3: printf("Mart, 31 dan\n");
                    break;
44          case 4: printf("April, 30 dana\n");
                    break;
46          case 5: printf("Maj, 31 dan\n");
                    break;
48          case 6: printf("Jun, 30 dana\n");
                    break;
50          case 7: printf("Jul, 31 dan\n");
                    break;
52          case 8: printf("Avgust, 31 dan\n");
                    break;
54          case 9: printf("Septembar, 30 dana\n");
                    break;
56          case 10: printf("Oktobar, 31 dan\n");
                    break;
58          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 2.39

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

14  if(dan < 0 || godina < 0)
        printf("Lose unet datum!\n");
16
    switch(mesec) // Dodati provere za redni broj dana!
18  {
        case 1: printf("Zima\n");
20              break;

22      case 2: printf("Zima\n");
                break;
24
        case 3:
26              if(dan < 21)
                    printf("Zima\n");
28              else
                    printf("Prolece\n");
30              break;

32      case 4: printf("Prolece\n");
                break;
34
        case 5: printf("Prolece\n");
36              break;

38      case 6:
                if(dan < 21)
40                  printf("Prolece\n");
                else
42                  printf("Leto\n");
```

```
                break;

        case 7: printf("Leto\n");
                break;

        case 8: printf("Leto\n");
                break;

        case 9:
                if(dan < 23)
                    printf("Leto\n");
                else
                    printf("Jesen\n");
                break;

        case 10: printf("Jesen\n");
                 break;

        case 11: printf("Jesen\n");
                 break;

        case 12:
                if(dan < 22)
                    printf("Jesen\n");
                else
                    printf("Zima\n");
                break;

        default: printf("Lose unet redni broj meseca!\n");
    }

    return 0;
}
```

2.3 Petlje

2.3.1 Ispis podataka

Zadatak 2.40 Napisati program koji 10 puta ispisuje tekst `Mi volimo da programiramo u programskom jeziku C`.

[Rešenje A.28]

Zadatak 2.41 Napisati program koji učitava ceo broj n i ispisuje n puta tekst `Mi volimo da programiramo u programskom jeziku C` korišćenjem `while`, `for` i `do-while` petlje. Obratiti pažnju na rezultat kada je $n \leq 0$.

[Rešenje 2.41]

Zadatak 2.42 Napisati program koji učitava pozitivan ceo broj n a potom ispisuje sve cele brojeve od 0 do $n-1$.

[Rešenje 2.42]

Zadatak 2.43 Napisati program koji učitava dva cela broja n i m ispisuje sve cele brojeve iz intervala $[n, m]$ korišćenjem `while`, `for` i `do-while` petlje. Obratiti pažnju na rezultat kada je $n > m$.

[Rešenje 2.43]

Zadatak 2.44 Napisati program koji učitava dva realna broja dvostruke tačnosti, a i b, za koje važi $a < b$. Napisati program koji ispisuje vrednost funkcije $cos(x)$ u 10 ravnomerno razmaknutih tačaka intervala $[a, b]$ Pri ispisu vrednosti se zaokružuju na 4 decimale. Za neispravan unos, program ispisuje `Greska!`.

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 2.107]

Zadatak 2.45 Fibonačijev niz počinje ciframa 1 i 1, a svaki član se dobija zbirom prethdona dva. Napisati program koji učitava ceo neoznačen broj n i određuje i na standatni izlaz ispisuje n-ti član Fibonačijevog niza.

[Rešenje 2.127]

*** Zadatak 2.46** Niz prirodnih brojeva formira se prema sledećem pravilu:

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

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

[Rešenje 2.46]

*** Zadatak 2.47** 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 uneti neoznačen broj 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 2.107]

2.3.2 Obrada celih brojeva, rad sa ciframa broja

Zadatak 2.48 Napisati program koji učitava ceo broj i ispisuje njegove cifre u obrnutom poretku.

[Rešenje 2.48]

Zadatak 2.49 Napisati program koji učitava ceo pozitivan broj n i ispisuje sve (danijela:prave delioce?? sta znaci ovo prave) delioce unetog broja. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

[Rešenje 2.49]

Zadatak 2.50 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 2.50]

Zadatak 2.51 Sa standarnog ulaza unosi se ceo broj. Napisati program koji na standardni izlaz ispisuje odgovor da li je uneti prirodan broj deljiv sumom svojih cifara.

[Rešenje 2.107]

Zadatak 2.52 Napisati program koji učitava ceo neoznačen broj i uklanja sve nule sa desne strane unetog broja. 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 2.52]

Zadatak 2.53 Napisati program koji učitava neoznačeni ceo broj i transformiše ga tako što svaku parnu cifru u zapisu broja uveća za 1. Novodobijeni broj ispisati na standarni izlaz.

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 2.53]

Zadatak 2.54 Sa standardnog ulaza unosi se neoznačen ceo broj. Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre polaznog broja. Cifre se posmatraju sa desna na levo.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 2.54]

*** Zadatak 2.55** Sa standardnog ulaza unosi se neoznačen ceo broj. Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem cifara koje su jednake zbiru svojih suseda. Cifre se posmatraju sa desna na levo.

Primer 1

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

Primer 2

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

Primer 3

```
Interakcija sa programom:
 Unesite broj: 242
 22
```

[Rešenje 2.55]

*** Zadatak 2.56** Broj je palindrom ako se isto čita i sa leve i sa desne strane. Napisati program koji učitava ceo neoznačen broj i proverava da li je učitani broj palindrom.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 2.56]

2.3.3 Unos i obrada veće količine podatka (unos i obrada niza brojeva?, nije sjajno zbog nizova)

Zadatak 2.57 Napisati program koji učitava pozitivan ceo broj n, a zatim učitava n celih brojeva i na standarni izlaz ispisuje sumu pozitivnih i sumu negativnih unetih brojeva.

[Rešenje 2.57]

Zadatak 2.58 Napisati program koji učitava cele cele brojeve sve dok se ne unese nula. Nakon toga ispisati proizvod onih unetih brojeva koji su pozitivni.

[Rešenje 2.58]

Zadatak 2.59 U prodavnici se nalazi n artikala čije cene su realni brojevi. Napisati program koji učitava n, a potom i cenu svakog od n artikala i određuje i na standarni izlaz ispisuje najmanju cenu.

[Rešenje 2.59]

Zadatak 2.60 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 2.60]

Zadatak 2.61 Sa standardnog ulaza unosi se ceo pozitivan broj n, a potom i n celih brojeva. Izračunati i ispisati zbir onih brojeva koji su neparni i negativni.

Primer 1

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

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 4
 Unesite n brojeva: -1 1 0 3
 -1
```

Primer 3

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

[Rešenje 2.61]

Zadatak 2.62 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 2.62]

Zadatak 2.63 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 pri unosu podataka ispisati `Greska!`.

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 2.107]

Zadatak 2.64 Sa standarnog ulaza unosi se ceo broj n, a potom n realnih brojeva. Odrediti koliko puta je prilikom unosa došlo do promene znaka. Ispisati dobijenu vrednost na standarni izlaz.

[Rešenje 2.107]

Zadatak 2.65 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 2.65]

Zadatak 2.66 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 2.66]

Zadatak 2.67 Sa standardnog ulaza se unosi ceo pozitivan broj n, a zatim i n celih brojeva. Napisati program koji ispisuje broj sa najvećom vodećom cifrom. Vodeća cifra je prva cifra iz zapisa broja. Ukoliko ima više takvih, ispisati prvi.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 5
 Unesite n brojeva: 8 964 32 511 27
 964
```

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 3
 Unesite n brojeva: 41 669 8
 8
```

[Rešenje 2.67]

Zadatak 2.68 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 2.68]

Zadatak 2.69 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 2.69]

2.3.4 Rad sa karakterima

Zadatak 2.70 Napisati program koji učitava karaktere dok se ne unese karakter tačka i ako je karakter malo slovo, ispisuje odgovarajuće veliko, ako je karakter veliko slovo ispisuje odgovarajuće malo, a u suprotnom ispisuje isti karakter kao i uneti.

[Rešenje 2.70]

Zadatak 2.71 Napisati program koji učitava karaktere dok se ne unese `EOF` a potom ispisuje broj velikih slova, broj malih slova, broj cifara, broj belina i zbir unetih cifara.

[Rešenje 2.71]

Zadatak 2.72 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. Ne praviti razliku između malih i velikih slova. Uputstvo: *Prilikom implementacije koristiti* `switch` *naredbu.*

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 2.72]

Zadatak 2.73 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 2.73]

2.3.5 Računanje sume i proizvoda

Zadatak 2.74 Napisati program koji učitava ceo pozitivan broj i izračunava njegov faktorijel. Testirati program za različite vrednosti unetog broja. U slučaju neispravnog unosa ispisati poruku `Greska!`. Uputstvo: *Obratiti pažnju da počev od broja* 23 *dolazi do prekoračenja prilikom računanja faktorijela.*

[Rešenje 2.74]

Zadatak 2.75 Sa standradnog ulaza unose se realan broj x i ceo neoznačen broj n. Napisati program koji izračunava stepen broja, odnosno 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 2.75]

Zadatak 2.76 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 **??**]

Zadatak 2.77 Napisati program koji učitava ceo pozitivan broj n i ispisuje vrednost tražene sume. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

a) Odrediti sumu kubova brojeva od 1 do n, odnosno $s = 1 + 2^3 + 3^3 + \ldots + n^3$.

b) Modifikovati program tako da ispisuje sumu kubova, $s = 1+2^3+3^3+\ldots+k^3$, za svaku vrednost $k = 1, \ldots, n$.

[Rešenje 2.77]

Zadatak 2.78 Sa standardnog ulaza unose se realan broj x i ceo neoznačen broj n. Napisati program koji izračunava i na standarni izlaz ispisuje sumu $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \ldots + n \cdot x^n$.

Primer 1

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

Primer 2

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

[Rešenje 2.78]

Zadatak 2.79 Sa standardnog ulaza unose se realan broj x i ceo neoznačen broj n. Napisati program koji izračunava i na standarni izlaz ispisuje sumu $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \ldots \frac{1}{x^n}$.

Primer 1

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

Primer 2

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

[Rešenje 2.79]

*** Zadatak 2.80** Napisati program koji učitava realane brojeve x i *eps* i sa zadatom tačnošću *eps* izračunava i na standarni izlaz ispisuje sumu $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots$. UPUTSTVO: *Prilikom računanja sume koristiti prethodni izračunati član sume u računanju sledećeg člana sume. Naime, ako je izračunat član sume $\frac{x^n}{n!}$ na osnovu njega se lako može dobiti član $\frac{x^{n+1}}{(n+1)!}$. Nikako ne računati stepen i faktorijel odvojeno zbog mogućnosti prekoračenja.*
Izračunati sumu u odnosu na tačnost eps znači uporediti poslednji član sume sa eps i ukoliko je taj poslednji član manji od eps prekinuti dalja izračunavanja.

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 2.80]

*** Zadatak 2.81** Napisati program koji učitava realane brojeve x i *eps* i sa zadatom tačnošću *eps* izračunava i na standarni izlaz ispisuje sumu $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} \ldots$. Voditi računa o mogućem prekoračenju, pogledati uputstvo prethodnog zadatka.

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 2.81]

Zadatak 2.82 Napisati program koji učitava realan broj x i prirodan broj n izračunava sumu $S = (1+\cos(x))\cdot(1+\cos(x^2))\cdot\ldots\cdot(1+\cos(x^n))$. NAPOMENA: *Voditi računa o efikasnosti problema.*

[Rešenje 2.107]

*** Zadatak 2.83** Napisati program koji učitava ceo neoznačen broj n, a na standarni izlaz ispisuje vrednost razlomka

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

[Rešenje 2.107]

*** Zadatak 2.84** Program učitava cele brojeve x i n. Napisati program koji računa sumu

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

NAPOMENA: *Voditi računa o prekoračenju i efikasnosti rešenja.*

[Rešenje 2.107]

*** Zadatak 2.85** Sa standardnog ulaza unosi se ceo pozitivan broj n veći od 0. Napisati program koji računa proizvod

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

U slučaju greške pri unosu podataka ispisati `Greska!`. NAPOMENA: *Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.*

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 2.107]

*** Zadatak 2.86** Sa standardnog ulaza unosi se ceo pozitivan neparan broj n. Napisati program koji za uneto n izračunava:

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

U slučaju greške pri unosu podataka ispisati `Greska!`. NAPOMENA: *Voditi računa o efikasnosti rešenja.*

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 2.107]

Zadatak 2.87 Sa standardnog ulaza unose se realni brojevi x i a i ceo pozitivan broj n veći od 0. Napisati program koji izračunava:

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

U slučaju greške pri unosu podataka ispisati `Greska!`.

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 2.107]

2.3.6 Dvostruka petlja i ispisivanje slike

*** Zadatak 2.88** Sa standardnog ulaza unosi se ceo pozitivan broj n. Napisati program koji ispisuje brojeve od 1 do n, zatim od 2 do $n-1$, 3 do $n-2$, itd. Ispis se završava kada nije moguće ispisati ni jedan broj. Za neispravan unos, program ispisuje `Greska!`.

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 2.107]

Zadatak 2.89 Sa standarnog ulaza učitava se ceo neoznačen broj n. 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 2.107]

*** Zadatak 2.90** Napisati program koji učitava ceo pozitivan broj n i ispisuje sve brojeve od 1 do n, zatim svaki drugi broj od 1 do n, zatim svaki treći broj od 1 do n itd., završavajući sa svakim n-tim (tj. samo sa 1). U slučaju greške pri unosu podataka ištampati `Greska!`.

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 2.107]

Zadatak 2.91 Napisati program koji učitava ceo broj n, a potom ispisuje kvadrat stranice n sastavljen od zvezdica:

Primer 1

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

[Rešenje 2.107]

Zadatak 2.92 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 2.92]

Zadatak 2.93 Sa standardnog ulaza unosi se neoznačcen 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 2.107]

Zadatak 2.94 Napisati program koji učitava ceo broj n, a potom ispisuje obrnuti pravougli trougao čija kateta je dužine n sastavljen od zvezdica.

Primer 1

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

[Rešenje 2.107]

Zadatak 2.95 Napisati program koji učitava ceo broj n, a potom ispisuje pravougli trougao čija kateta je dužine n sastavljen od zvezdica.

Primer 1

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

[Rešenje 2.107]

Zadatak 2.96 Napisati program koji učitava ceo broj n, a potom ispisuje trougao koji se dobija spajanjem dva pravougla trougla čija kateta je dužine n.

Primer 1

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

[Rešenje 2.107]

Zadatak 2.97 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 2.97]

Zadatak 2.98 Napisati program koji učitava ceo broj n, a potom ispisuje pravougli trougao čija kateta je dužine n sastavljen od zvezdica. Trougao je desno poravnat.

Primer 1

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

[Rešenje 2.107]

Zadatak 2.99 Napisati program koji učitava ceo broj n, a potom ispisuje trougao koji se dobija spajanjem dva pravougla trougla čija kateta je dužine n. Trougao je desno poravnat.

Primer 1

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

[Rešenje 2.107]

Zadatak 2.100 Napisati program koji učitava ceo broj n, a potom ispisuje rub jednakostraničnog trougla čija stranica je dužine n.

Primer 1

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

[Rešenje 2.107]

Zadatak 2.101 Napisati program koji učitava ceo broj n, i ispisuje sliku koja se dobija spajanjem dva jednakostranična trougla čija stranica je dužine n.

Primer 1

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

[Rešenje 2.107]

*** Zadatak 2.102** 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 2.102]

*** Zadatak 2.103** 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 2.103]

*** Zadatak 2.104** Napisati program koji učitava ceo broj n, i ispisuje sliku koja se dobija na sledeći način: u prvom redu je jedna zvezdica, u drugom redu su dve zvezdice razdvojene razmakom, treći red je sastavljen od zvezdica i iste je dužine kao i drugi red, četvrti red se sastoji od tri zvezdice razdvojene razmakom, a peti red je sastavljen od zvezdica i iste je dužine kao i četvrti red itd. Ukupna visina slike je n.

Primer 1

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

[Rešenje 2.107]

**** Zadatak 2.105** Sa standarnog 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 prazninom.

Primer 1

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

Primer 2

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

[Rešenje 2.107]

*** Zadatak 2.106** Sa standarnog ulaza unosi se ceo neoznačen broj n. Napisati program koji štampa romb sastavljen od minusa u pravougaoniku sastavljenim od zvezdica. Pogledati sliku u test primerima.

Primer 1

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

Primer 2

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

[Rešenje 2.107]

Zadatak 2.107 Napisati program koji učitava ceo broj n ($n \geq 2$). Napisati program koji na standardni izlaz ispisuje sliku kuće sa krovom (kuća je kocka stranice n, a krov jednakostranični trougao stranice n):

Primer 1

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

[Rešenje 2.107]

2.4 Rešenja

Rešenje A.28

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

```c
#include<stdio.h>

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

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

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

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

    ili

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

    ili

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

       Brojanje pocev od 0 uz koriscenje stroge nejednakosti
       je u duhu programskog jezika C i zato cemo ovaj nacin
       brojanja najcesce koristiti
   */
   return 0;
}
```

Rešenje 2.41

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

#include <stdio.h>

int main()
{

   int n,m;
   int i;

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

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

   i=0;
   while (i<n)          /* uslov petlje se proverava pre ulaska u
    telo petlje */
   {
      printf("We love C programming\n");
      i++;
   }

   printf("\n");

   /* 2. nacin - for petlja */
   printf("for: ");
                        /* naredba i=0 se izvrsava jednom, pre prve
     iteracije */
   for(i=0;i<n;i++)      /* uslov petlje i<=m se proverava pre svake
    iteracije */
      printf("We love C programming\n");   /* naredba i++ se izvrsava
     nakon svake iteracije */

   printf("\n");

   /* 3. nacin - do while petlja */
   printf("do while: ");  /* uslov petlje se proverava na kraju svake
     iteracije */
                        /* zbog toga se do while petlja izvrsava
    bar jednom, cak i u slucaju */
                        /* da uslov petlje nikada nije ispunjen */
   i=0;
   do                                   /* petlja se zapocinje bez
```

```
     provere uslova */
44   {
        printf("We love C programming\n");  /* stampa se dati tekst */
46      i++;                                /* uvecava se vrednost
     promenljive i */
     }
48   while(i<n);                           /* proverava se uslov i
     ukoliko je ispunjen, nastavlja se sa sledecom iteracijom */
                                           /* u suprotnom, petlja se
     zavrsava i program se nastavlja od prve naredbe koja sledi za
     petljom */
50   printf("\n");

52   return 0;

54 }
```

Rešenje 2.42

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

6 #include<stdio.h>

8 int main()
  {
10    int x;
      int n;
12
      printf("Unesi pozitivan ceo broj:");
14    scanf("%d", &n);
      x=0;
16    while (x<n)
      {
18        printf("%d\n", x);
          x++;
20    }
      return 0;
22 }
```

Rešenje 2.43

```
/*
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
```

```
   na rezultat kada je n>m.
*/

#include <stdio.h>

int main()
{

   int n,m;
   int i;

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

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

   i=n;
   while (i<=m)          /* uslov petlje se proverava pre ulaska u
    telo petlje */
   {
      printf("%d ", i);
      i++;
   }

   printf("\n");

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

   printf("\n");

   /* 3. nacin - do while petlja */
   printf("do while: ");  /* uslov petlje se proverava na kraju svake
     iteracije */
                          /* zbog toga se do while petlja izvrsava
    bar jednom, cak i u slucaju */
                          /* da uslov petlje nikada nije ispunjen */
   i=n;
   do                    /* petlja se zapocinje bez provere uslova */
   {
      printf("%d ",i);   /* stampa se vrednost promenljive i */
      i++;               /* uvecava se vrednost promenljive i */
   }
   while(i<=m);          /* proverava se uslov i ukoliko je ispunjen,
    nastavlja se sa sledecom iteracijom */
```

```
                        /* u suprotnom, petlja se zavrsava i program
    se nastavlja od prve naredbe koja sledi za petljom */
50    printf("\n");
    return 0;
52 }
```

Rešenje 2.107

Rešenje 2.127

Rešenje 2.46

```
1 /* Niz prirodnih brojeva formira se na sledeci nacin:
  an+1 = an/2 ako je an parno
3 an+1 = (3*an+1)/2 ako je an neparno
  Napisati program koji za uneti pocetni clan niza a0 stampa niz
      brojeva sve do prvog clana jednakog
5 1.
  */
7 #include<stdio.h>
  int main()
9 {
    int a0;
11  int an,an1;

13  printf("Unesi pocetni clan niza brojeva:");
    scanf("%d",&a0);
15
    if (a0>0)
17  {
      printf("%d\n", a0);
19
      an=a0;
21    while(an!=1)
      {
23      if (an%2) /* ukoliko je vrednost izraza an%2 razlicita od nule,
      */
          {         /* izraz se tumaci kao tacan i izvrsavaju se naredbe
      iz if grane */
25        an1=(3*an+1)/2;
        }
27      else /* u suprotnom, ukoliko je vrednost izraza an%2 jednaka
      nuli, izraz */
        {    /* se tumaci kao netacan i izvrsavaju se naredbe iz else
      grane */
29        an1=an/2;
        }
31      printf("%d\n",an1);
        an=an1;
```

```
33      }
    }
35  else
        printf("Nekorektan unos\n");
37
    return 0;
39  }
```

Rešenje 2.107

Rešenje 2.48

```
1   /*
        Napisati program koji za uneti ceo broj ispisuje njegove cifre
3       u obrnutom poretku.
    */
5
    #include<stdio.h>
7   #include<stdlib.h>
    int main()
9   {
        int x;
11      char cifra;
        printf("Unesi ceo broj:");
13      scanf("%d", &x);

15      x = abs(x); /* pretvaranje u apsolutnu vrednost se vrsi za slucaj
         kada je unet
                        negativan broj kako bismo osigurali da ce nam
         izdvojene cifre
17                biti pozitivne
                       */
19
        while(x>0)
21      {
            cifra=x%10;             /* izdvajamo poslednju cifru broja x */
23          printf("%d\n", cifra);
            x/=10;                  /* ako je npr x=1582, x%10 ce biti 2,
25                                                    a x/10 ce biti 158;
                                                npr x=5, x%10 ce biti 5
27                                                 a x/10 ce biti 0 */
        }
29
        return 0;
31  }
```

Rešenje 2.49

```c
/*
Napisati program koji ispisuje sve prave delioce unetog pozitivnog
    celog broja.

*/

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

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

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

   /* 1. nacin */
   printf("------------ 1. nacin ------------\n");
   for(i=2;i<x;i++)
   {
      printf("proveravam za %d...\n",i);
      if (x%i==0)
         printf("\t delilac:%d \n",i);
   }
   /* 2. nacin (brzi) */
   printf("------------ 2. nacin ------------\n");
   for(i=2;i<=sqrt(x);i++)
   {
      printf("proveravam za %d...\n",i);
      if (x%i==0)
        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
    */
         printf("\t delioci:%d %d \n",i,x/i);
   }
   return 0;
}
```

Rešenje 2.50

```c
/* Sa standardnog ulaza unosi se ceo neoznacen broj. Napisati program
     koji
```

```c
proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu ili ne
    . */

#include <stdio.h>

int main(){
    int n, cifra;
    int indikator=0;

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

    /* Sve dok imamo cifara u zapisu broja */
    while(n>0){

      /* Izdvajamo posledjnju cifru broja */
      cifra=n%10;

      /* Proveravamo da li je bas ona jednaka broju 5 */
      if(cifra==5){
        /* Ako jeste postavljamo indikator na vrednost 1 tako da
    znamo da smo
         * pronasli peticu i prekidamo sa izvrsavanjem petlje */
        indikator=1;
        break;
      }
      /* Ako izvdvojena cifra nije jednaka broju 5, broj delimo sa 10
     kako bi
         mogli da izdvojimo i preostale cifre broja na isti nacin */
      n=n/10;
    }

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

    return 0;
}
```

Rešenje 2.107

Rešenje 2.52

```c
/* Napisati program koji unetom broju uklanja nule sa desne strane.
    Novodobijeni
```

```
broj ispisati na standardni izlaz. */

#include <stdio.h>

int main(){
    int n;

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

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

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


    return 0;
}
```

Rešenje 2.53

```
#include <stdio.h>

int main() {

  unsigned int x;
  int pozicija;     // da li se radi o cifri jedinici, desetici,
    stotini itd...
  int cifra;        // trenutna izdvojena cifra iz broja x
  unsigned int y;   // broj dobijen nakon transformacije

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


  if(x > 0) {

    /* Posto pocinjemo sa izdvajanjem cifara od cifre jedinica,
      postavljamo tezinu (stepen) pozicije na 1 */
    pozicija = 1;
    y = 0;
```

```
21      /* Sve dok imamo cifara u zapisu broja */
        while(x > 0) {
23
          /* Izdvajamo poslednju cifru iz zapisa */
25        cifra = x % 10;

27        /* Proveravamo da li je cifra parna */
          if(cifra % 2 == 0){
29          /* I ako jeste, uvecavamo je */
            cifra++;
31
          }
33
          /* Novi broj formiramo tako sto izdvojenu cifru pomnozimo
        odgovarajucom
35            tezinom (stepenom) pozicije */
          y += cifra*pozicija;
37
          /* Pripremamo broj za izdvajanje naredne cifre */
39        x /= 10;

41        /* I uvecavamo tezinu (stepen) pozicije */
          pozicija *= 10;
43      }

45      /* Ispisujemo izracunatu vrednost */
        printf("%d\n", y);
47    }
      else
49      printf("Nekorektan unos.\n");

51    return 0;
  }
```

Rešenje 2.54

```
1   /* Sa standardnog ulaza unosi se neoznacen ceo broj. Napisati program
        koji
    formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre
        polaznog
3   broja. Cifre se posmatraju sa desna na levo.
    */
5
    #include <stdio.h>
7   #include <math.h>

9   int main() {

11    unsigned int x;
```

```
int stepen_deset; // da li se radi o cifri jedinici, desetici,
  stotini itd...
int cifra;        // trenutna izdvojena cifra iz broja x
int rbr; // redni broj cifre koju trenutno obradjujemo, gledano s
  desna na  levo
unsigned int y;   // broj dobijen nakon transformacije

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

if(x > 0) {
  /* Postavljamo vrednost stepena na 0 - to znaci da cemo prvo
  mnoziti sa
   * 10^0=1 */
  stepen_deset = 0;

  /* Postavljamo vrednost broja koji se formira na 0 */
  y = 0;
  /* Postavljamo redni broj pozicije na 0 */
  rbr = 0;

  /* Sve dok imamo cifara u zapisu broja */
  while(x > 0) {

    /* Izdvajamo cifru */
    cifra = x%10;

    /* Proveravamo da li je pozicija izdvojene cifre parna -
     * cifre na parnim pozicijama zadrzavamo
     */
    if(rbr % 2 == 0) {
      /* I ako jeste */

      /* Dodajemo izdvojenu cifru novom broju */
      /* Neophodno je izvrsiti "kastovanje" tipova, jer je double
  povratni tip
       * funkcije pow */
      y += cifra * ((int) pow(10, stepen_deset));

      /* Uvecavamo stepen zbog naredne cifre */
      stepen_deset++;
    }

    /* Azuriramo redni broj cifre */
    rbr++;
    /* I pripremamo broj za naredno izdvajanje */
    x /= 10;
  }

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

```
  }
  else
    printf("Nekorektan unos.\n");

  return 0;
}
```

Rešenje 2.55

```
/* Sa standardnog ulaza unosi se neoznacen 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. */

#include <stdio.h>

int main(){
  unsigned n, novo_n;
  int stepen;
  int cifra_levo, cifra_sredina, cifra_desno;

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


  /* Stepen broja 10 sa kojim cemo mnoziti cifre izdvojenog broja */
  stepen=1;

  /* Nova vrednost broja */
  novo_n=0;

  /* Sve dok u zapisu broja imamo barem tri cifre  */
  while(n>99){
    /* Izdvajamo srednju cifru, cifru desno od nje i cifru levo od
    nje:
    npr. za trojku 583 8 je srednja cifra, 3 je cifra desno, a 5
    cifra levo */
    cifra_desno=n%10;
    cifra_sredina=(n/10)%10;
    cifra_levo=(n/100)%10;

    /* U novi broj smestamo desnu cifru */
    novo_n+=cifra_desno*stepen;

    /* Azuriramo vrednost stepena */
    stepen=stepen*10;

    /* Ako je srednja cifra jednaka zbiru leve i desne cifre */
    if(cifra_levo+cifra_desno==cifra_sredina){
```

```
39        /* Treba izbaciti srednju cifru, pa broj n azuriramo tako sto
      ga podelimo sa 100 */
          n=n/100;
41      }
      else{
43
          /* Inace, zadrzavamo srednju cifru i odbacujemo samo poslednju
      */
45        n=n/10;
      }
47    }

49    /* Na novi broj dodajemo preostali dvocifreni ili jednocifreni broj
         */
      novo_n=n*stepen+novo_n;
51
      /* I ispisujemo rezultat */
53    printf("%d\n", novo_n);


55
      return 0;
57
}
```

Rešenje 2.56

```
1  /* Napisati program koji proverava da li je dati prirodan broj
       palindrom. Broj
   je palindrom ako se isto cita i sa leve i sa desne strane. */
3
   #include <stdio.h>
5  #include <math.h>

7  int main() {

9    int x;
     int broj_cifara;
11   int min_stepen, max_stepen;
     int pom;
13   int leva_cifra, desna_cifra;
     int indikator;
15
     printf("Unesite broj: ");
17   scanf("%d", &x);

19   /* Ako je korisnik uneo negativan broj, analiziramo njegovu
       apsolutnu
      * vrednost
21    */
     if(x < 0)
23     x=-x;
```

```
25
    /* Odredjujemo broj cifara u zapisu broja x
27    kako bismo mogli da izdvajamo istovremeno cifre i sa leve i sa
    desne
      strane
29  */
    broj_cifara = 0;
31  pom = x;
    while(pom > 0) {
33    pom /= 10;
      broj_cifara++;
35  }

37  /* Odredjujemo stepen koji stoji uz krajnju levu cifru broja */
    max_stepen = (int) pow (10, broj_cifara-1);
39
    /* Indikator je promenljiva koja ce nam ukazivati da li je broj
41   * palindrom ili ne
     */
43  indikator=1;
    while(x!=0 && indikator==1){
45      /* Izdvajamo levu cifru */
        leva_cifra=x/max_stepen;
47      /* Izdvajamo desnu cifru */
        desna_cifra=x%10;
49      /* Ako su cifre razlicite, odmah mozemo da zakljucimo da
         * broj nije palindrom i da prekinemo izvrsavanje petlje */
51      if(leva_cifra!=desna_cifra){
          indikator=0;
53        break;
        }
55      /* Formiramo novu vrednost broja x tako sto odbacujemo
         * krajnju levu i krajnju desnu cifru */
57      x=(x%max_stepen-x%10)/10;
        /* I korigujemo maksimalan stepen tako dobijenog broja -
59       * delimo sa 100 jer smo odbacili 2 cifre */
        max_stepen=max_stepen/100;
61  }

63  /* Ispisujemo rezultat */
    if(indikator==1)
65    printf("Broj je palindrom!\n");
    else
67    printf("Broj nije palindrom!\n");

69
  return 0;
71}
```

Rešenje 2.57

```c
/*
   Napisati program koji poziva korisnika da unese pozitivan ceo broj
     n,
   a zatim za unetih n celih brojeva ispisuje sumu pozitivnih i sumu
   negativnih brojeva.

*/

#include<stdio.h>

int main()
{
   int n;
   int x;
   int suma_poz;
   int suma_neg;
   int i;

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

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

   while(i<n)
   {
       printf("Unesi ceo broj:");
       scanf("%d", &x);

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

       i++;
   }

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

Rešenje 2.58

```c
/*
  Napisati program koji omogucava korisniku da unosi cele brojeve dok
  ne unese nulu. Nakon toga ispisati proizvod onih unetih brojeva
    koji
```

```
  su pozitivni.
*/

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

  p=1;
  while (1) /* izraz 1 je konstantan; razlicit je od nule sto znaci
    da ga tumacimo kao tacnog */
  {
     printf("Unesi jedan ceo broj:");
     scanf("%d", &x);
     if (x==0)  /* ukoliko je uneta nula */
        break;  /* break prekidamo petlju; izvrsavanje se nastavlja
    od prve naredbe nakon petlje  */

     if (x<0)     /* ukoliko je unet negativan broj, tu vrednost ne
    zelimo da pomnozimo sa ukupnim proizvodom p; zato moramo
    nastaviti dalje */
        continue;  /* sa izvrsavanjem petlje; continue prekida
    trenutnu iteraciju petlje tako sto preskace sve naredbe
                    koje nakon njega slede; izvrsavanje se
    nastavlja od provere uslova petlje  */
     p=p*x;
  }

  printf("Proizvod unetih brojeva je %d\n",p);

  return 0;
}
```

Rešenje 2.59

```
/*
Program izracunava minimum n unetih brojeva.
Npr. za n=4 i brojeve 3 8 2 9 program ispisuje 2
*/
#include <stdio.h>
int main()
{
    int n, i;
    float x, min;


    printf("Unesi n>0:");
    scanf("%d", &n);
    if (n<=0)                        /* ako je unos neispravan */
    {
```

```
16          printf("Neispravan unos\n");
            return -1;                    /* prekidamo izvrsavanje
        programa pomocu naredbe return */
        }                                 /* u slucaju greske kao sto je
        neispravan unos vracamo 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:");
28        scanf("%f", &x);
          if(x<min)
30           min=x;
          i++;
32      }
        printf("Minimum je: %f\n", min);
34      return 0;
}
```

Rešenje 2.60

```
1  /* Sa standardnog ulaza se unose realni brojevi sve do unosa broja 0.
       Napisati program koji izracunava i ispisuje
   aritmeticku sredinu unetih brojeva. */
3
   #include <stdio.h>
5  #include <math.h>

7  int main(){

9    float x;
     int broj_brojeva;
11   float suma;

13   /* Inicijalizujemo vrednosti */
     broj_brojeva=0;
15   suma=0;


17
     printf("Unesite brojeve: ");
19
     /* U petlji */
21   while(1){
       /* Ucitavamo broj sa ulaza */
```

```
23      scanf("%f", &x);

25      /* Ako je korisnik uneo 0, prekidamo sa petljom */
        if(x==0)
27        break;

29      /* Inace .. */

31      /* Procitani broj dodajemo na sumu */
        suma+=x;
33      /* I uvecavamo broj procitanih brojeva */
        broj_brojeva++;
35    }

37    /* Ispisujemo trazeni rezultat */
      printf("Aritmeticka sredina: %.4f\n", suma/broj_brojeva);
39
      return 0;
41  }
```

Rešenje 2.61

```
/* Sa standardnog ulaza unosi se ceo pozitivan broj n, a potom i n
    celih
2 brojeva. Izracunati i ispisati zbir onih brojeva koji su neparni i
    negativni. */

4 #include <stdio.h>

6 int main(){
    int n, i, x;
8   int zbir=0;

10  printf("Unesite broj n: ");
    scanf("%d", &n);
12
      printf("Unesite n brojeva: ");
14
      /* Inicijalizujemo brojac kojim kontrolisemo broj ucitavanja -
16     * treba da ih bude tacno n
       */
18  i=0;
    while(i<n){
20        /* Ucitavamo broj */
      scanf("%d", &x);
22
      /* Proveravamo da li broj negativan i neparan */
24    if(x<0 && x%2!=0){
              /* Ako jeste, dodajemo ga na zbir */
26      zbir=zbir+x;
      }
```

```
28
        /* Uvecavamo brojac iteracija */
30      i++;
    }
32
    /* Ispisujemo rezultat */
34  printf("%d\n", zbir);

36  return 0;
}
```

Rešenje 2.62

```
1  /* Sa standardnog ulaza unosi se realan broj m, ceo pozitivan broj n
       i n realnih
   brojeva. Izracunati i ispisati koliko je brojeva medju unetima manje
       od zadatog
3  broja m. */

5  #include <stdio.h>

7  int main(){

9      float m, x;
       int n, i;
11     int broj_brojeva=0;

13     printf("Unesite broj m: ");
       scanf("%f", &m);
15
       printf("Unesite broj n: ");
17     scanf("%d", &n);

19     printf("Unesite n brojeva: ");
       /* Inicijalizujemo brojac kojim kontrolisemo broj ucitavanja -
21      * treba da ih bude tacno n
        */
23     i=0;
       while(i<n){
25         /* Ucitavamo broj */
           scanf("%f", &x);
27
           /* Proveravamo da li je broj manji od zadatog broja m */
29         if(x<m){
               /* Ako jeste, uvecavamo brojac brojeva za 1 */
31             broj_brojeva++;
           }
33
           /* Uvecavamo brojac iteracija */
35         i++;
       }
```

```
37
    /* Ispisujemo rezultat */
39  printf("%d\n", broj_brojeva);

41  return 0;
}
```

Rešenje 2.107

Rešenje 2.107

Rešenje 2.65

```
/* Sa standardnog ulaza se unosi ceo pozitivan broj n, a zatim i n
    celih brojeva. Napisati program koji ispisuje
2 broj sa najvecom cifrom desetica. Ukoliko ima vise takvih, ispisati
    prvi. */

4 #include <stdio.h>
  #include <math.h>
6
  int main(){
8
    int n;
10  int x, x_desetica;
    int max_desetica, broj;
12  int i;

14  /* Citamo vrednost sa ulaza */
    printf("Unesite broj n: ");
16  scanf("%d", &n);

18  /* Postavljamo maksimalnu cifru desetice na 0 - 0 je svakako
      najmanja cifra pa je pocetna vrednost neutralna  tj.
    ne moze da utice na maksimum koji izracunavamo. Nije uvek zgodno
      pretpostaviti da je maksimalna vrednost 0. Na primer,
20  ako trazimo maksimum celih brojeva, a korisnik unese -32 -7 i -22,
      maksimalni je broj -7 */
    max_desetica=0;
22
    /* Ucitavamo broj po broj */
24  printf("Unesite n brojeva: ");
    for(i=0; i<n; i++){
26    scanf("%d", &x);

28    /* Izdvajamo cifru desetica procitanog broja */
      x_desetica=(abs(x)/10)%10;
30
      /* Ako je ona veca od maksimalne cifre desetica */
```

```
32      if(x_desetica>max_desetica){
          /* Cuvamo je */
34        max_desetica=x_desetica;
          /* Ali zbog ispisa, cuvamo i broj u kojem se ona pojavljuje */
36        broj=x;
        }
38    }

40    /* Ispisujemo rezultat */
      printf("%d\n", broj);
42
      return 0;
44
}
```

Rešenje 2.66

```
/* Sa standardnog ulaza se unosi ceo pozitivan broj n, a zatim i n
     celih brojeva. Napisati program koji ispisuje
2 broj sa najvecim brojem cifara. Ukoliko ima vise takvih, ispisati
     prvi. */

4 #include <stdio.h>
  #include <math.h>
6
  int main(){
8
    int n;
10  int x, x_kopija, br_cifara;
    int max_br_cifara, broj;
12  int i;

14  /* Citamo vrednost sa ulaza */
    printf("Unesite broj n: ");
16  scanf("%d", &n);

18  /* Postavljamo maksimalan broj cifara na 0  - svaki broj ima vise
      od 0 cifara pa je ova vrednost neutralna */
    max_br_cifara=0;
20
    /* Ucitavamo broj po broj */
22  printf("Unesite n brojeva: ");
    for(i=0; i<n; i++){
24    scanf("%d", &x);

26    /* Odredjujemo broj cifara unetog broja x */
      x_kopija=abs(x);
28    br_cifara=0;
      while(x_kopija!=0){
30      x_kopija=x_kopija/10;
        br_cifara++;
```

```
32      }
        /* Ako je broj cifara unetog broja veci od maksimalnog */
34      if(br_cifara>max_br_cifara){
          /* Cuvamo ga */
36        max_br_cifara=br_cifara;
          /* I zbog ispisa rezultata, cuvamo i originalni broj */
38        /* Zbog ovoga smo morali i da racunamo broj cifara nad kopijom
        broja x kako ne bismo promenili njegovu vrednost */
          broj=x;
40      }
    }
42
    /* Ispisujemo rezultat */
44  printf("%d\n", broj);

46  return 0;

48 }
```

Rešenje 2.67

```
/* Sa standardnog ulaza se unosi ceo pozitivan broj n, a zatim i n
     celih brojeva. Napisati program koji ispisuje
2 broj sa najvecom vodecom cifrom. Vodeca cifra je prva cifra iz zapisa
      broja. Ukoliko ima vise takvih, ispisati
  prvi. */
4
  #include <stdio.h>
6 #include <math.h>

8 int main(){

10  int n;
    int x, x_kopija;
12  int broj;
    int vodeca_cifra, max_vodeca_cifra;
14  int i;

16  /* Citamo vrednost sa ulaza */
    printf("Unesite broj n: ");
18  scanf("%d", &n);

20  /* Postavljamo maksimalnu vodecu cifru na 0 - cifre broja su vece
      ili jednake od 0 pa je ova vrednost neutralna */
    max_vodeca_cifra=0;
22
    /* Ucitavamo broj po broj */
24  printf("Unesite n brojeva: ");
    for(i=0; i<n; i++){
26    scanf("%d", &x);
```

```
28      /* Odredjujemo vodecu cifru broja */
        x_kopija=abs(x);
30      while(x_kopija>10){
          x_kopija=x_kopija/10;
32      }
        vodeca_cifra=x_kopija;
34
        /* Ako je izdvojena cifra veca od maksimalne vodece cifre */
36      if(vodeca_cifra>max_vodeca_cifra){
          /* Cuvamo je */
38        max_vodeca_cifra=vodeca_cifra;
          /* I zbog ispisa, cuvamo i broj u kojem se ona pojavljuje */
40        /* Zbog ovoga smo morali i da racunamo vodecu cifru nad kopijom
         broja x kako ne bismo promenili njegovu vrednost */
          broj=x;
42      }
    }
44
    /* Ispisujemo rezultat */
46  printf("%d\n", broj);

48  return 0;

50 }
```

Rešenje 2.68

```
/* Sa standardnog ulaza se unose celi pozitivni brojevi n (n > 1) i d
     , a zatim i n celih brojeva. Napisati program
2 koji izracunava koliko ima parova uzastopnih brojeva medju unetim
     brojevima koji se nalaze na rastojanju d.
Rastojanje izmedu brojeva je definisano sa d(x, y) = |y - x|. Rezultat
      ispisati na standardni izlaz. */
4
#include <stdio.h>
6 #include <math.h>

8 int main(){

10   int n;
     int d;
12   int x, y;
     int broj_parova;
14   int i;

16   /* Ucitavamo vrednosti sa ulaza */
     printf("Unesite brojeve n i d: ");
18   scanf("%d %d", &n, &d);

20   /* Inicijalizujemo broj parova */
     broj_parova=0;
```

```
22
   printf("Unesite n brojeva: ");
24
   /* Ucitavamo prvi broj */
26 scanf("%d", &x);

28 for(i=1; i<n; i++){
     /* Ucitavamo naredni broj */
30   scanf("%d", &y);

32   /* Ako su brojevi na rastojanju d */
     if(abs(y-x)==d)
34     /* Treba uvecati broj parova */
       broj_parova++;
36
     /* Cuvamo broj iz tekuce iteracije kako bismo mogli da ga
     upotrebimo u narednoj iteraciji */
38   x=y;
   }
40
   /* Ispisujemo rezultat */
42 printf("Broj parova: %d\n", broj_parova);

44 return 0;

46 }
```

Rešenje 2.69

```
1 /* Sa standardnog ulaza se unose celi brojevi sve do unosa broja 0.
     Napisati program koji izracunava i ispisuje
  razliku najveceg i najmanjeg unetog broja. */
3
  #include <stdio.h>
5 #include <math.h>

7 int main(){

9   int x;
    int min, max;
11
    printf("Unesite brojeve: ");
13
    /* Prvi broj ucitavamo izvan petlje zbog inicijalizacije maksimuma
       i minimuma */
15  scanf("%d", &x);
    max=x;
17  min=x;

19  /* U petlji smo sve dok ne procitamo broj 0 */
    while(x!=0){
```

```
21
    /* Proveravamo da li je procitani broj veci od aktuelnog
    maksimuma */
23  if(x>max)
      max=x;
25  /* Proveravamo da li je procitani broj manji od aktuelnog
    minimuma */
    if(x<min)
27    min=x;

29  /* Ucitavamo naredni broj */
    scanf("%d", &x);
31 }

33 /* Ispisujemo razliku najveceg i najmanjeg broja */
   printf("Razlika: %d\n", max-min);
35
   return 0;
37 }
```

Rešenje 2.70

```
1  /*
   Napisati program koji omogucava korisniku da unosi karaktere dok ne
       zada tacku i ukoliko je karakter malo slovo,
3  ispisuje odgovarajuce veliko, ukoliko je karakter veliko slovo
       ispisuje odgovarajuce malo, a u suprotnom ispisuje
   isti karakter kao i uneti.
5  */

7  #include <stdio.h>

9  int main()
   {
11   int c;

13    /* funkcija getchar ucitava jedan karakter.
        naredbom dodele (c=getchar()) promenljivoj c bice dodeljena
     vrednost
15     ascii koda unetog karaktera
       obratiti paznju na zagrade!
17   */
     while((c=getchar())!='.')
19   {
       if (c>='A' && c<='Z')
21       putchar(c+'a'-'A'); /* Razlika izmedju ascii koda svakog malog
     i odgovarajuceg velikog slova
                                je konstanta koja se moze sracunati
     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;
}
```

Rešenje 2.71

```
/*
2     Napisati program koji omogucava 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
#include <stdio.h>
8
int main()
10 {
    /* promenljivoj c dodelicemo povratnu vrednost funkcije getchar()
12     funkcija getchar() ucitava jedan karakter sa standardnog ulaza
       i vraca njegov ascii kod; povratna vrednost funkcije getchar je
14     int, pa i promenljiva c mora biti tipa int
    */
16  int c;

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

26  while((c=getchar())!=EOF)              /* petlja se zavrsava kada
      korisnik ne unese karakter, vec zada konstantu EOF */
    {                                      /* ova konstanta se zadaje
      kombinacijom tastera CTRL+D. U tom slucaju, getchar() vraca -1*/
28    if (c>='A' && c<='Z')
        br_v++; /* <=> br_v = br_v+1; */
30    else if (c>='a' && c<='z')
        br_m++;
32    else if (c>='0' && c<='9')
      {
34      br_c++;
        suma=suma+c-'0';             /* funkcija getchar() vraca ascii
      kod unetog karaktera; ascii kodovi cifara 0,1,...,9
36                                     su redom 48,49,...,57; Na primer,
      za unetu 1
```

```
                                    promenljiva c ce imati vrednost
    49. Zbog toga bi bilo pogresno racunati
38                  zbir kao zbir=zbir+c. Promenljivu zbir zato
    racunamo kao zbir=zbir+(c-'0')
                    jer c-'0' ce za unetu 0 proizvesti 48-'0' sto je
    0,
40                  za unetu 1 49-'0' sto je 1, za unetu 2 50-'0' sto
    je 2, ...*/
    }
42    else if (c=='\t' || c=='\n' || c==' ')
        br_b++;
44
      br_k++;
46  }

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

Rešenje 2.72

Rešenje 2.73

```
1 /* Sa standardnog ulaza se unosi ceo broj n, a zatim i n karaktera.
      Napisati program koji proverava da li se od
  unetih karaktera moze napisati rec Zima. */
3
  #include <stdio.h>
5 #include <math.h>

7 int main(){

9   int n;
    int broj_Z, broj_i, broj_m, broj_a;
11  char novi_red, c;
    int i;
13
    broj_Z=0;
15  broj_i=0;
    broj_m=0;
17  broj_a=0;


19
    /* Ucitavamo broj karaktera */
21  printf("Unesite broj: ");
    scanf("%d", &n);
23
```

```
  /* Ucitavamo karakter po karakter */
  for(i=0; i<n; i++){
    printf("Unestite %d. karakter: ", i+1);
    /*
      Prvo citamo znak za novi red koji je ostao neprocitan nakon
    pritiska Enter tastera
      posle prethodnog unosa, pa tek onda citamo karakter koji treba
    obradjivati
    */
    scanf("%c%c", &novi_red, &c);

    /* Analiziramo karakter */
    switch(c){
      case 'Z':
        broj_Z++;
        break;
      case 'i':
        broj_i++;
        break;
      case 'm':
        broj_m++;
        break;
      case 'a':
        broj_a++;
        break;
    }
  }

  /* Ako imamo barem jedno veliko slovo z i barem po jedno malo slovo
     i, m i a */
  if(broj_Z && broj_i && broj_m && broj_a){
    /* Zakljucujemo da se rec moze napisati */
    printf("Moze se napisati rec Zima.\n");
  }
  else{
    /* Inace, obavestavamo korisnika da je to nemoguce */
    printf("Ne moze se napisati rec Zima.\n");
  }

  return 0;
}
```

Rešenje 2.74

```
/*
   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.
*/
```

```c
#include<stdio.h>

int main()
{
  int x;
  unsigned long f;
  int i;
  int original;

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

  original=x;
  f=1;
  if (x<0)
    printf("Nekorektan unos\n");
  else
  {
     while (x>1)
     {
        f=f*x; /* vrednost izraza sa desne strane naredbe dodele
                  dodeljujemo promenljivoj sa leve strane naredbe
    dodele
                */
        x--;  /* operator -- umanjuje vrednost promenljive x za 1
                 naredba x--; ima isti efekat kao x-=1;
                 ili x=x-1;
               */
     }
     printf("%d! = %lu\n",x,f);        /* nekorektno: vrednost
    promenljive x je unistena */
     printf("%d! = %lu\n",original,f); /* korektno: promenljiva
    original sadrzi vrednost promenljive x pre ulaska u petlju */

  }


  return 0;
}
```

Rešenje 2.75

```c
/* Sa standradnog ulaza unose se realan broj x i ceo neoznacen broj n
    . Napisati
program koji izracunava x^n */

#include <stdio.h>

int main(){

  int n;
```

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

  /* Ucitavaju se brojevi x i n */
  printf("Unesite redom brojeve x i n: ");
  scanf("%f %d", &x, &n );

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

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

  /* Stampamo rezultat */
  printf("%f\n",vrednost);

  return 0;
}
```

Rešenje 2.76

```
/* Sa standradnog ulaza unose se realan broj x i ceo broj n. Napisati
program koji izracunava x^n */

#include <stdio.h>

int main(void){

    int n, n_abs;
    float x;
    float vrednost;
    unsigned exp;

    /* Ucitavaju se brojevi x i n */
    printf("Unesite redom brojeve x i n: ");
    scanf("%f %d", &x, &n );

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

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

    /* Stampamo rezultat */
    if(n<0){
      printf("%.3f\n",1/vrednost);
    }
    else{
```

```
        printf("%.3f\n",vrednost);
31    }

33    return 0;
}
```

Rešenje 2.77

```
/*
   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
      treba da bude:
      Suma kubova od 1 do 4 je 100
   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
bude:
i=1, n=1
i=2, n=9
i=3, n=36
i=4, n=100

*/

#include <stdio.h>

int main()
{
 int n;
 int i;
 int s;


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

 if (n<0)
  return -1;

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

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

```
 return 0;
}
```

Rešenje 2.78

```
/* Sa standardnog ulaza unose se realan broj x i ceo neoznacen broj n
    . Napisati
program koji izracunava sumu S=x+2*x^2+3*x^3+...+n*x^n */

#include <stdio.h>

int main(){
  unsigned n, i;
  float x, S, stepen;

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

    /* Inicijalizujemo sumu koju racunamo */
  S=0;

    /* Stepen promenljiva ce sadrzati vrednosti stepena x^n -
     * pocetna vrednost joj je 1 */
  stepen=1;

  for(i=1; i<=n; i++){
    stepen=stepen*x;
    S=S+i*stepen;
  }

  printf("S=%f\n", S);

  return 0;
}
```

Rešenje 2.79

```
/* Sa standardnog ulaza unose se realan broj x i ceo neoznacen broj n
    .
Napisati program koji izracunava sumu S=1+1/x+1/x^2+1/x^3+...+1/x^n
    */

#include <stdio.h>
int main(){
  unsigned n, i;
  float x, S, stepen;

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

```
11
   S=1;
13 stepen=1;
   for(i=1; i<=n; i++){
15   stepen=stepen*x;
     S=S+1/stepen;
17 }

19 printf("S=%f\n", S);

21 return 0;
}
```

Rešenje 2.80

```
/* Napisati program koji sa zadatom tacnoscu izracunava sumu
2 S=1+x+x^2/2!+x^3/3!+...+x^n/n! + ...*/
/* Napomena: ovo je razvoj funkcije e^x */
4
#include <stdio.h>
6 #include <math.h>
int main(){
8     int n, i, faktorijel;
      float  S;
10    float x, eps, stepen;

12    printf("Unesite x: ");
      scanf("%f", &x);
14
      printf("Unesite tacnost eps: ");
16    scanf("%f", &eps);


18
      /* Tacnost izracunavanja je zadovoljena ako je apsolutna vrednost
20     * razlika suma
       * u dvema uzastopnim iteracijama manja od zadate tacnosti;
22     * Odavde se izvodi da apsolutna vrednost opsteg clana sume
       * mora da bude manja od zadate tacnosti da bi uslov bio ispunjen
24     */

26    S=1;
      faktorijel=1;
28    stepen=x;
      i=2;
30    while(fabs(stepen/faktorijel)>eps){
          S=S+stepen/faktorijel;
32        stepen=stepen*x;
          faktorijel=faktorijel*i;
34        i++;
      }
36
```

```
    printf("S=%f\n", S);

    return 0;
}
```

Rešenje 2.81

```
/* Napisati program koji sa zadatom tacnosu izracunava sumu
 S=1-x+x^2/2!-x^3/3!+... */
/* razvoj funkcije sin(x) */

#include <stdio.h>
#include <math.h>
int main(){
  int n, i, faktorijel, znak;
  float S;
  float x, eps, stepen;

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

  printf("Unesite tacnost eps: ");
  scanf("%f", &eps);

    /* Tacnost izracunavanja je zadovoljena ako je apsolutna vrednost
     * razlika suma
     * u dvema uzastopnim iteracijama manja od zadate tacnosti;
     * Odavde se izvodi da apsolutna vrednost opsteg clana sume
     * mora da bude manja od zadate tacnosti da bi uslov bio ispunjen
     */

    S=1;
  faktorijel=1;
  stepen=x;
  i=2;
  znak=-1;
  while(fabs(stepen/faktorijel)>eps){
    S=S+znak*stepen/faktorijel;
        stepen=stepen*x;
        faktorijel=faktorijel*i;
    znak=-znak;
    i++;
  }

  printf("S=%f\n", S);

  return 0;
}
```

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.92

```
#include <stdio.h>

int main(){

  int n;
  int k, v;

  /* Unosi se dimenzija kvadrata */
  printf("Unesite broj n: ");
  scanf("%d", &n);

  /* Kvadrat iscrtavamo vrstu po vrstu */
  for(v=1; v<=n; v++){

    /* Ukoliko je u pitanju prva ili poslednja vrsta */
    if(v==1 || v==n){

      /* Stampamo n zvezdica */
      for(k=1; k<=n; k++)
        putchar('*');
```

```
    }
    else
    {
      /* Ukoliko vrsta nije ni prva ni poslednja */

      /* Stampamo prvu zvezdu */
      putchar('*');

      /* Zatim stampamo n-2 blanko znaka */
      for(k=1; k<=n-2; k++)
        putchar(' ');

      /* I stampamo poslednju zvezdu */
      putchar('*');

    }

    /* Posto smo odstampali sadrzaj vrste, stampamo i znak za novi
    red */
    putchar('\n');
  }

  return 0;
}
```

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.97

```
#include <stdio.h>

int main(){

  int n;
  char novi_red, c;
  int i, j;

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

  scanf("%c", &novi_red);
```

```
14    printf("Unesite karakter c: ");
      scanf("%c", &c);
16
      /* Iscrtavamo trougao red po red */
18    for(i=1; i<=n; i++){
20      /* Ukoliko je u pitanju prvi ili poslednji red */
        if(i==1 || i==n){
22        /* Stampamo i puta karakter c */
          for(j=1; j<=i; j++)
24          putchar(c);
        }
26      else{
          /* Inace... */
28
          /* Stampamo prvi karakter */
30        putchar(c);
32        /* Stampamo i-2 blanko znaka */
          for(j=1; j<=i-2; j++)
34          putchar(' ');
36        /* Stampamo poslednji karakter */
          putchar(c);
38      }
40      /* Stampamo znak za novi red */
        printf("\n");
42    }
44    return 0;
  }
```

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.102

```
1 #include <stdio.h>

3 int main(){
```

```
5   int n, i, j;

7   printf("Unesite broj n: ");
    scanf("%d", &n);
9
    /* Krstice koje iscrtavamo mozemo posmatrati kao dijagonale
       kvadrata dimenzije n */
11
    /* Prolazimo kroz sve vrste kvadrata */
13  for(i=1; i<=n; i++){

15    /* Prolazimo kroz sve kolone kvadrata */
      for(j=1; j<=n; j++){
17
        /* Ako se nalazimo na glavnoj ili sporednoj dijagonali */
19      if(i==j || i+j==n+1)
          /* Stampamo zvezdu */
21        putchar('*');
        else
23        /* Inace, stampamo blanko znak */
          putchar(' ');
25    }

27    /* Nakon uspesno iscrtane vrste, stampamo znak za novi red */
      putchar('\n');
29  }

31  return 0;
}
```

Rešenje 2.103

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

Rešenje 2.107

2.5 Funkcije

TODO Potpisi funkcija ne treba da budu ni verb ni $ vec u okviru taga , kao sto to pise u uputstvima u okviru kartice za formatiranje teksta

Zadatak 2.108 Napisati funkcije `int kvadrat(int x)` i `int kub(int x)` koje računaju, redom, kvadrat i kub datog broja. Napisati program koji testira rad ovih funkcija.

[Rešenje 2.108]

Zadatak 2.109 Napisati funkciju `float stepen(float x, int n)` koja računa x^n. Napisati program koji testira rad ove funkcije.

[Rešenje 2.109]

Zadatak 2.110 Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najveći zajednički delilac primenom Euklidovog algoritma. Napisati program koji testira rad ove funkcije.

[Rešenje 2.110]

Zadatak 2.111 Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato n vraca zbir recipročnih vrednosti brojeva od 1 do n. Napisati program koji testira rad ove funkcije. Rezultat zaokružiti na dve decimale.

[Rešenje 2.111]

Zadatak 2.112 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 2.112]

Zadatak 2.113 Napisati funkciju `void ispis(float x, float y, unsigned n)` koja za dva realna broja x i y i jedan neoznačeni ceo broj n ispisuje vrednosti

funkcije *sin* u n ravnomerno raspoređenih tačaka intervala [x,y]. Napisati program koji testira rad ove funkcije.

[Rešenje 2.113]

Zadatak 2.114 Napisati funkciju `int broj_ncifara(int x)` koja broji neparne cifre u zapisu datog celog broja. Testirati rad ove funkcije u programu koji učitava cele brojeve dok se ne unese nula i ispisuje broj neparnih cifara svakog unetog broja.

[Rešenje 2.114]

Zadatak 2.115 Napisati funkciju $int\ min(int\ x, int\ y, int\ z)$ koja izračunava minimum tri broja. Napisati program koji sa standardnog ulaza učitava tri cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
Interakcija sa programom:
 Unesite brojeve: 19 8 14
 Minimum je: 8
```

Primer 2

```
Interakcija sa programom:
 Unesite brojeve: -6 11 -12
 Minimum je: -12
```

[Rešenje 2.115]

Zadatak 2.116 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 2.116]

Zadatak 2.117 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 2.117]

Zadatak 2.118 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 2.118]

Zadatak 2.119 Napisati funkciju *void grafikon_h(int a, int b, int c, int d)* koja iscrtava horizontalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i prikazuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite vrednosti: 4 1 7 5
 ****
 *
 *******
 *****
```

Primer 2

```
Interakcija sa programom:
 Unesite vrednosti: 8 -2 5 4
 Greska: pogresan unos!
```

Primer 3

```
Interakcija sa programom:
 Unesite vrednosti: 5 2 2 10
 *****
 **
 **
 **********
```

[Rešenje 2.119]

Zadatak 2.120 Napisati funkciju *void grafikon_v(int a, int b, int c, int d)* koja iscrtava vertikalni prikaz zadatih vrednosti. Napisati program koji učitava

četiri pozitivna cela broja i ispisuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

```
Interakcija sa programom:
 Unesite vrednosti: 4 1 7 5
   *
   *
   **
 * **
 * **
 * **
 ****
```

Primer 2

```
Interakcija sa programom:
 Unesite vrednosti: 8 -2 5 4
 Greska: pogresan unos!
```

Primer 3

```
Interakcija sa programom:
 Unesite vrednosti: 5 2 2 4
 *
 *  *
 *  *
 ****
 ****
```

[Rešenje 2.120]

Zadatak 2.121 Napisati funkciju *int prestupna*(*int godina*) koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati 1 ako je godina prestupna ili 0 ako nije. Napisati program koji učitava dva cela broja $g1$ i $g2$ i ispisuje sve godine iz intervala $[g1, g2]$ koje su prestupne.

Primer 1

```
Interakcija sa programom:
 Unesite dve godine: 2001 2010
 Prestupne godine su: 2004 2008
```

Primer 2

```
Interakcija sa programom:
 Unesite dve godine: 2005 2015
 Prestupne godine su: 2008 2012
```

Primer 3

```
Interakcija sa programom:
 Unesite dve godine: 2010 2001
 Greska: pogresan unos!
```

Primer 4

```
Interakcija sa programom:
 Unesite dve godine: 2001 2002
 Nema prestupnih godina u ovom intervalu!
```

[Rešenje 2.121]

Zadatak 2.122 Napisati funkciju `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu. Napisati program koji testira ovu funkciju. U slučaju nekorektnog unosa ispisati odgovarajuću poruku o grešci.

[Rešenje 2.127]

Zadatak 2.123 Napisati funkciju `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan. Napisati program koji testira ovu funkciju.

[Rešenje 2.127]

Zadatak 2.124 Napisati funkciju `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum određuje datum sledećeg dana. Napisati program koji testira ovu funkciju.

[Rešenje 2.127]

Zadatak 2.125 Napisati funkciju `int broj_dana1(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do do datog datuma. Napisati glavni program koji testira napisanu funkciju.

[Rešenje 2.127]

Zadatak 2.126 Napisati funkciju `int broj_dana2(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati glavni program koji testira napisanu funkciju.

[Rešenje 2.127]

Zadatak 2.127 Napisati funkciju `int broj_dana3(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati glavni program koji testira napisanu funkciju.

[Rešenje 2.127]

Zadatak 2.128 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 2.128]

Zadatak 2.129 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 2.129]

Zadatak 2.130 Napisati funkciju *int rotacija(int n)* koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sa 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 2.130]

Zadatak 2.131 Napisati funkciju `int prost (int x)` koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati 1 ako je broj prost i 0 u suprotnom.

Testirati rad funkcije u programu koji za uneti ceo broj n ispisuje prvih n prostih brojeva.

[Rešenje 2.131]

Zadatak 2.132 Napisati funkciju `int sadrzi(int x, int c)` koja ispituje da li se cifra c nalazi u zapisu celog broja x. Napisati program koji testira rad ove funkcije.

[Rešenje 2.132]

Zadatak 2.133 Napisati program za ispitivanje svojstava cifara datog celog broja.

a) Napisati funkciju `sve_parne_cifre` koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.

b) Napisati funkciju `sve_cifre_jednake` koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

c) Testirati napisane funkcije na unetom celom broju i ispisati odgovarajuću poruku.

[Rešenje 2.133]

Zadatak 2.134 Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva uneta neoznačena broja x i n utvrdjuje da li je x neki stepen broja n. Ukoliko jeste, funkcija vraca izložilac stepena, a u suprotnom vraća -1. Napisati program koji testira rad ove funkcije.

[Rešenje 2.134]

Zadatak 2.135 Napisati funkciju `double e_na_x(double x, double eps)` koja računa vrednost e^x kao parcijalnu sumu reda $\sum_{n=0}^{\infty} \frac{x^n}{n!}$, pri čemu se sumiranje vrši dok je razlika sabiraka u redu po apsolutnoj vrednosti manja od ε. Napisati program koji testira rad ove funkcije.

[Rešenje 2.135]

Zadatak 2.136 Za dati broj moze se formirati niz tako da je svaki sledeći član niza dobijen kao suma cifara prethodnog člana niza. Broj je srećan ako se dati niz završava jedinicom. Napisati funkciju `int srecan(int x)` koja vraca 1 ako je broj srećan, a 0 u suprotnom. Napisati program koji za uneti prirodan broj n ispisuje sve srećne brojeve od 1 do n.

[Rešenje 2.136]

Zadatak 2.137 Napisati funkciju `int konverzija (int c)` koja prebacuje veliko slovo u ekvivalentno malo i obrnuto. Napisati program koji testira ovu funkciju na karakterima koji se unose sa standardnog ulaza do pojave EOF.

[Rešenje 2.137]

Zadatak 2.138 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 2.138]

Zadatak 2.139 Napisati funkciju $int\ faktorijel(int\ n)$ koja računa faktorijel broja n. Napisati i program koji učitava dva cela broja x i y ($0 \leq x, y \leq 12$) i ispisuje 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 2.139]

Zadatak 2.140 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 2.140]

Zadatak 2.141 Broj je Armstrongov ako je jednak sumi nekog stepena svojih cifara.

a) Napisati funkciju *int stepen*(*int x*, *int n*) koja izračunava *n*-ti stepen broja *x*.

b) Napisati funkciju *int armstrong*(*int x*) koja vraća 1 ako je broj Armstrongov, odnosno 0 ako nije.

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 2.141]

Zadatak 2.142 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 2.142]

Zadatak 2.143 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 2.143]

Zadatak 2.144 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 2.144]

Zadatak 2.145 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 2.145]

2.6 Rešenja

Rešenje 2.108

```
1 #include <stdio.h>

3 int kvadrat(int x)
  {
5    /* promenljive u listi argumenata funkcije, kao i one
```

```
      deklarisane u samoj funkciji, lokalne su za tu funkciju
      sto znaci da se promenljive x i y nece "videti" nigde izvan
      funkcije kvadrat (ni u funkciji main ni u funkciji kub)
   */

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

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

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

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

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

Rešenje 2.109

```
/*
Napisati program koji za uneti realan broj x i ceo broj n ispisuje
vrednost stepena x^n. Unosenje promenljivih, racunanje stepena i
ispis promenljivih realizovati u posebnim funkcijama.
*/

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

float stepen(float a, int b)
```

```
{
  float s=1;
  int i;

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

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

}

int main()
{
  int n;
  float x;
  float s;


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

  s = stepen(x,n);


  printf("%f^%d=%f\n",x,n,s);

  return 0;
}
```

Rešenje 2.110

```
/*
  Napisati funkciju koja za dva data cela broja odredjuje
  najveci zajednicki delilac. Napisati potom glavni program
  koji testira ovu funkciju.
*/

#include <stdio.h>

int euklid(int x, int y)
{
  int r;
  /* Euklidov algoritam */
  while(y)    /* algoritam se zaustavlja kada vrednost */
  {           /* promenljive y postane nula */
    r=x%y;
    x=y;
    y=r;
  }
```

```
19
  return x; /* nzd je sacuvan u promenljivoj x */
21 }

23 int main()
{
25   int a,b;
  int nzd;
27
  printf("unesi dva cela broja:");
29   scanf("%d%d", &a,&b);

31   nzd = euklid(a,b);  /* promenljivoj nzd dodeljujemo povratnu
       vrednost funkcije euklid */

33   printf("najveci zajednicki delilac za %d i %d je %d\n", a,b,nzd);

35   return 0;
}
```

Rešenje 2.111

```
/*
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 koristeci funkciju float zbir_reciprocnih
    (int n). Rezultat zaokruziti
na dve decimale.
6 */

8 #include <stdio.h>

10 float zbir_reciprocnih(int n)
{
12   float z=0;
  int i;
14   for(i=1;i<=n;i++)
    z+=1.0/i;  /* da bismo dobili reciprocnu vrednost broja, vazno je
     da izbegnemo celobrojno deljenje dva cela broja */
16   return z;    /* tako sto ce npr deljenik biti 1.0 umesto 1 */
}
18
int main()
20 {
  int n;
22   printf("Unesi jedan pozitivan ceo broj:\n");
  scanf("%d", &n);
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
      naredbe printf i umesto specifikatora %.2f bice ispisana
     povratna vrednost funkcije
      zbir_reciprocnih zaokruzena na dve decimale */
  return 0;
}
```

Rešenje 2.112

```
/*
Napisati funkciju koja racuna aritmeticku sredinu cifara datog celog
    broja.
Napisati potom glavni program koji omogucava korisniku da unese ceo
    broj
i racuna aritmeticku sredinu njegovih cifara primenom napisane
    funkcije. Ispisati
izracunatu vrednost zaokruzenu na dve decimale.
*/

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

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

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


   x=abs(x); /* uzimamo apsolutnu vrednost broja za slucaj da je
    negativan */

   while(x)
   {
      cifra=x%10;

      broj_cifara++;
      zbir_cifara+=cifra;

      x/=10;
   }

   return (0.0+zbir_cifara)/broj_cifara;  /* posto su zbir_cifara i
    broj_cifara celobrojne vrednosti,
                                          neophodno je da bar
    jednu od njih konvertujemo u realnu
```

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

Rešenje 2.113

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

31   if (n<=1 || a==b)
```

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


43


45  ispis(a,b,n);

47  return 0;
}
```

Rešenje 2.114

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

10 int broj_ncifara(int x)
{
12    int s=0;
      char cifra;
14    x = abs(x);

16    while(x)
      {
18       cifra = x%10;
         s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
      neparna,
20                         a 0 kada je cifra parna */
         x/=10;
22    }

24    return s;
}
26
int main()
```

```
{
  int x;
  do
  {
     scanf("%d",&x);
     printf("Broj neparnih cifara u zapisu broja %d: %d\n", x,
    broj_ncifara(x));
  } while(x!=0);

  return 0;
}
```

Rešenje 2.115

```
#include <stdio.h>

/*
  Funkcija koja racuna minimum tri cela broja
*/
int min(int x, int y, int z){
  int min;

  min=x;

  if(min>y)
    min=y;

  if(min>z)
    min=z;

  return min;
}

int main(){
  int x,y,z;

  /* Ucitavamo brojeve */
  printf("Unesite brojeve: ");
  scanf("%d%d%d", &x, &y, &z);

  /* Pozivamo funkciju i ispisujemo rezultat */
  printf("Minimum je: %d\n", min(x,y,z));

  return 0;
}
```

Rešenje 2.116

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

```c
/* Funkcija koja racuna apsolutnu vrednost */
unsigned int apsolutna_vrednost(int x){
  /* Kako funkcija vraca unsigned, a x je tipa int, vrsimo kastovanje
     rezultata u tip unsigned */
  return (unsigned)(x<0?-x:x);
}

int main(){
  int n;

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

  /* Ispisujemo njegovu apsolutnu vrednost */
  printf("Apsolutna vrednost: %u\n", apsolutna_vrednost(n));

  return 0;
}
```

Rešenje 2.117

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

/* Funkcija koja vraca razlomljeni deo prosledjenog broja */
float razlomljeni_deo(float x){

  /* Funkcija fabs vraca apsolutnu vrednost realnog broja
   * NAPOMENA: funkcija fabs se nalazi u zaglavlju math.h
   * NAPOMENA2: funkcija abs se nalazi u zaglavlju stdlib.h, ali se
     koristi samo za cele brojeve!
  */
  x = fabs(x);

  /* Razlomljeni deo broja dobijamo tako sto od samog broja oduzmemo
     njegov ceo deo*/
  return x - (int)x;
}

int main(){
  float n;

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

  /* Ispisujemo rezultat */
  printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));
```

```
27    return 0;
}
```

Rešenje 2.118

```
#include<stdio.h>

/* Funkcija koja iscrtava romb */
void romb(int n){
  int i, j;

  /* U svakoj liniji */
  for(i=0; i<n; i++){

  /* Prvo ispisujemo n-i-1 razmaka */
  for(j=0; j<n-i-1; j++)
    printf(" ");

  /* Zatim ispisujemo n zvezdica */
  for(j=0; j<n; j++)
    printf("*");

  /* Na kraju svake linije stoji oznaka za novi red */
  printf("\n");
  }

}

int main(){
  int n;

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

  /* Proveravamo korektnost ulaza i ispisujemo rezultat */
  if(n<=0)
  printf("Greska: pogresna dimenzija!\n");
  else
  romb(n);

  return 0;
}
```

Rešenje 2.119

```
#include<stdio.h>

```

```c
/* Funkcija koja stampa n zvezdica za kojima sledi znak za novi red
    */
void stampaj_zvezdice(int n){
  int i;
  for(i=0; i<n; i++)
    printf("*");

  printf("\n");
}

/* Funkcija koja crta grafikon */
void grafikon_h(int a, int b, int c, int d)
{
  int i;

  /* Prvo ispisujemo a zvezdica */
  stampaj_zvezdice(a);

  /* Zatim u sledecem redu b zvezdica */
  stampaj_zvezdice(b);

  /* Zatim u sledecem redu c zvezdica */
  stampaj_zvezdice(c);

  /* Zatim u poslednjem redu d zvezdica */
  stampaj_zvezdice(d);

}

int main(){
  int a,b,c,d;

  /* Ucitavamo vrednosti a,b,c,d */
  printf("Unesite vrednosti: ");
  scanf("%d%d%d%d", &a, &b, &c, &d);

  /* Proveravamo korektnost ulaza i ispisujemo rezultat */
  if(a <0 || b<0 || c<0 || d<0){
  printf("Greska: pogresan unos!\n");
  }else{
  grafikon_h(a,b,c,d);
  }

  return 0;
}
```

Rešenje 2.120

```c
#include<stdio.h>

int maksimum(int a, int b, int c, int d){
```

```
  int max;

  max=a;
  if(b>max)
    max=b;
  if(c>max)
    max=c;
  if(d>max)
    max=d;

  return max;
}

/* Funkcija koja iscrtava vertikalni grafikon */
void grafikon_v(int a, int b, int c, int d){
  int i, max;

  /* Na pocetku je potrebno pronaci najvecu od ove cetiri vrednosti
     */
  max=maksimum(a, b, c, d);

  /* Grafikon ukupno ima max horizontalnih linija */
  for(i=0; i<max; i++){

    /* U svakoj od horizontalnih linija se nalazi po 4 polja:
    polje za a,b,c i d uspravnu liniju.
    U svako od polja treba da se upise ili * ili belina,
    u zavisnosti od vrednosti a i toga u kojoj liniji se trenutno
    nalazimo
   */

  /* Proveravamo uslov za polje a */
  if(i<max-a)
    printf(" ");
  else
    printf("*");

  /* Proveravamo uslov za polje b */
  if(i<max-b)
    printf(" ");
  else
    printf("*");

  /* Proveravamo uslov za polje c */
  if(i<max-c)
    printf(" ");
  else
    printf("*");

  /* Proveravamo uslov za polje d */
  if(i<max-d)
    printf(" ");
```

```
  else
    printf("*");

  /* Na kraju svake horizontalne linije stampamo novi red */
  printf("\n");
  }
}

int main(){
  int a,b,c,d;

  /* Ucitavamo vrednosti a,b,c,d */
  printf("Unesite vrednosti: ");
  scanf("%d%d%d%d", &a, &b, &c, &d);

  /* Proveravamo korektnost ulaza i stampamo grafikon */
  if(a <0 || b<0 || c<0 || d<0)
  printf("Greska: pogresan unos!\n");
  else
  grafikon_v(a,b,c,d);

  return 0;
}
```

Rešenje 2.121

```
#include<stdio.h>

/* Funkcija koja proverava da li je godina prestupna */
int prestupna(int godina){
  if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
  return 1;
  else
  return 0;
}

/* Funkcija koja proverava da li postoji prestupna godina u datom
    intervalu */
int postoji_prestupna(int g1, int g2){
  for(; g1<=g2; g1++){
  if(prestupna(g1))
    return 1;
  }
  return 0;
}

int main(){

  int g1, g2;

  /* Ucitavamo godine */
```

```
25    printf("Unesite dve godine: ");
      scanf("%d%d", &g1, &g2);
27
      /* Proveravamo korektnost ulaza */
29    if(g1 < 0 || g2 < 0 || g1>g2){
      printf("Greska: pogresan unos!\n");
31    }
      else{
33
      /* Proveravamo da li uopste postoji prestupna godina u datom
        intervalu */
35    if(postoji_prestupna(g1,g2)){
        /* Ako postoje, ispisujemo ih */
37      printf("Prestupne godine su: ");
        for(; g1<=g2; g1++){
39      if(prestupna(g1))
          printf("%d ", g1);
41      }
        printf("\n");
43    }else{
        /* U suprotnom, stampamo odgovarajucu poruku */
45      printf("Nema prestupnih godina u ovom intervalu!\n");
      }
47    }
      return 0;
49  }
```

Rešenje 2.127

```
1   #include <stdio.h>

3   /* Funkcija koja racuna zbir delilaca broja x */
    int zbir_delilaca(int x){
5     int i=0;

7     /* Na pocetku zbir inicijalizujemo na 0 */
      int zbir = 0;
9
      /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
11    for(i=1; i<=x; i++){
      if(x % i == 0)
13      zbir += i;
      }
15
      /* Vracamo dobijeni zbir */
17    return zbir;
    }
19
    int main(){
21
      int k, i;
```

```
23
   /* Ucitavamo broj k */
25 printf("Unesite broj k:");
   scanf("%d", &k);
27
   /* Proveravamo korektnost ulaza */
29 if(k <= 0)
     printf("Greska: pogresan unos!\n");
31 else{

33   /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
     for(i=1; i<=k; i++)
35   printf("%d ", zbir_delilaca(i));

37   printf("\n");
   }
39

41 return 0;
}
```

Rešenje 2.127

Rešenje 2.127

Rešenje 2.127

Rešenje 2.127

Rešenje 2.127

Rešenje 2.128

```
1 #include <stdio.h>

3 /* Funkcija koja racuna zbir delilaca broja x */
int zbir_delilaca(int x){
5   int i=0;

7   /* Na pocetku zbir inicijalizujemo na 0 */
    int zbir = 0;
9
   /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
11 for(i=1; i<=x; i++){
   if(x % i == 0)
13   zbir += i;
```

```
  }

  /* Vracamo dobijeni zbir */
  return zbir;
}

int main(){

  int k, i;

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

  /* Proveravamo korektnost ulaza */
  if(k <= 0)
    printf("Greska: pogresan unos!\n");
  else{

    /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
    for(i=1; i<=k; i++)
    printf("%d ", zbir_delilaca(i));

    printf("\n");
  }


  return 0;
}
```

Rešenje 2.129

```
#include <stdio.h>

/* Funkcija koja uklanja broj stotina iz broja n */
int ukloni_stotine(int n){

  /* Ako je broj izmedju -100 i 100 nema cifru desetica pa onda
     vracamo isti taj broj */
  if(n>-100 && n<100)
  return n;
  else
  {
  /* U suprotnom vracamo broj sa uklonjenom cifrom stotina */

  /* Odredjujemo znak broja */
  int znak=(n<0)? -1 : 1;

  /* I nadalje radimo sa apsolutnom vrednoscu broja */
  n=abs(n);

```

```
  return znak*((n/1000)*100 + n%100);
  }
}

/* Funkcija koja vraca znak broja */
int znak(int broj){
  return broj<0?-1:1;
}

int main(){

  int broj;

  while(1){

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

  /* Broj 0 oznacava kraj rada */
  if(broj == 0)
    break;

  /* Ispisujemo rezultat, vodeci racuna da program treba da radi
    ispravno i za negativne brojeve */
  printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
  }


  return 0;
}
```

Rešenje 2.130

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

int rotacija(int n){

  /* U promenljivoj broj pamtimo originalnu vrednost n */
  int broj, br = 0, znak;

  /* Odredjujemo znak broja */
  znak=(n<0) ? -1: 1;

  /* I nadalje radimo sa apsolutnom vrednoscu broja */
  n=abs(n);

  /* U promenljivoj broj cuvamo kopiju broja n */
  broj=n;
```

```c
  /* Ako je broj jednocifren, nema potrebe da ga rotiramo. */
  if(n>-10 && n < 10)
    return n;

  /* Petljom izdvajamo cifru po cifru, kako bismo dosli do krajnje
    leve cifre broja
   (one koja treba da postane krajnje desna), npr za n = 1234, treba
    da dobijemo 1,
   zatim da "pomerimo" 234 u levo i da na kraj nalepimo 1 = 2341 */

  /* Na kraju ove petlje, u n se nalazi najlevlja cifra broja (koja
    treba da postane krajnje desna),
   dok se u br nalazi broj cifara unetog broja */
  while(n >=10){
    n/=10;
    br++;
  }

  /*
   Levi deo (234) dobijamo kao n%(10^broj_cifara)
   Zatim levi deo pomnozimo sa 10, da bi dobili 2340
   Zatim na levi deo dodamo desni deo (1) koja se nalazi u
    promenljivoj n
   */

  return znak* ((broj%(int)pow(10, br))*10 + n);
}

int main(){

  int n;
  while(1){

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

  /* Ako je uneta 0, izlazimo iz petlje */
  if(n == 0)
    break;

  /* Stampamo broj rotiran za jedno mesto u levo */
  printf("%d\n", rotacija(n));
  }


  return 0;
}
```

Rešenje 2.131

```c
/*
Napisati funkciju koja ispituje da li je dati ceo broj prost.
    Funkcija treba
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.
*/

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

int prost (int x) /* 1-broj je prost, 0-broj nije prost */
{
  int i;

  if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
    return 1;

  if (x%2==0)       /* parni brojevi nisu prosti */
    return 0;

  for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
    if (x%i==0) /* ako je pronadjen, to znaci da broj nije prost */
      return 0; /* zavrsavamo funkciju */

  /* ukoliko izvrsavanje funkcije dodje do poslednje naredbe return,
     to znaci da broj nije ispunio nijedan od prethodnih uslova
     (nije ni 2, ni 3, ni paran, niti ima ijednog delioca), odakle
     sledi da je prost i zbog toga vracamo 1
  */
  return 1;
}

int main()
{
  int n;
  scanf("%d",&n);
  int i,j;

  i=1; /* kandidat za prost broj */
  j=0; /* brojac prostih brojeva */
  while(j<n)
  {
     if (prost(i))         /* ako je broj prost */
     {
        printf("%d\n", i); /* stampamo ga i */
        j++;               /* uvecavamo brojac prostih brojeva */
     }
     i++; /* bilo da je i prost ili ne, uvecavamo ga za 1 i
    nastavljamo sa sledecom iteracijom */
  }
```

```
49
   return 0;
51 }
```

Rešenje 2.132

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

7  #include<stdio.h>
   #include<stdlib.h>
9
   int sadrzi(int x, int c)
11 {
      char cifra;
13    x=abs(x);
      while(x)
15    {
         cifra = x%10;
17       if (cifra==c)
            return 1;
19       x/=10;
      }
21    return 0;
   }
23 int main()
   {
25   int x;
     int c;
27   printf("Unesi jedan ceo broj i jednu cifru:");
     scanf("%d%d",&x,&c);
29   if (sadrzi(x,c))
        printf("Cifra %d se nalazi u zapisu broja %d\n",c,x);
31   else
        printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
33   return 0;
   }
```

Rešenje 2.133

```
/*
2

a) Napisati funkciju sve_parne_cifre koja ispituje da li se dati ceo
    broj sastoji iskljucivo iz parnih cifara. Funkcija treba
```

```
da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.

b) Napisati funkciju sve_cifre_jednake koja ispituje da li su sve
    cifre datog celog broja jednake. Funkcija treba
da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

c) Napisati potom glavni program koji na uneti ceo broj primenjuje
    napisane funkcije i ispisuje odgovarajuce poruke.

Na primer, za uneti broj 222, program treba da ispise:
Sve cifre broja su parne.
Sve cifre broja su jednake.

A za uneti broj -284:
Sve cifre broja su parne.
Broj sadrzi razlicite cifre

*/
#include <stdio.h>
#include <stdlib.h>

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

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

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

  return 1;      /* ukoliko se while petlja zavrsila, to znaci da
    uslov d%2==1 nije
                    nijednom bio ispunjen i da su sve cifre broja
    parne; zbog toga
                    vracamo 1
                */

}

int sve_cifre_jednake(int x) /* funkcija vraca 1 ako su sve cifre
    broja jednake i 0 u suprotnom*/
{
  char d;
```

```
  char prva_cifra;
48  x=abs(x);
  prva_cifra = x%10; /* izdvajamo prvu cifru broja */
50  x/=10;             /* broj delimo sa 10 jer smo prvu cifru vec
      izdvojili */

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

56    if (d!=prva_cifra)
        return 0;
58
      x/=10;
60  }

62  return 1;
}
64 main()
{
66  int x;
  int d;
68
  printf("unesi ceo broj:");
70  scanf("%d", &x);

72  if (sve_parne_cifre(x))
      printf("Sve cifre broja su parne\n");
74  else
      printf("Broj sadrzi bar jednu neparnu cifru\n");
76
  if (sve_cifre_jednake(x))
78    printf("Sve cifre broja su jednake\n");
  else
80    printf("Broj sadrzi razlicite cifre \n");

82 }
```

Rešenje 2.134

```
/*
2 Napisati funkciju koja za dva uneta neoznacena broja x i n utvrdjuje
      da li je x neki stepen
broja n. Ukoliko jeste, funkcija vraca izlozilac stepena, a u
      suprotnom vraca -1. Napisati
4 potom glavni program koji testira ovu funkciju.
*/
6
#include <stdio.h>
8
```

```
int je_stepen(unsigned x, unsigned n) /* funkcija vraca izlozilac
    stepena ukoliko broj x jeste neki stepen broja n */
{
   int i=1;
   int s=n;

   while(s<x)
   {
      s=s*n;
      i++;
   }

   if (s==x)
      return i;

   return -1;
}

int main()
{
  unsigned x;
  unsigned n;
  int st;

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

  st = je_stepen(x,n);

  if (st!=-1)
    printf("%u=%u^%d\n",x,n,st);
  else
    printf("%u nije stepen broja %u\n",x,n);

  return 0;
}
```

Rešenje 2.135

```
/*

 Napisati funkciju

 double e_na_x(double x, double eps)

 koja racuna vrednost e^x kao parcijalnu sumu reda
 suma(x^n/n!), gde indeks n ide od
 od 0 do beskonacno, pri cemu se sumiranje vrsi dok
 je razlika sabiraka u redu po apsolutnoj vrednosti
 manja od eps. Napisati potom program koji omogucuje
 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 2.136

```
/*
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 zavrsava sa
jedinicom. Napisati program koji za uneti broj odredjuje da li je srecan.
Na primer:
- 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.
- broj 991 je srecan, zbir njegovih cifara je 19, zbir cifara broja 19 je 10, zbir cifara
broja 10 je 1.
- broj 372 nije srecan, zbir njegovih cifara je 12, zbir cifara broja 12 je 3

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

Napisati program koji omogucava korisnuku da unese prirodan broj, poziva funkciju
i ispisuje da li je dati broj srecan. Potom traziti od korisnika da unese prirodan
broj n i ispisati sve srecne brojeve od 1 do n.
*/

#include<stdio.h>

int zbir_cifara(int x)
{
   int s=0;
   char cifra;
   while(x)
   {
      cifra = x%10;
      s+=cifra;
      x/=10;
   }
   return s;
}

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

   do
   {
     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 neoznacen 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 2.137

```
   /*
2  . a) Napisati funkciju

4    int konverzija (int c)

6  koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.

8  b) Napisati program koji omogucava korisniku da unese niz karaktera
   sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
10 Na primer, za uneti tekst "Kolokvijum iz Prog1 je 1.12." program
   treba da ispise "kOLOVKIJUM IZ pROG1 JE 1.12."
12
   */
14 #include <stdio.h>

16 int konverzija(int c)
   {
18   /* kljucna rec return vraca povratnu vrednost funkcije (ako je ima)
        */
     /* i zavrsava izvrsavanje funkcije */
20
     if (c>='A' && c<='Z')
22     return c+'a'-'A';

24   if (c>='a' && c<='z')
       return c-'a'+'A';
26
     return c;
```

```
28 }

30 int main()
   {
32   int c;

34   while((c=getchar())!=EOF)  /* korisnik unosi karakter po karakter
       do konstante EOF */
       putchar(konverzija(c));  /* funkcija putchar ispisuje jedan
36                                 karakter na standardni izlaz */

38   return 0;
   }
```

Rešenje 2.138

```
1  #include <stdio.h>

3  /* Funkcija int zapis(int x, int y) proverava da li su dva cela broja
        napisana
    * pomocu istih cifara, kao i da li se te cifre pojavljuju
5   * isti broj puta.
    * Ideja je sledeca:
7   *  iz broja x izdvajaju se redom cifra po cifra s kraja,
    *  a zatim se svaka takva cifra trazi i u broju y.
9   *  Ukoliko postoji, eliminise se prvi put kada se pojavi (dakle,
       samo jednom).
    *  Ukoliko su sve cifre iste (***redosled nije bitan***),
11  *  na kraju ce i iz x i iz y biti sve cifre eliminisane",
    *  te ostaju nule u oba broja.
13  *
    *  Broj novo_y formira se, zbog jednostavnosti, pomocu Heronovog
       obrasca.
15  *  Ovaj postupak obradjen je u okviru funkcije int izbaci_cifru(int
       y, int cifra).
    */
17
   int izbaci_cifru(int y, int cifra) {
19
     int novo_y = 0;
21   int indikator = 0;
     int izdvojena_cifra;
23
     while(y) {
25
       izdvojena_cifra = y % 10;
27     /* U slucaju da se cifra razlikuje od one koju treba eliminisati,
        * ili ukoliko je jedna cifra vec elimisana =>
29      * tekucu cifru ukljuciti prilikom formiranja novog y
        * */
31     if(izdvojena_cifra != cifra || indikator)
```

```
33        /* Heronov obrazac.
          * Menja poredak cifara, ali on u ovom slucaju i nije bitan.
35        */
          novo_y = novo_y*10 + izdvojena_cifra;
37      else

39        /* U slucaju da je cifra vec eliminisana,
          * ne treba je opet eliminisati.
41        * Za svaku pojavu cifre iz x,
          * eliminise se jedna odgovarajuca pojava
43        * te cifre iz y.
          */
45        indikator = 1;

47      y /= 10;
    }
49
    return novo_y;
51  }

53  int zapis(int x, int y) {

55    /* Cifra koja se izdvaja iz x, a onda eliminise iz y */
    int cifra;
57
    /* U slucaju da su prosledjeni brojevi negativni */
59  x = abs(x);
    y = abs(y);
61
    while(x) {
63
      cifra = x % 10;
65    x /= 10;

67    y = izbaci_cifru(y, cifra);

69    /* otkomentarisati donju liniju radi lakseg pracenja rada
      programa: */
      // printf("Iz x izdvojeno: %d\n\tx = %d, y = %d\n\n", cifra, x, y
      );
71  }

73  return (x == 0 && y == 0);
}
75
int main() {
77
    int x, y;
79  printf("Unesite dva cela broja: ");
    scanf("%d%d", &x, &y);
81
```

```
  if(zapis(x, y))
83  printf("Uslov je ispunjen!\n");
  else
85  printf("Uslov nije ispunjen!\n");

87  return 0;
}
```

Rešenje 2.139

```
1  #include <stdio.h>

3  /* Funkcija racuna faktorijel broja.
    * Faktorijel formiramo mnozenjem sa trenutnom vrednoscu broja x,
5   * a zatim smanjujuci tu vrednost za 1.
    * Ukoliko je x = 5, f = 5 * 4 * 3 * 2 * 1
7   */
  int faktorijel(int x) {
9
    int f = 1;
11  while(x) {
      f *= x;
13    x--;
    }
15  return f;
  }
17
  int main() {
19
    int x, y;
21
    printf("Unesite dva broja: ");
23  scanf("%d%d", &x, &y);

25  /* Provera uslova.
     *
27   * Faktorijel je veoma brza funkcija, tj.
     * s povecanjem broja x, drasticno brzo uvecava se i vrednost x!.
29   * Tip podatka int ima ogranicenje u velicini broja koji moze da
       sadrzi.
     * Za 13! i vece, int ne bi mogao da sacuva sve cifre potrebne za
       zapis tako velikog broja,
31   * te bi doslo do prekoracenja.
     *
33   * Slicno, faktorijel nije definisan nad skupom negativnih celih
       brojeva.
     */
35  if(x < 0 || y < 0 || x > 12 || y > 12) {
      printf("Greska: pogresan unos!\n");
37  }
    else{
```

```
39    printf("%d\n", faktorijel(x) + faktorijel(y));
      }
41    return 0;
}
```

Rešenje 2.140

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

/* Funkcija proverava da li se
 * cifre u zapisu broja nalaze u rastucem poretku.
 *
 * Situacija od interesa je kada za dve uzastopne cifre to nije
    slucaj.
 * Tada ne treba proveravati i za ostale cifre,
 * vec odmah prekinuti izvrsavanje funkcije.
 *
 * Ukoliko funkcija nije ranije prekinuta,
 * to znaci da cifre jesu u rastucem poretku
 * (odnosno, kako izdvajamo cifre od nazad, u stvari proveravamo
    opadajuci poredak),
 * te treba vratiti 1.
 */

int rastuce(int n) {

  int tekuca_cifra;
  int prethodna_cifra;

  n = abs(n);

  /* Prvu cifru (odnosno, poslednju u zapisu broja)
   * izdvajamo izvan petlje
   * kako bismo mogli da je poredimo sa narednom
   */
  tekuca_cifra = n % 10;
  n /= 10;

  while(n) {

    /* Cifra koja je bila tekuca u prethodnoj iteraciji petlje,
     * u novoj iteraciji postaje prethodna.
     *
     * Novoizdvojena cifra je tekuca.
     */
    prethodna_cifra = tekuca_cifra;
    tekuca_cifra = n % 10;

    /* Ukoliko smo naisli na cifre koje kvare rastuci poredak,
```

```
     * prekidamo izvrsavanje funkcije sa odgovarajucom povratnom
    vrednoscu 0.
     */
     if(prethodna_cifra < tekuca_cifra)
  return 0;

    /* Inace, nastavljamo sa izdvajanjem cifara */
    n /= 10;
  }

  return 1;
}

int main() {

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

  if(rastuce(x))
    printf("Cifre su u rastucem poretku!\n");
  else
    printf("Cifre nisu u rastucem poretku!\n");

  return 0;
}
```

Rešenje 2.141

```
#include <stdio.h>

/* Funkcija racuna broj x na n-ti stepen */
int stepen(int x, int n) {

  int i;
  /* Promenljiva u kojoj se cuva proizvod broja x sa samim sobom, n
     puta */
  int st = 1;

  for(i = 1; i <= n; i++)
    st *= x;

  return st;
}

/* Funkcija proverava da li je broj Armstrongov. */
int armstrong(int x) {

  /* u y se cuva zbir i-tih stepena cifara */
  int y;
  /* stepen za koji se proverava */
```

```
  int i = 1;
  /* prilikom izdvajanja cifara, broj x se menja,
   * te treba imati promenlju koja cuva pravu vrednost x
   */
  int original = x;

  do {

    y = 0;
    /* Racunamo i-te stepene za svaku cifru,
     * i istovremeno te stepen sabiramo.
     * Rezultat pamtimo u promenljivoj y.
     */
    while(x) {

      y += stepen(x % 10, i);
      x /= 10;
    }

    /* x je sada promenjen, pa ga treba vratiti na pravu vrednost. */
    x = original;
    i++;

  } while(y < x); /* Petlju vrtimo sve dok je zbir stepena cifara
    manji od datog broja. */

  /* Ukoliko smo nasli i, takvo da je zbir i-tih stepena cifara
   * jednak upravo broju x, takav broj je Armstrongov,
   * te izraz x == y vraca 1.
   *
   * Inace, vraca 0, tj. broj nije Armstrongov.
   */
  return x == y;
}

int main() {

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

  if(armstrong(x))
    printf("Broj je Armstrongov!\n");
  else
    printf("Broj nije Armstrongov!\n");

  return 0;
}
```

Rešenje 2.142

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

/* Funkcija proverava da li su dve uzastopne cifre
 * razlicite parnosti.
 *
 * Interesantna situacija je ukoliko su dve uzastopne cifre
 * obe parne, odnosno obe neparne.
 * Ovaj uslov svodimo na poredjenje njihovih ostataka pri deljenju sa 2:
 * ukoliko su ostaci isti, cifre su iste parnosti,
 * te ne treba dalje proverati da li je uslov zadovoljen,
 * vec odmah prekinuti sa izvrsavanjem funkcije.
 *
 * Ukoliko dve uzastopne cifre ni u jednom slucaju nisu bile iste parnosti,
 * a izdvojene su sve cifre iz broja x,
 * uslov je ispunjen, pa funkcija vraca 1.
 */
int par_nepar(int x) {

  int prethodna_cifra;
  int tekuca_cifra;

  /* u slucaju da je uneti broj negativan */
  x = abs(x);

  /* jednu cifru izdvajamo van petlje
   * kako bismo mogli da je odmah u petlji poredimo sa narednom
   */
  prethodna_cifra = x % 10;
  x /= 10;

  while(x) {

    tekuca_cifra = x % 10;

    if(tekuca_cifra % 2 == prethodna_cifra % 2)
      return 0;

    /* tekuca cifra postaje prethodna cifra za narednu iteraciju */
    prethodna_cifra = tekuca_cifra;
    x /= 10;
  }

  return 1;
}

int main() {

  int x;
```

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

  if(par_nepar(x))
    printf("Broj ispunjava uslov!\n");
  else
    printf("Broj ne ispunjava uslov!\n");

  return 0;
}
```

Rešenje 2.143

```
#include <stdio.h>

/* Funkcija broji koliko puta se realan broj x
 * javlja u nizu unetih brojeva sa tastature.
 *
 * Brojevi se unose sve do pojave 0,
 * pa treba koristiti do..while petlju,
 * kako bi bar jedan broj bio unet (makar bio i 0).
 */
int prebrojavanje(float x) {

  /* y prihvata uneti broj sa tastature */
  float y;
  /* br_pojavljivanja je brojac koji broji koliko puta se broj x
     javlja u unetom nizu brojeva */
  int br_pojavljivanja = 0;

  printf("Unesite brojeve: ");
  do {

    /* Unosimo broj. */
    scanf("%f", &y);

    /* Poredimo uneti broj sa datim brojem.
     * Ukoliko je unet bas trazeni broj,
     * uvecavamo brojac.
     * */
    if(x == y)
      br_pojavljivanja++;

  } while(y); /* Sve dok nije uneta 0 */

  return br_pojavljivanja;
}

int main() {

  float x;
```

```
38    int br_pojavljivanja;

40    printf("Unesite broj x: ");
      scanf("%f", &x);
42
      br_pojavljivanja = prebrojavanje(x);
44    printf("Broj pojavljivanja broja %.2f je: %d\n", x,
        br_pojavljivanja);

46    return 0;
  }
```

Rešenje 2.144

```
1   #include <stdio.h>

3   /* Funkcija racuna n-ti clan Fibonacijevog niza.
     * Clanovi ovog niza zadaju se rekurzivno tj. u zavisnosti od
        prethodnih clanova.
5    * Fibonacijevi brojevi od 0. do 47. se mogu smestiti u tip int, a
        kako n moze uzimati vrednosti
     * od 1 do 50, povratni tip funkcije je long int.
7    */
  long int fibonaci(int n) {
9
      int i;
11
      /* f0 i f1 su prva dva clana niza */
13    int f0 = 1;
      int f1 = 1;
15    /* promenljiva u kojoj se cuvaju opsti clanovi: n+2, n+1. i n-ti
        clan */
      long int fn2, fn1, fn;
17
      /* ukoliko treba vratiti nulti ili prvi clan,
19     * njih ne treba racunati
       * jer su vec dati.
21     */
      if(n == 0 || n == 1)
23      return 1;

25    /* postavljamo prethodne clanove niza */
      fn = f0;
27    fn1 = f1;
      /* racunamo od drugog clana, pa dok ne dodjemo do n-tog */
29    for(i = 2; i <= n; i++) {

31      /* izracunamo n+2-i clan niza sabiranjem prethodna dva clana */
        fn2 = fn1 + fn;
33      /* promenimo prethodne clanove niza, zbog naredne iteracije */
        fn =  fn1;
```

```
35      fn1 = fn2;
    }
37
    return fn2;
39  }

41  int main() {

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

47    /* Provera vrednosti za broj n */
    if(n <0 || n > 50) {
49      printf("Greska: nedozvoljena vrednost!\n");
    }
51    else{
      printf("%ld\n", fibonaci(n));
53    }

55    return 0;
  }
```

Rešenje 2.145

```
1   #include <stdio.h>

3   /* Funkcija vraca karakter koji se u abecedi
     * nalazi k mesta pre datog karaktera c
5    */
  char sifra(char c, int k) {
7
    /* Ukoliko je uneto malo slovo ... */
9   if(c >= 'a' && c <= 'z')
      /* Pri tome karakter koji je k pozicija pre datog karaktera
      ispada iz opsega malih slova ... */
11    if(c-k < 'a')
        /* Treba krenuti s drugog kraja abecede, racunajuci i
      preskocena slova.
13         *
           * Na primer, ukoliko je c = 'b' i k = 2
15         * Jedan karakter pre 'b' je 'a'.
           * Dva karaktera pre 'b' je 'z' (kruzno).
17         *
           * Karakter iz prvog dela abecede, koji je preskocen, je 'a'.
19         * Broj preskocenih karaktera iz prvog dela abecede
           * racunamo tako sto izracunamo c - 'a' (rastojanje od datog
      karaktera do malog slova a)
21         * sto je u ovom slucaju 'b' - 'a' = 1.
           *
23         * Ostatak karaktera do k ispisujemo, ali gledavsi unazad od z.
```

```
        * Zato racunamo k - (c - 'a') - 1.
25      *
        * Od k oduzimamo rastojanje izmedju c i 'a',
27      * kako bismo dobili preostali broj karaktera koji treba
    preskociti.
        */
29      return 'z' - (k - (c -  'a') - 1);
    else
31      /* U suprotnom, karakter ne ispada iz opsega malih slova, te je
     dovoljno bas njega i vratiti */
        return c-k;
33
  /* Ukoliko je uneto veliko slovo ... */
35  else if(c >= 'A' && c <= 'Z')
    if(c-k < 'A')
37      return 'Z' - (k - (c -  'A') - 1);
    else
39      return c-k;


41
  return c;
43}

45int main() {

47  int k;
  char c;
49
  printf("Unesite broj k: ");
51  scanf("%d", &k);

53  printf("Unesite tekst (CTRL + D za prekid): ");
  while((c = getchar()) != EOF)
55    putchar(sifra(c, k));

57  return 0;
}
```

3

Predstavljanje podataka

3.1 Nizovi

Zadatak 3.1 Skalarni proizvod dva vektora $a = (a_1, \ldots, a_n)$ i $b = (b_1, \ldots, b_n)$ je suma $a_1 \cdot b_1 + \ldots a_n \cdot b_n$. Napisati program koji računa skalarni proizvod dva vektora. Svaki vektor je zadat kao celobrojni niz sa najviše 100 elemenata. Program treba da učita dimenziju nizova (oba niza su iste dimenzije), zatim jedan po jedan element niza i da ispise njihov skalarni proizvod na standardni izlaz. Ako nizovi nisu iste dužine ispisati poruku `Greska!`.

[Rešenje 3.1]

Zadatak 3.2 Napisati program koji učitava broj elemenata niza (ne veći od 100), a zatim učitava elemente niza i ispisuje:

a) elemente niza koji se nalaze na parnim indeksima

b) parne elemente niza

Ukoliko je broj elemenata niza manji od 1 ili veči od 100 ispisati poruku `Greska!`.

[Rešenje 3.2]

Zadatak 3.3 Napisati program koji učitava jedan ceo broj a zatim ispisuje koliko puta svaka cifra učestvuje u zapisu tog broja. Nije potrebno ispisivati da se neka cifra pojavila 0 puta. UPUTSTVO: *Za evidenciju broja pojavljivanja svake cifre koristiti niz.*

[Rešenje 3.3]

Zadatak 3.4 Napisati program koji učitava karakter po karakter do znaka `EOF` i ispisuje koliko se puta u unetom tekstu pojavila svaka cifra, svako malo slovo i svako veliko slovo. Ispisati broj pojavljivanja samo za ona mala slova, velika slova i cifre koji su se u unetom tekstu pojavili više od 0 puta. UPUTSTVO: *Za evidenciju broja pojavljivanja cifara, malih i velih slova koristiti tri niza.*

[Rešenje 3.4]

Zadatak 3.5 Napisati program koji učitava dimenziju n dva celobrojna niza a i b (oba niza su iste dimenzije), zatim učitava elemente oba niza (prvo se unose elementi niza a, a potom elementi niza b) i formira treći niz c tako što naizmenično raspoređuje elemente nizova a i b unutar njega: $a_0, b_0, a_1, b_1, \ldots, a_{n-1}, b_{n-1}$. Program treba da ispiše elemente novog niza c na standardni izlaz. Maksimalni broj elemenata u nizovima a i b 100. U slučaju neispravnog unosa ispisati `Greska!`.

[Rešenje 3.5]

Zadatak 3.6 Napisati program koji učitava dimenziju n celobrojnog niza a i njegove elemente, a zatim iz niza a izbacuje sve elemente koji nisu deljivi svojom poslednjom cifrom, izuzev elemenata čija je poslednja cifra 0 koje treba zadržati. Program treba da ispiše izmenjeni niz na standardni izlaz. Niz a sadrzi najviše 100 elemenata.

[Rešenje 3.6]

Zadatak 3.7 Napisati funkcije za rad sa nizovima celih brojeva. U programu učitati dimenziju niza (ne veću od 100) i testirati rad napisanih funkcija. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

a) Napisati funkciju `void ucitaj(int niz[], int dimenzija)` koja učitava sadržaj niza.

b) Napisati funkciju `void stampaj(int niz[], int dimenzija)` koja štampa sadržaj niza.

c) Napisati funkciju koja računa sumu elemenata niza.

d) Napisati funkciju koja računa prosečnu vrednost elemenata niza.

e) Napisati funkciju koja izračunava minimum elemenata niza.

f) Napisati funkciju koja izračunava poziciju maksimalnog elementa u nizu.

[Rešenje 3.7]

Zadatak 3.8 Napisati funkcije za rad sa nizovima celih brojeva. U programu učitati dimenziju niza (ne veću od 100) i korišćenjem funkcije `ucitaj` iz zadatka 3.7 učitati elemente niza. Potom učitati ceo broj m i testirati rad napisanih funkcija. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

a) Napisati funkciju koja proverava da li niz sadrži neku vrednost m.

b) Napisati funkciju koja vraća vrednost prve pozicije na kojoj se nalazi element koji ima vrednost m, ili -1 ukoliko element nije u nizu.

c) Napisati funkciju koja vraća vrednost poslednje pozicije na kojoj se nalazi element koji ima vrednost m, ili -1 ukoliko element nije u nizu.

d) Napisati funkciju koja proverava da li elementi niza čine palindrom.

e) Napisati funkciju koja proverava da li su elementi niza uređeni neopadajuće.

f) Napisati funkciju koja izračunava najdužu uzastopnu seriju jednakih elemenata u nizu.

[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 3.114]

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 3.115]

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 3.116]

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
```

Primer 3

```
Interakcija sa programom:
 Unesite karakter: U
 Unesite karakter: 4
 Unesite karakter: a
 Unesite karakter: u
 Unesite karakter: *
 u a 4 U
```

[Rešenje 3.117]

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 nizove 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 3.118]

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 3.119]

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 3.17]

Zadatak 3.18 Sa standardnog ulaza se, redom, učitavaju dimenzija i elementi dvaju nizova 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 3.18]

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 3.19]

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 3.20]

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 3.21]

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
```

[Rešenje 3.22]

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 3.23]

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 3.24]

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 3.25]

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 3.26]

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

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 prosleđenom 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 duzine 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 izmedu 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 rezulata na standarni 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 izmedju 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:
 Ixxx 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

(a) 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 izmeu 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`.

(b) 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

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

#include <stdio.h>
#define MAX 100

/*

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

MAX se u ovom zadatku koristi kao maksimalni broj elemenata niza.
    Ukoliko bismo zeleli
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
```

```
strane, da nismo koristili pretprocesorsku direktivu i da smo svaki
    put
umesto MAX direktno navodili vrednost 100, morali bismo da je
    izmenimo na svakom
mestu u kodu.

*/
int main()
{
  int a[MAX];
  int b[MAX];
  int n;
  int i;
  int s;


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

  if (n<1 || n>100)
  {
    printf("Neispravan unos\n");
    return -1;
  }

  /*
     prvi element niza ima indeks 0, a poslednji n-1,
     gde je n broj elemenata niza; elementima niza pristupamo
     preko indeksa; na primer, ako niz a ima 5 elemenata, mozemo
     im pristupiti pomocu
     a[0], a[1], a[2], a[3], a[4]

  */

  for (i=0; i<n; i++)
  {
    printf("a[%d]=",i);
    scanf("%d", &a[i]);
  }

  for (i=0; i<n; i++)
  {
    printf("b[%d]=",i);
    scanf("%d", &b[i]);
  }

  s=0;

  for (i=0; i<n; i++)
    s = s + a[i]*b[i];

  printf("Skalarni proizvod: %d\n",s);
```

```
  return 0;
}
```

Rešenje 3.2

```
/*
   Napisati program koji ucitava broj elemenata niza (n<=100),
   zatim ucitava elemente niza i ispisuje:
   a) elemente niza koji se nalaze na parnim indeksima
   b) parne elemente niza

*/

#include <stdio.h>
#define MAX 100

int main()
{
  int a[MAX];
  int n;
  int i;


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

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



  for (i=0; i<n; i++)
  {
    printf("a[%d]=",i);
    scanf("%d", &a[i]); /* ucitavamo jedan po jedan element niza */
  }

  printf("Elementi sa parnim indeksima:\n");
  for (i=0; i<n; i+=2)
    printf("a[%d]=%d\n",i,a[i]);

  printf("Parni elementi:\n");
  for (i=0; i<n; i++)
    if (a[i]%2==0)
      printf("a[%d]=%d\n",i,a[i]);


  return 0;
```

```
}
```

Rešenje 3.3

```
/*
   Napisati program koji ucitava jedan ceo broj a zatim ispisuje
    koliko puta koja cifra ucestvuje
   u zapisu tog broja. Nije potrebno ispisivati da se neka cifra
    pojavila 0 puta.

   Na primer, za uneti broj 4611, izlaz treba da bude:

   U zapisu broja 4611, cifra 1 se pojavljuje 2 puta
   U zapisu broja 4611, cifra 4 se pojavljuje 1 puta
   U zapisu broja 4611, cifra 6 se pojavljuje 1 puta

   A za uneti broj -252

   U zapisu broja -252, cifra 2 se pojavljuje 2 puta
   U zapisu broja -252, cifra 5 se pojavljuje 1 puta

*/

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

int main()
{
   int x;
   int brojaci[10];
   char cifra;
   int original;
   int i;

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

   /*
      svaki element niza brojaci predstavlja
      brojac za jednu cifru:
      brojac[0] sadrzi broj nula
      brojac[1] sadrzi broj jedinica
      ...
      brojac[9] sadrzi broj devetki

      brojaci se inicijalizuju na vrednost 0
   */

   for(i=0;i<10;i++)
      brojaci[i]=0;
```

```
47      /*
            vrednost promenljive x ce biti unistena
49          u while petlji jer je u svakom koraku delimo
            sa 10; njenu vrednost cuvamo u promenljivoj
51          original kako bismo mogli da je iskoristimo
            na kraju prilikom ispisa
53      */
        original = x;
55
        /*
57          Uzimamo apsolutnu vrednost broja za slucaj
            da je uneti broj negativan
59      */
        x=abs(x);
61
        /* Izdvajanje cifara broja */
63      do
        {
65          cifra = x%10;
            brojaci[cifra]++;  /* Uvecavamo brojac odgovarajuce 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
    int main()
11  {
      /* Za svaku dekadnu cifru definisemo jedan brojac (tj. imamo niz
```

```
13      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 brojaci se inicijalizuju nulama.
    */
17  int cifre[10];
    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++)
27  {
      mala[i]=0;
29    velika[i]=0;
    }
31
    while((c = getchar()) != EOF)
33  {

35    if (c>='A' && c<='Z')
        velika[c-'A']++;
37    else if (c>='a' && c<='z')
        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
        /*
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 brojaci (jer niz ima 10 elemenata,
     pa su indeksi od 0 do 9). Time postizemo da brojaci[0] broji
47   karaktere '0', itd. Isto vazi i za brojace za mala i velika slova.
        */
49  }

51  /* Prikazujemo elemente niza, tj. vrednosti brojaca: */
    for(i = 0; i < 10; i++)
53    if (cifre[i]!=0)
        printf("Karakter %c se pojavljuje %d puta\n", '0' + i,
55      cifre[i]);

57   for(i = 0; i < 26; i++)
      if (mala[i]!=0)
59      printf("Karakter %c se pojavljuje %d puta\n", 'a' + i,
       mala[i]);
61
```

```
    for(i = 0; i < 26; i++)
63    if (velika[i]!=0)
        printf("Karakter %c se pojavljuje %d puta\n", 'A' + i,
65     velika[i]);

67  return 0;
}
```

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
6   broj elemenata u nizovima a i b 100.
*/
8
#include <stdio.h>
10 #define MAX 100

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

18  int n;
    int i,j;
20
    printf("Unesi dimenziju niza:");
22  scanf("%d", &n);

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


30
    printf("\nUnesi elemente niza a:\n");
32  for(i=0;i<n;i++)
    {
34    printf("a[%d]=",i);
      scanf("%d", &a[i]);
36  }
```

```
38    printf("\nUnesi elemente niza b:\n");
      for(i=0;i<n;i++)
40    {
        printf("b[%d]=",i);
42      scanf("%d", &b[i]);
      }
44
      /*
46      Koristimo dva indeksa:
        1. i, sa kojim pristupamo
48         elementima niza a i b, i koji uvecavamo za 1
           nakon svake iteracije,
50      2  j, sa kojim pristupamo
           elementima niza c; s obzirom da u svakoj
52         iteraciji dodeljujemo vrednost za dva
           elementa niza c (c[j] i c[j+1]), indeks
54         j uvecavamo za 2 nakon svake iteracije
      */
56    for(i=0,j=0;i<n;i++,j+=2)
      {
58      c[j]=a[i];
        c[j+1]=b[i];
60    }

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

66    return 0;
  }
```

Rešenje 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 zadrzati. Program treba da
      ispise
     izmenjeni niz na standardni izlaz. Mozemo pretpostaviti da niz a
7    sadrzi najvise 100 elemenata.
  */
9
  #include <stdio.h>
11 #define MAX 100

13 int main()
  {
15
    int a[MAX];
17
```

```
    int n;
19  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]);
    }
39


41  /*
      Dodatni indeks j se uvecava u slucaju da element na indeksu
43    i treba da ostane u nizu, tj da je deljiv svojim
      indeksom i; u suprotnom, j se nece uvecati i
45    element i ce u narednoj iteraciji biti zamenjen elementom koji
      jeste deljiv svojim indeksom
47  */

49  for(i=0,j=0;i<n;i++)
    {
51    poslednja_cifra = a[i]%10;

53    /*
         zbog lenjog izracunavanja, ako je prvi uslov
55       u disjunkciji tacan, drugi se nece ispitivati
         (jer  ce tada disjunkcija biti tacna bez obzira
57       da li je drugi uslov tacan ili ne)
      */
59    if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
      {
61        a[j]=a[i];
          j++;
63    }
    }
65  /*
      Izbacivanjem elemenata dimenzija niza se menja, odnosno
67    smanjuje se za broj izbacenih elemenata
    */
69  novo_n=j;
```

```
  printf("Nakon izmena:\n");
  for(i=0;i<novo_n;i++)
    printf("a[%d]=%d\n",i,a[i]);

  return 0;
}
```

Rešenje 3.7

```
/*
   a) Napisati funkciju koja ucitava sadrzaj niza.
   b) Napisati funkciju koja stampa sadrzaj niza.
   c) Napisati funkciju koja racuna sumu elemenata niza.
   d) Napisati funkciju koja racuna prosecnu vrednost elemenata niza.
   e) Napisati funkciju koja izracunava minimum elemenata niza.
   f) Napisati funkciju koja izracunava poziciju maksimalnog elementa
     u nizu.
   g) Napisati program koji testira prethodne funkcije.

*/

#include <stdio.h>
#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]);
   }
}

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

/* c) */
int suma(int a[], int n)
{
   int i;
   int s=0;
   for(i=0;i<n;i++)
```

```
        s+=a[i];
42    return s;
}
44

46  /* d) */
float prosek(int a[], int n)
48  {
    int i;
50  int s = suma(a,n);
    return (float) s/n;
52  }

54
    /* e) */
56  int minimum (int a[],int n)
    {
58  int m;
    int i;
60  m = a[0];

62  /*
        minimum inicijalizujemo na prvi element niza (a[0])
64      u svakom koraku poredimo vrednost minimuma
        sa jednim elementom niza, iduci redom; s obzirom
66      da je minimum inicijalizovan na a[0], nema potrebe
        porediti a[0] sa a[0] i zbog toga indeksiranje krece
68      od 1
    */
70
    for(i=1;i<n;i++)
72      if (m>a[i])
            m = a[i];
74
    return m;
76  }

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

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

```
            m_poz=i;
94        }

96    return m_poz;
}
98



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

108 if (n<1 || n>MAX)
    {
110    printf("Nekorektan unos\n");
       return -1;
112 }

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

118 printf("Suma elemenata niza: %d\n", suma(a,n));
    printf("Prosecna vrednost elemenata niza: %.2f\n", prosek(a,n));
120 printf("Minimumalni element niza: %d\n", minimum(a,n));
    printf("Indeks maksimalnog elementa niza: %d\n", max_pozicija(a,n)
     );
122
    return 0;
124 }
```

Rešenje 3.8

```
/*
2    a) Napisati funkciju koja ucitava sadrzaj niza.
     b) Napisati funkciju koja stampa sadrzaj niza.
4    c) Napisati funkciju koja proverava da li niz sadrzi neku vrednost
       m.
     d) Napisati funkciju koja vraca vrednost prve pozicije na kojoj se
6 nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
     nizu.
     e) Napisati funkciju koja vraca vrednost poslednje pozicije na
     kojoj se
8 nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
     nizu.
     f) Napisati funkciju koja proverava da li elementi niza cine
     palindrom.
```

```
   g) Napisati funkciju koja proverava da li su elementi niza
    uredjeni
neopadajuce.
   h) Napisati funkciju koja izracunava najduzu uzastopnu seriju
    jednakih
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.
   i) Napisati program koji testira prethodne funkcije.
*/
#include <stdio.h>
#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]);
   }
}

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


/* c) */
int sadrzi(int a[], int n, int m)
{
   int i;
   /*
     poredimo jedan po jedan element niza a sa datim m; ukoliko
     ustanovimo jednakost, to znaci da niz sadrzi element jednak
     m i vracamo 1
   */
   for(i=0;i<n;i++)
      if (a[i]==m)
         return 1;

   /*
     ukoliko se petlja zavrsi a uslov a[i]==m nijednom nije bio
    ispunjen,
     to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
    vrati 0
```

```
   */
58 return 0;
}
60
/* d) */
62 int prvo_pojavljivanje(int a[], int n, int m)
{
64 int i;
   /*
66   poredimo jedan po jedan element niza a sa datim m; ukoliko
     ustanovimo jednakost, vracamo indeks elementa niza a koji
68   je jednak sa m
   */
70 for(i=0;i<n;i++)
      if (a[i]==m)
72       return i;

74 /*
     ukoliko se petlja zavrsi a uslov a[i]==m nijednom nije bio
    ispunjen,
76   to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
    vrati -1
   */
78 return -1;
}
80
/* e) */
82 int poslednje_pojavljivanje(int a[], int n, int m)
{
84 int i;
   /*
86   krecemo od indeksa poslednjeg elementa, n-1
   */
88 for(i=n-1;i>=0;i--)
      if (a[i]==m)
90       return i;

92 return -1;
}
94
/* f) */
96 int palindrom(int a[], int n)
{
98
   int i,j;
100
   /*
102  uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
     uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
104  .
     .
106  i tako redom dok je prva pozicija manja od druge
```

```
    */
    for(i=0,j=n-1;i<j;i++,j--)
      if(a[i]!=a[j])
         return 0;

    return 1;
}

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

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

    Sortiranost proveravamo na sledeci nacin: za svaki par susednih
     elemenata
    a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
     proveravamo
    da li vazi da je drugi clan para manji od prvog. Ako naidjemo na
     par za koji
    to ne vazi, niz nije sortiran i funkcija vraca 0. Ukoliko se
     petlja zavrsi
    a da pritom uslov a[i]<a[i-1] nije nijednom bio ispunjen, to znaci
      da je
    niz sortiran i funkcija vraca 1

    */
    for(i=1; i<n; i++)
       if (a[i]<a[i-1])
          return 0;

    return 1;
}

/* h) */
int najduza_konstanta(int a[], int n)
{
    int i; /* indeks niza */
    int j; /* duzina intervala */
    int duzina;
    int max_duzina=0;


    for(i=0,j=0;i<n-1;i++)
    {
       if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
       {
          j++;          /* uvecavamo duzinu konstantnog intervala */
```

```
154          /*
               ako se niz zavrsava konstantnim intervalom (nalazimo se u
     poslednjoj
156            iteraciji petlje i tada je i==n-2), ispitujemo da li je
     taj konstantni
               interval maksimalne duzine
158          */
             if(i==n-2)
160          {
                j++;
162             if(j>max_duzina)
                   max_duzina=j;
164          }
           }
166        else
           {
168           /*
                izasli smo iz konstantnog intervala
170
                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++;

182          /* 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()
{
   int a[MAX];
   int n;
   int m;
   int i;

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

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

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

   printf("Unesi jedan ceo broj:");
   scanf("%d",&m);


   if(sadrzi(a,n,m))
      printf("Niz sadrzi element cija je vrednost %d\n", m);
   else
      printf("Niz ne sadrzi element cija je vrednost %d\n", m);

   i = prvo_pojavljivanje(a,n,m);
   if(i!=-1)
      printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
     prvog pojavljivanja u nizu je %d\n", m,i);
   else
      printf("Niz ne sadrzi element cija je vrednost %d\n", m);


   i = poslednje_pojavljivanje(a,n,m);
   if(i!=-1)
      printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
     poslednjeg pojavljivanja u nizu je %d\n", m,i);
   else
      printf("Niz ne sadrzi element cija je vrednost %d\n", m);

   if(palindrom(a,n))
      printf("Elementi niza cine palindrom\n");
   else
      printf("Elementi niza ne cine palindrom\n");

   if(neopadajuci(a,n))
      printf("Niz je sortiran neopadajuce\n");
   else
```

```
        printf("Niz nije sortiran neopadajuce\n");
252
254     printf("Duzina najduzeg konstantnog intervala: %d\n",
         najduza_konstanta(a,n));
256     return 0;
}
```

Rešenje 3.9

```
1  /*
       a) Napisati funkciju koja ucitava sadrzaj niza.
3      b) Napisati funkciju koja stampa sadrzaj niza.
       c) Napisati funkciju koja proverava da li niz sadrzi neku vrednost
          m.
5      d) Napisati funkciju koja vraca vrednost prve pozicije na kojoj se
   nalazi element koji ima vrednost m, ili -1 ukoliko element nije u
         nizu.
7      e) Napisati funkciju koja vraca 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 neopadajuce.
       h) Napisati funkciju koja izracunava najduzu uzastopnu seriju
         jednakih
13 elemenata u nizu. Na primer, za uneti niz 1 2 3 4 4 4 5 6 7 8 9 9
         funkcija
   treba da vrati 3.
15     i) Napisati program koji testira prethodne funkcije.
   */
17 #include <stdio.h>
   #define MAX 100
19
   /* a) */
21 void ucitaj(int a[], int n)
   {
23     int i;
       for(i=0;i<n;i++)
25     {
           printf("Unesi element na poziciji %d:",i);
27         scanf("%d",&a[i]);
       }
29 }

31 /* b) */
   void stampaj(int a[], int n)
```

```
{
   int i;
   for(i=0;i<n;i++)
      printf("%d\t",a[i]);
   printf("\n");
}


/* c) */
int sadrzi(int a[], int n, int m)
{
   int i;
   /*
     poredimo jedan po jedan element niza a sa datim m; ukoliko
     ustanovimo jednakost, to znaci da niz sadrzi element jednak
     m i vracamo 1
   */
   for(i=0;i<n;i++)
      if (a[i]==m)
         return 1;

   /*
     ukoliko se petlja zavrsi a uslov a[i]==m nijednom nije bio
    ispunjen,
     to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
    vrati 0
   */
   return 0;
}

/* d) */
int prvo_pojavljivanje(int a[], int n, int m)
{
   int i;
   /*
     poredimo jedan po jedan element niza a sa datim m; ukoliko
     ustanovimo jednakost, vracamo indeks elementa niza a koji
     je jednak sa m
   */
   for(i=0;i<n;i++)
      if (a[i]==m)
         return i;

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

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

   return -1;
}

/* f) */
int palindrom(int a[], int n)
{

   int i,j;

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

   return 1;
}

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

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

   Sortiranost proveravamo na sledeci nacin: za svaki par susednih
    elemenata
   a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
    proveravamo
   da li vazi da je drugi clan para manji od prvog. Ako naidjemo na
    par za koji
   to ne vazi, niz nije sortiran i funkcija vraca 0. Ukoliko se
    petlja zavrsi
```

```
   a da pritom uslov a[i]<a[i-1] nije nijednom bio ispunjen, to znaci
     da je
129    niz sortiran i funkcija vraca 1

131    */
       for(i=1; i<n; i++)
133       if (a[i]<a[i-1])
             return 0;
135
       return 1;
137 }

139 /* h) */
    int najduza_konstanta(int a[], int n)
141 {
       int i; /* indeks niza */
143    int j; /* duzina intervala */
       int duzina;
145    int max_duzina=0;


147
       for(i=0,j=0;i<n-1;i++)
149    {
          if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
151       {
             j++;          /* uvecavamo duzinu konstantnog intervala */
153
             /*
155            ako se niz zavrsava konstantnim intervalom (nalazimo se u
     poslednjoj
               iteraciji petlje i tada je i==n-2), ispitujemo da li je
     taj konstantni
157            interval maksimalne duzine
             */
159          if(i==n-2)
             {
161             j++;
                if(j>max_duzina)
163                max_duzina=j;
             }
165       }
          else
167       {
             /*
169             izasli smo iz konstantnog intervala

171             ukoliko smo imali bar dva elementa u konstantnom
     intervalu,
                vrednost promenljive j ce biti 1, a duzina tog intervala
     je 2;
173             zbog toga je neophodno takve (pozitivne) j uvecati za 1;
```

```
175             sa druge strane, ako su a[i] i a[i+1] razliciti,
                duzina tog intervala je 0
177         */

179         if (j>0)
                j++;
181
            /* azuriramo maksimalnu duzinu uspona */
183         if(j>max_duzina)
                max_duzina=j;
185         /*
                duzina uspona se postavlja na nulu
187             kako bi mogli da je iskoristimo
                za naredni uspon
189         */
            j=0;
191
        }
193


195     }

197     return max_duzina;
}
199

201 int main()
{
203     int a[MAX];
        int n;
205     int m;
        int i;
207
        printf("Unesi dimenziju niza:");
209     scanf("%d",&n);

211     if (n<1 || n>MAX)
        {
213         printf("Nekorektan unos\n");
            return -1;
215     }

217     ucitaj(a,n);
        printf("Ucitani niz:");
219     stampaj(a,n);

221     printf("Unesi jedan ceo broj:");
        scanf("%d",&m);
223


225     if(sadrzi(a,n,m))
            printf("Niz sadrzi element cija je vrednost %d\n", m);
```

```
227     else
            printf("Niz ne sadrzi element cija je vrednost %d\n", m);
229
        i = prvo_pojavljivanje(a,n,m);
231     if(i!=-1)
            printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
           prvog pojavljivanja u nizu je %d\n", m,i);
233     else
            printf("Niz ne sadrzi element cija je vrednost %d\n", m);
235

237     i = poslednje_pojavljivanje(a,n,m);
        if(i!=-1)
239         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
           poslednjeg pojavljivanja u nizu je %d\n", m,i);
        else
241         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

243     if(palindrom(a,n))
            printf("Elementi niza cine palindrom\n");
245     else
            printf("Elementi niza ne cine palindrom\n");
247
        if(neopadajuci(a,n))
249         printf("Niz je sortiran neopadajuce\n");
        else
251         printf("Niz nije sortiran neopadajuce\n");

253
        printf("Duzina najduzeg konstantnog intervala: %d\n",
          najduza_konstanta(a,n));
255
        return 0;
257 }
```

Rešenje 3.10

```
1  /*
     a) Napisati funkciju koja sve vrednosti niza uvecava za vrednost m.
3    b) Napisati funkciju koja obrce vrednosti elementima niza.
     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.
     e) Napisati program koji testira prethodne funkcije.
7
     Napisati potom glavni program koji testira ovu funkciju.
9  */

11 #include<stdio.h>
   #define MAX 100
13
```

```
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]);
   }
}

void stampaj(int a[], int n)
{
   int i;
   for(i=0;i<n;i++)
        printf("%d\t",a[i]);
   printf("\n");
}


void uvecaj(int a[], int n, int m)
{
   int i;
   for(i=0;i<n;i++)
        a[i]+=m;
}


void obrni(int a[], int n)
{

   int t;
   int i,j;
   /*
    Niz obrcemo tako sto razmenimo vrednosti elemenata na pozicijama
    0 i n-1,
    zatom 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
     druge
   */

   for(i=0,j=n-1;i<j;i++, j--)
   {
        t = a[i];
        a[i] = a[j];
        a[j] = t;
   }

}

void rotiraj1(int a[], int n)
{
   int i;
   int tmp;
```

```
    tmp=a[0]; /* izdvajamo prvi element */
    for(i=0;i<n-1;i++)
         a[i]=a[i+1]; /* pomeramo preostale elemente */
    a[n-1] = tmp;  /* poslednjem elementu dodeljujemo
                      sacuvanu vrednost prvog elementa */
}

void rotirajk(int a[], int n, int k)
{
    int i;
    /*
       k puta rotiramo niz za jednu poziciju
       ulevo
    */
    for(i=0;i<k;i++)
         rotiraj1(a,n);
}

int main()
{
  int a[MAX];
  int n;
  int i;
  int k;
  int m;

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

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

  ucitaj(a,n);

  printf("Unesi jedan ceo broj:");
  scanf("%d", &m);

  uvecaj(a,n,m);
  printf("Elementi niza nakon uvecanja za %d:\n",m);
  stampaj(a,n);

  obrni(a,n);
  printf("Elementi niza nakon obrtanja:\n");
  stampaj(a,n);

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

  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 3.114

```
1  #include <stdio.h>

3  #define MAX 100

5  int main()
   {
7    float brojevi[MAX];
     int n, i;
9
     printf("Unesite broj elemenata niza: ");
11   scanf("%d", &n);

13   if(n<1 || n>100)
     {
15     printf("Greska: pogresan unos!\n");
     }else{
17
     printf("Unesite elemente niza:\n");
19   for(i=0;i<n;i++)
       scanf("%f", &brojevi[i]);
21
     /*
23     Ukoliko je i element niza brojevi[i] negativan broj,
       kvadriramo ga tako sto ga pomnozimo sa samim sobom.
25   */

27   for(i=0;i<n;i++)
       if(brojevi[i]<0)
29       brojevi[i] *= brojevi[i];

31   /* Ispisujemo sve elemente niza. */

33   for(i=0;i<n;i++)
```

```
    printf("%g ", brojevi[i]);
}
  return 0;
}
```

Rešenje 3.115

```
#include <stdio.h>

#define MAX 100

int main()
{
  int brojevi[MAX];
  int n, i, k, indikator;

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

  if(n<1 || n>100)
  {
    printf("Greska: pogresan unos!\n");
  }
  else{
  printf("Unesite elemente niza: ");

  for(i=0;i<n;i++)
    scanf("%d", &brojevi[i]);

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

  if(k == 0)
  {
    printf("Greska: pogresan unos!\n");
  }
  else{
  /*
     Promenljiva koja nam cuva informaciju o tome
     da li je u nizu postojao element koji je deljiv brojem k.
     Inicijalno je postavimo na nulu.
  */

  indikator = 0;

  /*
    Ukoliko je element niza deljiv brojem k, postavljamo indikator na
     1
    i ispisujemo indeks tog elementa.
  */

```

```
  for(i=0;i<n;i++)
    if(brojevi[i]%k == 0)
      {
        indikator = 1;
        printf("%d ",i);
      }

  /*
    Ukoliko je indikator jednak nuli to znaci da ne postoji element u
     nizu koji je deljiv brojem k.
  */

  if(indikator == 0)
    printf("U nizu nema elemenata koji su deljivi brojem %d!\n",k);
  }
  }
  return 0;
}
```

Rešenje 3.116

```
#include <stdio.h>

#define MAX 100

int main()
{
  int brojevi[MAX];
  int n, i, poz_max, poz_min, max, min, tmp;

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

  if(n<1 || n>100)
  {
    printf("Greska: pogresan unos!\n");
    return 0;
  }

  printf("Unesite elemente niza:\n");
  for(i=0;i<n;i++)
    scanf("%d", &brojevi[i]);

  /*
    Maksimum tj. minimum pre ulaska u petlju postavimo da budu prvi
    element niza.
    Pozicije maksimuma tj. minimuma postavimo na 0.
  */
  max = brojevi[0];
  min = brojevi[0];
  poz_max = 0;
```

```
  poz_min = 0;

  /*
    Pronadjemo maksimalni tj. minimalni element tako sto u petlji
    prodjemo kroz sve elemente i ukoliko naletimo na element veci od
    maksimuma
    tj. manji od minimuma, promenimo tako da sada maksimum tj.
    minimum budu taj element
    i promenimo njihove pozicije.
  */
  for(i=1;i<n;i++)
  {
    if(brojevi[i] > max)
    {
      max = brojevi[i];
      poz_max = i;
    }

    if(brojevi[i] < min)
    {
      min = brojevi[i];
      poz_min = i;
    }
  }

  /*
    Zamenimo minimalni i maksimalni element na pozicijama poz_min i
    poz_max.
    Koristimo pomocnu promenljivu tmp kako bismo sacuvali vrednost
    maksimalnog elementa.
  */
  tmp = max;
  brojevi[poz_max] = min;
  brojevi[poz_min] = tmp;

  for(i=0;i<n;i++)
    printf("%d ", brojevi[i]);

  return 0;
}
```

Rešenje 3.117

```
#include <stdio.h>

#define MAX 100

int main()
{
  char karakteri[MAX];
  char c;
```

```
  int i, n;


  for(i=0;i<MAX;i++)
  {
    /*
      Ucitavamo karakter po karakter dok ne unesemo * ili ne
    prekoracimo 100 karaktera
      i upisujemo ih u niz.
    */
    printf("Unesite karakter: ");
    scanf("%c", &c);

    /*
      Citamo belinu nakon unesenog karaktera.
    */
    getchar();

    /*
      Ukoliko smo uneli * izlazimo iz petlje
    */
    if(c == '*')
      break;

    /*
      Stavljamo karakter u niz.
    */
    karakteri[i] = c;
  }

  /*
    Broj unetih karaktera je nakon prolaska kroz petlju i-1.
  */
  n = i-1;

  /*
    Ispisujemo karaktere u obrnutom redosledu.
  */
  for(i=n;i>=0;i--)
  {
    printf("%c ", karakteri[i]);
  }

  return 0;
}
```

Rešenje 3.118

```
#include <stdio.h>

int main()
```

```
{
  char c;
  int cifrex[10], cifrey[10];
  int x, y, i, indikator;

  printf("Unesite dva broja: ");
  scanf("%d%d", &x, &y);

  /*
    Uzmemo apsolutnu vrednost brojeva za slucaj da su negativni.
  */
  x=abs(x);
  y=abs(y);

  /*
    Niz cifrex nam predstavlja brojace za cifre broja x, na pocetku
    ga inicijalizujemo na 0.
    Analogno za cifrey.
  */
  for(i=0;i<10;i++)
  {
      cifrex[i] = 0;
      cifrey[i] = 0;
  }

  /*
    Skidamo jednu po jednu cifru broja x i povecavamo njen brojac u
    nizu cifrex.
  */
  while(x)
  {
    c = x%10;
    cifrex[c]++;
    x /= 10;
  }

  /*
    Isto radimo i za broj y.
  */
  while(y)
  {
    c = y%10;
    cifrey[c]++;
    y /= 10;
  }

  /*
    Promenljiva koja nam sluzi za proveru da li su oba broja
    sastavljena od istih cifara.
    Pretpostavicemo da jesu i postaviti indikator na 1.
    Nakon toga u petlji prolazimo kroz nizove cifrex i cifrey u
    kojima se nalaze
```

```
    brojevi pojavljivanja svih cifri 0-9 u broju x i y, i prvi put
    kada naletimo na
    neku cifru koja se ne pojavljuje isti broj puta u oba broja x i y
    ,
    postavljamo promenljivu indikator na 0 (brojevi x i y nisu
    zapisani sa istim ciframa)
    i izlazimo iz petlje.
  */
  indikator = 1;
  for(i=0;i<10;i++)
    if(cifrey[i] != cifrex[i])
    {
     indikator = 0;
     break;
    }

  /*
    Ako je promenljiva indikator ostala 1, to znaci da u petlji nismo
     pronasli cifru
    koja se ne pojavljuje isti broj puta u brojevima x i y, sto znaci
     da se oni zapisuju istim ciframa.
  */
  if(indikator)
    printf("Brojevi se zapisuju istim ciframa!\n");
  else
    printf("Brojevi se ne zapisuju istim ciframa!\n");

  return 0;
}
```

Rešenje 3.119

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX], b[MAX], c[2*MAX];
  int i, n;

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

  if(n<1 || n>100)
  {
    printf("Greska: pogresan unos!\n");
    return 0;
  }

  printf("Unesite elemente niza a: ");
```

```
  for(i=0;i<n;i++)
    scanf("%d", &a[i]);

  printf("Unesite elemente niza b: ");
  for(i=0;i<n;i++)
    scanf("%d", &b[i]);

  /*
    Niz c ima 2*n elemenata. Prvih n elemenata niza b, i nakon toga n
     elemenata niza a.
    Elementi iz niza a se nalaze na pozicijama 0,1,2,...n-1, a
    elementi niza b na pozicijama
    n,n+1,...2*n. Jednim prolaskom kroz petlju na poziciju i u nizu c
     stavljamo element niza b - b[i],
    a na poziciju n+i element niza a - a[i].
  */
  for(i=0;i<n;i++)
  {
    c[i] = b[i];
    c[n+i] = a[i];
  }

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

  return 0;
}
```

Rešenje 3.17

```
#include <stdio.h>

#define MAX 100

/*
  Funkcija koja vraca broj pojavljivanja broja x u nizu.
*/

int broj_pojavljivanja(int niz[], int n, int x)
{
  int i, rezultat = 0;

  /*
    Kada naidjemo na element niza koji je jednak broju x, povecamo
    brojac rezultat.
  */
  for(i=0;i<n;i++)
    if(niz[i] == x)
      rezultat++;

  return rezultat;
```

```
}

int main()
{
  int a[MAX], b[MAX];
  int i, j, n, n_b;

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

  if(n<1 || n>100)
  {
    printf("Greska: pogresan unos!\n");
    return -1;
  }

  printf("Unesite elemente niza a: ");
  for(i=0;i<n;i++)
    scanf("%d", &a[i]);

  /*
    Brojac elemenata rezultujuceg niza b.
  */
  j = 0;
  for(i=0;i<n;i++)
  {
    /*
      Ukoliko se element niza pojavljuje tacno tri puta i ne postoji
    u nizu b koji trenutno ima j elemenata
      (nismo ga jos uvek dodali) dodajemo ga u niz b i povecavamo
    brojac j.
    */
    if(broj_pojavljivanja(a, n, a[i])==3 && broj_pojavljivanja(b, j,
    a[i])==0)
    {
      b[j] = a[i];
      j++;
    }
  }
  /*
    Broj elemenata u nizu b je j.
  */
  n_b = j;

  for(i=0;i<n_b;i++)
    printf("%d ", b[i]);

  return 0;
}
```

Rešenje 3.18

```c
#include <stdio.h>

#define MAX 100

/*
  Funkcija koja vraca 1 ukoliko broj x postoji u nizu, 0 inace.
*/

int postoji(int niz[], int n, int x)
{
  int i;

  for(i=0;i<n;i++)
    if(niz[i] == x)
      return 1;

  return 0;
}

int main()
{
  int a[MAX], b[MAX], unija[2*MAX], presek[MAX], razlika[MAX];
  int i, j, n_a, n_b, n_u, n_p, n_r, indikator;

  printf("Unesite broj elemenata niza a: ");
  scanf("%d", &n_a);

  if(n_a<1 || n_a>100)
  {
    printf("Greska: pogresan unos!\n");
    return -1;
  }

  printf("Unesite elemente niza a: ");
  for(i=0;i<n_a;i++)
    scanf("%d", &a[i]);

  printf("Unesite broj elemenata niza b: ");
  scanf("%d", &n_b);

  if(n_b<1 || n_b>100)
  {
    printf("Greska: pogresan unos!\n");
    return -1;
  }

  printf("Unesite elemente niza b: ");
  for(i=0;i<n_b;i++)
    scanf("%d", &b[i]);

  /*
```

```
52      Brojaci elemenata u nizovima unija, presek i razlika.
    */
54  n_u = 0;
    n_p = 0;
56  n_r = 0;

58  for(i=0;i<n_a;i++)
    {
60    /*
        Ukoliko se element a[i] ne nalazi u uniji, dodajemo ga u uniju
      i povecamo brojac elemenata u nizu unija.
62    */
      if(postoji(unija,n_u,a[i]) == 0)
64    {
        unija[n_u] = a[i];
66      n_u++;
      }
68
      /*
70      Ukoliko se element a[i] postoji u nizu b i ne postoji u nizu
      presek, dodajemo ga u presek i povecavamo brojac elemenata u nizu
       presek.
      */
72    if(postoji(b, n_b, a[i])==1 && postoji(presek, n_p, a[i])==0)
      {
74      presek[n_p] = a[i];
        n_p++;
76    }

78    /*
        Ukoliko element a[i] ne postoji u nizu b i ne postoji u nizu
      razlika, dodajemo ga u razliku i povecavamo brojac elemenata u
      nizu razlika.
80    */
      if(postoji(b, n_b, a[i])==0 && postoji(razlika, n_r, a[i])==0)
82    {
        razlika[n_r] = a[i];
84      n_r++;
      }
86  }

88  /*
      Elemente niza b koji ne postoje u uniji dodajemo u uniju.
90  */
    for(i=0;i<n_b;i++)
92    if(postoji(unija, n_u, b[i]))
      {
94      unija[n_u] = b[i];
        n_u++;
96    }

98  printf("Unija: ");
```

```
  for(i=0;i<n_u;i++)
    printf("%d ", unija[i]);

  printf("\nPresek: ");
  for(i=0;i<n_p;i++)
    printf("%d ", presek[i]);

  printf("\nRazlika: ");
  for(i=0;i<n_r;i++)
    printf("%d ", razlika[i]);

  return 0;
}
```

Rešenje 3.19

```
#include <stdio.h>

#define MAX 100

int main()
{
  int a[MAX], b[MAX];
  int i, j, n_a, n_b;

  printf("Unesite broj elemenata niza: ");
  scanf("%d", &n_a);

  if(n_a<1 || n_a>100)
  {
    printf("Greska: pogresan unos!\n");
    return -1;
  }

  printf("Unesite elemente niza: ");
  for(i=0;i<n_a;i++)
    scanf("%d", &a[i]);

  /*
   1. nacin

   J nam predstavlja brojac prve slobodne pozicije na koju mozemo
    upisati element niza koji treba da ostane u nizu.
   Kada naletimo na element koji je paran, kopiramo ga na mesto a[j]
    i povecamo brojac j.
   Ukoliko naletimo na element koji je neparan, njega samo preskocimo
    .
  */

  for(i=0, j=0;i<n_a;i++)
  {
```

```
    if(a[i]%2 == 0)
    {
      a[j] = a[i];
      j++;
    }
  }

  /*
    Na pozicijama od 0...j-1 se sada nalaze elementi koji su parni,
    te je nova dimenzija niza sada j.
  */
  n_a=j;

  for(i=0;i<n_a;i++)
    printf("%d ", a[i]);

  /*
    2. nacin

    Kada naletimo na element niza koji je paran, kopiramo ga u niz b
    i povecamo j - brojac elemenata niza b.

  for(i=0, j=0;i<n_a;i++)
    if(a[i]%2 == 0)
    {
      b[j] = a[i];
      j++;
    }

  n_b = j;

  for(i=0;i<n_b;i++)
    printf("%d ", b[i]);

  */
  return 0;
}
```

Rešenje 3.20

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

#define MAX 100

/*
  Funkcija koja proverava da li je broj prost.
  Vraca 1 ukoliko broj jeste prost, inace 0.
*/
int prost(int x)
{
```

```
12    int i;
14    if(x == 2 || x == 3)
        return 1;
16
      if(x%2 == 0)
18      return 0;
20    for(i=3;i<=sqrt(x);i+=2)
        if(x%i == 0)
22        return 0;
24    return 1;
  }
26
  int main()
28{
    int a[MAX], b[MAX];
30  int i, j, n_a, n_b;
32  printf("Unesite broj elemenata niza: ");
    scanf("%d", &n_a);
34
    if(n_a<1 || n_a>100)
36  {
      printf("Greska: pogresan unos!\n");
38    return -1;
    }
40
    printf("Unesite elemente niza: ");
42  for(i=0;i<n_a;i++)
      scanf("%d", &a[i]);
44
    /*
46    1. nacin

48    J nam predstavlja brojac prve slobodne pozicije na koju mozemo
      upisati element niza koji treba da ostane u nizu.
      Kada naletimo na element koji nije prost, kopiramo ga na mesto a[
      j] i povecamo brojac j.
50    Ukoliko naletimo na element koji je prost, njega samo preskocimo.

52  */

54  for(i=0, j=0;i<n_a;i++)
    {
56    if(prost(a[i]) == 0)
      {
58      a[j] = a[i];
        j++;
60    }
    }
```

```
62
   n_a=j;
64
   for(i=0;i<n_a;i++)
66   printf("%d ", a[i]);

68 /*
     2. nacin
70
     Prolazimo kroz niz a i svaki broj koji nije prost kopiramo u niz
     b i povecamo j - brojac elemenata u nizu b.
72
   for(i=0, j=0;i<n_a;i++)
74   if(prost(a[i]) == 0)
     {
76     b[j] = a[i];
       j++;
78   }

80 n_b = j;

82 for(i=0;i<n_b;i++)
     printf("%d ", b[i]);
84 */
   return 0;
86 }
```

Rešenje 3.21

```
/*
2  Napisati funkciju int prebrojavanje(int a[], int n) koja izracunava
       broj elemenata niza celih brojeva a duzine n
koji su manji od poslednjeg elemenata niza. Napisati i program koji
     testira rad funkcije. Pretpostaviti da duzina
4 niza nece biti veca od 100.

6  */

8 #include <stdio.h>
  #define MAX 100
10
  /*
12 * Funkcija prebrojavanje vraca broj elemenata niza koji su manji od
       poslednjeg
   * NAPOMENA: Poslednji element niza se nalazi na poziciji n-1
14 */
  int prebrojavanje(int a[], int n)
16 {
     int i;
18   /*Inicijalizujemo brojac na 0*/
     int br=0;
```

```
20
   /*
22  * Petljom prolazimo kroz sve clanove niza,
    * poredimo ih sa poslednjim elementom i
24  * ukoliko su manji, uvecavamo brojac
    */
26 for(i=0;i<n-1;i++){
     if(a[i]<a[n-1]){
28     br++;
   }
30 }

32 /*Vracamo izracunatu vrednost*/
   return br;
34 }

36 int main()
   {
38   int a[MAX];
     int n;
40   int i;

42   printf("Unesite broj elemenata niza:");
     scanf("%d", &n);
44
     /*Provera korektnosti ulaznih podataka*/
46   if(n<=0 || n>100)
     {
48      printf("Greska: pogresan unos!\n");
        return 0;
50   }

52   /*Ucitavanje niza*/
     printf("Unesite elemente niza:");
54   for(i=0;i<n;i++)
      scanf("%d",&a[i]);
56
     /*Ispis rezultata*/
58   printf("%d\n", prebrojavanje(a,n));

60   return 0;
   }
```

Rešenje 3.22

```
/*
2 Napisati funkciju int prebrojavanje(int a[], int n) koja izracunava
      broj parnih elemenata niza celih brojeva a
duzine n koji prethode maksimalnom elementu niza. Napisati i program
      koji testira rad funkcije. Pretpostaviti
4 da duzina niza nece biti veca od 100.
```

```
 */

#include <stdio.h>
#define MAX 100

/*Funkcija koja vraca broj parnih elemenata niza koji se nalaze
    ispred najveceg elementa u nizu
 * Ideja je da prvo pronadjemo najveci element niza i njegovu
    poziciju, a zatim da jos jednom
 * prodjemo kroz niz sa ciljem da nadjemo sve parne brojeve koje
    prethode maksimalnom.
 */
int prebrojavanje(int a[], int n)
{
  int i;
  int max;
  int max_ind;
  int br = 0;

  /*Na pocetku postavljamo da je maksimalni element a[0] i da je
    odgovarajuca pozicija 0*/
  max = a[0];
  max_ind=0;

  /*Pronalazimo maksimum niza i pamtimo i vrednost i poziciju*/
  for(i=1;i<n-1;i++)
    if(a[i]>max)
    {
      max = a[i];
      max_ind = i;
    }

  /* Krecemo od pocetka niza i idemo do pozicije na kojoj se nalazi
    najveci element
   * i pronalazimo sve parne brojeve
   */
  for(i=0;i<max_ind;i++)
    if(a[i]%2==0)
      br++;

  return br;
}

int main()
{
  int a[MAX];
  int n;
  int i;

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

```
52    /*Vrsimo proveru korektnosti ulaza*/
      if(n<=0 || n>100)
54    {
          printf("Greska: pogresan unos!\n");
56        return 0;
      }
58
      /*Ucitavamo elemente niza*/
60    printf("Unesite elemente niza:");
      for(i=0;i<n;i++)
62     scanf("%d",&a[i]);

64    /*Ispisujemo rezultat*/
      printf("%d\n", prebrojavanje(a,n));
66    return 0;
  }
```

Rešenje 3.23

```
1  /*
     Napisati funkciju int prebrojavanje_cifre(char s[], int n) koja
       izracunava broj cifara u nizu karaktera a duzine n.
3    Napisati i program koji testira rad funkcije. Pretpostaviti da
       duzina niza nece biti veca od 100.
   */
5
   #include <stdio.h>
7  #include <ctype.h>
   #define MAX 100
9
   /* Funkcija koja prebrojava koliko ima cifara u datom nizu karaktera
       */
11 int prebrojavanje(char a[], int n)
   {
13   int i;
     int br = 0;
15
     /*Prolazimo kroz niz i proveravamo da li je trenutni karakter cifra
         i ukoliko jeste, uvecavamo brojac*/
17   /*Funkcija isdigit vraca 1 ukoliko je prosledjeni karakter cifra, a
         0 u suprotnom
     i nalazi se u zaglavlju ctype.h
19    */
     for(i=0;i<n;i++)
21     if(isdigit(a[i]))
         br++;
23
     return br;
25 }

27 int main()
```

```
{
29   char a[MAX];
     int n;
31   int i;

33   printf("Unesite broj elemenata niza:");
     scanf("%d", &n);
35
     /*Vrsimo proveru korektnosti ulaza*/
37   if(n<=0 || n>100)
     {
39      printf("Greska: pogresan unos!\n");
        return 0;
41   }

43   /*Ucitavamo elemente niza*/
     printf("Unesite elemente niza:");
45   for(i=0;i<n;i++)
     {
47   /*Kako su elementi niza karakteri, neophodno je da u svakoj
       iteraciji preskocimo karakter koji oznacava belinu ili novi red*/
      getchar();
49    /*A da zatim ucitamo sam karakter u niz*/
      scanf("%c",&a[i]);
51   }

53   /*Ispisujemo rezultat*/
     printf("Broj cifara je: %d\n", prebrojavanje(a,n));
55   return 0;
}
```

Rešenje 3.24

```
1  /*
    Napisati funkciju int zbir(int a[], int n, int i, int j) koja racuna
        zbir elemenata niza celih brojeva a duzine n od pozicije i do
       pozicije j.
3   Napisati i program koji testira rad funkcije. Pretpostaviti da
       duzina niza nece biti veca od 100.

5   */

7  #include<stdio.h>
   #define MAX 100
9
   /*Funkcija koja vraca zbir elemenata koji se nalaze izmedju pozicija
       i i j*/
11 int zbir(int a[], int n, int i, int j){
     /*Na pocetku incijalizujemo sumu na 0*/
13   int k, s=0;
```

```
15    /*Krecemo od pozicije i i idemo do pozicije j i dodajemo na sumu
        tekuci element niza*/
      for(k=i; k<=j; k++)
17    s+=a[k];

19    /*Na kraju vracamo izracunatu sumu*/
      return s;
21  }

23  int main(){

25    int n, i, j;
      int a[MAX];
27
      printf("Unesite broj elemenata niza: ");
29    scanf("%d", &n);

31    /*Proveravamo korektnost ulaza*/
      if(n <=0 || n>100)
33    {
      printf("Greska: pogresan unos!\n");
35    return 0;
      }
37
      /*Ucitavamo elemente niza*/
39    printf("Unesite elemente niza:");
      for(i=0; i<n; i++)
41    scanf("%d", &a[i]);

43    /*Ucitavamo interval [i,j]*/
      printf("Unesite vrednosti za i i j: ");
45    scanf("%d%d", &i, &j);

47    /*Proveravamo korektnost zadatog intervala */
      if(i > n-1 || j > n-1 || i > j){
49    printf("Greska: nekorektne vrednosti granica!\n");
      return 0;
51    }

53    /*Ispisujemo rezultat*/
      printf("Zbir je: %d", zbir(a,n,i,j));
55
      return 0;
57  }
```

Rešenje 3.25

```
/*
2   Napisati funkciju float zbir_pozitivnih(float a[], int n, int k)
      koja izracunava zbir prvih k pozitivnih elemenata realnog niza a
      duzine n.
```

```
   Napisati i program koji testira rad funkcije. Pretpostaviti da
     duzina niza nece biti veca od 100.
 */

#include<stdio.h>
#define MAX 100

/*Funckija racuna zbir prvih k pozitivnih clanova niza a*/
float zbir_pozitivnih(float a[], int n, int k){
  int i;

  /*Na pocetku inicijalizujemo sumu na 0*/
  float s=0;

  /*Prolazimo kroz niz brojeva i zaustavljamo se ili ako smo dosli do
     kraja ili ukoliko smo sabrali k brojeva */
  for(i=0; i<n && k>0; i++){
  if(a[i] >= 0){
    /*Kada naidjemo na pozitivan element, uvecavamo sumu i smanjujemo
     k*/
    s+=a[i];
    k--;
  }
  }

  /*Na kraju vracamo izracunatu sumu */
  return s;
}

int main(){
  int n, i, k;
  float a[MAX];

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

  /*Proveravamo korektnost ulaza*/
  if(n<=0 || n> MAX){
  printf("Greska: pogresan unos!\n");
  return 0;
  }

  /*Ucitavamo elemente niza*/
  printf("Unesite elemente niza: ");
  for(i=0; i<n; i++)
  scanf("%f", &a[i]);

  /*Ucitavamo k*/
  printf("Unesite vrednost za k: ");
  scanf("%d", &k);

  /*Proveravamo korektnost za k*/
```

```
52    if(k<0){
      printf("Greska: pogresan unos!");
54    return 0;
      }
56
      /*Ispisujemo rezultat*/
58    printf("Zbir je: %.2f\n", zbir_pozitivnih(a,n,k));
      return 0;
60  }
```

Rešenje 3.26

```
1   /*
      Napisati funkciju void kvadriranje(float a[], int n) koja kvadrira
        elemente realnog niza a duzine n koji se nalaze na parnim
        pozicijama.
3     Napisati i program koji testira rad funkcije. Pretpostaviti da
        duzina niza nece biti veca od 100.

5    */

7   #include<stdio.h>
    #define MAX 100
9
    /*Funkcija kvadirraj menja niz a tako sto kvadrira sve elemente na
        parnim pozicijama. */
11  void kvadriraj(float a[], int n){
      int i;
13
      /*Petljom prolazimo kroz niz i ukoliko je pozicija parna, a[i]
        postaje a[i]*a[i]*/
15    for(i=0; i<n; i++){
      if(i%2 ==0)
17      a[i]*=a[i]; //skraceno od a[i] = a[i]*a[i]
      }
19
    }
21
    int main(){
23
      int n, i, j;
25    float a[MAX];

27    printf("Unesite broj elemenata niza: ");
      scanf("%d", &n);
29
      /*Proveravamo korektnost ulaza*/
31    if(n <=0 || n>100)
      {
33    printf("Greska: pogresan unos!\n");
      return 0;
```

```
35    }

37    /*Ucitavamo elemente niza*/
      printf("Unesite elemente niza:");
39    for(i=0; i<n; i++)
      scanf("%f", &a[i]);
41
      /*Pozivamo funkciju koja kvadrira odgovarajuce elemente*/
43    kvadriraj(a,n);

45    /*Stampamo rezultat
       NAPOMENA: Kada koristimo %g za stampanje realnih brojeva,
47     oni ce biti istampani na najoptimalniji nacin
       (imace onoliko decimalnih mesta koliko ima i sam broj)
49     */
      for(i=0; i<n; i++)
51    printf("%g ", a[i]);

53    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

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 koristi 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 3.44]

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 3.45]

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 3.46]

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 3.47]

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 3.48]

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 zamanjuje 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 3.49]

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
```

Primer 2

```
Interakcija sa programom:
 Unesite tekst:
 sunce
 Broj tacaka: 0
 Broj zareza: 0
```

[Rešenje 3.50]

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 3.51]

Zadatak 3.52 Napisati funckiju $void\ min\_max(float\ a[],\ int\ n,\ float*\ min,\ float*\ max)$ 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 izmeu `a` i `b`. Ištampati niz `A` na standarni izlaz. Maksimalan broj elemenata niza `A` je 200. Ukoliko nisu zadati svi argumenti komandne linije ili ne zadovoljavaju potrebna

svojstva ispisati poruku o grešci.

[Rešenje 3.56]

3.4 Rešenja

Rešenje 3.56

```
/*
  Napisati funkciju uredi koja uredjuje svoja dva
  celobrojna argumenta tako da se u prvom nalazi manji
  a u drugom veci. Napisati potom glavni program koji
  ucitava dva cela broja i uredjuje njihove vrednosti
  primenom napisane funkcije. Na primer, ako su ucitane
  promenljive x=5 i y=2, njihove vrednosti nakon
  primene funkcije uredi treba da budu x=2 i y=5.
*/

#include <stdio.h>

/*
   Argumenti funkcije uredi_pogresno, promenljive a i b,
   predstavljaju lokalne promenljive za ovu funkciju
   i prestaju da postoje po zavrsetku funkcije. Zbog toga
   se efekti razmene vrednosti promenljivih a i b u slucaju
   da je a>b vide u funkciji, ali se ne vide u glavnom programu.
*/
void uredi_pogresno(int a, int b)
{
  int t;

  if (a>b)
  {
     t = a;
     a = b;
     b = t;
  }
  printf("uredi_pogresno :: a=%d, b=%d\n", a, b);
  printf("uredi_pogresno :: &a=%p, &b=%p\n", &a, &b);
}

/*
   Argumenti funkcije uredi_tacno, promenljive pa i pb,
   takodje su lokalne promenljive za ovu funkciju i
   prestaju da postoje kada se funkcija zavrsi.
   Njima prosledjujemo adrese promenljivih a i b koje zelimo
   da razmenimo u slucaju da je a>b.
```

```
40
    Promenljivoj a pristupamo preko pokazivacke promenljive
42  pa sa *pa i slicno, promenljivoj pb pristupamo sa *pb.

44  Vrednosti promenljivih *pa i *pb razmenjujemo kao
    i vrednosti bilo koje dve celobrojne promenljive.
46
*/
48 void uredi_tacno(int * pa, int * pb)
{
50   int t;
   if (*pa>*pb)
52   {
       t = *pa;
54     *pa = *pb;
       *pb = t;
56   }
   printf("uredi_tacno :: *pa=%d, *pb=%d\n ", *pa, *pb);
58 printf("uredi_tacno :: pa=%p, pb=%p\n ", pa, pb);

60 }
int main()
62 {
   int a,b;
64
   printf("Unesi dve celobrojne promenljive:");
66 scanf("%d%d",&a,&b);

68 printf("main :: a=%d, b=%d\n", a,b);
   printf("main :: &a=%p, &b=%p\n", &a, &b);
70 uredi_pogresno(a,b);
   printf("main :: nakon uredi_pogresno, a=%d, b=%d\n", a, b);
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:
  C = 1 - ( R / 255 )
  M = 1 - ( G / 255 )
  Y = 1 - ( B / 255 )

  Napisati program koji ucitava boju u rgb formatu,
  primenjuje odgovarajucu funkciju i ispisuje boju u cmy formatu.

*/

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

void rgb_to_cmy(float* a, float* b, float* c)
{
  /* Zagrade su neophodne jer aritmeticke operacije
     imaju veci prioritet od operatora dereferenciranja (*).
  */
  *a=1-(*a)/255;
  *b=1-(*b)/255;
  *c=1-(*c)/255;

  /*
  Pomocu return ne mozemo vratiti vise od jedne vrednosti.

  Ceste greske:
  return a,b,c;          return vraca samo jednu vrednost
  return a; return b; return c; return ce vratiti samo a

  Zato je neophodno da promenljive ciju vrednost
  zelimo da promenimo prenesemo preko pokazivaca.
  */
}


int rgb_korektno(float a)
{
   if(a<0 || a>255)
      return 0;
   return 1;
}


int main()
{
  float a,b,c;

  /*
      Argumenti funkcije rgb_to_cmy su
      pokazivaci na float. Njima prosledjujemo
      adrese promenljivih a, b i c.
  */
```

```
55
   printf("Unesi boju u rgb formatu (vrednosti izmedju 0 i 255:");
57 scanf("%f%f%f",&a,&b,&c);

59 if(rgb_korektno(a) && rgb_korektno(b) && rgb_korektno(c))
      rgb_to_cmy(&a,&b,&c);
61 else
   {
63    printf("Nekorektan unos\n");
      return -1;
65 }

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

69 return 0;
}
```

Rešenje 3.40

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

10 #include<stdio.h>

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

18 Indikator funkcija vraca kao povratnu vrednost, preko kljucne reci
   return.
20
   Koordinate presecne tacke (ako postoji) funkcija vraca preko
22 liste argumenata, zbog cega promenljive kojima ce koordinate
   biti dodeljene prenosimo preko pokazivaca (promenljive px i py)
24
   Promenljive koje sadrze podatke o pravama (k1,n1,k2,n2) se ne
26 menjaju u funkciji i zbog toga ih ne moramo prenositi preko
   pokazivaca.
28 */

30 int presek(float k1, float n1, float k2, float n2, float* px, float*
       py)
{
```

```
32    if (k1==k2)
        return 0;
34
      *px = -(n1-n2)/(k1-k2);
36    *py = k1*(*px)+n1;
      return 1;
38  }

40  int main()
    {
42    float k1,k2,n1,n2;
      float x,y;
44
      printf("Unesi k i n za prvu pravu:");
46    scanf("%f%f",&k1,&n1);

48    printf("Unesi k i n za drugu pravu:");
      scanf("%f%f",&k2,&n2);
50
      if(presek(k1,n1,k2,n2,&x,&y))
52       printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
      else
54       printf("Prave su paralelne\n");

56    return 0;
    }
```

Rešenje 3.41

```
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
      return 0;
27 }
```

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
   #include <stdio.h>
23 #include <string.h>

25 /*
       Funkcija strspn(str,accept) je ugradjena funkcija koja vraca broj
        karaktera
27     stringa str koji se nalaze u stringu accept, pocev od pocetka
        stringa str.

29     Funkcija strspn se nalazi u zaglavlju string.h.

31     Funkcija strspn_klon je jedna implementacija funkcije strspn.

33     U zadacima cemo uvek koristiti ugradjenu funkciju strspn osim ako
        u tekstu zadatka
       nije naglaseno da se ona ne sme koristiti. Funkcija strspn_klon
        sluzi da pokaze na koji
35     nacin radi ugradjena funkcija strspn.

37     Ugradjena funkcija strspn poziva se na isti nacin kao funkcija
        strspn_klon:
```

```
   strspn(s1,s2)

*/

int strspn_klon(char str[], char accept[])
{
   int br=0;
   int i;

   for(i=0; str[i];i++)
      if(strchr(accept, str[i])!=NULL)
         br++;
      else      /* ako pronadjemo karakter u stringu str koji nije */
         break; /* u stringu accept, prekidamo petlju */

   return br;
}

int main(int argc, char* argv[])
{

   int br;

   if(argc<3)
   {
      printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
     arg2\n");
      return -1;
   }

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

Rešenje 3.43

```
/*
   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
   slovo koje je u abecedi od njega udaljeno k pozicija, i to
   k pozicija ulevo, ako je karakter c jednak karakteru 'L' ili
    udesno
   ako je karakter c jednak karakteru 'D'. Sifrovanje treba da bude
    kruzno. Ako string
   s sadrzi karakter koji nije alfanumericki, ostaviti ga
    nesifriranog.

```

```
   Napisati potom glavni program koji testira napisanu funkciju za
    string i prirodan
   broj koji se unose kao argumenti komandne linije dok se pravac
    sifrovanja unosi
   kao opcija -p koja moze imati vrednosti 'L' ili 'D'. Ukoliko
    opcija -p nije
   navedena, podrazumevani pravac je udesno.

   Mozemo podrazumevati da string sadrzi najvise 30 karaktera.

   Primeri upotrebe:

   1:
   ./a.out abcd 2
   cdef

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

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

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

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

   6:
   ./a.out ab12cd 2 -p D
   cd12ef

*/

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define MAX 31

void sifruj(char s[], char c, int k)
{
   int i;
   int znak;
   char t;

   /*
      S obzirom da ce korektnost unosa podataka
      biti ispitana pre poziva funkcije, promenljiva
```

```
58          c ce imati vrednost 'L' ili 'D'.

60          Promenljiva znak ima vrednost 1 ili -1
            i sluzi kao pomocna promenljiva u slucaju
62          da prilikom sifriranja konvertovani
            karakter izadje iz opsega malih ili velikih slova.
64
       */
66     znak=1;
       if (c=='L')
68          znak = -1;


70
       for(i=0; s[i];i++)
72          if(isalpha(s[i]))
            {
74              /*
                   Promenljiva t predstavlja sifrirani karakter s[i].
76                 Ako je promenljiva t izvan opsega malih ili velikih slova
        ,
                   dodajemo joj ili oduzimamo ukupan broj slova u abecedi
        (26),
78                 u zavisnosti od pravca sifriranja, kako bismo omogucili
                   kruzno sifriranje.
80              */
                t = s[i]+znak*k;
82              if((islower(s[i]) && (t<'a' || t>'z')) || (isupper(s[i]) &&
        (t<'A' || t>'Z')))
                   s[i]=t-znak*26;
84              else
                   s[i]=t;
86          }
    }
88
    int main(int argc, char* argv[])
90  {

92      int k;
        char pravac;
94      char rec[MAX];

96      /*
            Program mozemo pozivati na dva nacina:
98          ./a.out abcd 2
            ili
100         ./a.out abcd 2 -p D

102         Zbog toga, broj argumenata moze biti 3 ili 5.
        */
104
        if (argc!=3 && argc!=5)
106     {
```

```
        printf("Nekorektan unos: broj argumenata moze biti 3 ili 5\n");
108     return -1;
    }
110
    /*
112     Argumenti komandne linije su stringovi. Ako program pokrecemo
        na sledeci nacin:
114     ./a.out abcd 2 -p D
        to znaci da je argument koji odgovara dvojci u stvari
116     string "2". Da bismo string konvertovali u ceo broj,
        koristimo ugradjenu funkciju atoi iz biblioteke stdlib.h.
118 */

120 k = atoi(argv[2]);

122 /*
        Ispitujemo korektnost datih podataka:
124 */
    if (k<=0)
126 {
        printf("Nekorektan unos: broj pozicija mora biti pozitivan ceo
     broj\n");
128     return -1;
    }
130
    /* Korektnost unosa je ispitana, sto znaci da
132 argc moze biti 3 ili 5 */

134 if (argc==3) /* Ako je argc 3: */
        pravac='D';
136 else         /* Ako argc nije 3, tada je sigurno 5, jer je */
    {            /* korektnost unosa ispitana, a unos je korektan
     jedino za argc==3 ili argc==5 */
138   /*
          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         pravac=argv[4][0];
        else
164     {
            printf("Nekorektan unos: pravac moze biti L ili D\n");
166         return -1;
        }
168   }

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

Rešenje 3.44

```
1  #include <stdio.h>

3  int main(int argc, char* argv[]) {

5    int i;
     int s = 0;
7
     /* char *argv[] <--- niz niski koje predstavljaju argumente
        navedene iza poziva programa
9     int argc      <--- ukupan broj niski (sa sve nazivom programa)
       navedenih prilikom pozivanja

11    Ukoliko je program pozvan sa ./a.out 12 abc 6 5 3ab

13    argv[0] = "./a.out".
      argv[1] = "12"
15    argv[2] = "abc"
      argv[3] = "6"
17    argv[4] = "5"
      argv[5] = "3ab"
19
```

```
    argc iznosi 6

    Kako je argv[] po prirodi niz,
    koristimo tzv. brojacku odnosno
    for petlju
    i obradjujemo svaki od argumenata.
   */

   /* Funkcija atoi() prihvata nisku,
    i racuna dekadnu vrednost prosledjene niske,
    dokle god se ona moze racunati.
    Na primer, ukoliko je niska "-123",
    atoi() vraca broj -123.
    Ako je, pak, niska "123abc",
    atoi() ce vratiti 123
    (prilikom prve pojave karaktera koji nije cifra, funkcija prekida
     izracunavanje).

    To za posledicu ima da, ukoliko je funkciji
    prosledjeno nesto
    sto se ne moze pretvoriti u broj,
    na primer niska "abcd",
    funkcija atoi() vraca 0.
   */

   for(i = 1; i < argc; i++)
     s += atoi(argv[i]); /* Zbog nacina rada funkcije atoi(), mozemo
     je pozvati nad svim argumentima
                komandne linije, i sabrati odgovarajuce dekadne
     vrednosti.
                Ukoliko neki argument i nije broj, to ne predstavlja
     problem
                jer ce u tom slucaju odgovarajuci sabirak biti 0
              */

     printf("Zbir numerickih argumenata: %d\n", s);

   return 0;
}
```

Rešenje 3.45

```
#include <stdio.h>

int main(int argc, char* argv[]) {

  int i;

  /* Prolazimo for petljom kroz niz argumenata,
```

```
   i trazimo one niske ciji je prvi karakter bas 'z'.
   Ukoliko je trenutni argument koji se ispituje
   argv[i],
   kako je on sam po sebi niska,
   do prvog karaktera dolazimo kao i pri dosadasnjem
   radu sa niskama --> argv[i][0]
                       ^
                       |
                  index prvog karaktera u niski argv[i]
  */

  for(i = 1; i < argc; i++)
    if(argv[i][0] == 'z')
      printf("%s ", argv[i]);

  putchar('\n');

  return 0;
}
```

Rešenje 3.46

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

int main(int argc, char* argv[]) {

  int i;
  int br = 0;

  /* Da bismo proverili da li se karakter 'z' (tj. 'Z')
   nalazi u niski argv[i],
   to mozemo uciniti koriscenjem funkcije
   strchr() koja se nalazi u string.h.

   Ukoliko je karakter sadrzan u okviru niske,
   strchr() vraca pokazivac na taj karakter
   unutar same niske.
   Inace, ukoliko se karakter ne nalazi u niski,
   funkcija vraca NULL.
  */

  for(i = 1; i < argc; i++)
    if(strchr(argv[i], 'z') != NULL || strchr(argv[i], 'Z') != NULL)
      br++;

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

  return 0;
}
```

Rešenje 3.47

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

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

  /*
    Ispisujemo gresku ukoliko nema dovoljno argumenata komandne
    linije.
  */
  if(argc != 2)
  {
    printf("Greska: nedostaje argument komandne linije!\n");
    return -1;
  }

  /*
    Pretvaramo argument komandne linije koji je string u ceo broj
    koriscenjem funkcije atoi
  */
  n = atoi(argv[1]);
  n = abs(n);

  for(i=(-1)*n;i<=n;i++)
    printf("%d ",i);

  return 0;
}
```

Rešenje 3.48

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

int main(int argc, char *argv[])
{
  int indikator = 0;
  int i,j;
  /*
    Ukoliko imamo samo jedan argument komandne linije,
    ispisujemo da nema istih i zavrsavamo program.
  */
  if(argc < 2)
  {
    printf("Medju argumentima nema istih.\n");
    return -1;
  }
```

```
17
   /*
19   Prolazimo kroz niz argumenata i za svaki posebno proverimo
     da li medju ostalima postoji neki koji mu je jednak i ako postoji
21   ispisujemo poruku i zavrsavamo program.
     Ako smo izasli iz prve petlje to znaci da nismo pronasli dva ista
      elementa
23   i ispisujemo odgovarajucu poruku.
   */
25 for(i=0;i<argc;i++)
   {
27   for(j=0;j != i && j<argc; j++)
       if(strcmp(argv[i], argv[j]) == 0)
29     {
         printf("Medju argumentima ima istih.\n");
31       return 0;
       }
33 }

35 printf("Medju argumentima nema istih.\n");
   return 0;
37 }
```

Rešenje 3.49

```
   #include <stdio.h>
2
   #define MAX 21
4
   void modifikacija(char *s, char *t, int *br_modifikacija)
6  {
     int i;
8    for(i=0;s[i];i++)
       if(s[i]>='a' && s[i]<='z')
10     {
         t[i] = toupper(s[i]);
12       (*br_modifikacija)++;
       }
14     else
         t[i] = s[i];
16 }

18 int main()
   {
20   char s[MAX], t[MAX];
     int br_modifikacija = 0;
22
     printf("Unesite nisku: ");
24   scanf("%s", s);

26   modifikacija(s, t, &br_modifikacija);
```

```
28    printf("Modifikovana niska je: %s\nBroj modifikacija je: %d\n", t,
        br_modifikacija);

30    return 0;
}
```

Rešenje 3.50

```
1  /*
   Napisati funkciju
3    void interpunkcija(int * br_tacaka, int * br_zareza)
   koja za tekst koji se unosi sa standardnog ulaza sve do kraja ulaza
       prebrojava
5  broj tacaka i zareza. Napisati zatim program koji testira napisanu
       funkciju.
   */
7

9  #include <stdio.h>

11 void interpunkcija(int* br_tacaka, int* br_zareza){

13   int tacke=0, zarezi=0;
     char c;
15
     while((c=getchar())!=EOF){
17     if(c=='.')
         tacke++;
19
       if(c==',')
21       zarezi++;
     }
23
     *br_tacaka=tacke;
25   *br_zareza=zarezi;

27 }

29 int main(){
     int br_tacaka, br_zareza;
31
     printf("Unesite tekst: \n");
33
     interpunkcija(&br_tacaka, &br_zareza);
35
     printf("Broj tacaka: %d\n", br_tacaka);
37   printf("Broj zareza: %d\n", br_zareza);

39   return 0;
}
```

Rešenje 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. Pokazivaci
    pn i nn
redom treba da sadrze broj elemenata niza parnih tj. niza neparnih
    elemenata.
Pretpostaviti da duzina niza a nece biti veca od 50. Napisati program
     koji
testira napisanu funkciju.
*/

#include <stdio.h>
#define MAX 50

void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
    int* nn){

  int i, j, k;

  /* i - brojac niza a */
  /* j - brojac niza parnih brojeva */
  /* k - brojac niza neparnih brojeva */

  for(i=0, j=0, k=0; i<n; i++){
      /* Ako je element niza paran */
      if(a[i]%2==0){
          /* Smestamo ga u niz parnih brojeva i uvecavamo indeks niza
     j */
          parni[j]=a[i];
          j++;
      }
      else{
          /* Inace, smestamo ga u niz neparnih brojeva i uvecavamo
    indeks niza k */
          neparni[k]=a[i];
          k++;
      }
  }

  *pn=j;
  *nn=k;

}

int main(){
```

```
  int n, i, j, pn, nn;
42  int a[MAX], parni[MAX], neparni[MAX];

44  /* Ucitavamo dimenziju niza */
  printf("Unesite broj elemenata niza: ");
46  scanf("%d", &n);

48  if(n<0 || n>MAX){
      printf("Greska: pogresna dimenzija niza!\n");
50    return 0;
  }
52
  /* Ucitavamo elemente niza */
54  printf("Unesite elemente niza: ");
  for(i=0; i<n; i++){
56    scanf("%d", &a[i]);
  }
58
  /* Pozivamo funkciju koja razbija zadati niz na niz parnih i niz
     neparnih */
60  par_nepar(a, n, parni, &pn, neparni, &nn);


62
  /* Ispisujemo dobijene nizove */
64  printf("Niz parnih brojeva: ");
  for(i=0; i<pn; i++)
66    printf("%d ", parni[i]);
  printf("\n");
68
  printf("Niz neparnih brojeva: ");
70  for(i=0; i<nn; i++)
    printf("%d ", neparni[i]);
72  printf("\n");

74  return 0;
}
```

Rešenje 3.52

```
/*
2   Napisati funckiju
      void min_max(float a[], int n, float* min, float* max)
4   koja izracunava minimalni i maksimalni element niza a duzine n.
    Napisati zatim i program koji ucitava niz realnih brojeva
      maksimalne
6   duzine 50 i ispisuje vrednosti minimuma i maksimuma na tri decimale
      .

8 */

10 #include<stdio.h>
```

```c
#define MAX 50

void min_max(float a[], int n, float* min, float* max){

  int i;

  /* Inicijalizujemo vrednosti minimuma i maksimuma */
  *min=a[0];
  *max=a[0];

  /* Obilazimo preostale elemente niza */
  for(i=1; i<n; i++){

    /* Ako je tekuca vrednost veca od maksimalne, azuriramo maksimum
    */
    if(a[i]>*max){
      *max=a[i];
    }

    /* Ako je tekuca vrednost manja od minimalne, azuriramo minimum
    */
    if(a[i]<*min){
      *min=a[i];
    }
  }
}

int main(){
  int i, n;
  float a[MAX], min, max;

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

  if(n<0 || n>MAX){
    printf("Greska: pogresna dimenzija niza!\n");
    return 0;
  }

  /* Ucitavamo elemente niza */
  printf("Unesite elemente niza:\n");
  for(i=0; i<n; i++){
    scanf("%f", &a[i]);
  }

  /* Pozivamo funkciju za racunanje maksimuma i minimuma */
  min_max(a, n, &min, &max);

  /* Ispisujemo rezultat */
  printf("Minimum: %.3f\n", min);
  printf("Maksimum: %.3f\n", max);
```

```
62    return 0;

64  }
```

Rešenje 3.56

```
1   #include <stdio.h>

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


5
    int main()
7   {
        int a,b,s;
9
        scanf("%d%d",&a,&b);
11
        suma(a,b,&s);
13
        printf("suma: %d\n",s);
15
        return 0;
17  }

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

Rešenje 3.56

```
1   #include <stdio.h>

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


5
    int main()
7   {
        int a,b,s;
9
        scanf("%d%d",&a,&b);
11
        suma(a,b,&s);
13
        printf("suma: %d\n",s);
15
        return 0;
17  }
```

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

Rešenje 3.56

```
1  #include <stdio.h>

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


5
   int main()
7  {
       int a,b,s;
9
       scanf("%d%d",&a,&b);
11
       suma(a,b,&s);
13
       printf("suma: %d\n",s);
15
       return 0;
17 }

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

Rešenje 3.56

```
   #include <stdio.h>
2
   void suma(int a, int b, int *s);
4


6  int main()
   {
8      int a,b,s;

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

12     suma(a,b,&s);

14     printf("suma: %d\n",s);

16     return 0;
```

```
}

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

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

a) Napisati funkciju *int samoglasnik*(*char c*) koja proverava da li je zadati karakter samoglasnik. Funkcija treba da vrati vrednost 1 ako karakter c jeste samoglasnik, odnosno 0 ako nije.

b) Napisati funkciju *int samoglasnik_na_kraju(char s[])* koja proverava da li se niska *s* završava samoglasnikom (koristiti funkciju iz tačke a)).

c) Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje da li završava samoglasnikom ili ne.

Primer 1

```
Interakcija sa programom:
 Unesite nisku: abcde
 Niska se zavrsava samoglasnikom!
```

Primer 2

```
Interakcija sa programom:
 Unesite nisku: AaBb+cCdD
 Niska se ne zavrsava samoglasnikom!
```

Primer 3

```
Interakcija sa programom:
 Unesite nisku: pRograMiranjE
 Niska se zavrsava samoglasnikom!
```

[Rešenje 3.65]

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 3.66]

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 3.67]

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 3.68]

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 3.69]

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 3.70]

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 3.71]

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:  aaiioo124
 Unesite nisku s:  aeiou
 6
```

Primer 3

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

[Rešenje 3.72]

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 3.73]

Zadatak 3.74

- Napisati funkciju

  ```
  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 izmeu svojih samoglasnika. (Reč je niz karaktera izmeu dve praznine; razmak izmeu samoglasnika reči `danas` i `jutro` je 2, a razmak izmedju `sutrk` i `mnozenje` 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 odreuje poziciju poslednjeg karaktera niske `s` sadržanog u okviru niske `t`, zanemarujući pri tom razliku izmeu 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 1101010111100
```

Primer 2

```
Interakcija sa programom:
 012 435 0x64FE
 1010 110110011 110010011111110
```

Primer 3

```
Interakcija sa programom:
 123 0777
 1111011 111111111
```

Primer 4

```
Interakcija sa programom:
 981
 1111010101
```

[Rešenje 3.91]

Zadatak 3.80 Implementirati funkciju int `str_str(char s[], char t[])` koja proverava da li niska `s` sadrzi nisku `t`. Zatim napisati program koji sa standardnog ulaza učitava pet redova (svaki red ima najvise 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*nprogram
 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 `kljuca` 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 `kljuc` duži od reci.

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 sledece 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 najvišse 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 "uvwbcdnoppqr" a reč "zec" sa "abcfghdef". Napisati program koji šifruje unetu reč sa standardnog ulaza i štampa dobijeni rezultat na standardni izlaz. Za reč pretpostaviti da nije duža od 20 karaktera. Unos reči ostvariti koristeci 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 izmeų 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 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 takoe 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. Dopušteno 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 svIma
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

a) Napisati C funkciju `int procitaj_recenicu(char *s, int max_len)`, koja sa standarnog 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.

b) 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.

c) 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 celobrojnom vrednoscu `b` koja odreuje broj pozicija pomeranja, kao i karakterom `'L'` ili `'D'` koji odreuje smer pomeranja (levo ili desno).

b) Napisati funkciju `void sifruj(char rec[],Sifra s)` koja transformiše zadatu 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 standarnog 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 standarnog ulaza (maksimalna dužina reči je 20 karaktera). Ispisati reči na standarni izlaz nakon primenjenih svih zadatih 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

```
/*
   Napisati funkciju koja konvertuje dati string tako sto
   mala slova menja u velika a velika u mala. Napisati
   potom glavni program koji ucitava string, poziva napisanu
   funkciju i ispisuje konvertovani string. Mozemo pretpostaviti
   da string ne sadrzi vise od 10 karaktera.
*/
```

```c

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

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

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

      Konverzija malog slova u veliko bez upotrebe funkcije toupper:
          s[i] = s[i]-'a'+'A';
      Konverzija velikog slova u malo bez upotrebe funkcije tolower:
          s[i] = s[i]+'a'-'A';

   */
}

int main()
{
   /*
      Poslednji karakter svakog stringa je terminirajuca
      nula '\0', specijalni karakter ciji je ASCII kod 0.

      Ukoliko pretpostavljamo da string sadrzi najvise 30
      karaktera, neophodno je deklarisati niz od 31 karaktera,
      pri cemu se dodatni izdvaja za terminirajucu nulu.

   */
   char s[31];
   printf("Unesi string:");

   /*
      Za razliku od nizova koji se ucitavaju i stampaju
      element po element, stringovi se mogu ucitati i
      odstampati pomocu jedne scanf/printf naredbe koriscenjem
      specifikatora %s.
```

```
58      Funkcija scanf ucitava 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 skrati koja uklanja beline sa
    kraja datog stringa.
4
    Napisati glavni program koji testira napisanu
6   funkciju na stringu "rep belina                    ".

8 */

10 #include <stdio.h>
   #include <ctype.h>
12
   /*
14     Funkcija koja racuna duzinu niza
       ne racunajuci '\0'.
16
       U biblioteci string.h definisan je veliki
18     broj funkcija za rad sa stringovima,
       ukljucujuci i funkciju strlen koja racunana
20     duzinu stringa.

22     Funkcija strlen_klon predstavlja jednu
       implementaciju funkcije strlen.
24
       U zadacima cemo uvek koristiti ugradjenu
26     funkciju strlen osim ako u tekstu zadatka
       nije naglaseno da se ona ne sme koristiti.
28     Funkcija strlen_klon sluzi da pokaze na koji
       nacin radi ugradjena funkcija strlen.
30
       Ugradjena funkcija strlen poziva se na
32     isti nacin kao funkcija strlen_klon:
       strlen(s1)
34
   */
36 int strlen_klon(char s[])
   {
```

```
38    int i=0;
      while(s[i]) /* ASCII kod karaktera '\0' je jednak nuli. */
40       i++;

42    return i;
  }
44
  void skrati(char s[])
46 {
    /*
48       Poslednji karakter stringa s(ne racunajuci '\0') ima
       indeks strlen_klon(s)-1; ideja je da pocnemo od poslednjeg
50     karaktera stringa i da smanjujemo indeks dokle god
       je karakter na poziciji i blanko znak.
52
    */
54  int i;
    for(i=strlen_klon(s)-1;i>=0;i--)
56      if (!isspace (s[i])) /* Ako s[i] nije blanko znak, prekidamo
      petlju. */
           break;
58
    s[i+1]='\0';  /* DOdajemo terminirajucu nulu iza indeksa i (prvi
      neblanko karakter gledano sdesna nalevo).*/
60
    /*
62       Ugradjena funkcija isspace nalazi se u biblioteci ctype.h i
       vraca 1 ako
       je dati karakter blanko znak a 0 u suprotnom.
64
       Unarni logicki operator ! oznacava negaciju.
66
    */
68 }

70 int main()
  {
72
    /*
74       Ukoliko string ne zelimo da ucitavamo po pokretanju programa
       vec da ga unapred zadamo, to mozemo uraditi na sledeci nacin:
76  */
    char s[]="rep belina                              ";
78  /* U ovom slucaju nije neophodno navoditi dimenziju stringa vec
       ce ona biti automatski postavljena na broj karaktera u stringu +
       1 za
80     terminirajucu nulu. */


82
    printf("Pre skracivanja: *%s*\n", s);
84  skrati(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
*/
44 void strcpy_klon(char kopija[], char original[])
{
```

```
  int i;
  for(i=0; original[i]; i++)
    kopija[i]=original[i];

  kopija[i] = '\0';
}

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

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

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

  strcpy_klon(dst,src);

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

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

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

  return 0;
}
```

Rešenje 3.60

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

   Napisati potom glavni program koji ucitava linije
   do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko
```

```
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)
59 {
     int i=0;
```

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

Rešenje 3.61

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

#include <stdio.h>
#include <ctype.h>
#define MAX 10
/*
  String b se sastoji od karaktera koji
  cine jedan ceo broj, onim redom kojim
  se karakteri pojavljuju u zapisu broja.

  Ako je prvi karakter stringa b '-',
  to znaci da je broj negativan i
  funkcija znak_broja vraca -1

  U suprotnom, broj je pozitivan i
  funkcija znak_broja vraca 1

*/

int znak_broja(char b[])
{
   if(b[0]=='-')
      return -1;
   return 1;
}


/*
  Funkcija formiraj_broj na osnovu
  karaktera koji cine broj iz stringa
  b vraca ceo broj koji odgovara
  zapisu datom u stringu b.

  Ako su cifre broja a,b,c i d, tada
  broj mozemo kreirati kao:
  a*10^3 + b*10^2 + c*10^1 + d*10^0

  Medjutim, efikasnije je koristiti
  Hornerovu semu:

  10*(10*(10*(10*0 + a)+b)+c)+d

*/

int formiraj_broj(char b[])
{
```

```
    int i;
52  int n=0;
    int znak = znak_broja(b);
54
    /*
56    Ako je broj negativan, cifre u nizu b
      pocinju od indeksa 1
58  */

60  i=0;
    if(znak==-1)
62     i=1;

64  /*
      Funkcija isdigit proverava da li je broj
66    cifra. Nalazi se u biblioteci ctype.h

68    Proveravamo da li je karakter u zapisu
      broja cifra kako bismo se osigurali
70    od nekorektnog unosa, npr ako korisnik
      unese -123abc. Ovaj unos je moguc jer
72    se vrsi sa scanf("%s",broj), gde unosimo
      karaktere do prvog blanko znaka
74
      Ako naidjemo na karakter koji nije cifra,
76    prekidamo petlju

78  */
    for(; b[i]!='\0'; i++)
80     if(isdigit(b[i]))
          n = n*10 + b[i] - '0';
82     else
          break;
84
    /* Formirani broj mnozimo znakom: */
86
    n*=znak;
88  return n;

90 }

92 int main()
   {
94  char broj[MAX];
    int n;
96
    /* Ucitavamo broj: */
98  scanf("%s", broj);

100 /* Ispisujemo rezultat: */
    printf("Broj zapisan kao int: %d\n", formiraj_broj(broj));
102
```

```
    return 0;
}
```

Rešenje 3.62

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

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

   Funkcija transformisi_negativan vraca
   1 ako je broj negativan i 0 u suprotnom, a
   uz to, ako broj jeste negativan, funkcija
   treba da ga konvertuje u  njegovu apsolutnu
   vrednost. S obzirom da funkcija treba da vrati dve
   vrednosti, to realizujemo na sledeci nacin:
   1. indikator da li je broj negativan
   ce vratiti kao povratnu vrednost
   2. apsolutnu vrednost broja ce vratiti
   preko liste argumenata, zbog cega broj
   prenosimo preko pokazivaca

*/
int transformisi_negativan(int* pn)
{
   if(*pn<0)
   {
      *pn = -(*pn);
      return 1;
   }
   return 0;
}

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

   do
   {
      cifra = n%10;

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

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

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

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

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

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

void obrni(char s[])
{

    char t;
    int i,j;
    /*
     Karaktere stringa obrcemo tako sto razmenimo karaktere na
     pozicijama 0 i n-1,
     zatom 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
       druge
    */

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

}

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

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

```
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.
102       Na primer, za broj 234 niz ce sadrzati cifre 4 3 2. */
        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 utvrdjuje da
       li su strinovi
14       s i t jednaki. Ukoliko jesu, vraca 0, a u suprotnom vraca
       razliku
         ASCII kodova prva dva razlicita karaktera na istim pozicijama
```

```
16        (npr strcmp("aa","ab") ce vratiti -1 a strcmp("ab","aa") 1).

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
      strcmp_klon sluzi da pokaze na koji
24        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[])
   {
34   int i;
     for(i=0; s1[i]==s2[i];i++)
36     if (s1[i]=='\0')
         return 0;
38
     return s1[i] - s2[i];
40 }

42 int main()
   {
44    char s1[21];
      char s2[21];
46    char* p;

48    printf("Unesi dva stringa:");
      scanf("%s%s",s1,s2);
50
      /*
52        Funkcija strstr(s,t) je ugradjena funkcija koja utvrdjuje da
       li je string t
          podstring stringa t i ako jeste, vraca pokazivac (char*) na
       karakter
54        stringa s odakle pocinje prvo pojavljivanje stringa t, a NULL
      u suprotnom.

56        NULL je pokazivac koji ne pokazuje ni na sta, odnosno ne
      sadrzi adresu
          nijedne promenljive.
58
          Podsetimo se veze nizova(a time i stringova) i pokazivaca:
60        ako je string deklarisan sa s1[21], tada je njegov naziv s1
```

```
        ekvivalentan adresi prvog karaktera stringa:
        s1 <=> &s1[0]
        i nadalje redom:
        s1+1 <=> &s1[1]
        ...
        u opstem slucaju:
        s1+i <=> &s1[i]

        To znaci da se indeks elementa na koji pokazuje s1+i moze
        dobiti tako sto od s1+i oduzmemo pokazivac na pocetak niza:
        s1+i-s1 <=> i. Ovako od pokazivaca na karakter u stringu
        dobijamo njegov indeks u stringu.

    */

    p = strstr(s1,s2);

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

    return 0;
}
```

Rešenje 3.64

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

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

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

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

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

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

36  */

38  /*
       Funkcija strchr(s,c) je ugradjena funkcija koja vraca pokazivac
40     na prvi karakter u stringu s koji je jednak karakteru c, ako
     takav
       postoji, a NULL u suprotnom.
42
       Indeks od pokazivaca dobijamo na isti nacin kao u prethodnom
     zadatku
44     sa strstr.
  */
46
  p = strchr(s,c);
48 if(p!=NULL)
     printf("%c se pojavljuje u %s na poziciji %d\n", c, s, p-s);
50 else
     printf("%c se NE pojavljuje u %s\n",c, s);
52
  return 0;
54}
```

Rešenje 3.65

```
1 /*
  a) Napisati funkciju int samoglasnik(char c) koja proverava da li je
       zadati karakter samoglasnik. Funkcija
3 treba da vrati vrednost 1 ako karakter c jeste samoglasnik, odnosno 0
       ako nije.
  b) Napisati funkciju int samoglasnik_na_kraju(char s[]) koja
       proverava da li se niska s zavrsava samoglasnikom
5 (koristiti funkciju iz tacke a)).
  c) Napisati program koji ucitava nisku maksimalne duzine 20 karaktera
       i ispisuje da li zavrsava samoglasnikom ili ne.
7 */

9 #include <stdio.h>
```

```c
#include <ctype.h>
#include <string.h>
#define MAX_DUZINA 20

/* Funkcija proverava da li je karakter c samoglasnik */
int samoglasnik(char c){

  char C;

  /* Konvertujemo karakter u veliko slovo kako bismo smanjili broj
     provera */
  C=toupper(c);

  /* Samoglasnici su slova a, e, i, o i u */
  if(C=='A' || C=='E' || C=='I' || C=='O' || C=='U')
    return 1;

  return 0;
}

/* Funkcija koja proverava da li se niska s zavrsava samoglasnikom */
int samoglasnik_na_kraju(char s[]){
  int duzina;

  /* Odredjujemo duzinu niske */
  duzina=strlen(s);

  /* Proveravamo da li je niska prazna */
  if(duzina==0)
    return 0;

  /* Ako niska nije prazna, proveravamo da li se samoglasnik nalazi
     na kraju */
  /* Numeracija karaktera u niski pocinje nulom pa zato proveravamo
     poziciju duzina -1 */
  return samoglasnik(s[duzina-1]);
}


int main(){
  char s[MAX_DUZINA+1];

  /* Ucitavamo nisku */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Proveravamo da li se zavrsava samoglasnikom i ispisujemo
     odgovarajucu poruku */
  if(samoglasnik_na_kraju(s))
    printf("Niska se zavrsava samoglasnikom!\n");
  else
    printf("Niska se ne zavrsava samoglasnikom!\n");
```

```
  return 0;
}
```

Rešenje 3.66

```
/*
Napisati funkciju void kopiraj_n(char t[], char s[], int n) koja
    kopira najvise n karaktera niske s u nisku t.
Napisati i program koji ucitava nisku maksimalne duzine 20 karaktera
    i jedan ceo broj i testira rad napisane funkcije.
*/

#include <stdio.h>
#define MAX_DUZINA 20

void kopiraj_n(char t[], char s[], int n){
  int i;

  /* Brojac i oznacava tekucu poziciju u niski */
  /* Uslov i<n je neophodan zbog kopiranja najvise n karaktera */
  /* Uslov s[i]!='\0' (ili skraceno samo s[i]) je neophodan kako bi
    bili sigurni da na poziciji i postoji karakter u niski s */
  for(i=0; i<n && s[i]!='\0'; i++){
    t[i]=s[i];
  }

  /* Upisujemo terminirajucu nulu u novodobijenu nisku */
  t[i]='\0';
}


int main(){
  int n;
  char s[MAX_DUZINA+1], t[MAX_DUZINA+1];

  /* Ucitavamo nisku */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Ucitavamo broj n i proveravamo korektnost unosa */
  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<0 || n>MAX){
    printf("Greska: pogresan unos!\n");
    return 0;
  }

  /* Pozivamo funkciju za kopiranje */
  kopiraj_n(t, s, n);

```

```
  /* Ispisujemo dobijenu nisku */
  printf("Rezultujuca niska: %s\n", t);

  return 0;
}
```

Rešenje 3.67

```
/*
Napisati funkciju void dupliranje(char t[], char s[]) koja na osnovu
    niske s formira nisku t tako sto duplira svaki
karakter niske s. Napisati i program koji ucitava nisku maksimalne
    duzine 20 karaktera i testira rad napisane funkcije.
*/

#include <stdio.h>
#define MAX_DUZINA 20

void dupliranje(char t[], char s[]){
  int i, j;

  /* Brojac i oznacava tekucu poziciju u niski s */
  /* Brojac j oznacava tekucu poziciju u niski t */
  for(i=0, j=0; s[i]!='\0'; i++, j+=2){
    t[j]=s[i];
    t[j+1]=s[i];
  }

  /* Upisujemo terminirajucu nulu u novodobijenu nisku */
  t[j]='\0';
}


int main(){
  int n;
  char s[MAX_DUZINA+1], t[2*MAX_DUZINA+1];

  /* Ucitavamo nisku */
  printf("Unesite nisku: ");
  scanf("%s", s);

  /* Pozivamo funkciju za dupliranje */
  dupliranje(t, s);

  /* Ispisujemo dobijenu nisku */
  printf("%s\n", t);

  return 0;
}
```

Rešenje 3.68

```c
/*
 Napisati funkciju int heksa_broj(char s[]) koja proverava da li je
    niskom s zadat korektan heksadekadni broj.
 Heksadekadni broj je korektno zadat ako pocinje prefiksom 0x ili 0X
    i ako sadrzi samo cifre i mala ili velika slova A, B, C, D, E i F
    .
 Funkcija treba da vrati vrednost 1 ako je niska korektan
    heksadekadni broj, odnosno 0 ako nije.
 Napisati i program koji ucitava nisku maksimalne duzine 7 karaktera
    i ispisuje rezultat rada funkcije.
 */

#include<stdio.h>
#define MAX 8


/*
 Funkcija koja proverava da li je prosledjeni karakter ispravna
    heksadekadna cifra (broj ili slovo a,b,c,d,e,f)
 Ukoliko jeste, funkcija vraca 1, u suprotnom 0.
 */
int heksa_cifra(char c){
  /*Pretvaramo karakter c u veliko slovo*/
  c = toupper(c);

  /*Proveravamo da li je u pitanju cifra ili slovo A,B,C,D,E,F i
    ukoliko jeste, vracamo 1*/
  if(isdigit(c) || (c >= 'A' && c <= 'F'))
  return 1;

  /*Ukoliko nije, vracamo 0*/
  return 0;
}

/*Funkcija koja proverava da li prosledjena niska s predstavlja
    ispravan heksadekadni broj */
int heksa_broj(char s[]){
  int i;

  /*Kako heksadekadni brojevi pocinju sa 0x ili 0X, prvo proveravamo
    da li je taj uslov ispunjen,
  tj. da li je s[0] jednak 0 i da li je s[1] jednak X i ako taj uslov
     nije ispunjen, onda
  niska s ne predstavlja korektan heksadekadni broj */
   if(s[0] != '0' || toupper(s[1]) != 'X')
   return 0;

   /*Prolazimo kroz nisku, pocev od pozicije 2 (jer su prve dve
    pozicije 0x) i za svaki karakter do kraja
   niske proveravamo da li je ispravna heksadekadna cifra.
```

```
   Ako naidjemo na bilo koji koji ne ispunjava taj uslov, onda niska
    s nije korektan heksadekadni broj
41   i vracamo 0. */
   for(i=2; s[i] != '\0'; i++)
43   if(!heksa_cifra(s[i]))
       return 0;
45
   /*Ako smo stigli do kraja, znaci da su svi karakteri date niske
    ispravne heksadekadne cifre
47    i zato vracamo 1 */
   return 1;
49 }

51 int main(){
   char s[MAX];
53
   /*Ucitavamo nisku*/
55 printf("Unesite nisku:");
   scanf("%s", s);
57
   /*Pozivamo funckiju i stampamo odgovarajucu poruku*/
59 if(heksa_broj(s))
   printf("Korektan heksadekadni broj!\n");
61 else
   printf("Nekorektan heksadekadni broj!\n");
63
   return 0;
65 }
```

Rešenje 3.69

```
1 /* Napisati funkciju int heksa_broj(char s[]) koja izracunava dekadnu
       vrednost heksadekadnog broja zadatog niskom s.
    * Napisati i program koji ucitava nisku maksimalne duzine 7
      karaktera i ispisuje rezultat rada funkcije.
3  * Pretpostaviti da je uneta niska korektan heksadekadni broj. */

5 #include<stdio.h>
  #include<string.h>
7
  #define MAX 8
9
  /*Funkcija koja racuna dekadnu vrednost heksadekadnog broja*/
11 int heksa_broj(char s[]){
   int i,k;
13 char c;

15 /*Racunamo duzinu niske koja predstavlja heksadekadni broj*/
   int n = strlen(s);
17
   /*Inicijalizujemo vrednost v na 0*/
```

```
19    int v = 0;

21    /*Prolazimo petljom kroz nisku, krenuvsi sa desne strane
      npr: 1a8e = e*1 + 8*16 + a*256 + 1*4096
23    Promenljiva k ce nam biti mnozilac i ona uzima vrednosti 1, 16,
        256, 4096, ...
      Promenljiva c ce nam cuvati trenutnu heksadekadnu cifru (u nasem
        primeru e, 8, a, 1)
25    U svakom koraku treba na ispravan nacin da pomnozimo c i k
      */
27    for(i=n-1, k=1; i>=2; i--, k*=16)
      {
29    /*U c smestamo trenutnu heksadekadnu cifru.
       Pozivamo funkciju toupper da bi obezbedili da radimo samo sa
        velikim slovima.
31     Ako je s[i] cifra, funkcija toupper je nece promeniti.
       */
33    c = toupper(s[i]);

35    if(isdigit(c)){
        /*Ako je c cifra, onda samu vrednost te cifre dobijamo sa c-'0'
37       NAPOMENA: Nije ispravno napisati c*k jer je c karakter!
         */
39      v += (c-'0')*k;
      }else{
41      /*Ako je c slovo izmedju A i F, mi treba da dobijemo odgovarajucu
         vrednost izmedju 10 i 15.
         Ova vrednost se dobija sa 10 + c - 'A'. npr. za A ce biti 10 + '
        A' - 'A' = 10, za B: 10 + 'B' - 'A' = 11, ...*/
43      v += (c - 'A' + 10)*k;
      }
45    }

47    /*Na kraju vracamo izracunatu vrednost */
       return v;
49  }

51  int main(){
      char s[MAX];
53
      /*Ucitavamo nisku*/
55    printf("Unesite nisku:");
      scanf("%s", s);
57
      /*Ispisujemo rezultat*/
59    printf("%d\n", heksa_broj(s));

61    return 0;
    }
```

Rešenje 3.70

```c
/*
 Napisati funkciju int podniska(char s[], char t[]) koja proverava da
     li je niska t podniska niske s.
 Napisati i program koji ucitava dve niske maksimalne duzine 10
    karaktera i testira rad napisane funkcije.

 Napomena: u biblioteci string.h postoji funkcija strstr
 char* strstr(const char* s, const char* t)
 koja vraca adresu pocetka prvog pojavljivanja niske t u niski s ili
    NULL ako se niska t ne javlja
 u niski s.
 */

#include<stdio.h>
#define MAX 11

/*Funkcija koja proverava da li je t podniska od s*/
int podniska(char s[], char t[]){
  int i, j, k;

  /*Spoljna petlja ide redom po niski s*/
  for(i=0; s[i] != '\0'; i++){

  /*Unutrasnja petlja ide redom po niski t*/
  /*Promenljiva k pamti vrednost i, sluzi za poredjenje s[k] i t[j] i
   *zatim se vrsi pomeranje i po niski s i po niski t (k++, j++) */
  /*Cim naidjemo na situaciju da se karakteri ne poklapaju, izlazimo
    iz  unutrasnje petlje*/
  for(j=0, k = i; t[j] != '\0'; j++, k++){
    if(s[k] != t[j])
    break;
  }

  /*Ako smo prosli celu unutrasnju petlju (do kraja niske t), to
    znaci da su se svi karakteri iz t poklopili
   sa karakterima iz s i onda vracamo 1*/
  if(t[j] == '\0')
    return 1;
  }

  /*Na kraju vracamo 0*/
  return 0;
}


int main(){
  char s[MAX], t[MAX];

  /*Ucitavamo prvu nisku*/
  printf("Unesite nisku s: ");
  scanf("%s", s);
```

```
48    /*Ucitavamo drugu nisku*/
      printf("Unesite nisku t: ");
50    scanf("%s", t);

52    /*Pozivamo funkciju i ispisujemo odgovarajucu poruku*/
      if(podniska(s,t))
54    printf("t je podniska niske s!\n");
      else
56    printf("t nije podniska niske s!\n");

58    return 0;
  }
```

Rešenje 3.71

Rešenje 3.72

Rešenje 3.73

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 refleksivna(int a[][MAX], int n)`
kojom se za relaciju zadatom matricom `a` (matruca je kvadratna) ispitije da li je refleksivna.

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 refleksivana ako sa svako `i` važi: `i@i=1`
Relacija je simetricna ako za svako `i` i `j` važi: `i@j=1` => `j@i=1`
Relacija je tranzitivna ako za svako `i`, `j` i `k` važi: `i@j=1` i `j@k=1` => `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 (refleksivna, simetrična i tranzitivna).

[Rešenje 3.108]

Zadatak 3.93 Napisati funkciju `float sumD(float a[][max], int n)` koja odreuje sumu elememenata iznad glavne dijagonale. Potom napisati funkciju `float sumd(float a[][max], int n)` koja odreuje sumu elememenata 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 odreuje 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 elemeti k-te i t-te vrste matrice(k i t su argumenti funkcije). Funkciju testirati pozivom u main-u i ispisom novodobijene matrice na standarni izlaz. Sa standarnog 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 standarnog 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 standari 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 standarnog 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:

(a) Učitava matricu i ispisuje je na izlaz. U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

(b) 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 standarnog 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 zbirovi elemenata kolona matrice uredjeni 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 standarnog 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 standarni 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 $a = (a_1, a_2, \ldots, a_n)$ i $b = (b_1, b_2, \ldots, b_n)$ je $a_1 \cdot b_1 + a_2 \cdot b_2 + \ldots + a_n \cdot b_n$.
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 100×100.

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]

Zadatak 3.107 Sa standarnog 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) izmedju 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 linje 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 50×50. Ukoliko nisu zadati svi potrebni argumenti komadne 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 Definisati strukturu kojom se predstavlja kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja, a zati 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 3.114]

Zadatak 3.115 Definisati strukturu *Lopta* sa poljima *poluprecnik* (ceo broj u centimetrima) i *boja* (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Zatim učitati informacije o n lopti ($0<n<50$) i ispisati ukupnu zapreminu, kao i broj crvenih lopti. *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 3.115]

Zadatak 3.116 Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vocka* sadrži ime voćke (nisku maksimalne dužine 20 karaktera) i količinu vitamina C u miligramima (realan broj). Napisati program koji sa standardnog ulaza učitava podatke o voćkama sve do unosa reči KRAJ i ispisuje ime voćke sa

najviše vitamina C. Pretpostaviti da broj voćki neće biti veći od 50. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
 Unesite ime voćke i njenu količinu vitamina C:   jabuka 4.6
 Unesite ime voćke i njenu količinu vitamina C:   limun 51
 Unesite ime voćke i njenu količinu vitamina C:   kivi 92.7
 Unesite ime voćke i njenu količinu vitamina C:   banana 8.7
 Unesite ime voćke i njenu količinu vitamina C:   pomorandza 53.2
 Unesite ime voćke i njenu količinu vitamina C:   KRAJ
 Voce sa najvise C vitamina je: kivi
```

[Rešenje 3.116]

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.*

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 3.117]

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 3.118]

Zadatak 3.119 Definisati 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 3.119]

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 standarnog ulaza (najviše 100). Ispisati njihov zbir, raliku 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 standarnog 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
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 recniku.* 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 Definisati strukturu koja opisuje dete atributima ime deteta (ne vece 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 standarnog 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 standarni 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 izmedju 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

```
/*

  Data je struktura koja opisuje koordinate
  tacke u ravni:

  typedef struct point
  {
    float x;
    float y;
  } POINT;

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

  Testirati napisane funkcije u glavnom programu.

*/

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

typedef struct point
{
  float x;
  float y;
```

```
} POINT;

/*
  Poljima strukture pristupamo pomocu
  operatora .

  Ako je promenljiva a tipa POINT,
  njenim koordinatama pristupamo
  pomocu a.x i a.y
*/

float rastojanje (POINT a, POINT b)
{
  return sqrt(pow(a.x-b.x,2)+pow(a.y-b.y,2));
}

POINT sredina (POINT a, POINT b)
{
  POINT s;
  s.x = (a.x+b.x)/2;
  s.y = (a.y+b.y)/2;
  return s;
}


int main()
{

  POINT a = {1,2};
  POINT b;
  POINT sredina_a_b;

  /* Ispisujemo koordinate tacke a. */
  printf("Tacka a ima koordinate %.2f,%.2f\n", a.x, a.y);

  /* Ucitavamo koordinate tacke b. */
  printf("Unesi prvu koordinatu tacke: ");
  scanf("%f", &b.x);
  printf("Unesi drugu koordinatu tacke: ");
  scanf("%f", &b.y);
  printf("Tacka b ima koordinate %.2f,%.2f\n", b.x, b.y);

  /* Strukture kao argumenti funkcije - prenos po vrednosti. */
  printf("Rastojanje izmedju tacaka a i b je %.2f\n", rastojanje(a,b)
    );

  /* Struktura kao povratna vrednost funkcije. */
  sredina_a_b=sredina(a,b);
  printf("Tacka na sredini izmedju tacaka a i b je %.2f,%.2f\n",
    sredina_a_b.x, sredina_a_b.y);

  return 0;
```

```
}
```

Rešenje 3.110

```
/*
  Data je struktura
    typedef struct Student
    {
      char ime[MAX];
      char prezime[MAX];
      char smer;
      float prosek;
    } STUDENT;

I   Napisati funkciju koja ucitava sa standardnog ulaza podatke o
    studentu. Mozemo pretpostaviti da
    ime i prezime studenta ne sadrze vise od 30 karaktera.
II  Napisati funkciju koja ispisuje podatke o studentu na standardni
    izlaz.
III Ucitati niz od n studenata i :
      a) ispisati imena i prezimena onih koji su na smeru R
      b) ispisati podatke za studenta sa najvecim prosekom; ako ima
    vise takvih studenata, ispisati
          1) sve njih
   2) prvog
   3) poslednjeg
*/

#include <stdio.h>
#define MAXST 100
#define MAX 31

typedef struct Student
{
  char ime[MAX];
  char prezime[MAX];
  char smer;
  float prosek;
} STUDENT;

/*
   I

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

   Zagrade su neophodne zbog prioriteta operatora:
   operator * ima veci prioritet nego opetator . .

   Operator -> pruza skraceni zapis za prethodno
```

```
45    navedeni pristup poljima:
      s->ime je skraceno za (*s).ime
47    s->prezime je skraceno za (*s).prezime
      itd.
49
*/
51 void ucitaj(STUDENT* s)
{
53   /* printf("Ime:"); */
     scanf("%s",s->ime);
55   /* printf("Prezime:"); */
     scanf("%s",s->prezime);
57   getchar();
     /* printf("Smer:"); */
59   scanf("%c",&s->smer);
     /* printf("Prosek:");*/
61   scanf("%f", &s->prosek);
}
63
/* II */
65 /*
     Kada neku promenljivu prenosimo u funkciju kao argument, obicno
67   je prenosimo po vrednosti (bez pokazivaca), ako se ona nece menjati
        u funkciji
     ili po adresi (preko pokazivaca), ako ce se njena vrednost
       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 zavrsetku
     funkcije. S obzirom da se strukuture sastoje od vise polja,
       zauzimaju
73   vise memorije nego nestrukturne promenljive. Zbog toga je za
       njihovo kopiranje
     potrebno vise vremena i vise memorijskih resursa nego za kopiranje
       nestrukturnih
75   promenljivih.

77   Da bismo ucinili program efikasnijim, korisno je da strukturu uvek
       kao
     argument funkcije prenosimo po adresi (preko pokazivaca), bez
       obzira
79   da li ce se struktura u toj funkciji menjati ili ne. Pokazivac na
       strukturu
     zauzima manje memorije nego sama struktura pa je izrada njegove
       kopije
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 mogucnost da je u funkciji menjamo. Ukoliko zelimo da
        onemogucimo promenu,
      uz argument dodajemo kljucnu rec const. Ako pokusamo da promenimo
        argument
87    funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
         s->smer='X';),
      kompajler ce prijaviti gresku. Na ovaj nacin obezbedjujemo da
        promenljiva
89    koju smo preneli po adresi ne da bismo je promenili vec radi
        povecanja efikasnosti programa,
      ne bude, cak ni slucajno, izmenjena u funkciji.
91
    */
93
    void ispisi(const STUDENT* s)
95  {
      printf("%s %s, %c, %.2f\n",s->ime, s->prezime, s->smer, s->prosek);
97  }


99
    float najveci_prosek(STUDENT studenti[], int n)
101 {
      float m;
103   int i;
      /* Pretpostavimo da student sa indeksom 0 ima
105      maksimalni prosek. */
      m = studenti[0].prosek;
107   for(i=1;i<n;i++)
        if(m<studenti[i].prosek) /* Ako student sa indeksom i ima veci
        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;
119   int i;
      for(i=0;i<n;i++)
121     if(m==studenti[i].prosek) /* Ako naidjemo na studenta sa
        maksimalnim prosekom, prekidamo petlju. */
        {
123        /*
             Na strukture se moze primenjivati
125          naredba dodele.
           */
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:
    - strukturu kao povratnu vrednost
153    - prenos strukture preko pokazivaca u funkciju, s obzirom da ce se
       promeniti u funkciji

155    Prilikom izrade zadataka moze biti izabran bilo koji od opisanih
     nacina rada, osim
    ako neki nacin nije posebno naglasen u tekstu zadatka.
157

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

169 /*  printf("Unesi broj studenata:"); */
  scanf("%d", &n);
171

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

```
    }

/*  printf("Unesi podatke o studentima:"); */
  for(i=0;i<n;i++)
  {
/*    printf("%d. student:\n", i); */
    ucitaj(&studenti[i]);
  }

  printf("Studenti sa R smera:\n");
  for(i=0;i<n;i++)
     if(studenti[i].smer == 'R')
        ispisi(&studenti[i]);
  printf("---------------------\n");


  /* b)1)

     Stampamo podatke o svim studentima sa
     maksimalnim prosekom.
  */

  max_prosek = najveci_prosek(studenti, n);
  printf("Svi studenti koji imaju maksimalni prosek:");
  for(i=0;i<n;i++)
     if(studenti[i].prosek==max_prosek)
        ispisi(&studenti[i]);

  /* b)2) */
  student_sa_max_prosekom = prvi_student_sa_najvecim_prosekom(
    studenti,n,max_prosek);

  printf("Prvi student u nizu sa najvecim prosekom: ");
  ispisi(&student_sa_max_prosekom);

  /* b)3) */
  poslednji_student_sa_najvecim_prosekom(studenti,n,max_prosek,&
    student_sa_max_prosekom);

  printf("Poslednji student u nizu sa najvecim prosekom: ");
  ispisi(&student_sa_max_prosekom);

  return 0;
}
```

Rešenje 3.111

```
/*

Napisati program koji ucitava reci sa standardnog ulaza dok korisnik
    ne zada EOF i ispisuje
```

```
ih na standardni izlaz svaku u posebnom redu, poravnatu udesno u
    odnosu
na poslednji karakter najduze reci. Koristiti
strukturu typedef struct rec
    {
       char s[21];
       int duzina;
          }REC;
Na primer, ako su unesene sledece reci:
Danas imamo ispit iz programiranja1.
Nadam se da nece biti tesko!
onda ispis izgleda ovako:
          Danas
          imamo
          ispit
             iz
programiranja1.
          Nadam
             se
             da
           nece
           biti
         tesko!

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

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

*/

#include<stdio.h>
#include<string.h>
#define MAXRECI 100
#define MAX 31

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


void ucitaj_rec(REC* rec)
{
   scanf("%s", rec->s);
   rec->duzina = strlen(rec->s);
```

```
}

/*
   U funkciji ucitaj_niz_reci argument n oznacava broj
   elemenata niza reci, koji ce biti poznat tek po
   zavrsetku funkcije. Ova promenljiva ce dobiti svoju
   vrednost u funkciji i zbog toga mora biti prenesena
   preko pokazivaca.

*/

void ucitaj_niz_reci(REC reci[], int* pn, int granica)
{
   int i=0;
   do
   {
      ucitaj_rec(&reci[i]);
      i++;
   }
   while(reci[i-1].duzina>0 && (i-1)<granica);

   /*
      S obzirom da se promenljiva i ucitava
      pre ispitivanja uslova, uslov ispitujemo
      za rec sa indeksom i-1
   */

   *pn = i-1;

   /*
      S obzirom da se vrednost promenljive i
      ucitava i kada je unesen EOF, dimenzija
      niza odgovarace vrednosti i-1
   */
}

int max_duzina(REC reci[], int n)
{
   int najveca_duzina;
   int i;

   /*
      Najvecu duzinu inicijalizujemo na duzinu
      prve reci.
   */
   najveca_duzina = reci[0].duzina;

   for(i=1;i<n;i++)
      if(reci[i].duzina>najveca_duzina)   /* Ukoliko u nizu naidjemo
    na rec duzine vece od najvece duzine, */
         najveca_duzina = reci[i].duzina; /* menjamo vrednost
    promenljive najveca_duzina. */
```

```
105     return najveca_duzina;
}
107
/*
109     Da bismo realizovali ispis u trazenom formatu, pre
    svake reci ispisujemo onoliko razmaka koliko iznosi
111     razlika maksimalne duzine i duzine date reci.
*/
113
void ispis(REC reci[], int n, int max_d)
115 {
    int i,j;
117
    for(i=0;i<n;i++)
119     {
        for(j=0;j<max_d-reci[i].duzina;j++)
121         printf(" ");
        printf("%s\n", reci[i].s);
123     }

125 }

127 int main(int argc, char* argv[])
{
129     REC reci[MAXRECI];
    int najveca_duzina;
131     int n;

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

137     return 0;
}
```

Rešenje 3.112

```
/*
2     Napisati program koji izracunava prosecnu cenu jedne potrosacke
    korpe. Potrosacka korpa se sastoji od broja kupljenih artikala i
4     niza kupljenih artikala. Svaki artikal odredjen je svojim nazivom,
    kolicinom i cenom. Program treba da ucita broj potrosaca n (
     najvise 100),
6     zatim podatke za n potrosackih korpi i da na osnovu ucitanih
     podataka
    izracuna prosecnu cenu potrosacke korpe. Ucitavanje se vrsi sa
     standarnog
8     ulaza pri cemu se prvo zada broj artikala, a zatim za svaki
     artikal naziv,
    kolicina i cena.  Mozemo pretpostaviti da nijedan
```

```
10    potrosac nece kupiti vise od 20 artikala, kao i da naziv svakog
       artikla
      sadrzi maksimalno 30 karaktera.
12
  */
14
  #include <stdio.h>
16 #define MAXART 20
  #define MAXPOT 100
18 #define MAXNAZIV 31

20 typedef struct artikal
  {
22    char naziv[MAXNAZIV];
      int kolicina;
24    float cena;
  } ARTIKAL;
26
  typedef struct korpa
28 {
      int br_art;
30    ARTIKAL artikli[MAXART];
  } KORPA;
32
  /*
34    Funkcija ucitaj_artikal ucitava podatke za jedan
      artikal i vraca 1 ako je ucitavanje bilo uspesno
36    a 0 u suprotnom. Ucitavanje je neuspesno ukoliko
      kolicina nekog artikla ili njegova cena nisu pozitivni
38    brojevi.

40    S obzirom da funkcija ucitaj artikal treba da vrati
      dve vrednosti (ucitanu strukturu i indikator uspesnosti),
42    strukturu ARTIKAL prenosimo preko pokazivaca a
      indikator uspesnosti vracamo kao povratnu vrednost.
44
  */
46
  int ucitaj_artikal(ARTIKAL* a)
48 {

50    scanf("%s", a->naziv);
      scanf("%d", &a->kolicina);
52
      if (a->kolicina<=0)
54    {
         printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina
       );
56       return 0;
      }
58

      scanf("%f",&a->cena);
```

```
60    if (a->cena<0)
      {
62       printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
         return 0;
64    }

66    return 1;
}
68
/*
70    Funkcija izracunaj_racun izracunava racun date
      potrosacke korpe u kojoj su inicijalizovani
72    podaci o broju artikala i o svakom pojedinacnom
      artiklu.
74 */
float izracunaj_racun(const KORPA* k)
76 {
      int i;
78    float racun=0;
      for(i=0;i<k->br_art;i++)
80       racun+=k->artikli[i].kolicina * k->artikli[i].cena;
      return racun;
82 }

84 /*
      Pri ucitavanju korpe, zadaje se broj artikala a zatim
86    podaci za svaki artikal.

88    Funkcija ucitaj_korpu vraca 1 ako je ucitavanje uspesno
      i 0 u suprotnom. Do neuspesnog ucitavanja moze doci
90    ako broj artikala u korpi nije pozitivan ili ako dodje
      do neuspesnog ucitavanja nekog artikla.
92 */

94 int ucitaj_korpu(KORPA* k)
{
96    int i;
      scanf("%d", &k->br_art);
98    if (k->br_art<=0)
      {
100      printf("Nekorektan unos za broj artikala: %d\n", k->br_art);
         return 0;
102   }
      for(i=0; i<k->br_art;i++)
104      if (ucitaj_artikal(&k->artikli[i])==0)
            return 0;
106
      return 1;
108 }

110 /*
      Funkcija ucitaj_niz_korpi ucitava podatke
```

```
112    za niz od n potrosackih korpi. Funkcija
       vraca 1 ako je ucitavanje uspesno i 0 ako
114    nije. Ucitavanje je neuspesno ukoliko ne uspe
       ucitavanje jedne od korpi.
116 */

118 int ucitaj_niz_korpi(KORPA korpe[], int n)
    {
120    int i,j;
       for(i=0; i<n; i++)
122       if(ucitaj_korpu(&korpe[i])==0)
             return 0;
124
       return 1;
126
    }
128
    /*
130    Funkcija stampaj_racun ispisuje na
       standardni izlaz racun za datu korpu
132    tako sto za svaki artikal ispise
       naziv, cenu i kolicinu i na kraju
134    ukupnu cenu za kupljene artikle.
    */
136
    void stampaj_racun(const KORPA* k)
138 {
       int i,j;
140    for(i=0;i<k->br_art;i++)
          printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
        kolicina, k->artikli[i].cena);
142    printf("------------------------\n");
       printf("\tukupno: %.2f\n", izracunaj_racun(k));
144
    }
146
    /*
148    Funkcija stampaj_racune_za_korpe
       ispisuje na standardni izlaz racune
150    za svaku korpu u nizu potrosackih
       korpi
152 */

154 void stampaj_racune_za_korpe(KORPA korpe[], int n)
    {
156    int i;
       for (i=0;i<n;i++)
158    {
          printf("\nKorpa %d:\n",i);
160       stampaj_racun(&korpe[i]);
       }
162 }
```

```
164 /*
       Funkcija prosek racuna prosecnu cenu
166    potrosacke korpe za dati niz potrosackih
       korpi
168 */
    float prosek(KORPA korpe[], int n)
170 {
       int i;
172    float p;

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

180 int main()
    {
182    int n;
       KORPA korpe[MAXPOT];
184
       printf("Unesi broj potrosackih korpi:");
186    scanf("%d", &n);

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

194    if (ucitaj_niz_korpi(korpe, n)==0)
          return -1;
196
       stampaj_racune_za_korpe(korpe,n);
198    printf("Prosecna cena potrosacke korpe: %.2f\n", prosek(korpe, n))
         ;

200    return 0;
    }
```

Rešenje 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 vraca 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
21   treba da ucita sa standardnog ulaza operaciju i dva
     operanda u polja o, x i y strukture IZRAZ. Funkcija vraca
23   1 ako je ucitavanje bilo uspesno, tj. ako je izraz bio
     korektno zadat ili 0 u suprotnom.
25
     d) Napisati funkciju koja stampa dati izraz infiksno, u obliku
27   x o y = vr. Na primer, za izraz + 4 17 ispis treba
     da bude 4+17=21
29


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
#include <stdio.h>
43 #define MAX 1000

45 typedef struct izraz
{
47   char o;
     int x;
49   int y;
} IZRAZ;
51


53 /*
     Funkcija korektan_izraz vraca 1 ako je izraz korektan a 0
55   u suprotnom. Izraz je korektan ukoliko se sastoji od
```

```
   aritmetickih operacija +,-,* ili /, i ukoliko je u slucaju
57 operacije deljenja drugi operand razlicit od nule.
*/
59 int korektan_izraz(const IZRAZ* izraz)
{
61 if(izraz->o!='+' && izraz->o!='-' && izraz->o!='*' && izraz->o!='/
    ')
   {
63    printf("Nedozvoljena operacija!\n");
      return 0;
65 }
   if(izraz->o=='/' && izraz->y==0)
67 {
      printf("Deljenje nulom!\n");
69    return 0;
   }
71 return 1;
}
73
/*
75 Promenljiva izraz ce se promeniti u funkciji
   vrednost tako sto ce njenom neinicijalizovanom
77 polju vr biti dodeljena vrednost izraza. Zbog
   toga ovu promenljivu funkciji prosledjujemo
79 po adresi, preko pokazivaca
*/
81
int vrednost(const IZRAZ* izraz)
83 {
   int v;
85
   switch (izraz->o)
87 {
      case '+':
89       v=izraz->x+izraz->y;
         break;
91    case '-':
         v=izraz->x-izraz->y;
93       break;
      case '*':
95       v=izraz->x*izraz->y;
         break;
97    case '/':
         v=izraz->x/izraz->y;
99       break;
   }
101 return v;
}
103


105 /*
   Promenljiva izraz ce se promeniti u funkciji
```

```
107     ucitaj_izraz tako sto ce njenim neinicijalizovanim
        poljima o,x,y biti dodeljene vrednosti ucitane
109     sa standardnog ulaza. Zbog toga ovu promenljivu
        funkciji prosledjujemo po adresi, preko pokazivaca.
111
        S obzirom da ucitavanje karaktera nije prvo
113     ucitavanje koje se obavlja u programu, funkcijom
        getchar() "pokupimo" karakter kojim razdvajamo
115     unos karaktera od prethodnog unosa (najcesce blanko
        znak)
117
    */
119
    int ucitaj_izraz(IZRAZ* izraz)
121 {
        getchar();
123     scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
        if (!korektan_izraz(izraz))
125         return 0;
        return 1;
127 }


129
    void stampaj_izraz(const IZRAZ* izraz)
131 {
        printf("%d %c %d = %d\n", izraz->x, izraz->o, izraz->y, vrednost(
         izraz));
133 }

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

151 int main()
    {
153     int n;
        IZRAZ izrazi[MAX];
155     int max;
        int i;
157
```

```
    /* Ucitavamo broj izraza: */
159 scanf("%d", &n);
    if(n<0 || n>MAX)
161 {
       printf("Nekorektna vrednost broja n!\n");
163    return -1;
    }
165

167  /* U petlji ucitavamo jedan po jedan izraz: */
    for(i=0; i<n; i++)
169    if(ucitaj_izraz(&izrazi[i])==0)
       {
171       printf("Nekorektan unos\n");
          return -1;
173    }

175
    printf("Svi izrazi:\n");
177 for(i=0; i<n; i++)
          stampaj_izraz(&izrazi[i]);
179
    max = max_vr(izrazi, n);
181 printf("Maksimalna vrednost izraza:%d\n", max);

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
      manja od polovine maksimalne, ispisujemo ga. */
187       stampaj_izraz(&izrazi[i]);

189 return 0;
}
```

Rešenje 3.114

```
1  #include <stdio.h>

3  /* Struktura koja opisuje kompleksni broj obuhvata polje za realni i
       polje za imaginarni deo broja. */
   typedef struct Complex {
5
     float re;
7    float im;
   } Complex;
9
   /* Funkcija kojom se izracunava zbir kompleksnih brojeva */
11 Complex saberi(Complex *a, Complex *b) {
```

```
13    Complex c;
      c.re = a->re + b->re;
15    c.im = a->im + b->im;
      return c;
17  }

19  /* Funkcija kojom se izracunava razlika kompleksnih brojeva */
    Complex oduzmi(Complex *a, Complex *b) {
21
      Complex c;
23    c.re = a->re - b->re;
      c.im = a->im - b->im;
25    return c;
    }
27
    /* Funkcija kojom se izracunava proizvod kompleksnih brojeva */
29  Complex pomnozi(Complex *a, Complex *b) {

31    Complex c;
      c.re = a->re * b->re - a->im * b->im;
33    c.im = b->re * a->im + a->re * b->im;
      return c;
35  }

37  /* Funkcija kojom se izracunava kolicnik kompleksnih brojeva */
    Complex podeli(Complex *a, Complex *b) {
39
      Complex c;
41    c.re = (a->re * b->re + a->im * b->im) / (b->re*b->re + b->im*b->im
        );
      c.im = (b->re * a->im - a->re * b->im) / (b->re*b->re + b->im*b->im
        );
43    return c;
    }
45
    int main() {
47
      Complex a, b;
49    Complex c;

51    /* Ucitavamo kompleksne brojeve */
      printf("Unesite realni i imaginarni deo prvog broja: ");
53    scanf("%f%f", &a.re, &a.im);

55    printf("Unesite realni i imaginarni deo drugog broja: ");
      scanf("%f%f", &b.re, &b.im);
57
      c = saberi(&a, &b);
59    printf("Zbir: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im);
        /* Ukoliko je imaginarni deo negativan,
                                        njegov zapis vec ukljucuje znak,
61                                      te to treba proveriti.
```

```
                                        Inace, broj je oblika a+b*i
63                                    */

65   c = oduzmi(&a, &b);
     printf("Razlika: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im)
       ;
67
     c = pomnozi(&a, &b);
69   printf("Proizvod: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.im
       );

71   if(b.re != 0 || b.im != 0) {
       c = podeli(&a, &b);
73     printf("Kolicnik: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : ' ', c.
       im);
     }
75   else
       printf("Kolicnik ne postoji.\n"); /* Ni u polju kompleksnih
       brojeva
77                     nije dozvoljeno deljenje nulom
                        */
79

81   return 0;
}
```

Rešenje 3.115

```
1  #include <stdio.h>
   #include <math.h>
3
   #define MAX 50
5
   typedef struct lopta {
7    int poluprecnik;
     enum {plava, zuta, crvena, zelena} boja; /* tip "boja" je
       nabrajajuci tip,
9                          a efekat nabrajanja mogucih vrednosti
                           {plava, zuta, zelena, crvena}
11                         ekvivalentan je definisanju
                           4 celobrojne konstante direktivom #define
13                       */
   } LOPTA;
15
   /*  Funkcija koja odredjuje zapreminu lopte */
17 float zapremina(LOPTA* l) {
    return pow(l->poluprecnik, 3)*4/3*M_PI;
19 }

21 /* Pomocna funkcija koja racuna zapreminu svih lopti */
   float ukupna_zapremina(LOPTA lopte[], int n) {
```

```
23
   int i;
25 float z = 0;

27 for(i = 0; i < n; i++)
     z += zapremina(&lopte[i]);
29
   return z;
31 }

33 /*
   Funkcija je opstija od trazene i broji sve lopte odredjene boje u
     nizu lopti.
35 U zavisnosti od prosledjene boje funkciji, funkcija vraca
     odgovarajuci broj.
*/
37 int broj_lopti_u_boji(LOPTA lopte[], int n, int boja) {

39 int br = 0;
   int i;
41 for(i = 0; i < n; i++)
     if(lopte[i].boja == boja)
43     br++;
   return br;
45 }

47 int main() {

49 LOPTA lopte[MAX];
   int n;
51 int i;
   int boja;
53
   printf("Unesite broj lopti: ");
55 scanf("%d", &n);

57 if(n < 1 || n > MAX) {

59   printf("Nekorektan unos.\n");
     return 0;
61 }

63 printf("Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta,
     3-crvena, 4-zelena):\n");
   for(i = 0; i < n; i++) {
65
     printf("%d. lopta: ", i+1);
67   scanf("%d%d", &lopte[i].poluprecnik, &boja);

69 /* U zavisnosti od unetog
      celog broja,
71    bira se boja lopte.
```

```
   */
73   switch(boja) {

75     case 1: lopte[i].boja = plava; break;
       case 2: lopte[i].boja = zuta; break;
77     case 3: lopte[i].boja = crvena; break;
       case 4: lopte[i].boja = zelena; break;
79     default:
       printf("Nekorektan unos.\n");
81     return 0;
     }
83 }

85 printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));

87 printf("Ukupno crvenih lopti: %d\n", broj_lopti_u_boji(lopte, n,
     crvena));

89 return 0;
}
```

Rešenje 3.116

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

4 #define MAX_DUZINA 21
  #define MAX_BR_VOCKI 50
6
  typedef struct vocka
8 {
    char ime[MAX_DUZINA];
10  float vitamin;
  } VOCKA;
12


14 int main()
  {
16  VOCKA vocke[MAX_BR_VOCKI];
    int i = 0, n, max_vocka;
18  char ime[MAX_DUZINA];

20  /*
      Ucitavamo podatke o vockama i smestamo ih u niz
22    sve dok ne unesemo rec KRAJ ili ucitamo MAX_BR_VOCKI vocki.
    */
24  do
    {
26    printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
      scanf("%s",ime);
28    /*
```

```
      Kada unesemo rec KRAJ prekidamo petlju.
    */
    if(strcmp(ime, "KRAJ") == 0)
      break;

    /*
      Inace ucitavamo i kolicinu vitamina
      i tu vrednost smestamo u vocku na poziciji i
    */
    strcpy(vocke[i].ime,ime);
    scanf("%f",&vocke[i].vitamin);
    i++;
  }
  while(i<MAX_BR_VOCKI);

  n = i;

  /*
    Pretpostavicemo da prva vocka ima najvise vitamina.
    Procicemo kroz niz vocki i ukoliko naidjemo na vocku koja ima
    vise vitamina
    od one koja trenutno ima najvise, azuriracemo vrednosti
    maksimalne vocke.

    Sve vreme cuvamo indeks vocke sa najvise vitamina C.
  */

  max_vocka = 0;
  for(i=1;i<n;i++)
    if(vocke[i].vitamin > vocke[max_vocka].vitamin)
    {
      max_vocka = i;
    }

  printf("Voce sa najvise C vitamina je: %s\n", vocke[max_vocka].ime)
    ;

  return 0;
}
```

Rešenje 3.117

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

#define MAX_IME_PREZIME 51
#define MAX_ZELJA 101
#define MAX_BR_STUDENATA 100

typedef struct student
{
```

```
10    char imeipre[MAX_IME_PREZIME];
      char zelja[MAX_ZELJA];
12  } STUDENT;


14
    int main()
16  {
      STUDENT studenti[MAX_BR_STUDENATA];
18    char odgovor[3];
      char imeiprezime[MAX_IME_PREZIME];
20    int i = 0, n;
      int broj_pronalazaka;
22


24    /*
        Ucitavamo studente i njihove zelje i upisujemo ih u niz
26       sve dok to zelimo ili dok ne unesemo MAX_BR_STUDENATA studenata.
      */
28    while(i < MAX_BR_STUDENATA)
      {
30      printf("Ime i prezime studenta: \n");
        /*
32        Funkcija fgets ucitava jednu liniju i smesta je promenljivu
        koju zadajemo kao njen prvi argument.
          Drugi argument je maksimalna duzina linije.
34        Treci argument je kod nas stdin sto predstavlja standardni ulaz
        .
        */
36
        if(fgets(studenti[i].imeipre, MAX_IME_PREZIME, stdin) == NULL)
38      {
          printf("Greska: nismo dobro ucitali ime i prezime studenta.");
40        return 0;
        }
42
        printf("Njegova zelja: \n");
44
        if(fgets(studenti[i].zelja, MAX_ZELJA, stdin) == NULL)
46      {
            printf("Greska: nismo dobro ucitali zelju studenta.");
48          return 0;
        }
50
        i++;
52
        printf("Jos vrednih studenta (da/ne)?\n");
54
        scanf("%s", odgovor);
56
        /*
58        Moramo da pokupimo karakter koji unesemo nakon odgovora
          kako ga ne bismo ucitali u sledecoj iteraciji petlje.
```

```
60      */
        getchar();
62      /*
          Ukoliko je nas odgovor "ne" prekidamo petlju.
64        A ukoliko nije ni "da" ni "ne" onda nismo dobro uneli odgovor.
        */
66      if(strcmp(odgovor,"ne") == 0)
          break;
68      else if(strcmp(odgovor,"da") != 0)
        {
70        printf("Greska: odgovor mora biti u obliku (da/ne)!");
          return 0;
72      }

74    }

76    /*
        Postavljamo dimenziju niza.
78    */
      n = i;
80    printf("Za podsecanje uneti ime i prezime: \n");
      if(fgets(imeiprezime, MAX_IME_PREZIME, stdin) == NULL)
82    {
        printf("Greska: nismo dobro ucitali ime i prezime studenta.");
84      return -1;
      }
86
      /* Prolazimo kroz listu studenta i ispisujemo zelje studenta cije
         ime i prezime smo uneli. */
88    broj_pronalazaka=0;
      for(i=0;i<n;i++){
90      if(strcmp(imeiprezime, studenti[i].imeipre) == 0){
          broj_pronalazaka++;
92        printf("Novogodisnja zelja: %s\n",studenti[i].zelja);
        }
94    }

96    /* Za slucaj da nismo pronasli studenta sa trazenim imenom */
      if(broj_pronalazaka==0){
98      printf("Trazeni student ne postoji - mozda ce mu poklon odneti
        drugi Deda Mraz\n");
      }
100
      return 0;
102 }
```

Rešenje 3.118

```
1   /*
      Definisati strukturu Grad u kojoj se nalazi ime grada (niska duzine
          20 karaktera) i prosecna temperatura u
```

```
  toku decembra (realan broj). Napisati program koji ucitava imena n
    (0<n<50) gradova i njihove prosecne
  temperature, a zatim ispisuje one gradove koji imaju idealnu
    temperaturu za klizanje: od 3 do 8 stepeni.
  Napomena: probati sa testiranjem zadataka pomocu preusmeravanja.
*/

#include <stdio.h>
#define MAX_DUZINA 20
#define MAX_BR_GRADOVA 50

typedef struct Grad{
  char ime_grada[MAX_DUZINA+1];
  float temperatura;
}Grad;


int main(){
  int n, i;
  Grad grad[MAX_BR_GRADOVA];

  printf("Unesite broj n: ");
  scanf("%d", &n);
  if(n<0 || n>MAX_BR_GRADOVA){
    printf("Greska: pogresan unos!\n");
    return 0;
  }

  for(i=0; i<n; i++){
    printf("Unesite grad i temperaturu: ");
    scanf("%s %f", grad[i].ime_grada, &grad[i].temperatura);
  }

  printf("Gradovi sa idealnom temperaturom za klizanje u decembru:\n"
    );
  for(i=0; i<n; i++){
    if(grad[i].temperatura>=3 && grad[i].temperatura<=8){
      printf("%s\n", grad[i].ime_grada);
    }
  }

  return 0;
}
```

Rešenje 3.119

```
/*
  Definisati strukturu ParReci koja sadrzi rec na srpskom jeziku i
    odgovarajuci prevod na engleski jezik. Zatim
  sa standardnog ulaza sve do kraja ulaza ucitavati parove reci i,
    posebno, za recenicu koja se zadaje sa ulaza
```

```
  ispisati prevod - ako je rec u recenici nepoznata umesto nje
    ispisati odgovarajuci broj zvezdica. Reci nece biti
  duze od 50 karaktera, a ukupan broj reci nece biti veci od 100.
    Napomena: probati sa testiranjem zadataka
  pomocu preusmeravanja.
*/

#include <stdio.h>
#include <string.h>
#define MAX_DUZINA 20
#define MAX_BR_RECI 100
#define MAX_DUZINA_RECENICE 100

typedef struct ParReci{
  char sr[MAX_DUZINA+1];
  char en[MAX_DUZINA+1];
}ParReci;


/*
  Funkcija koja u recniku koji sadrzi n reci trazi prevod reci rec i
    upisuje ga u prevod.
  Ukoliko se rec ne nalazi u recniku, prevod se sastoji od zvezdica
    pri cemu broj zvezdica odgovara
  duzini nepoznate reci.
*/

void pronadji_prevod(ParReci recnik[], int n, char rec[], char prevod
    []){
  int i;

  /* Pretrazujemo recnik i trazimo zadatu rec */
  for(i=0; i<n; i++){

    /* Ukoliko se rec nalazi u recniku */
    if(strcmp(recnik[i].sr, rec)==0){
      /* Ocitavamo njen prevod */
      strcpy(prevod, recnik[i].en);
      /* I obustavljamo pretragu */
      return;
    }
  }

  /* Ukoliko rec nije pronadjena, formiramo prevod reci koji se
    sastoji od zvezdica */
  for(i=0; rec[i]; i++){
    prevod[i]='*';
  }
  prevod[i]='\0';
}


```

```
int main(){
  ParReci recnik[MAX_BR_RECI];
  int n;
  char sr[MAX_DUZINA+1];
  char en[MAX_DUZINA+1];
  int i, j, k;
  char recenica[MAX_DUZINA_RECENICE+1];
  char rec[MAX_DUZINA+1];
  char prevod[MAX_DUZINA+1];
  int citamo_rec;
  char* novi_red;

  /* Ucitavamo parove reci sa standardnog ulaza sve do kraja ulaza*/
  i=0;
  while(scanf("%s %s", sr, en)!=EOF){
    if(i==MAX_BR_RECI)
      break;

    strcpy(recnik[i].sr, sr);
    strcpy(recnik[i].en, en);

    i++;
  }
  /* Broj parova reci cuvamo u promenljivoj n */
  n=i;

  /* Ucitavamo recenicu - nisku karaktera sve do pojave znaka za novi
     red */
  printf("Unesite recenicu za prevod:\n");
  fgets(recenica, MAX_DUZINA_RECENICE, stdin);

  /* Ako postoji, zamenjujemo znak za novi red terminirajucom nulom
    */
  novi_red=strchr(recenica, '\n');
  if(novi_red!=NULL)
    *novi_red='\0';


  /* Izdvajamo rec po rec unesene recenice */
  /* j oznacava tekuci karakter recenice koji se obradjuje */
  j=0;
  /* citamo_rec sa mogucim vrednostima 1 i 0 ce biti indikator koji
    pokazuje da li citamo rec ili ne */
  citamo_rec=0;

  while(1){
    /* Proveravamo da li smo stigli do kraja recenice */
    if(recenica[j]=='\0')
      break;

    /* Ukoliko smo procitali karakter koji je sastavni deo reci (nije
     belina) */
```

```
    if(recenica[j]!=' ' && recenica[j]!='\n' && recenica[j]!='\t'){
99      /* Smestamo ga u rec */
        if(citamo_rec==0){
101       citamo_rec=1;
          k=0;
103     }
        rec[k]=recenica[j];
105     k++;
    }
107 else{
      /* Inace, procitali smo karakter koji ne treba ukljuciti u rec
    */
109   /* Ako smo pre toga citali rec */
      if(citamo_rec==1){
111     /* Prekidamo citanje */
        citamo_rec=0;
113     rec[k]='\0';

115     /* I trazimo i ispisujemo odgovarajuci prevod reci */
        pronadji_prevod(recnik, n, rec, prevod);
117     printf("%s ", prevod);
      }
119 }

121 /* Prelazimo na sledeci karakter recenice */
    j++;
123 }

125 /* Za slucaj da nije obradjena, obradjujemo poslednju rec i
      ispisujemo njen prevod */
  if(citamo_rec){
127   rec[k]='\0';
    pronadji_prevod(recnik, n, rec, prevod);
129 printf("%s\n", prevod);
  }
131

133 return 0;

135 }
```

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 4.7]

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 4.8]

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 obradjivane:
 Petlje
 Funkcije
 Nizovi
 Strukture

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

Primer 2

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

IZLAZ:
 Greska: Pogresan broj argumenata!
```

[Rešenje 4.9]

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 4.10]

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 4.11]

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
 mi14005 8 8 9 8 10
 ml15112 9 8 8 7 10
 mr15007 10 10 10 10 10
 mn13208 7 7 9 6 10

IZLAZ:
 korisnicko ime: mr15007, prosek ocena: 10.00
```

[Rešenje 4.12]

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 4.13]

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 4.14]

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 4.15]

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

Primer 1

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

Primer 2

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

Primer 3

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

IZLAZ:
 -1
```

[Rešenje 4.33]

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

Primer 1

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

Primer 2

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

Primer 3

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

IZLAZ:
 -1
```

[Rešenje 4.33]

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 drugo na standarnom ulazu, radi sledeće: za cifru u prvoj datoteci, u drugu datoteku se upisuje 0, za slovo se upisuje 1, a za sve ostale karaktere se upisuje 2. Maksimalna dužina naziva datoteka je 20 karaktera.

Primer 2

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

     11112111121120021111211112112000211112111111112112000211112111111112112000
```

[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 standarnog ulaza se učitava ime tekstualne datoteke i prirodan broj k. Podrazumeva se da zadata datoteka sadrži samo slova i beline i

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

[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 neoznacen
  ceo broj. Formirati novi broj koji se dobija
  izbacivanjem svake druge cifre iz polaznog
      broja.
IZLAZ.TXT
  standardnog izbacivanjem
```

Primer 2

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

Primer 3

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

Primer 4

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

[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 izmeų belina i koji nije duži od 200 karaktera. Dozvoljeno je korišćenje specifikatora za čitanje reči. Maksimalan broj reči nije poznat. U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

Primer 1

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

Primer 2

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

[Rešenje 4.33]

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

(a) ispisuje ga [3],

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

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

Primer 1

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

Primer 2

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

[Rešenje 4.33]

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

(a) ispisuje ga, [3]

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

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

Primer 1

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

Primer 2

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

[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 reč `r` pojavljuje bar `n` puta, gde je n prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu `broj_pojavljivanja: linija`. Linije brojati počevši od `1`. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.33]

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 pov cevši od `1`.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 2

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

[Rešenje 4.33]

Zadatak 4.28 Definisati strukturu

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

Primer 5

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

[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 komadne linije ispisati poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 4.33]

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
      studenti.txt
STUDENTI.TXT
 Marko Markovic 5 6 7 8 9 0
 Jelena Jankovic 10 10 10 0
 Filip Viskovic 10 9 8 7 6 0
 Jana Peric 10 10 9 9 8 8 7 7
     0
IZLAZ:
 Jelena Jankovic 10 10 10 0
     10.00
```

Primer 2

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

[Rešenje 4.33]

Zadatak 4.31

a) Napisati C funkciju `int unesiSkup(char *s, FILE* f)` kojom se unosi skup elemenata iz datoteke F. Skup se predstavlja kao niz karaktera, pri čemu su dozvoljeni elementi skupa mala i velika slova abecede, kao i cifre. Unos se prekida kada se naiẹ na znak za novi red ili nedozvoljeni karakter za skup (maksimalan broj elemenata skupa je 1000). Funkcija vraća broj elemenata skupa koji su uspesno učitani.

b) Napisati funkciju `void prebroj(char *s, int *br_slova,int *br_cifara)` kojom se odreụje broj slovnih elemenata skupa (velikih ili malih slova) kao i broj cifara u skupu.

c) Napisati glavni program gde se unose podaci o skupu elemenata. Ime datoteke se zadaje kao argument komandne linije. Na stadardni izlaz ispisati informacije o broju slova i cifara (koristiti funkcije pod a) i b)).

Primer 1

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

Primer 2

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

Primer 3

```
POKRETANJE: ./a.out skup3.txt
SKUP3.TXT
  broJ3
   broj5
IZLAZ:
  broj slova: 4
  broj cifara: 1
```

Primer 4

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

[Rešenje 4.33]

Zadatak 4.32 Definisati strukturu

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

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

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

U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[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

```
/*
    Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
    datoteku izlaz.txt karakter po karakter.
*/

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

int main()
{
   int c;
   FILE *ulaz, *izlaz;

   /*
      Promenljive ulaz i izlaz predstavljaju
      pokazivace na ugradjenu strukturu FILE.
      Unutar ove strukture nalaze se polja neophodna
      za rad sa datotekama.

      Kada zelimo da radimo sa nekom datotekom,
      moramo je prvo otvoriti. Ugradjena funkcija
```

```
   fopen(dat, mode) otvara datoteku sa nazivom
   dat. Datoteka moze biti otvorena za citanje,
   pisanje ili nadovezivanje, sto odredjuje
   argument mode koji moze imati vrednost "r" (read),
   "w"(write) ili "a"(append).
*/

ulaz=fopen("ulaz.txt","r");

/*
   Do neuspesnog otvaranja datoteke moze doci
   ukoliko ne postoji datoteka sa datim nazivom
   ili je putanja do datoteke pogresna. U tom
   slucaju, funkcija fopen vraca pokazivac na NULL
   i tada treba prijaviti gresku. Datoteka stderr
   predstavlja standardnu datoteku u koju se upisuju
   greske. Stderr je podrazumevano postavljen
   na standardni izlaz.

   Ugradjena funkcija exit prouzrokuje zavrsetak programa.
   Argument ove funkcije je jedna od konstanti definisanih
   u biblioteci stdlib.h koje pokazuju da li se program
   zavrsio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).

*/
if(ulaz==NULL)
{
   fprintf(stderr,"error fopen(): Neuspelo otvoranje datoteke ulaz
 .txt za citanje.\n");
   exit(EXIT_FAILURE);
}

izlaz= fopen("izlaz.txt", "w");
if(izlaz==NULL)
{
   fprintf(stderr,"error fopen():  Neuspelo otvoranje datoteke
 izlaz.txt za citanje.\n");
   exit(EXIT_FAILURE);
}

/*
   Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
   Povratna vrednost ove funkcije je ascii kod unetog
   karaktera.

   Funkcija fputc ispisuje karakter c u datoteku izlaz.
*/

while((c=fgetc(ulaz))!=EOF)
   fputc(c,izlaz);

/*
```

```
      Nakon zavrsetka rada sa datotekama, neophodno ih je
      zatvoriti pomocu ugradjene funkcije fclose.
   */
   fclose(ulaz);
   fclose(izlaz);
   return 0;
}
```

Rešenje 4.2

```
/*
   Napisati program koji u datoteci cije se ime navodi kao prvi
   argument komandne linije odredjuje liniju maksimalne duzine i
   ispisuje je na standarni izlaz. Ukoliko ima vise takvih linija,
   ispisati onu koja je leksikografski prva. Mozemo pretpostaviti
   da datoteka ne sadrzi linije duze od 80 karaktera.
*/
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAX_LEN 81

int main(int argc, char* argv[])
{
   char linija[MAX_LEN];
   char max_linija[MAX_LEN];
   int duzina;
   int max_duzina;

   FILE *ulaz, *izlaz;

   /*
     Proveravamo da li poziv programa ima dovoljan broj argumenata.
   */
   if(argc!=2)
   {
      fprintf(stderr,"Greska: program se pokrece sa: %s
    ime_ulazne_datoteke\n", argv[0]);
      exit(EXIT_FAILURE);
   }

   ulaz=fopen(argv[1],"r");
   if(ulaz==NULL)
   {
      fprintf(stderr,"error fopen(): Neuspelo otvoranje datoteke %s
    za citanje.\n", argv[1]);
      exit(EXIT_FAILURE);
   }

   /*
```

```
      Funkcija fgets ucitava jednu liniju teksta maksimalne duzine MAX_LEN
      iz datoteke ulaz u string linija. Ukoliko ucitavanje ne uspe (na primer,
      zato sto smo dosli do kraja datoteke), povratna vrednost ove funkcije
      bice prazan pokazivac (NULL).
*/

max_duzina=0;
while(fgets(linija, MAX_LEN, ulaz)!=NULL)
{
   duzina = strlen(linija);
   /*
      Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u petlju.
      Ovu promenljivu menjamo kada je duzina ucitana linije
      veca od max_duzina ili kada su jednake, ali je ucitana linija
      leksikografski ispred trenutne linije sa maksimalnom duzinom.

      Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii kodova prva dva
      razlicita karaktera stringova s1 i s2 na istim indeksima, ukoliko oni
      postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga osetljiva
      na mala i velika slova (npr 'D' je leksikografski ispred 'p').

   */

   if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija, max_linija)<0))
   {
      strcpy(max_linija, linija);
      max_duzina=duzina;
   }
}

/*
   Funkcija fputs ispisuje string koji je njen prvi argument u datoteku
   koja je njen drugi argument. Sve funkcije za ucitavanje iz datoteka i
   upis u datoteke (fgetc, fputc, fgets, fputs, ...) mozemo koristiti
   i kada radimo sa standardnim ulazom i standardnim izlazom. Kao nazive
   datoteka tada navodimo stdin i stdout.
*/
```

```
76    fputs(max_linija, stdout);

78    fclose(ulaz);
      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>
    #include <stdlib.h>
10  #include <math.h>

12  /*
      Funkcija ucitaj_i_prebroj ucitava brojeve
14    iz datoteke na koju pokazuje f i prebrojava
      koliko je medju njima k-tocifrenih brojeva
16  */
    int ucitaj_i_prebroj(FILE* f, int k)
18  {
      int n;
20    int x;
      int i;
22    int br;

24    /* U datoteci je prvo naveden ukupan broj brojeva. */
      fscanf(f,"%d",&n);
26
      /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
       */
28    if(n<=0)
      {
30      fprintf(stderr, "Greska: broj n mora biti prirodan\n");
        exit(EXIT_FAILURE);
32    }

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

```
40      }
42      return br;
}
44
int broj_cifara(int x)
46 {
    int br_c;
48
    br_c=0;
50
    /*
52      Do while petlja je pogodnija od petlji sa preduslovom
        jer tacno racuna broj cifara i za broj 0.
54  */
    do
56  {
      br_c++;
58    x/=10;
    } while(x);
60
    return br_c;
62 }
64 int main(int argc, char* argv[])
{
66  int n;
    int k;
68  FILE* f;
    int br;
70
    if(argc!=3)
72  {
        fprintf(stderr,"Greska: program se pokrece sa: %s
     naziv_datoteke k \n", argv[0]);
74      exit(EXIT_FAILURE);
    }
76
    f=fopen(argv[1], "r");
78
    if(f==NULL)
80  {
        fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
      .\n", argv[1]);
82      exit(EXIT_FAILURE);
    }
84
    /* Argumenti komandne linije su stringovi. Da bismo konvertovali
     string
86      u ceo broj koristimo ugradjenu funkciju atoi. */
    k = atoi(argv[2]);
88
```

```
    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
            je prestao da vazi uslov petlje, odnosno karakter t[j] je
39          jednak '\0'. Ako vazi drugi slucaj, to znaci da se string
            t nalazi u stringu s pocev od indeksa i i potrebno je
     uvecati
41          brojac br.
       */
43      if (t[j]=='\0')
            br++;
45   }

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

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

63   in= fopen(argv[1],"r");
     if(in==NULL)
65   {
        fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
     .\n", argv[1] );
67      exit(EXIT_FAILURE);
     }
69
     out= fopen(argv[2],"w");
71   if(out==NULL)
     {
73      fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
     .\n", argv[2] );
        exit(EXIT_FAILURE);
75   }

77   printf("Unesi n:");
     scanf("%d", &n);
79
     if(n<=0)
81   {
        fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
83      exit(EXIT_FAILURE);
     }
```

```
85
    fscanf(in,"%s",rec);
87
    while(fgets(linija,MAXL,in)!=NULL)
89  {
       br = broj_pojavljivanja(linija,rec);
91     if (br>=n)
          fprintf(out,"%d: %s\n", br, linija);
93  }
    fclose(in);
95  fclose(out);
    return 0;
97 }
```

Rešenje 4.5

```
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        reciprocni razlomak za svaki razlomak iz niza (npr. za 2/3
    treba upisati 3/2)
      b) ukoliko je navedena opcija y, upisati u datoteku cije je ime
     drugi argument komandne linije
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
15   datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
     obe opcije, sto znaci da je minimalni broj argumenata 3.
17
     Moguci nacini pokretanja:
19   ./a.out ulaz.txt izlaz.txt -x
     ./a.out ulaz.txt izlaz.txt -y
21   ./a.out ulaz.txt izlaz.txt -yx
     ./a.out ulaz.txt izlaz.txt -xy
23
   */
25

   #include <stdio.h>
```

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

#define MAX 100

typedef struct razlomak
{
  int br;
  int im;
} RAZLOMAK;

/*
   Funkcija ucitaj_razlomke ucitava razlomke iz datoteke
   na koju pokazuje f u niz. Dimenzija niza, na koju
   pokazuje pokazivac dim, nije poznata. Prva vrednost
   u datoteci je ukupan broj razlomaka i tu vrednost
   ucitavamo u promenljivu dim.

   Funkcija fscanf se koristi isto kao i funkcija scanf
   uz dodatni prvi argument koji predstavlja naziv
   datoteke iz koje se vrsi ucitavanje.

*/
int ucitaj_razlomke(RAZLOMAK niz[], int* dim, FILE* f)
{
   int i;

   fscanf(f,"%d", dim);
   for (i=0; i<*dim; i++)
   {
      fscanf(f,"%d %d", &niz[i].br, &niz[i].im);
      if (niz[i].im==0)
         return 0;
   }
   return 1;
}

RAZLOMAK reciprocni(RAZLOMAK* r)
{
   RAZLOMAK rec;
   rec.im = r->br;
   rec.br = r->im;
   return rec;
}

float vrednost(RAZLOMAK* r)
{
   return 1.0*r->br/r->im;
}

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
      if(argc!=4)
103   {
         fprintf(stderr,"Greska: program se pokrece sa: %s
       ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n", argv[0]);
105      exit(EXIT_FAILURE);
      }
107

109   in= fopen(argv[1],"r");
      if(in==NULL)
111   {
         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
       .\n", argv[1] );
113      exit(EXIT_FAILURE);
      }
115
      out= fopen(argv[2],"w");
117   if(out==NULL)
      {
119      fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
       .\n", argv[2] );
         exit(EXIT_FAILURE);
121   }

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

125   if (argv[3][0] != '-')
      {
```

```
127       fprintf(stderr, "Greska u zadavanju opcija: program se pokrece
      sa: %s ime_ulazne_datoteke ime_izlazne_datoteke [ -x | -y]\n",
      argv[0]);
129       exit(EXIT_FAILURE);
    }

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

    for(j=1;argv[3][j]!='\0';j++)
135     switch(argv[3][j])
        {
137        case 'x': xoption=1;
                     break;
139        case 'y': yoption=1;
                     break;
141        default:
                     fprintf(stderr, "Greska: nedozvoljeni karakter\n"
      );
143                  exit(EXIT_FAILURE);
        }
145

147   if(ucitaj_razlomke(razlomci, &dim, in)==0)
    {
149      fprintf(stderr, "Greska pri zadavanju razlomaka\n");
         exit(EXIT_FAILURE);
151 }

153   /*
         U zavisnosti od datih opcija, vrsimo upis reciprocnih
155      razlomaka u trazenom formatu.

157      Funkcija fprintf se koristi na isti nacin kao
         funkcija printf uz dodatni prvi argument koji
159      oznacava naziv datoteke u koju se vrsi upis.
    */
161 for (i=0; i<dim;i++)
    {
163    /*
          Ukoliko je brojilac razlomka jednak nuli,
165       nema smisla traziti njegovu reciprocnu vrednost
       */
167    if (razlomci[i].br==0)
          continue;
169
       r = reciprocni(&razlomci[i]);
171
       if (xoption)
173       fprintf(out,"%d/%d ", r.br, r.im);
```

```
175         if (yoption)
               fprintf(out, "%f ", vrednost(&r));
177
            fprintf(out,"\n");
179     }

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

Rešenje 4.6

```
1  /*
     Za svaki automobil poznati su marka, model i cena. Iz datoteke cije
3    se ime zadaje sa standardnog ulaza ucitava se broj automobila a
       potom
     i podaci za svaki automobil. Program treba da:
5    a) izracuna prosecnu cenu po marki kola
     b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
       zadaje
7    kao argument komandne linije, da ispise automobile u tom cenovnom
     rangu zajednu sa prosecnom cenom odgovarajuce marke
9
     Mozemo pretpostaviti da se model i marka sastoje od jedne reci i
11   da svaka od njih sadrzi najvise 30 karaktera kao i da se u datoteci
     nalaze podaci za najvise 100 automobila.
13
  */
15
  #include <stdio.h>
17 #include <stdlib.h>
  #include <string.h>
19 #define MAX 31
  #define MAXA 100
21
  typedef struct automobil
23 {
      char marka[MAX];
25    char model[MAX];
      float cena;
27 } AUTOMOBIL;

29 /*
      Struktura INFO sadrzi naziv
31    marke automobila, prosek cena
      za tu marku i broj automobila
33    te marke
  */
35 typedef struct info
```

```
{
37    char marka[MAX];
      float vrednost;
39    int n;
} INFO;

41
int ucitaj_podatke(FILE* f, AUTOMOBIL a[], int* pn, int max)
43 {
      int i;
45
      fscanf(f,"%d", pn);
47    if (*pn<=0 || *pn>max)
      {
49       printf("Nekorektan unos dimenzije niza automobila\n");
         return 0;
51    }
      for(i=0;i<*pn;i++)
53       fscanf(f,"%s %s %f", a[i].marka, a[i].model, &a[i].cena);

55    return 1;
}
57
/*
59    Funkcija sadrzi ispituje da li se u nizu proseka po marki
      nalazi prosek za marku m. Posto podatak o marki automobila
61    predstavlja string, poredjenje vrsimo pomocu funkcije strcmp.

63    Povratna vrednost ove funkcije je indeks pojavljivanja, ukoliko
      se marka m pojavljuje u nizu proseka, ili -1 u suprotnom.
65 */

67 int sadrzi(INFO p[], int n, char m[])
{
69    int i;
      for(i=0;i<n;i++)
71       if(strcmp(p[i].marka,m)==0)
            return i;
73
      return -1;
75 }

77 /*
      Funkcija informacije_o_markama za niz automobila a dimenzije n
79    racuna proseke cena automobila po markama i smesta ih u niz
      p. Na dimenziju niza p pokazuje pokazivac pn.
81
      Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
83    sumu cena automobila te marke (koju cemo smestiti u polje vrednost
        strukture
      INFO), i broj automobila te marke (koju cemo smestiti u polje
85    n strukture INFO) i da na kraju podelimo ove dve promenljive
      i tako dobijemo prosecnu vrednost cene.
```

```
87
   Za svaki automobil a[i] proveravamo da li se njegova marka vec
89 nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
   vredost cene automobila a[i] i uvecavamo broj automobila sa
91 tom markom. U suprotnom, dodajemo novi element u niz p. Posto
   ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
93 na koju pokazuje pokazivac *pn.
*/
95 void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
{
97    int i,j;
   int ind;
99    for(i=0;i<n;i++)
   {
101      /* Proveravamo da li se marka automobila a[i] vec nalazi u
            nizu p (niz proseka po markama) */
103      ind = sadrzi(p,*pn1,a[i].marka);
         if(ind==-1) /* Ako se ne nalazi, uvodimo novi element niza na
     kraj, na poziciju *pn. */
105      {
            strcpy(p[*pn1].marka, a[i].marka);
107         p[*pn1].vrednost = a[i].cena;
            p[*pn1].n = 1;
109         (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
       */
         }
111      else /* Ako se nalazi, azuriramo polja strukture. */
         {
113         p[ind].vrednost+=a[i].cena;
            p[ind].n++;
115      }
   }
117
   /* Na osnovu sume cena i broja automobila racunamo prosecnu
     vrednost. */
119 for(i=0;i<*pn1;i++)
      p[i].vrednost = p[i].vrednost/p[i].n;
121
}
123
void stampaj_informacije(INFO p[], int n)
125 {
   printf("Informacije o broju automobila i prosecnoj ceni po markama
     :\n");
127 int i;
   for(i=0;i<n;i++)
129    printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
}
131
/*
133 Funkcija stampa automobile cija je cena manja od maksimalne
   cene koju je korisnik naveo u komandnoj liniji da je spreman
```

```
135    da plati, zajedno sa prosecnom cenom za tu marku automobila
   */
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
       n1)
   {
139    /*
          S obzirom da je niz p formiran na osnovu niza a, marka svakog
141       automobila iz niza a se sigurno nalazi u nizu p. Zbog toga
          nije neophodno proveravati da li je povratna vrednost funkcije
143       sadrzi razlicita od -1.
       */
145    int i;
       printf("Kola u vasem cenovnom rangu:\n");
147    for(i=0;i<n;i++)
          if(a[i].cena<g)
149          printf("%s %s %.2f\n", a[i].marka, a[i].model, p[sadrzi(p,n1
       ,a[i].marka)].vrednost);
   }
151
   int main(int argc, char* argv[])
153 {
       AUTOMOBIL kola[MAXA];
155    FILE* f;
       char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
        ulaza. */
157    float granica; /* Maksimalna cena koju je korisnik spreman da
        plati.
                          Zadaje se kao argument komandne linije.
159                    */
       INFO infos[MAXA];
161    int dim_kola,dim_infos;
       int i;
163
       if(argc!=2)
165    {
          fprintf(stderr,"Greska: program se pokrece sa: %s
        gornja_granica_cene \n", argv[0]);
167       exit(EXIT_FAILURE);
       }
169
       /* Argumenti komandne linije su stringovi. Da bismo od stringa
        dobili
171       realan broj, koristimo ugradjenu funkciju atof. */
       granica = atof(argv[1]);
173
       printf("Unesi naziv datoteke:");
175    scanf("%s", dat);

177    f=fopen(dat, "r");

179    if(f==NULL)
       {
```

```
181         fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
        .\n", dat);
            exit(EXIT_FAILURE);
183     }

185     if (ucitaj_podatke(f,kola,&dim_kola,MAXA)==0)
        {
187         fprintf(stderr, "Greska pri ucitavanju podataka\n");
            exit(EXIT_FAILURE);
189     }


191
        informacije_o_markama(kola, dim_kola, infos, &dim_infos);
193
        stampaj_informacije(infos,dim_infos);
195
        stampaj_kandidate(kola, dim_kola, granica, infos, dim_infos);
197
        fclose(f);
199     return 0;
}
```

Rešenje 4.7

```
1  /* Napisati program koji prebrojava mala slova u datoteci test.txt */

3  #include<stdio.h>

5  int main(){

7    FILE* in;
     int c, broj_malih=0;
9
     /*Otvaramo datoteku test.txt za citanje i proveravamo da li smo je
       uspesno otvorili*/
11   in = fopen("test.txt", "r");
     if(in == NULL){
13   printf("Greska!");
     return 0;
15   }

17   /*Citamo karakter po karakter, i ukoliko je procitani
      *karakter  malo slovo, uvecevamo brojac*/
19   while((c=fgetc(in))!=EOF){
     if(islower(c))
21     broj_malih++;
     }
23
     /*Ispisujemo rezultat*/
25   printf("Broj malih slova je: %d\n", broj_malih);
```

```
27    /*Zatvaramo datoteku*/
      fclose(in);
29
      return 0;
31  }
```

Rešenje 4.8

```
/* Napisati program koji prepisuje svaki treci karakter datoteke ulaz
    :txt u datoteku izlaz.txt */
2
#include<stdio.h>
4
int main(){
6
  FILE *in, *out;
8 int c;
  int rbr_karaktera;
10

12 /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
     uspesno otvorili*/
  in = fopen("ulaz.txt", "r");
14 if(in == NULL){
  printf("Greska!");
16 return 0;
  }
18
  /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo dali smo je
     uspesno otvorili*/
20 out = fopen("izlaz.txt", "w");
  if(out == NULL){
22 printf("Greska!");
  return 0;
24 }

26 /* Inicijalizujemo redni broj karaktera koji se cita */
  rbr_karaktera=0;
28
  /* Citamo karakter po karakter iz datoteke sve dok ne stignemo do
     kraja datoteke */
30 while((c=fgetc(in)) != EOF){

32   /* Ukoliko je procitani karakter na poziciji koja je deljiva sa 3
      prepisujemo ga */
     if(rbr_karaktera%3==0)
34   fputc(c, out);

36 /* Uvecavamo redni broj karaktera */
  rbr_karaktera++;
38 }
```

```
40   /*Zatvaramo obe datoteke koje smo otvorili*/
     fclose(out);
42   fclose(in);
     return 0;
44 }
```

Rešenje 4.9

```
/* Kao argumenti komandne linije se zadaju ime datoteke i ceo broj k.
     Napisati program koji na standardni izlaz
2 ispisuje sve linije zadate datoteke cija je duzina veca od k. Moze se
     pretpostaviti da duzina linije nece biti veca
od 80 karaktera */
4
#include<stdio.h>
6 #include<string.h>

8 #define MAXL 81

10 int main(int argc, char* argv[]){

12   FILE* in;
     char linija[MAXL];
14   int k;

16   /*Proveravamo da li je program ispravno pozvan*/
     if(argc!=3){
18   printf("Greska: pogresan broj argumenata!");
     return 0;
20   }

22   /*Otvaramo za citanje datoteku koja se navodi kao prvi argument
       komandne linije*/
     in = fopen(argv[1], "r");
24   if(in == NULL){
     printf("Greska: neuspesno otvaranje datoteke!");
26   return 0;
     }
28
     /*Uzimamo brojevnu vrednost drugog argumenta komandne linije*/
30   k = atoi(argv[2]);

32   /*Citamo liniju po liniju i sve linije duze od k ispisujemo na
       standardni izlaz*/
     while(fgets(linija, MAXL, in) != NULL){
34   if(strlen(linija) > k)
       printf("%s", linija);
36   }
     printf("\n");
38
```

```
  /*Zatvaramo datoteku*/
  fclose(in);
  return 0;
}
```

Rešenje 4.10

```
/* Napisati program koji prebrojava koliko se linija datoteke ulaz.
    txt zavrsava niskom s koja se ucitava sa stan-
dardnog ulaza. Moze se pretpostaviti da duzina linije nece biti veca
    od 80 karaktera, kao i da duzina niske s
ne ce biti veca od 20 karaktera */

#include<stdio.h>
#include <string.h>
#define MAXL 81
#define MAXS 21

/*Funkcija brojLinija proverava koliko linija u datoteci in se
    zavrsava niskom s.
 Funkcija radi tako sto cita jednu po jednu liniju iz datoteke,
 i zatim kraj te linije poredi sa niskom s.*/
int brojLinija(FILE* in, char* s){

  char linija[MAXL];
  int broj_linija = 0;
  int duzina_s = strlen(s);
  int duzina_linije;

  while(fgets(linija, MAXL, in) != NULL){
  duzina_linije = strlen(linija);

  /* Ukoliko je znak za novi red bio indikacija kraja linije,
    uklanjamo ga kako bi mogli da izvrsimo
   *ispravno poredjenje (jer niska s nema novi red na kraju) */
  if(linija[duzina_linije-1]=='\n'){
    linija[duzina_linije-1] = '\0';
    duzina_linije--;
  }

  /*linija + duzina_linije ce nas odvesti na kraj tog stringa, a kada
     oduzmemo duzinu stringa s,
    a kada od toga oduzmemo duzinu niske s, dobicemo bas onoliko
    poslednjih karaktera, koliko
    nam i treba. U primeru uspravna crta (|) oznacava pokazivac
       s              ab
       duzina_s           2
       Linija:            aaabbbdfsssab
                    |
       Linija + duzina linije   aaabbbdfsssab
                           |
```

```
       Linija + duzina linije - 2 aaabbbdfsssab
                              |
      kada kazemo strcmp(linija + duzina_linije - duzina_s, s), mi
    cemo u nasem primeru zaista porediti "ab" i "ab".
   */
  if(strcmp(linija + duzina_linije - duzina_s, s) == 0)
    broj_linija++;
  }
  return broj_linija;
}

int main(){

  FILE* in;
  char s[MAXS];

  /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
    uspesno otvorili*/
  in = fopen("ulaz.txt", "r");
  if(in == NULL){
  printf("Greska: neuspesno otvaranje datoteke!\n");
  return 0;
  }

  /*Ucitavamo nisku*/
  printf("Unesite nisku s: ");
  scanf("%s", s);

  /*Ispisujemo koliko linija iz datoteke se zavrsava sa niskom s*/
  printf("Broj linija: %d\n", brojLinija(in, s));

  /*Zatvaramo datoteku*/
  fclose(in);

  return 0;
}
```

Rešenje 4.11

```
/* Napisati program koji pronalazi maksimum brojeva zapisanih u
    datoteci brojevi.txt */

#include<stdio.h>

int main(){

  FILE* in;
  float broj, max_broj;

  /*Otvaramo datoteku brojevi.txt za citanje i proveravamo da li smo
    je uspesno otvorili*/
```

```
  in = fopen("brojevi.txt", "r");
  if(in == NULL){
  printf("Greska pri otvaranju datoteke!");
  return 0;
  }

  /*
   Kako bismo inicijalizovali promenljivu max_broj,
   citamo jedan broj iz datoteke i smestamo ga u
   ovu promenljivu. */
  fscanf(in, "%f", &max_broj);

  /*U petlji citamo sve ostale brojeve i poredimo ih sa trenutnim
    maksimumom.*/
  while(fscanf(in, "%f", &broj) != EOF){
  if(broj > max_broj)
    max_broj = broj;
  }

  /*Ispisujemo rezultat*/
  printf("Najveci broj je: %.2f\n", max_broj);

  /*Zatvaramo datoteku brojevi.txt*/
  fclose(in);

  return 0;
}
```

Rešenje 4.12

```
/* U datoteci studenti. txt se nalaze informacije o studentima: prvo
    broj studenata, a zatim u pojedinacnim linijama
korisnicko ime i pet poslednjih ocena koje je student dobio. Napisati
     program koji pronalazi studenta koji je
ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da
    broj studenata nece biti veci od 100. */

#include<stdio.h>

#define MAXS 100

/*Definisemo strukturu za cuvanje studenata*/
typedef struct st{
  char korisnicko_ime[8];
  float prosek;
}STUDENT;

int main(){

  FILE *ulaz;
  STUDENT studenti[MAXS];
```

```
19
   int ocena1,ocena2,ocena3,ocena4,ocena5, i=0, i_max_prosek;
21 float max_prosek = 0;

23 /*Otvaramo datoteku studenti.txt za citanje*/
   ulaz = fopen("studenti.txt", "r");
25 if(ulaz == NULL){
   printf("Greska pri otvaranju datoteke!\n");
27 return 0;
   }
29
   /*Ucitavamo liniju po liniju iz datoteke, sve dok ne dodjemo do
     kraja.
31  Korisnicko ime smestamo u niz, a ocene ucitavamo u pomocne
     promenljive ocena1,...ocena5.
    Zatim, na osnovu ocena racunamo prosek.
33
    Ovde paralelno sa ucitavanjem pronalazimo i studenta sa najvecim
     prosekom i
35  pamtimo njegov prosek i njegovu poziciju u nizu studenata,
    Nismo morali ovako. Mogli smo i prvu da ucitamo sve studente, a
     zatim da prodjemo
37  jednom kroz niz i da nadjemo onog sa najvecim prosekom.

39  */
   while(fscanf(ulaz, "%s%d%d%d%d%d", studenti[i].korisnicko_ime, &
     ocena1, &ocena2, &ocena3, &ocena4, &ocena5) != EOF){
41 studenti[i].prosek = (ocena1 + ocena2 + ocena3 + ocena4 + ocena5)
     /5.0;

43 if(studenti[i].prosek > max_prosek){
      max_prosek= studenti[i].prosek;
45    i_max_prosek = i;
   }
47 i++;
   }
49
   /*Ispisujemo rezultat*/
51 printf("korisnicko ime: %s, prosek ocena: %.2f\n", studenti[
     i_max_prosek].korisnicko_ime, studenti[i_max_prosek].prosek);

53 /*Zatvaramo datoteku*/
   fclose(ulaz);
55
   return 0;
57 }
```

Rešenje 4.13

Rešenje 4.14

```c
/* Napisati program koji za rec s maksimalne duzine 20 karaktera koja
     se zadaje sa standardnog ulaza u datoteku
rotacije.txt upisuje sve rotacije reci s */

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

#define MAXS 21

/*Funkcija rotira nisku za jedno mesto u desno.
 Duzina niske n nije obavezan argument. Mogli smo
 i da je racunamo u okviru funkcije, ali kako ce sve niske
 sa kojima radimo biti iste duzine, efikasnije je da jednom
 izracunamo tu duzinu u glavnom programu,
 pa da je prosledjujemo kao argument.*/
void rotiraj_za_1(char* s, int n){
  int i;
  char c = s[0];
  for(i=0; i<n-1; i++){
  s[i] = s[i+1];
  }
  s[n-1] = c;
}

int main(){

  char s[MAXS];
  int n, i;
  FILE * izlaz;

  /*Otvaramo datoteku rotacije.txt za pisanje i proveravamo da li smo
     je uspesno otvorili*/
  izlaz = fopen("rotacije.txt", "w");
  if(izlaz == NULL){
  printf("Greska pri otvaranju fajla!");
  return 0;
  }

  /*Sa standardnog ulaza ucitavamo rec koju treba da rotiramo*/
  scanf("%s", s);

  /*Racunamo njenu duzinu*/
  n = strlen(s);

  /*U petlji, ispisujemo tu rec u datoteku, pa je rotiramo za jedno
    mesto u desno.*/
  for(i=0; i<n; i++){
  fprintf(izlaz, "%s\n", s);
  rotiraj_za_1(s,n);
  }
```

```
49    /*Zatvaramo datoteku rotacije.txt*/
      fclose(izlaz);
51
      return 0;
53
}
```

Rešenje 4.15

```
1  /* Napisati program koji linije koje se ucitavaju sa standardnog
       ulaza sve do kraja ulaza prepisuje u datoteku
   izlaz:txt i to, ako je prilikom pokretanja zadata opcija -v ili -V
       samo one linije koje pocinju velikim slovom,
3  ako je zadata opcija -m ili -M samo one linije koje pocinju malim
       slovom, a ako je opcija izostavljena sve linije.
   Pretpostaviti da linije nece biti duze od 80 karaktera.
5  */

7  #include<stdio.h>
   #include<string.h>
9  #include<ctype.h>

11 #define MAXL 81

13 int main(int argc, char* argv[]){

15    char linija[MAXL];
      FILE* izlaz;
17
      /*Indikatori koji oznacavaju koja opcija je navedena kao argument
        komandne linije
19    vind - ispisuju se recenice koje pocinju velikim slovom
      mind - ispisuju se recenice koje pocinju malim slovom
21     */
      int vind=0, mind = 0;
23
      /*Proveravamo da li je program ispravno pozvan*/
25    if(argc > 2){
      printf("Greska pri pozivanju programa!\n");
27    return 0;
      }
29
      /*Ako opcije nisu zadate, onda treba da se ispisuju sve recenice,
        pa postavljamo oba indikatora na 1*/
31    if(argc == 1){
      vind = mind = 1;
33    }else{

35    /*Proveravamo da li je postavljena neka od opcija -v,-V,-m, -M
       Ako jeste, postavljamo odgovarajuci indikator
37     Ako nije, onda ispisujemo poruku o gresci*/
```

```
  if(strcmp(argv[1], "-v") == 0 || strcmp(argv[1], "-V") == 0)
39    vind = 1;
  else if(strcmp(argv[1], "-m") == 0 || strcmp(argv[1], "-M") == 0)
41    mind = 1;
  else{
43    printf("Greska pri zadavanju opcije!\n");
      return 0;
45  }
  }
47

49  /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo da li smo je
       uspesno otvorili*/
  izlaz = fopen("izlaz.txt", "w");
51  if(izlaz == NULL){
  printf("Greska pri otvaranju datoteke!\n");
53  return 0;
  }
55

  /*Citamo liniju po liniju sa standardnog ulaza i ispisujemo je u
    datoteku.
57   Liniju ispisujemo ukoliko je ispunjen neki od dva uslova:
     1. Izabrana je opcija za ispis malih slova i linija pocinje malim
      slovom
59   2. Izabrana je opcija za velika slova i linija pocinje velikim
     slovom
    NAPOMENA: Kada dodje do kraja ulaza, funkcija fgets vraca NULL
61  */
  while(fgets(linija, MAXL, stdin) != NULL){
63  if( mind && islower(linija[0]) || vind && isupper(linija[0]) ||
    mind && vind)
    fputs(linija, izlaz);
65  }

67  /*Zatvaramo datoteku izlaz.txt*/
  fclose(izlaz);
69

  return 0;
71}
```

Rešenje 4.33

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 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 smera i moze biti mi, ma, mr, ms, mm, mv. GG je oznaka godine upisa. BBB je oznaka broja koji moze biti jednocifren, trocifren ili dvocifren sa vodećim nulama. Na osnovu ovih podataka na standarni

izlaz ispisati ime smera kome student pripada i indeks u obliku broj/godina. U slučaju greške (ukoliko OO kao oznaka smera 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 abedece 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 učešljavanjem 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:	Primer 2:	Primer 3:	Primer 4:
1234 5678	1342 2431	1234 4321	-1234 1234

```
0              1              1              -1
```

[Rešenje A.28]

A.2 Kvalifikacioni zadaci

A.3 Ispitni rokovi

A.3.1 Programiranje 1, i–smer, Završni ispit, januar, 23.01.2016.

Zadatak A.10 *(5 poena)* Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} 2(x-1) & x \geq 1 \\ 0 & \text{inače} \end{cases}$$

[Rešenje A.28]

Grupa I

Zadatak A.11 *(4 poena)* Napisati C program koji sa standardnog ulaza učitava pozitivan ceo broj **n** i na standardni izlaz ispisuje n-ti član niza:

$$a_n = \begin{cases} 1 & n = 1 \\ 3 & n = 2 \\ 2a_{n-1} + 3a_{n-2} + 4 & n \geq 3 \end{cases}$$

Neispravnim ulazom se smatra broj manji ili jednak nuli i u tom slučaju na standardni izlaz ispisati -1. Dozvoljeno je korišćenje nizova. Maksimalna vrednost za **n** je **2000**.

```
Primer 1:      Primer 2:    Primer 3:    Primer 4:
-123           1            4            10
-1             1            39           29523
```

[Rešenje A.28]

Zadatak A.12 *(7 poena)* Napisati funkciju

```
void f3(char s[], char* c, int* br)
```

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 standarni 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 "Minesweeperšastoji se od pravougaone table izdeljene 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)* Definisati strukturu kojim 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:                    Primer 2:                    Primer 3:
Ulaz:              Izlaz:    Ulaz:            Izlaz:      Ulaz             Izlaz:
4                  8         3                -1          3                0
autobus   27 18              AutobuS 65 23                tramvaj 7  29
trolejbus 28 15              Kombi   1  10                tramvaj 3  15
tramvaj   7  29              minibus 6  21                tramvaj 12 12
autobus   27 24             6                            14
27
```

```
Primer 4:                        Primer 5:
Ulaz:              Izlaz:        Ulaz:        Izlaz:
2                  -1            500          -1
autobus 26 20
tramvaj 9  32
```

[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:      Primer 2:          Primer 3:    Primer 4:   Primer 5:
Ulaz:         Ulaz:             Ulaz:       Ulaz:      Ulaz:
5            9                6           3          5
1 2 3 4 5     9 8 7 6 5 4 3 2 1    1 2 3 4 5 6   1 2 3      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 barkodu, 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 prosek zbirova 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 100×100. 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)* Definisati 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 x 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 i 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 pronalizi 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 standarni 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 standarnog 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.

[Rešenje A.28]

Zadatak A.28 Definisati strukturu *Tacka* za predstavljanje tačaka u ravni sa koordinatama tipa *double*. Sa standardnog ulaza se ucitava 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 učitanog niza. Rezultat ispisati na dve decimale. 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.

[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

Rešenje A.28