

PROGRAMIRANJE 1

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

PROGRAMIRANJE 1
Zbirka zadataka sa rešenjima

**Beograd
2016.**

Autori:

dr Milena Vujošević Jančić, docent na Matematičkom fakultetu u Beogradu

dr Jovana Kovačević, docent na Matematičkom fakultetu u Beogradu

Danijela Simić, asistent na Matematičkom fakultetu u Beogradu

Anđelka Zečević, asistent na Matematičkom fakultetu u Beogradu

PROGRAMIRANJE 1

Zbirka zadataka sa rešenjima

Sadržaj

1	Uvodni zadaci	1
1.1	Samo ispis	1
1.2	Celi brojevi	2
1.2.1	Prodavnica	2
1.2.2	Naredba dodele	3
1.2.3	Cifre	4
1.3	Realni brojevi	6
1.3.1	Geometrijski zadaci	8
1.4	Mesano celi i realni (kastovanje)	9
1.5	Ruzni zadaci	10
1.6	Zadaci sa operatorom ?:	10
1.7	Rešenja	12
2	Kontrola toka	29
2.1	Naredbe grananja	29
2.2	Rešenja	39
2.3	Petlje	76
2.3.1	Ispis podataka	76
2.3.2	Obrada celih brojeva, rad sa ciframa broja	79
2.3.3	Unos i obrada veće količine podatka (unos i obrada niza brojeva?, nije sjajno zbog nizova)	82
2.3.4	Rad sa karakterima	86
2.3.5	Računanje sume i proizvoda	87
2.3.6	Dvostruka petlja i ispisivanje slike	91
2.4	Rešenja	99
2.5	Funkcije	138
2.6	Rešenja	150

3	Predstavljanje podataka	187
3.1	Nizovi	187
3.2	Rešenja	201
3.3	Pokazivači	249
3.4	Rešenja	255
3.5	Niske	276
3.6	Rešenja	286
3.7	Višedimenzioni nizovi	310
3.8	Rešenja	317
3.9	Strukture	318
3.10	Rešenja	324
4	Ulaz i izlaz programa	355
4.1	Standardni tokovi	355
4.2	Argumenti komandne linije	355
4.3	Datoteke	355
4.4	Rešenja	369
5	Razni zadaci	397
5.1	Rešenja	397
A	Ispitni zadaci	399
A.1	Testovi/Kolokvijumi	399
A.1.1	Programiranje 1, i-smer, kolokvijum	399
A.2	Kvalifikacioni zadaci	403
A.3	Ispitni rokovi	403
A.3.1	Programiranje 1, i-smer, Završni ispit, januar, 23.01.2016.	403
A.3.2	Programiranje 1, i-smer, Završni ispit, februar, 11.02.2016.	408
A.3.3	1. Grupa, I smer, Programiranje 1 2015/2016, ispit, jun	410
A.3.4	Praktični deo ispita, jun	411
A.4	Rešenja	411

Predgovor

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

Autori

1

Uvodni zadaci

Dodati svuda da budu bar dva test primera, tj svuda gde to moze i ima smisla (za zdravo svete jasno je da nemogu da se napisu dva test primera.)

1.1 Samo ispis

Zadatak 1.1 Napisati program koji na standardni izlaz ispisuje tekst **Zdravo svete!**. **Milena:** Dodati test primer. Voditi racuna da nakon teksta zadatka ide prazan red pa test primer, a ne oznaka za prazan red sa dve crte pa test primer! To je vazno zbog uvlacenja i poravnavanja test primera.

Milena: TODO srediti resenje

[Rešenje 1.1]

Zadatak 1.2 **Jovana:** Za ovaj zadatak nema resenja. To do: dodati (Jovana). *

Milena: Andjelka je bila dala neki smislen predlog kako ovaj zadatak preformulisati da ima smisla.

Napisati program koji na standarni izlaz ispisuje sledeći tekst:

Primer 1

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

[Rešenje 1.2]

1.2 Celi brojevi

Milena: u resenjima komentare nikada ne pisati pored koda vec iznad koda. To je zbog poravnanja. Da bi nam ceo kod bio na isti nacin nazublje, pustacemo ga kroz indent sa određenim parametrima. indent nekako cudno pravna kod koji je tako sa strane i ne bude lepo. Zato je bolje da to u startu nemamo. Postojala je vec kartica na ovu temu na trell-u, a dodala sam jos jednu. Slobodno obrisi ovaj komentar kada ga procitas :-)

Zadatak 1.3 Napisati program za uneti ceo broj ispisuje taj broj, njegov kvadrat i njegov kub. **Nije dobro povezano sa resenjem, pojavljuju se znakovi pitanja. Dodati bar jos jedan test primer.**

Primer 1

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

[Rešenje ??]

Zadatak 1.4 Napisati program koji za uneta dva cela broja ispisuje najpre unete vrednosti, a zatim i njihov zbir, razliku, proizvod, ceo deo pri deljenju prvog broja drugim brojem i ostatak pri deljenju prvog broja drugim brojem. NAPOMENA: *Pretpostaviti da je unos korektan, tj. da druga uneta vrednost nije 0.*

Podesiti resenje. Izbaciti suvisne komentare iz resenja.

[Rešenje 1.4]

1.2.1 Prodavnica

Zadatak 1.5 Napisati program koji pomaže kasirki da izračuna ukupan račun ako su poznate cene dva kupljena artikla. NAPOMENA: *Pretpostaviti da su cene artikla pozitivni celi brojevi.*

Milena: dodati test primere.

[Rešenje 1.5]

Zadatak 1.6 Napisati program koji za unetu količinu jabuka u kilogramima i unetu cenu po kilogramu ispisuje ukupnu vrednost date količine jabuka. NAPOMENA: *Pretpostaviti da je cena jabuka pozitivan ceo broj.*

Primer 1

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

[Rešenje 1.6]

Zadatak 1.7 Napisati program koji pomaže kasirki da obračuna kusur koji treba da vrati kupcu. Za unetu cenu artikla, količinu artikla i iznos koji je kupac dao, program treba da ispiše vrednost kusura. NAPOMENA: *Pretpostaviti da su cene svih artikala pozitivni celi brojevi, kao i da su unete vrednosti ispravne, tj. da se može vratiti kusur.*

Primer 1

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

Milena: Dodati svuda gde može da budu bar dva test primera. U rešenju bi trebalo koristiti tip unsigned zbog pretpostavke da je u pitanju pozitivan broj.

[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. NAPOMENA: *Pretpostaviti da je cena proizvoda pozitivan ceo broj.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite cenu proizvoda: 8367
8367=8*1000+ 3*100 +1*50 +1*10 +7*1
```

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite cenu proizvoda: 934
934=0*1000+ 9*100 +0*50 +3*10 +4*1
```

[Rešenje 1.8]

1.2.2 Naredba dodele

1 Uvodni zadaci

Zadatak 1.9 Date su dve celobrojne promenljive. Napisati program koji razmenjuje njihove vrednosti.

Primer 1

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

[Rešenje 1.9]

Zadatak 1.10 Date su dve celobrojne promenljive a i b . Napisati program koji promenljivoj a dodeljuje njihovu sumu, a promenljivoj b njihovu razliku. NAPOMENA: *Ne koristiti pomoćne promenljive.*

Milena: ako ne zelimo da damo resenje, onda iskomentarisemo naredni red, da se ne bi stavljao link na resenje koje ne postoji.

1.2.3 Cifre

Kod svih zadataka dodato je da podrazumevamo ispravan unos

Broj - pozitivan ili prirodan? Cini mi se da je u R zadacima pozitivan a u I zadacima prirodan :)

Kog tipa da budu broj koji se unosi i cifre? Prosle godine: u uvodnim zadacima je sve bilo int da ih ne zbunjujemo previse. Od naredbe grananja smo poceli da cifre definisemo kao char a broj kao int pa uzmemo apsolutnu vrednost. Kako sada? Trenutno je u resenjima sve int.

Zadatak 1.11 Napisati program koji za uneti pozitivan trocifreni broj na standardni izlaz ispisuje njegove cifre jedinica, desetica i stotina. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Milena: Ako je pretpostavka da je broj pozitivan da onda tip u resenju treba da bude unsigned

Primer 1

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

Primer 1

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

[Rešenje 1.11]

Zadatak 1.12 Napisati program koji učitava pozitivan trocifreni broj sa standardnog ulaza i ispisuje broj dobijen obrtanjem njegovih cifara. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 1

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

[Rešenje 1.12] **Milena:** 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.

Milena: Sve zadatke koji se bave radom sa ciframa broja grupisati na jedno mesto.

Zadatak 1.13 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

NAPOMENA: *Pretpostaviti da je unos ispravan.* **Milena:** 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 1.13]

1 Uvodni zadaci

Zadatak 1.14 Napisati program koji ispisuje broj koji se dobija izbacivanjem cifre desetica u unetom prirodnom broju.

Primer 1

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

[Rešenje 1.14]

Zadatak 1.15 Da li je ovaj zadatak za uvodno potpoglavlje sa celim brojevima? Ima `pow` i kastovanje. Mozda pre da ide u zadatke gde su mesano celi i realni

Napisati program koji za unete prirodne brojeve x , c i p ispisuje broj koji se dobija ubacivanjem cifre c u broj x na poziciji p . Možemo podrazumevati da je broj p manji od ukupnog broja cifara broja x i da numeracija cifara počinje od 0. UPUTSTVO: Koristiti funkciju `pow` iz `math.h` biblioteke.

Milena: Izmenila bih da numeracija cifara pocinje od 0, jer se to uklapa sa tezijskim faktorom i nekako je logicnije. Izmenjeno. Izmeniti i resenja.

Primer 1

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

[Rešenje 1.15]

Zadatak 1.16 Isto i za ovaj zadatak: da li je ovaj zadatak za uvodno potpoglavlje sa celim brojevima? Ima `pow` i kastovanje. Mozda pre da ide u zadatke gde su mesano celi i realni. Ili bez `pow`.

Sa standardnog unosa se unosi prirodan broj n i cifre c_1 i c_2 . Napisati program ispisuje broj dobijen umetanjem cifara c_1 i c_2 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*. **Milena:** Meni ovo deluje kao tekst zadatka cije je resenje dato kod 1.33, mozda gresim. *ODGOVOR: jeste, to je sada povezano*

[Rešenje 1.16]

1.3 Realni brojevi

Zadatak 1.17 Napisati program koji učitava realnu vrednost izraženu u inčima, konvertuje tu vrednost u centimetre i ispisuje je zaokruženu na dve decimale. UPUTSTVO: *Jedan inč ima 2.54 centimetra.*

Primer 1

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

Primer 2

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

[Rešenje 1.17]

Jovana: Zadatak sa konverzijama - funte->kilogrami i slično su razdvojeni. Uz njih nema rešenje a mislim da tako treba i da ostane jer se svi resavaju isto kao in->cm. Da li bismo mogli da zadatak sa C->F da preformulisemo tako da je temperatura ceo broj? To bi bila lepa ilustracija za kastovanje.

Zadatak 1.18 Napisati program koji učitava dužinu izraženu u miljama, konvertuje tu vrednost u kilometre i ispisuje je zaokruženu na dve decimale. UPUTSTVO: *Jedna milja ima 1.609344 kilometara.*

[Rešenje ??]

Zadatak 1.19 Napisati program koji učitava težinu izraženu u funtama, konvertuje tu vrednost u kilograme i ispisuje je zaokruženu na dve decimale. UPUTSTVO: *Jedna funta ima 0.45359237 kilograma.*

[Rešenje ??]

Zadatak 1.20 Napisati program koji učitava temperaturu izraženu u farenhajtima, konvertuje tu vrednost u celzijuse i ispisuje je zaokruženu na dve decimale. UPUTSTVO: $F = \frac{9 \cdot C}{5} + 32$

[Rešenje ??]

Zadatak 1.21 Napisati program koji za unete realne vrednosti a_{11} , a_{12} , a_{21} , a_{22} ispisuje vrednost determinante matrice:

```
a11 a12
a21 a22
```

Pri ispisu vrednost zaokružiti na 4 decimale.

Milena: Umesto verbatim staviti odgovarajući format za prikaz matrice.

Jovana: A koji je to prikaz?

Milena: Milena: Pokusaj google: how to write matrix in latex. Bilo koja varijanta koja ti odgovara a ima onaj standardni matematički izgled je ok.

1 Uvodni zadaci

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 1.21]

1.3.1 Geometrijski zadaci

U svim zadacima dodata je pretpostavka da su duzine pozitivni realni brojevi.

Zadatak 1.22 Napisati program koji za unete dužine stranica pravougaonika ispisuje njegov obim i površinu. Ispisati tražene vrednosti zaokružene na dve decimalne. NAPOMENA: *Pretpostaviti da su sve dužine pozitivne realne vrednosti kao i da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.22]

Zadatak 1.23 Napisati program koji učitava dužinu poluprečnika kruga i ispisuje njegov obim i površinu zaokružene na dve decimalne. NAPOMENA: *Pretpostaviti da su sve dužine pozitivne realne vrednosti kao i da je unos ispravan.*

Primer 1

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

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite duzinu poluprecnika kruga: 14.932  
|| Obim: 93.82, povrsina: 700.46
```

[Rešenje 1.23]

Zadatak 1.24 Milena: 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 može da bude realan broj? Možda bi svi ti geometrijski trebalo da rade sa realnim brojevima? Tako bi bili konzistentni...

Uradjeno Napisati program koji za unetu dužinu stranice jednakostraničnog trougla ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je dužina pozitivna realna vrednost kao i da je unos ispravan.*

Primer 1

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

Primer 1

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

[Rešenje 1.24]

Zadatak 1.25 Pravougaonik čije su stranice paralelne koordinatnim osama zadat je koordinatama suprotnih temena (gornje levo i donje desno teme). Napisati program koji ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da su koordinate realne vrednosti.*

[Rešenje 1.25]

1.4 Mesano celi i realni (kastovanje)

Zadatak 1.26 Napisati program koji za tri uneta cela broja ispisuje njihovu aritmetičku sredinu zaokruženu na dve decimale.

Primer 1

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

Primer 2

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

[Rešenje 1.26]

Zadatak 1.27 Napisati program koji pomaže moleru da izračuna površinu zidova prostorije koju treba da okreči. Za unete dimenzije sobe u metrima (dužinu, širinu i visinu), program treba da ispiše površinu zidova za krećenje 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.

1 Uvodni zadaci

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 1.27]

1.5 Ruzni zadaci

Zadatak 1.28 Napisati program koji za uneta vremena poletanja i sletanja aviona ispisuje dužinu trajanja leta. NAPOMENA: *Pretpostaviti da su poletanje i sletanje u istom danu kao i da su sve vrednosti ispravno unete.*

Ili da ovaj zadatak prebacimo u if, pa da moze da se proverí 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.

Jovana: Mnogo me zamara ovaj zadatak. Nisam sredila resenja.

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 1.28]

1.6 Zadaci sa operatorom ?:

Jovana: Nema resenja ni za jedan od ovih zadataka. Oni su sa i smeru. Danijela, da li ih mozda ti imas?

Zadatak 1.29 Sve zadatke sa operatorom ? grupisati na kraju ovog poglavlja. Napisati program koji za uneta dva cela broja ispisuje njihov maksimum.

[Rešenje 1.29]

Zadatak 1.30 Data su dva cela broja a i b . Napisati program koji dodeljuje promenljivoj *rezultat* 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!

Jovana: Po dogovoru na sastanku, umesto a,b,c zadatak je preformulisan na dve vrednosti - samo a i b . Prilagoditi resenja.

- a) a i b su razliciti brojevi
- b) a i b su parni brojevi
- c) a i b su pozitivni brojevi, ne veci od 100

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

[Rešenje 1.30]

Jovana: Prema dogovoru sa sastanka, naredna 3 primera prelaze u if (pod komentarom su)

Zadatak 1.31 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 1.31]

1.7 Rešenja

Rešenje 1.1

```
1  /*
3      Navedeni program sastoji definise funkciju koja se zove main.
      Program moze da sadrzi vise funkcija,
5      ali obavezno mora da sadrzi funkciju koja se zove main i
      izvršavanje programa uvek pocinje od te funkcije. Pored naziva,
7      zapis svake funkcije cine i povratna vrednost funkcije (u ovom
      slucaju int), lista argumenata koje funkcija koristi (u ovom
9      slucaju prazne zagrade, ()) i telo funkcije koje je ograniceno
      viticastim zagradama ({ i }). O ovim pojmovima bice vise reci
11     u narednim poglavljima.

13     Unutar tela funkcije navode se naredbe. Unutar navedenog programa
      postoji jedna naredba koja predstavlja poziv funkcije printf.
15     Funkcija printf služi za ispis teksta na standardni izlaz (obicno
      ekran). Deklaracija ove funkcije data je u zaglavlju stdio.h
17     koje je potrebno ukljuciti direktivom #include na pocetku
      samog programa.

19

21     Da bismo pokrenuli program, prvo ga moramo prevesti u izvrsnu
      datoteku. Na primer, ako je navedeni program sacuvan kao zdravo.c,
      prevodjenje se vrsi naredbom:

23         gcc zdravo.c

25

27     Ukoliko nije bilo gresaka prilikom prevodjenja, bice generisana
      izvrsna datoteka pod nazivom a.out koja se pokrece navodjenjem
      sledece naredbe:

29         ./a.out

31

33     Ukoliko je bilo gresaka prilikom prevodjenja, one se moraju
      otkloniti a postupak prevodjenja se mora ponoviti.
```

```
35 */
#include<stdio.h>
37
int main()
39 {
    /* printf: funkcija pomocu koje se vrši ispis */
41    /* oznaka \n : prelazak u novi red */
    printf("Zdravo svete!\n");
43
    return 0;
45 }
```

Rešenje 1.2

Rešenje 1.3

```
2  #include <stdio.h>
4  int main()
{
6      /*
           Svaka promenljiva u programu mora biti deklarisan na
           pocetku main funkcije. Deklaracija se sastoji iz naziva
           promenljive (u ovom slucaju n) ispred kog se navodi tip
           promenljive (u ovom slucaju celobrojni tip, int). Svaka
           deklaracija završava se simbolom ";".
12     */
14     int n;
16
17     /*
           Vrednost promenljive se učitava pomocu funkcije scanf koja
           je, kao i funkcija printf, definisana u standardnoj biblioteci
           stdio.h. Argumenti funkcije scanf. koji se navode u zagradama
           ( i ) i razdvajaju zarezima, oznacavaju sledece:
           "%d" - format za tip podatka koji ce biti ucitan
           (%d za int, svaki tip ima svoj format)
           &n - adresa promenljive x (o adresama ce biti vise
           reci u narednim poglavljima).
26     Ucitavanje se vrši sa standardnog ulaza (obicno tastatura).
           */
28     printf("Unesite ceo broj: ");
    scanf("%d", &n);
30
32
    /*
```

1 Uvodni zadaci

```
34     Funkcija printf ispisuje tekst "Uneti broj: ", a nakon toga,
      umesto formata %d, ispisuje vrednost promenljive n.
36 */
printf("Uneti broj: %d\n", n);
38 printf("Kvadrat: %d\n", n*n); /* Umesto formata %d ispisuje se
      vrednost izraza n*n. */
40 printf("Kub: %d\n", n*n*n);   /* Umesto formata %d ispisuje
      se vrednost izraza n*n*n. */
42
44 return 0;
}
```

Rešenje 1.4

```
1
#include<stdio.h>
3
int main()
5 {
7     int x, y, rezultat; /* Promenljive istog tipa mogu se deklarirati
      jedna za drugom. */
9
printf("Unesi vrednost celobrojne promenljive x:");
11 scanf("%d", &x); /* "%d" - specifikator tipa koji treba uneti (%d
      za int)
      &x - adresa promenljive x
13 */
15 printf("Unesi vrednost celobrojne promenljive y:");
scanf("%d", &y);
17
/* 1) ispis unetih vrednosti */
19 printf("x=%d, y=%d\n", x,y); /* umesto prvog %d bice ispisana
      vrednost promenljive x */
      /* umesto drugog %d bice ispisana
      vrednost promenljive y */
21
/* 2) ispis zbira */
23 rezultat = x+y; /* dodelimo vrednost promenljivoj rezultat */
printf("Zbir je %d\n", rezultat);
25
/* 3) ispis razlike */
27 printf("Razlika je %d\n",x-y); /* mozemo ispisivati direktno
      vrednost izraza x-y i bez */
      /* njegovog dodeljivanja posebno
      promenljivoj */
29
```

```

31  /* 4) ispis proizvoda */
    printf("%d*%d=%d\n",x,y,x*y);
33
    /* 5) ispis kolicnika */
35  rezultat = x/y;
    printf("celobrojno deljenje: %d/%d=%d\n",x,y,rezultat); /*
        promenljiva rezultat je celobrojna (int) */
37
                                                /* ona ne
        moze sadrzati realan broj */
                                                /* ukoliko
        je x=7, a y=2, tada ce nakon naredbe */
39
                                                /*
        rezultat=x/y; promenljiva rezultat imati vrednost 2 */
                                                /* a ne
        2.5 */
41
    printf("ostatak pri celobrojnem deljenju: %d %% %d=%d\n",x,y,x%y);
43
                                                /*
        operator % izracunava ostatak pri celobrojnem deljenju */
                                                /* 7%2 ima
        vrednost 1 (jer je 7=3*2+1) */
45
                                                /* oznaku
        % u naredbi printf pisemo %% */
    return 0;
47 }

```

Rešenje 1.5

Rešenje ovog zadatka svodi se na rešenje zadatka 1.4, na deo koji se odnosi na izračunavanje zbira dva broja. Zbog pretpostavke da su cene artikala pozitivni celi brojevi, tip promenljivih za artikle treba da bude `unsigned int`.

Rešenje 1.6

Rešenje ovog zadatka svodi se na rešenje zadatka 1.4, na deo koji se odnosi na izračunavanje proizvoda dva broja. Zbog pretpostavke da su cene artikala pozitivni celi brojevi, tip promenljivih za artikle treba da bude `unsigned int`.

Rešenje 1.7

```

1
#include <stdio.h>
3
int main(){
5     int cena;
    int kolicina;
7     int iznos;
    int kusur;
9

```

1 Uvodni zadaci

```
11  /*
    Ucitavamo potrebne podatke. Unutar jednog scanf-a mozemo
    ucitati vise podataka odjednom. Za svaki treba navesti
13  odgovarajuci format za tip podataka koji se unosi.
    */
15  printf("Unesite redom cenu, kolicinu i iznos: ");
    scanf("%d %d %d", &cena, &kolicina, &iznos);
17
    /* Izracunavamo kusur: */
19  kusur=iznos - kolicina*cena;
21
    /* I ispisujemo trazenu vrednost: */
    printf("Kusur je %d dinara.\n", kusur);
23
    return 0;
25 }
```

Rešenje 1.8

```
1  #include <stdio.h>
3
4  int main()
5  {
6      int x;
7      printf("Unesi cenu:");
8      scanf("%d", &x);
9
10     /*
11     Na primer, neka je uneta cena 8347 dinara.
12     Vrednost x/1000 predstavlja broj novcanica
13     od 1000 dinara pomocu kojih mozemo sakupiti
14     celokupnu sumu. 8347 celobrojno deljeno sa
15     1000 (operacija / nad celim brojevima) iznosi 8.
16     */
17     printf("%d=%d*1000+ ", x,x/1000);
18     /*
19     Potrebno nam je 8 novcanica od
20     1000 dinara, a koliko nam je potrebno ostalih
21     novcanica? Za to moramo pristupiti preostaloj
22     sumi. Jedan nacin je da nadjemo ostatak pri deljenju
23     unete vrednosti x (u primeru 8347) sa 1000 (operacija %).
24     On iznosi 347. Ovu vrednost dodeljujemo promeljivoj x.
25     */
26     x=x%1000;
27     /*
28     Nastavljamo postupak trazanjem broja novcanica
29     od 100 dinara.
30     */
31     printf("%d*100 +", x/100);
32     x=x%100;
```



```

33 printf("%d*50 +",x/50);
    x=x%50;
35 printf("%d*10 +", x/10);
    x=x%10;
37 printf("%d*1\n", x);
    return 0;
39 }

```

Rešenje 1.9

```

2  #include<stdio.h>
   int main()
4  {
    int x,y;
6    int t;

8    printf("Unesi dve celobrojne vrednosti:");
    scanf("%d%d",&x,&y);

10   printf("pre zamene: x=%d, y=%d\n",x,y);
12   t=x; /* promenljiva t dobija vrednost promenljive x */
        x=y; /* promenljiva x dobija vrednost promenljive y */
14   y=t; /* promenljiva y dobija vrednost promenljive t */
    printf("posle zamene: x=%d, y=%d\n",x,y);
16   return 0;
   }

```

Rešenje 1.10

Rešenje 1.11

```

1  #include <stdio.h>
3  int main()
   {
5     int x;
     int cifra_jedinice;
7     int cifra_desetice;
     int cifra_stotine;

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

13    /*
15    Na primer, neka je uneti broj 374. Potrebno je da koriscenjem
    racunskih operacija za rad sa celim brojevima pristupimo
    njegovoj cifri jedinice, cifri desetice i cifri stotine.

```

1 Uvodni zadaci

```
17
18
19     Primetimo najpre sledece:
20     374/10 = 37
21     374%10 = 4
22     Dakle, operacijama celobrojnog deljenja i ostatka pri deljenju
23     mozemo iz svakog broja izdvojiti njegovu poslednju cifru (u
24     ovom slucaju 4) i broj sastavljen od svih cifara osim poslednje
25     (u ovom slucaju 37).
26
27     Cifri jedinice sada lako pristupamo koriscenjem ostatka pri
28     deljenju sa 10. Ona iznosi upravo 4.
29
30     Pri trazanju cifre desetice mozemo ponovo primeniti princip
31     izdvajanja poslednje cifre kao ostatka pri deljenju sa 10.
32     Razlika je sto ne mozemo deseticu izdvojiti ako primenimo %10
33     na 374 (tine dobijamo 4), vec %10 primenjujemo na 37, pri cemu
34     37 dobijamo kao ceo deo pri deljenju broja 374 brojem 10.
35     Dakle, cifru desetice dobijamo kao (374/10)%10.
36
37     S obzirom da znamo da je u pitanju trocifreni broj, cifru
38     stotine mozemo izdvojiti celobrojnim deljenjem sa 100: 374/100
39     iznosi upravo 3.
40
41     */
42     cifra_jedinice = x%10;
43     cifra_desetice = (x/10)%10;
44     cifra_stotine = x/100;
45
46     printf("jedinica %d, desetica %d, stotina %d\n", cifra_jedinice,
47           cifra_desetice,
48           cifra_stotine);
49
50     /*
51     2. nacin, bez uvodjenja dodatnih promenljivih cifra_jedinice,
52     cifra_desetice i cifra_stotine:
53
54     printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10, x
55     /100);
56     */
57     return 0;
58 }
```

Rešenje 1.12

```
1
2     #include <stdio.h>
3     int main()
4     {
5         int x;
6         int obrnuto_x;
7     }
```

```
9   int cifra_jedinice;
   int cifra_desetice;
   int cifra_stotine;

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

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

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

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

25   return 0;
}
```

Rešenje 1.13

```
1  #include <stdio.h>

3  int main(){

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

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

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

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

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

27     /* Izracunavamo sumu kvadrata cifara */
```

1 Uvodni zadaci

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

31
/* Odredjujemo broj zapisan istim ciframa ali u obrnutom redosledu
*/
33 broj_obrnuto= j*1000+d*100+s*10+h;
printf("Broj u obrnutom poretku: %d\n", broj_obrnuto);

35

37 /* Odredjujemo broj u kojem su cifra jedinica i
    cifra stotina zamenile mesta
39 */
    broj_zamena=h*1000+j*100+d*10+s;
41 printf("Broj sa zamenjenom cifrom jedinica i stotina: %d\n",
    broj_zamena);

43
    return 0;
45 }
```

Rešenje 1.14

Rešenje 1.15

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

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

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

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

17
    /* Odredjujemo deo broja koji se nalazi levo od pozicije p */
19     levo=x/(int)pow(10, p-1);

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

23
    /* Ispisujemo dobijenu vrednost */
25     printf("Rezultat je: %d\n", novo_x);
```

```
27  /* Završavamo sa programom */  
    return 0;  
29  }
```

Rešenje 1.16

```
#include <stdio.h>  
2  #include <math.h>  
    #include <limits.h>  
4  
/* u zaglavlju limits.h  
6 su definisane maksimalne i minimalne  
vrednosti za svaki tip podataka  
8 npr. INT_MAX konstanta je najveći ceo  
broj koji može da se stavi  
10 u promenljivu tipa int  
zbog toga za poslednji test primer  
12 ne dobijamo željeni broj  
jer je došlo do prekoracenja  
14 novibroj je veći od INT_MAX  
*/  
16  
/* test primeri:  
18 broj: 140  
c1: 2  
20 c2: 3  
  
22 novibroj: 13240  
-----  
24 broj: 526  
c1: 7  
26 c2: 4  
  
28 novibroj: 54726  
-----  
30 broj: 25  
c1: 9  
32 c2: 5  
  
34 novibroj: 5925  
-----  
36 test primer koji dovodi do prekoracenja, pa zbog toga  
ne dobijamo željeni rezultat:  
38  
39 broj: 100000000  
40 c1: 5  
c2: 1  
42  
43 novibroj: neočekivan rezultat ---> PREKORACENJE  
44
```

1 Uvodni zadaci

```
46  */
47  int main(){
48  int broj,c1,c2,z1,z2;
49  int novibroj;
50  int dostatak1, dostatak2 ;
51  printf("unesi broj: ");
52  scanf("%d", &broj);
53  printf("unesi c1: ");
54  scanf("%d", &c1);
55  printf("unesi c2: ");
56  scanf("%d", &c2);

57  /* najbolje odmah da se kastuje z1 jer se kasnije cesto
koristi u racunu pa da ne ponavljamo (int) */
58  // za stotine pozicija je 3 ---> z1 = (int)pow(10,3-1);
59  z1 = (int)pow(10,2);

60  dostatak1 = broj % z1;

61  /*
62  levi ostatak je u stvari ovaj deo --> broj / z1 * z1 * 10
63  inace taj deo moze da se racuna i kao --> (broj - broj % z1) * 10
64  */
65  novibroj = broj / z1 * z1 * 10 + z1 * c1 + dostatak1 ;

66  //sada u novibroj insertujemo cifru c2 na poziciju 4 - za hiljade
67
68  z2 = (int)pow(10,3);

69  dostatak2 = novibroj % z2;

70  /*
71  levi ostatak je u stvari ovaj deo --> broj / z2 * z2 * 10
72  inace taj deo moze da se racuna i kao --> (broj - broj % z2) * 10
73  */
74  novibroj = novibroj / z2 * z2 * 10 + z2 * c2 + dostatak2 ;

75
76  printf("Novi broj je: %d\n", novibroj);
77  printf("Maksimalna vrednost za int je: %d\n", INT_MAX);
78
79  return 0;
80  }
```

Rešenje 1.17

```
1  #include <stdio.h>

3  int main()
4  {
```

```
5      float in; /* float - realni tip jednostruke tacnosti */
7      float cm;

9      printf("Unesi broj inca: ");
10     scanf("%f", &in);
11     /* "%f" - format za unos/ispis float promenljivih */

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

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

17     return 0;
18 }
```

Rešenje 1.21

Rešenje 1.22

```
2  #include <stdio.h>

4  int main()
5  {
6      float a, b;
7      float obim, površina;

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

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

15     /* Površina */
16     površina=a*b;

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

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

Rešenje 1.23

1 Uvodni zadaci

```
1 #include <stdio.h>
  #include <math.h>
3
4 /* Biblioteka math.h sadrzi veliki broj matematickih
5    funkcija i konstanti. U ovom zadatku je koristimo
6    zbog konstante pi (M_PI)
7
8    Za prevodjenje je neophodno ukljuciti opciju -lm
9    npr. gcc primer.c -lm
10 */
11 int main()
12 {
13     float r;
14     float O;
15     float P;
16     printf("Unesite duzinu poluprecnika kruga:");
17     scanf("%f", &r);
18
19     O=2*r*M_PI;
20     P=r*r*M_PI;
21
22     printf("Obim: %.2f, površina: %.2f\n",O,P);
23
24     return 0;
25 }
```

Rešenje 1.24

```
#include <stdio.h>
2 #include <math.h>
3
4 int main(){
5     float a, b, c;
6     float obim, s, površina;
7
8     /* Ucitavamo potrebne podatke */
9     printf("Unesite duzine stranica trougla: ");
10    scanf("%f %f %f", &a, &b, &c);
11
12    /* Obim */
13    obim=a+b+c;
14
15    /* Površina - koristimo Heronov obrazac*/
16    s=obim/2;
17    površina=sqrt(s*(s-a)*(s-b)*(s-c));
18
19    /* Ispisujemo trazene vrednosti */
20    printf("Obim: %.2f\n", obim);
21    printf("Površina: %.2f\n", površina);
22 }
```



```
24     return 0;  
    }
```

Rešenje 1.25

Rešenje 1.26

```
1  #include<stdio.h>  
3  
4  int main()  
5  {  
6      int a, b, c;  
7      float as;  
9      printf("Unesite tri cela broja:");  
      scanf("%d%d%d",&a,&b,&c);  
11  
      /* pogresan nacin: as = (a+b+c)/3;  
13  
      Ukoliko podelimo zbir a+b+c sa 3, to ce biti primena  
15      operatora / na dva cela broja. Na ovaj nacin izracunacemo  
      koliko iznosi a+b+c celobrojno podeljeno sa 3. To znaci da  
17      ce za unete vrednosti 11, 5 i 4 aritmeticka sredina biti  
      6.00. Zaista, zbir 11+5+4 iznosi 20, a kada 20 celobrojno  
19      podelimo sa 3 dobijamo 6. Ovu celobrojnu vrednost dodeljujemo  
      realnoj promenljivoj as, cime se ona konvertuje u 6.000000 i  
21      ispisujemo je zaokruzenu na dve decimale. Izlaz iz programa bi  
      bio pogresan: 6.00.  
23  
      Da bismo dobili kolicnik prilikom primene operatora / na dva  
25      cela broja, a ne celobrojno deljenje, jedan argument mora da  
      bude realan broj. Jedan nacin je da umesto sa celobrojn  
27      trojkom (3) deljenje izvedemo sa realnom trojkom (3.0):  
      */  
29      as=(a+b+c)/3.0;  
31  
      /*  
33      Trazeni kolicnik mozemo dobiti na razne nacine:  
      as=1.0*(a+b+c)/3;  
35      ili  
      as=(0.0+a+b+c)/3;  
37      ili  
      as=((float)(a+b+c))/3;  
39      itd.  
      */  
41  
      printf("Aritmeticka sredina unetih brojeva je %.2f\n", as);  
43      return 0;
```

```
}
```

Rešenje 1.27

```
1  #include <stdio.h>
3
5  int main(){
7      int duzina, sirina, visina;
9      int cena;
11     float površina_za_krecenje;
13     float ukupna_cena;
15
17     /* Ucitavamo duzinu, sirinu i visinu sobe */
19     printf("Unesite dimenzije sobe: ");
21     scanf("%d %d %d", &duzina, &sirina, &visina);
23
25     /* Ucitavamo cenu krecenja */
27     printf("Unesite cenu po kvadratnom metru: ");
29     scanf("%d", &cena);
31
33     /* Povrsina za krecenje odgovara površini kvadra -
        bez poda jer se on ne kreće */
    površina_za_krecenje=0.8*(duzina*sirina+
                               2*duzina*visina+
                               2*sirina*visina);
    ukupna_cena=površina_za_krecenje*cena;
35
37     /* Ispisujemo tražene podatke */
39     printf("Moler treba da okreće %.2f kvadratna metra\n",
41           površina_za_krecenje);
43
45     printf("Cena krecenja je %.2f\n", ukupna_cena);
47
49     /* Završavamo sa programom */
51     return 0;
53 }
```

Rešenje 1.28

```
1  /* Napisati program koji učitava sa standardnog ulaza vreme poletanja
   i vreme
2  sletanja aviona, a potom ispisuje duzinu trajanja leta. Mozemo
   pretpostaviti da
3  su poletanje i sletanje u istom danu. */
4
5  #include <stdio.h>
6
7  int main(){
```

```
8
10 int poletanje, poletanje_sat, poletanje_minut, poletanje_sekund;
12 int sletanje, sletanje_sat, sletanje_minut, sletanje_sekund;
14 int duzina, duzina_sat, duzina_minut, duzina_sekund;

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

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

24 /* Pretvoricemo i vreme poletanja i vreme sletanja u sekunde */
26 poletanje=poletanje_sat*3600+poletanje_minut*60+poletanje_sekund;
28 sletanje=sletanje_sat*3600 + sletanje_minut*60 +sletanje_sekund;

30 /* I izracunati razliku u sekundama */
32 duzina=sletanje-poletanje;

34 /* Izdvajamo broj sati, broj minuta i broj sekundi */
36 duzina_sat=duzina/3600;
38 duzina_minut=(duzina%3600)/60;
    duzina_sekund=(duzina%3600)%60;

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

    return 0;
}
```

Rešenje 1.29

Rešenje 1.30

Rešenje 1.31

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 standardni 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 računa 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 unositi 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  
|| Najveća cifra je: 8
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 238  
|| Greška: Niste uneli četvorocifren 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 Armstrongov.
```

Primer 2

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

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 84  
|| Greška: 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 ($1 \leq k \leq 189$) ispisuje cifru koja se nalazi na k -toj poziciji datog niza.

Primer 1

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

Primer 2

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

[Rešenje 2.18]

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

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 standardnog 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 poprilično 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 standardnog ulaza se unosi karakter *c*. Napisati program koji:

2 Kontrola toka

- 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 najveći mogući šestocifren broj. Dobijeni šestocifreni broj ispisati na standardni izlaz. Ako neki od unetih brojeva nije trocifren, smatrati da ulaz nije ispravan. **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:

- dužinu zajedničkog dela ta dva intervala
- 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 standardnog ulaza se unosi realan broj x i broj k koje može biti 1, 2 ili 3. Napisati program koji izračunava vrednost funkcije $F(k, x) = f(f(f(\dots f(x))))$ gde je funkcija f prime-njena k -puta. U slučaju neispravnog ulaza, odštampati odgovarajuću poruku o grešci. **dobiti 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 is-pisuje 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. Napisati 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.me-sec.godina.* proverava da li je korektan.

Primer 1

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

Primer 2

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

[Rešenje 2.30]

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 2.101]

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 $P \text{ op1 } Q \text{ op2 } R$ 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

```
2  /*
   Napisati program koji za 2 cela broja uneta sa standardnog ulaza
   ispisuje njihov minimum na standardni izlaz.
4  */
6  #include <stdio.h>
   int main()
8  {
   int a,b;
10  int min1;
   int min2;
12  int min3;
```

2 Kontrola toka

```
14     scanf("%d%d",&a,&b);
16
17     /* 1. nacin */
18     if (a<b)
19         min1=a;
20     else
21         min1=b;
22
23     printf("Minimum unetih brojeva (1.nacin) je %d\n",min1);
24
25     /* 2. nacin */
26     min2 = (a<b) ? a : b;
27     printf("Minimum unetih brojeva (2.nacin) je %d\n",min2);
28
29     /* 3. nacin */
30     min3=a;
31     if (b<a)
32         min3 = b;
33     printf("Minimum unetih brojeva (3.nacin) je %d\n",min3);
34
35     return 0;
36 }
```

Rešenje 2.2

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()
9 {
10     int x;
11     printf("Unesi godinu:");
12     scanf("%d",&x);
14
15     if ((x%4==0 && x%100!=0) || x%400==0)
16         printf("Godina je prestupna\n");
17     else
18         printf("Godina nije prestupna\n");
19     return 0;
20 }
```


Rešenje 2.4

```
1  /*
   Napisati program koji za uneti ceo broj ispisuje njegovu reciprocnu
   vrednost.
3  Ukoliko je uneti broj jednak nuli, ispisati poruku "Nedozvoljeno
   deljenje nulom".
   */
5
6  #include <stdio.h>
7
8  int main()
9  {
10     int x;
11     float rx;
12
13     printf("Unesi jedan ceo broj:");
14     scanf("%d",&x);
15
16     /*
17      obratiti paznju:
18      x==0 - relacija jednakosti (da li je vrednost promenljive x
19      jednaka nuli)
20      x=0 - naredba dodele (promenljiva x dobija vrednost nula)
21     */
22
23     if (x==0)
24         printf("Nedozvoljeno deljenje nulom\n");
25     else
26     {
27         rx = 1.0/x;
28         printf("Recipročna vrednost unetog broja:%f\n",rx);
29     }
30     return 0;
31 }
```

Rešenje 2.5

```
1  #include <stdio.h>
2  /*
3  Napisati program koji za uneti ceo broj x ispisuje da li je jednak
   nuli,
   manji od nule ili veci od nule.
   */
5
6  int main()
7  {
8     int x;
9     printf("Unesi ceo broj:");
10     scanf("%d",&x);
```

2 Kontrola toka

```
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  */
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
25  return 0;
}
```

Rešenje 2.6

```
1  /*
        Napisati program koji za uneto vreme ispisuje koliko je sati i
        minuta ostalo
3  do ponoci.
4  */
5  #include<stdio.h>
6
7  int main()
8  {
9      int sati;
10     int minuti;
11     int preostali_sati;
12     int preostali_minuti;
13
14     printf("Unesi vreme (broj sati u intervalu [0,24), broj minuta u
        intervalu [0,60]):");
15     scanf("%d%d",&sati,&minuti);
16
17     preostali_sati = 24-sati-1;
18     preostali_minuti = 60-minuti;
19     if (preostali_minuti==60)
20     {
21         preostali_sati++;
22         preostali_minuti=0;
23     }
24
25     printf("Do ponoci je ostalo %d sati i %d minuta\n", 24-sati-1, 60-
        minuti);
26     return 0;
27 }
```

Rešenje 2.7

```

1  /*
3  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
5  artikal dobija za 1 dinar, napisati kolika je ukupna cena, kao i
       koliko
       dinara se uštedi zahvaljujući popustu.
7  */

9  #include <stdio.h>
10 int main()
11 {
12     int a,b,c;
13     int min;
14     int min1;
15     int min2;
16     int cena_bez_popusta, cena_sa_popustom;
17
18     scanf("%d%d%d",&a,&b,&c);
19
20
21     if (a<b)
22         if (a<c) /* poredak: a<b,a<c => a,b,c ili a,c,b */
23             min=a;
24         else /* poredak: a<b, a>=c => a<b, c<=a => c,a,b */
25             min=c;
26     else /* b<=a */
27         if (b<c) /* poredak: b<=a,b<c => b,a,c ili b,c,a */
28             min=b;
29         else /* poredak: b<=a, c<=b => c,b,a */
30             min=c;
31
32     printf("Minimum unetih brojeva (1.nacin) je %d\n",min);
33
34     /* 2. nacin */
35     /* najpre odredimo minimum brojeva a,b */
36     if (a<b)
37         min1=a;
38     else
39         min1=b;
40
41     if (c<min1)
42         min1=c;
43     printf("Minimum unetih brojeva (2.nacin) je %d\n",min1);
44
45     /* 3. nacin */
46     min2=a;
47     if(min2>b)
48         min2=b;

```

2 Kontrola toka

```
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
6  */
7  #include<stdio.h>
8  int main()
9  {
10     int a,b,c;
11     int s;
12     printf("Unesi prvi ceo broj:");
13     scanf("%d",&a);
14     printf("Unesi drugi ceo broj:");
15     scanf("%d",&b);
16     printf("Unesi treci ceo broj:");
17     scanf("%d",&c);
18
19     s=0; /* inicijalizujemo promenljivu s na nulu */
20
21     if (a>0)
        s=s+a; /* naredba dodele: vrednost izraza a desne strane znaka
jednakosti
22         dodeljujemo promenljivoj sa leve strane znaka
23         jednakosti.

```

2 Kontrola toka

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

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

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

33

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

Rešenje 2.11

```
1  /*
3  Napisati program koji za realan broj unet sa standardnog ulaza
   ispisuje njegovu apsolutnu vrednost.
5
6  */
7
8  #include<stdio.h>
9  #include<math.h>
10 #include<stdlib.h>
11 int main()
12 {
13     float x;
14     float y;
15
16     printf("Unesi jedan realan broj:");
17     scanf("%f",&x);
18
19     /* 1. nacin */
20     if (x>0)
21         y=x;
22     else
23         y=-x;
24     printf("Apsolutna vrednost broja %f je %f\n",x,y);
25
26     /* 2. nacin */
27     y=x;
28     if (y<0)
29         y=-y;
30
31     printf("Apsolutna vrednost broja %f je %f\n",x,y);
32
33     /* 3. nacin - pogresan!*/
```

```
    y=abs(x); /* funkcija abs vraca ceo broj! za racunanje apsolutne
35      vrednosti realnog broja treba koristiti funkciju fabs */
37      /* funkcija abs se nalazi u zaglavlju stdlib.h */
    printf("Apsolutna vrednost broja %f je %f\n",x,y);

    /* 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' :
17      case 'E' :
18      case 'I' :
19      case 'O' :
20      case 'U' :
21      case 'a' :
22      case 'e' :
23      case 'i' :
24      case 'o' :
25      case 'u' : printf("Uneli ste samoglasnik\n");
26                  break;
27      default : printf("Niste uneli samoglasnik\n");
28                  break;
29  }
30
       return 0;
32 }
```

Rešenje 2.13

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



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

Rešenje 2.14

```
1  /*
2  Napisati program koji od korisnika zahteva da unese
3  cetvorocifreni broj. Program za taj broj proverava
4  da li su cifre uredjene rastuce, opadajuce ili nisu
5  uredjene i stampa odgovarajucu poruku na standardni
6  izlaz. Voditi racuna o nekorektnim unosima. Na primer,
7  pokretanje programa moze da izgleda ovako:
8
9  Unesi jedan cetvorocifreni broj: -1357
10 Cifre su mu uredjene neopadajuce.
11
12 ili ovako
13
14 Unesi jedan cetvorocifreni broj: 9952
15 Cifre su mu uredjene nerastuce.
16
17 ili ovako
18
19 Unesi jedan cetvorocifreni broj: 9572
20 Cifre su mu nisu uredjene.
21
22 Unesi jedan cetvorocifreni broj: 123
23 Uneti broj nije cetvorocifren.
24
25 */
26
27 #include <stdio.h>
28 #include <stdlib.h>
29
30 int main()
31 {
32     int x;
33     char c1;    /* cifre su brojevi {0,1,2,3,4,5,6,7,8,9} */
34     char c10;
35     char c100;
36     char c1000;
37
38     printf("Unesi jedan cetvorocifreni broj:");
39     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
47 if (x<1000 || x>9999)
    printf("Uneti broj nije cetvorocifren\n");
49 else
    {
51     c1 = x%10;
53     c10 = (x/10)%10;
55     c100 = (x/100)%10;
57     c1000 = (x/1000)%10;
59
61     printf("Cifre broja: %d,%d,%d,%d\n",c1000,c100,c10,c1);
63
65     if (c1000<=c100 && c100<=c10 && c10<=c1)
        printf("Cifre su uredjene neopadajuce \n");
        else if (c1000>=c100 && c100>=c10 && c10>=c1)
            printf("Cifre su uredjene nerastuce \n");
        else
            printf("Cifre nisu uredjene\n");
    }
    return 0;
}
```

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 unetog karaktera
   ispisuje se odgovarajuca poruka na standardni izlaz:
5 k - proverava da li su date tacke temena pravougaonika cije su
   stranice paralelne koordinatnim osama i
   u slucaju da jesu, ispisuje obim datog pravougaonika; mozemo
   podrazumevati da ce korisnik koordinate tacaka
7 unosi redom A,B,C,D, pri cemu ABCD opisuje pravougaonik cije su
   stranice AB,BC,CD i DA, a dijagonale AC i BD
   na primer, tacke (1,1),(2,1),(2,2),(1,2) cine pravougaonik cije
   su stranice paralelne koordinatnim osama i ciji je obim 4
9 a tacke (1,1),(2,2),(3,3),(4,4) ne cine pravougaonik
h - proverava da li su unete tacke kolinearne i ukoliko jesu,
   ispisati jednacinu prave kojoj pripadaju
11 na primer, tacke (1,2),(2,3),(3,4),(4,5) su kolinearne i
   pripadaju pravoj y=x+1
   tacke (1,1),(1,2),(1,3),(1,4) su kolinearne i pripadaju pravoj x
   =1
13 a tacke (1,1),(2,1),(2,2),(1,2) nisu kolinearne
```

```

j - Kramerovim pravilom proverava da li je dati sistem jednačina
15 x1 * p + x2 * q = x4 - x3
y1 * p + y2 * q = y4 - y3
17   odredjen, neodredjen ili nema resenja, i u slucaju da je odredjen
   ispisati resenja.
   na primer, za unete koordinate (1,1),(1,1),(1,0),(2,2) sistem
   nema resenja
19       za unete koordinate (1,1),(1,1),(1,1),(1,1) sistem je
   neodredjen ili nema resenja
       za unete koordinate (6,1),(8,3),(10,-4),(9,1) sistem
   ima jedinstveno resenje 4.30, 3.10
21
*/
23
#include<stdio.h>
25 #include<math.h>
int main()
27 {
    char c;
29     float x1,y1,x2,y2,x3,y3,x4,y4;
    float kab,kbc,kad;
31     float dab,dad;
    float delta, deltap, deltaq;
33     float 0;
    float k,n;
35
    printf("Unesi jedan karakter:");
37     scanf("%c",&c);

    printf("Unesi realne koordinate 4 tacke:");
39     scanf("%f%f%f%f%f%f", &x1, &y1, &x2, &y2, &x3, &y3, &x4, &y4);
41
    switch (c)
43     {
        case 'k':
45         if (y1==y2 && y3==y4 && x1==x4 && x2==x3)
            {
47                 dab = sqrt(pow(x1-x2,2)+pow(y1-y2,2)); // funkcija pow(x
,y) racuna vrednost stepene funkcije x^y
                dad = sqrt(pow(x1-x4,2)+pow(y1-y4,2)); // x i y su
                realne vrednosti
49                0 = 2*dab + 2*dad;
                printf("Obim pravougaonika je %f\n",0);
51            }
            else
53                printf("Tacke ne cine pravougaonik sa stranicama koje su
                paralelne koordinatnim osama\n");
                break;
55            case 'h':
                if ((x1-x2)!=0) // ukoliko se tacke A(x1,y1) i B(x2,y2) ne
                nalaze na pravoj koja je paralelna x osi
57                {

```

```

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

        if (y3==x3*k+n && y4==x4*k+n) // proverimo da li tacke
61         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':
73         delta = x1*y2-x2*y1;
            deltap = x2*(y4-y3)-y2*(x4-x3);
75         deltaq = x1*(y4-y3)-y1*(x4-x3);
            if (delta!=0)
77             printf("Sistem ima jedinstveno resenje %.2f, %.2f\n",
deltap/delta, deltaq/delta);
            else if (deltap==0 && deltaq==0)
79             printf("Sistem je neodredjen ili nema resenja.\n");
            else
81             printf("Sistem nema resenja\n");
            break;
83         default:
            printf("Nekorektan unos\n");
85     }
    return 0;
87 }

```

Rešenje 2.16

```

1  /* Sa standardnog ulaza se unosi ceo cetvorocifren broj. Napisati
   program koji
   ispisuje njegovu najveću cifru na standardni izlaz. */
3
   #include <stdio.h>
5
   int main(){
7       int n, j, d, s, h, max;

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

```

```

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

13  /* Proveravamo da li se radi o cetvorocifrenom broju */
14  if(n<1000 || n>9999){
15      /* Ako broj nije cetvorocifren, prijavljujemo gresku */
16      printf("Greska: Niste uneli cetvorocifren broj!\n");
17  }
18  else{

19      /* Ako je broj cetvorocifren, izdvajamo cifre broja:
20       * j - jedinice, d - desetice, s - stotine i h - hiljade
21       */
22      j=n%10;
23      d=(n/10)%10;
24      s=(n/100)%10;
25      h=n/1000;

26      /* Odredjujemo maksimalnu cifru */
27      max=j;
28      if(d>max)
29          max=d;
30      if(s>max)
31          max=s;
32      if(h>max)
33          max=h;

34      /* II nacin:
35       * if(j>d && j>s && j>h)
36       *   max=j;
37       * if(d>j && d>s && d>h)
38       *   max=d;
39       * if(s>j && s>d && s>h)
40       *   max=s;
41       * if(h>j && h>d && h>s)
42       *   max=h;
43       */

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

46  }

47  return 0;
48  }

```

Rešenje 2.17

```

1  /* Napisati program koji za dati trocifren broj proverava da li je
2  Armstrongov. Broj je Armstrongov ako je jednak zbiru kubova svojih
3  cifara.
4  */

```

```
5 #include <stdio.h>

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

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

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

21        /* Ako je broj trocifre, izdvajamo cifre broja:
           j -jedinice, d - desetice, s - stotine
23        */
        j=n%10;
25        d=(n/10)%10;
        s=n/100;

27        /* Proveravamo da li je broj Armstrongov */
29        if(n==j*j*j+d*d*d+s*s*s){
            printf("Broj je Armstrongov.\n");
31        }
        else{
33            printf("Broj nije Armstrongov.\n");
        }
35    }

37    return 0;

39 }
```

Rešenje 2.18

```
1 /* Za ceo broj k izmedu 1 i 189 koji se unosi sa standardnog ulaza,
   odrediti
   cifru koja se nalazi na k-toj poziciji
3 niza 12345678910111213...9899 u kom su redom ispisani brojevi od 1
   do 99. */

5
7 #include <stdio.h>

9 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
                * brojeva */
29                 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
               zapoceto je sa zapisom sledeceg */
41                 /* 9 oduzimamo jer je 9 broj cifara potrebnih za zapis
               jednocifrenih
                * brojeva */
43                 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         }
55     else{
        printf("Greska: Nedozvoljena vrednost broja k!\n");
    }
```

```
    }  
57     return 0;  
59 }
```

Rešenje 2.19

```
1  /* Sa standardnog ulaza se unosi cetvorocifreni pozitivan broj.  
   Napisati program  
   koji racuna i ispisuje proizvod parnih cifara datog broja. Ukoliko  
   uneti broj  
3  nije pozitivna cetvorocifrena vrednost ispisati poruku Greska!. */  
  
5  #include <stdio.h>  
  
7  int main(){  
    int n, j, d, s, h;  
9    int broj_parnih, proizvod_parnih;  
  
11   /* Ucitavamo broj */  
    printf("Unesite broj: ");  
13    scanf("%d", &n);  
  
15   /* Proveravamo da li je unet cetvorocifreni broj */  
    if(n<1000 || n>9999){  
17       /* Ako nije, prijavljujemo gresku */  
        printf("Greska!\n");  
19    }  
    else{  
21  
        /* Ako jeste: */  
23  
        /* Izdvajamo cifre broja:  
25         j -jedinice, d - desetice, s - stotine i h - hiljade  
        */  
27         j=n%10;  
         d=(n/10)%10;  
29         s=(n/100)%10;  
         h=n/1000;  
31  
        /* Inicijalizujemo broj parnih cifara na 0 */  
33         broj_parnih=0;  
        /* Postavljamo proizvod parnih cifara na 1 (neutral za mnozenje)  
        */  
35         proizvod_parnih=1;  
  
37         /* Proveravamo da li je cifra jedinica parna */  
         if(j%2==0){  
39             proizvod_parnih=proizvod_parnih*j;  
             broj_parnih++;  
41         }
```



```

43  /* Proveravamo da li je cifra desetica parna */
44  if(d%2==0){
45      proizvod_parnih=proizvod_parnih*d;
46      broj_parnih++;
47  }
48
49  /* Proveravamo da li je cifra stotina parna */
50  if(s%2==0){
51      proizvod_parnih=proizvod_parnih*s;
52      broj_parnih++;
53  }
54
55  /* Proveravamo da li je cifra hiljada parna */
56  if(h%2==0){
57      proizvod_parnih=proizvod_parnih*h;
58      broj_parnih++;
59  }
60
61  /* Proveravamo da li u zapisu broja ima parnih cifara i
62     ispisujemo
63     rezultat */
64  if(broj_parnih==0){
65      printf("Proizvod parnih cifara: 0\n");
66  }
67  else{
68      printf("Proizvod parnih cifara: %d\n", proizvod_parnih);
69  }
70
71  }
72
73  return 0;
74
75  }

```

Rešenje 2.20

```

1  /* Sa standardnog ulaza unosi se 5 karaktera. Proveriti da li je prvi
2     karakter
3     veliko ili malo slovo a. Ako jeste, ispisati karaktere obrnutim
4     redosledom, a ako nije, nista ne ispisivati. */
5
6  #include <stdio.h>
7
8  int main(){
9
10     char c1, c2, c3, c4, c5;
11
12     /* Citamo karaktere */
13     /* Obratiti paznju na format ucitavanja */
14     printf("Unesite karaktere: ");

```

2 Kontrola toka

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

15
/* Proveravamo da li je prvi karakter malo ili veliko slova a */
17 if(c1=='a' || c1=='A'){
    /* I ako jeste, ispusujemo karaktere u obrnutom redosledu */
19     printf("%c %c %c %c %c\n", c5, c4, c3, c2, c1);
    }
21
    return 0;
23 }
```

Rešenje 2.21

```
1
/*
3 Napisati program koji za karakter koji ucitava jedan karakter i :
- u slucaju da je uneta cifra, ispisuje nju i njen ascii kod
5 - u slucaju da je uneto malo slovo, ispisuje njega, njegov ascii kod,
odgovarajuće veliko slovo i njegov ascii kod
- u slucaju da je uneto veliko slovo, ispisuje njega, njegov ascii
kod, odgovarajuće malo slovo i njegov ascii kod
7 - u ostalim slucajevima, ispisuje uneti karakter i njegov ascii kod
*/
9 #include <stdio.h>
int main()
11 {
    char c;
13     printf("Unesi jedan karakter:");
    scanf("%c", &c);
15
    if (c>='0' && c<='9')
17         printf("cifra:%c ascii:%d\n",c,c);
    else if (c>='A' && c<='Z')
19         printf("veliko slovo:%c ascii:%d odgovarajuće malo:%c, ascii:%d\n",c,c,c-'A'+ 'a',c-'A'+ 'a'); /* Razlika izmedju ascii koda
svakog malog i odgovarajućeg velikog slova
                                     je konstanta koja se moze
    sracunati izrazom 'a'-'A' (i iznosi 32) */
21     else if (c>='a' && c<='z')
        printf("malo slovo:%c ascii:%d odgovarajuće veliko:%c, ascii:%d\n",c,c,c-'a'+ 'A',c-'a'+ 'A');
23     else
        printf("karakter:%c ascii:%d\n",c,c);
25     return 0;
27 }
```

Rešenje 2.22

```
1  /* Sa standarnog ulaza unosi se jedan karakter. Ako je u pitanju malo
   slovo,
   zameniti ga odgovarajućim velikim slovom i ispisati na standardni
   izlaz. Ako je
3  u pitanju veliko slovo, zameniti ga odgovarajućim malim slovom
   i ispisati ga na standardni izlaz. Ako je u pitanju cifra ispisati
   poruku cifra.
5  Ako je u pitanju bilo koji drugi karakter, onda ga ispisati na
   standarni izlaz
   između dveju zvezdica. */
7
   #include <stdio.h>
9
   int main(){
11
       char c;
13
       /* Citamo karakter */
15       printf("Unesite karakter: ");
       scanf("%c", &c);
17
       /* Proveravamo da li je karakter malo slovo */
19       if(c>='a' && c<='z'){
           /* I ako jeste, ispusujemo odgovarajuće veliko slovo */
21           printf("%c\n", c-'a'+'A');
       }
23       else{
           /* Proveravamo da li je karakter veliko slovo */
25           if(c>='A' && c<='Z'){
               /* I ako jeste, ispusujemo odgovarajuće malo slovo */
27               printf("%c\n", c-'A'+'a');
           }
29           else{
               /* Proveravamo da li je karakter cifra */
31               if(c>='0' && c<='9'){
                   /* I ako jeste, ispusujemo odgovarajuću poruku */
33                   printf("cifra\n");
               }
35               else{
                   /* Inace ispisujemo karakter između dveju zvezdica */
37                   printf("'%c'",c);
               }
39           }
       }
41       return 0;
43 }
```

Rešenje 2.23

2 Kontrola toka

```
1  /* Sa standardnog ulaza se unosi 5 karaktera. Ispisati na izlazu broj
   unetih
   malih slova. */
3
   #include <stdio.h>
5
   int main(){
7
       char c1, c2, c3, c4, c5;
9       int broj_malih_slova=0;
11
       /* Citamo karaktere */
13       printf("Unesite karaktere: ");
       scanf("%c %c %c %c %c", &c1, &c2, &c3, &c4, &c5);
15
       /* Proveravamo da li je prvi karakter malo slovo */
17       if(c1>='a' && c1<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
19           broj_malih_slova++;
       }
21
       /* Proveravamo da li je drugi karakter malo slovo */
23       if(c2>='a' && c2<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
25           broj_malih_slova++;
       }
27
       /* Proveravamo da li je treci karakter malo slovo */
29       if(c3>='a' && c3<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
31           broj_malih_slova++;
       }
33
       /* Proveravamo da li je cetvrti karakter malo slovo */
35       if(c4>='a' && c4<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
37           broj_malih_slova++;
       }
39
       /* Proveravamo da li je peti karakter malo slovo */
41       if(c5>='a' && c5<='z'){
           /* I ako jeste, uvecavamo broj malih slova */
43           broj_malih_slova++;
       }
45
       /* Ispisujemo rezultat */
47       printf("Broj malih slova: %d\n", broj_malih_slova);
49
       return 0;
```

51 | }

Rešenje ??

```

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

```

```

    }
49
    if (d > max)
51    {
        max = d;
53        stepenMax = 1000;
    }
55    if (d < min)
    {
57        min = d;
        stepenMin = 1000;
59    }

61
    int rez;
63    /* Ideja:
        4179, mesta menjamo tako sto oduzmemo 9 i dodamo 1,
65        ^^  odnosno oduzemo 100 i dodamo 900 */

67    if (broj > 0)
        rez = broj - max*stepenMax + min*stepenMax
69              - min*stepenMin + max*stepenMin;
    else
71        rez = broj + max*stepenMax - min*stepenMax
              + min*stepenMin - max*stepenMin;
73
    printf("%d\n",rez);
75    return 0;
77 }
```

Rešenje 2.25

```

1  #include <stdio.h>

3  int main()
    {
5      int br1, br2, br3;
        scanf("%d%d%d", &br1, &br2, &br3);
7
        if (br1 > 999 || br1 < 100 || br2 > 999 || br2 < 100
9            || br3 > 999 || br3 < 100)
        {
11         printf("-1");
            return -1;
13     }

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

```

19     max1 = br3;

21     /* Ako je br1 vec najveći, onda pretragu
       za sledecim najvećim 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  odredjuje:
   a) duzinu zajednickog dela ta dva intervala
4  b) najveći interval sadržan u datim intervalima (presek), a ako on ne
       postoji
       dati odgovarajuću poruku.
6  c) duzinu realne prave koju pokrivaju ta dva intervala
   d) najmanji interval koji sadrži date intervale
8  */

10 #include <stdio.h>
   #include <stdlib.h>
12
14 int main() {
16     int a1, b1, a2, b2;
       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]
       * racuna se kao:
       * [a3, b3] = [max{a1,a2}, min{b1, b2}] */
24
26     a3 = a1 > a2 ? a1 : a2;
       b3 = b1 < b2 ? b1 : b2;
28

```

```
30  /* U ovom slucaju, presek je prazan */
31  if(a3 >= b3) {
32
33      duzina_zajednickog_dela = 0;
34      zajednicka_duzina = abs(b1-a1) + abs(b2-a2);
35  }
36  else {
37
38      duzina_zajednickog_dela = abs(b3-a3);
39      zajednicka_duzina = abs(b2-a1);
40  }
41
42  /* Racunanje "pokrivaca" */
43  x = a1 < a2 ? a1 : a2;
44  y = b1 > b2 ? b1 : b2;
45
46  printf("Duzina zajednickog dela: %d\n", duzina_zajednickog_dela);
47
48  if(a3 >= b3)
49      printf("Presek intervala: prazan\n");
50  else
51      printf("Presek intervala: [%d, %d]\n", a3, b3);
52
53  printf("Zajednicka duzina intervala: %d\n", zajednicka_duzina);
54  printf("Najmanji interval: [%d, %d]\n", x, y);
55
56  return 0;
57 }
```

Rešenje 2.27

```
1  /* Data je funkcija  $f(x) = 2 * \cos(x) - x*x$  . Sa standardnog ulaza
2     se unosi
3     realan broj x i broj k koje moze biti 1, 2 ili 3. Napisati program
4     koji
5     izracunava  $F(k, x) = f(f(f(...f(x))))$  gde je funkcija f
6     primenjena k-puta.
7     */
8
9  #include <stdio.h>
10 #include <math.h>
11
12 int main(){
13     float x;
14     int k;
15     float F;
16
17     printf("Unesite redom x i k: ");
18     scanf("%f %d", &x, &k);
19
20     /* Proveravamo vrednost za k */
```



```

17  if(k<1 || k>3){
18      printf("Greska: nedozvoljena vrednost za k!\n");
19      return 0;
20  }
21  printf("F(%f,%d)=", x, k);

22      /* Analiziramo moguće slučajeve */
23  if(k==1){
24      F=2*cos(x)-x*x*x;
25  }
26  else{
27      if(k==2){
28          x=2*cos(x)-x*x*x;
29          F=2*cos(x)-x*x*x;
30      }
31      else{
32          x=2*cos(x)-x*x*x;
33          x=2*cos(x)-x*x*x;
34          F=2*cos(x)-x*x*x;
35      }
36  }
37
38  }
39
40  /* Ispisujemo rezultat */
41  printf("%f\n", F);
42
43  return 0;
44  }

```

Rešenje 2.28

```

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

5  #include <stdio.h>

7  int main(){

9      int dan;

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

13     switch(dan){
14         case 1:
15             printf("ponedeljak\n");
16             break;
17         case 2:

```

```
19     printf("utorak\n");
20     break;
21 case 3:
22     printf("sreda\n");
23     break;
24 case 4:
25     printf("cetvrtak\n");
26     break;
27 case 5:
28     printf("petak\n");
29     break;
30 case 6:
31     printf("subota\n");
32     break;
33 case 7:
34     printf("nedelja\n");
35     break;
36 default:
37     printf("Greska: nedozvoljeni unos!\n");
38 }
39
40 return 0;
41 }
```

Rešenje 2.29

```
1 /* Sa standardnog ulaza se ucitavaju dva cela broja i jedan od
   karakteri +, -,
   *, / ili % koji predstavlja operaciju koju treba izvrstiti nad unetim
   brojevima.
3 Napisati program koji koriscenjem switch naredbe analizira o kom
   karakteru je
   rec i na standardni izlaz ispisuje rezultat. U slucaju pogresnog
   unosa ispisati
5 odgovarajuću poruku. */
7 #include <stdio.h>
9 int main(){
11     char op;
12     int x, y;
13
14     printf("Unesite operator i dva cela broja: ");
15     scanf("%c %d %d", &op, &x, &y);
17     switch(op){
18         case '+':
19             printf("Rezultat je: %d\n", x+y);
20             break;
21         case '-':
```

```

    printf("Rezultat je: %d\n", x-y);
23     break;
    case '*':
25     printf("Rezultat je: %d\n", x*y);
        break;
27     case '/':
        if(y==0)
29         printf("Greska: deljenje nulom nije dozvoljeno!\n");
        else
31         printf("Rezultat je: %f\n", x*1.0/y);
        break;
33     case '%':
        printf("Rezultat je: %d\n", x%y);
35         break;
        default:
37         printf("Greska: nepoznat operator!\n");
    }
39
    return 0;
41 }

```

Rešenje 2.30

```

1  /* Napisati program koji za uneti datum u formatu dan.mesec.godina.
   proverava da
   li je korektan. */
3
   #include <stdio.h>
5
   int main(){
7       int dan, mesec, godina, dozvoljen_broj_dana;

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

   /* Proveravamo godinu */
13  if(godina<0){
15      printf("Datum nije korektan (neispravna godina)!\n");
      return 0;
17  }

   /* Proveravamo mesec */
19  if(mesec<1 || mesec>12){
21      printf("Datum nije korektan (neispravan mesec)!\n");
      return 0;
23  }

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

```

```
29     case 3:
30     case 5:
31     case 7:
32     case 8:
33     case 10:
34     case 12:
35         /* Dozvoljeni broj dana za januar, mart, maj, jul, avgust,
36          * oktobar i decembar je 31 */
37         dozvoljen_broj_dana=31;
38         break;
39     case 2:
40         /* Proveravamo da li je godina prestupna */
41         if(godina%4==0 && godina%100!=0 || godina%400==0)
42             /* Ako jeste, dozvoljeni broj dana za februar je 29 */
43             dozvoljen_broj_dana=29;
44         else
45             /* Ako nije, dozvoljeni broj dana za februar je 28 */
46             dozvoljen_broj_dana=28;
47         break;
48     case 4:
49     case 6:
50     case 9:
51     case 11:
52         /* Dozvoljeni broj dana za april, jun, septembar i novembar
53          * je 30 */
54         dozvoljen_broj_dana=30;
55         break;
56     }
57     /* Proveravamo dan */
58     if(dan<0 || dan>dozvoljen_broj_dana){
59         printf("Datum nije korektan (neispravan dan)!\n");
60         return 0;
61     }
62
63     /* Sve provere su ispunjene pa zakljucujemo da je datum korektan
64     */
65     printf("Ispravan datum!\n");
66
67     return 0;
68 }
```

Rešenje 2.31

```
1 /* Napisati program koji za korektno unet datum u formatu dan.mesec.
2 godina.
3 ispisuje datum prethodnog dana. */
4
5 #include <stdio.h>
6
7 int main(){
```

```
7      int dan, mesec, godina;
      int prethodni_dan, prethodni_mesec, prethodni_godina;

9

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

13

15     /* Racunamo dan, mesec i godinu prethodnog dana */
      prethodni_dan=dan-1;
      prethodni_mesec=mesec;
      prethodni_godina=godina;

17

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;
              prethodni_godina=godina-1;
29         }

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

      /* Ispisujemo datum koji smo izracunali */
```

2 Kontrola toka

```
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 2.101

```
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()))
16         br_cif++;
17
18     printf("%d\n", br_cif);
19
20     return 0;
21 }
```

Rešenje 2.35

Rešenje 2.36

```
1  #include <stdio.h>
2  #include <ctype.h> // !!!
3
4  // Upotreba funkcija isalpha, isdigit, toupper, tolower
5
6  // isalpha( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
7  // je karakter slovo (malo ili veliko), inace 0
8  // isdigit( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
9  // je karakter cifra, inace 0
10 // isupper( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
11 // je karakter veliko slovo, inace 0
12 // islower( karakter ) - funkcija vraća vrednost razlicitu od 0 ako
13 // je karakter malo slovo, inace 0
14 // toupper( karakter ) - ukoliko je karakter malo slovo, funkcija
15 // vraća odgovarajuće veliko slovo,
16 // inace vraća isti karakter
17 // tolower( karakter ) - ukoliko je karakter veliko slovo, funkcija
18 // vraća odgovarajuće malo slovo,
19 // inace vraća isti karakter
20
21 int main()
22 {
23     char c;
24     char veliko_slovo;
25     char malo_slovo;
26
27     printf("Unesite karakter: ");
28     scanf("%c",&c);
29
30     if(isalpha(c))
31     {
32         printf("Karakter %c je slovo\n",c);
33
34         if(isupper(c))
35             printf("Veliko slovo\n");
36         else
37             printf("Malo slovo\n");
38     }
39 }
```

```
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

```
2  #include <stdio.h>
4  // Za uneti redni broj dana u nedelji ispisati njegov naziv
6  int main()
7  {
8      int broj_dana;
10     printf("Unesite broj dana: ");
11     scanf("%d",&broj_dana);
12
13     switch(broj_dana)
14     {
15         case 1: printf("Dan je ponedeljak\n");
16                 break; // Obavezan izlazak iz case-a!
17         case 2: printf("Dan je utorak\n");
18                 break; // Obavezan izlazak iz case-a!
19         case 3: printf("Dan je sreda\n");
20                 break; // Obavezan izlazak iz case-a!
21         case 4: printf("Dan je cetvrtak\n");
22                 break; // Obavezan izlazak iz case-a!
23         case 5: printf("Dan je petak\n");
24                 break; // Obavezan izlazak iz case-a!
25         case 6: printf("Dan je subota\n");
26                 break; // Obavezan izlazak iz case-a!
27         case 7: printf("Dan je nedelja\n");
28                 break; // Obavezan izlazak iz case-a!
29         default: printf("Lose unet broj!\n"); // Ako ni jedna provera
30                 ne prolazi
31     }
32
33     return 0;
34 }
```

Rešenje 2.38

```
2  #include <stdio.h>
3  #include <stdlib.h> // Potrebno za exit funkciju
4
5  // Za unetu godinu i mesec ispisuje se naziv meseca i koliko dana ima
6  // u tom mesecu te godine
7
8  int main()
9  {
10     int godina;
11     int mesec;
12
13     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)
24        // Provera da li je godina prestupna,
        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");
63     }
64
65     return 0;
66 }
```

Rešenje 2.39

```
#include <stdio.h>
2
// Za uneti datum određuje ispisuje se naziv godisnjeg doba kome
// datum pripada
4
int main()
6 {
    int godina;
    int mesec;
    int dan;
10
    printf("Unesite datum (DD MM GGGG): ");
12    scanf("%d%d%d", &dan, &mesec, &godina);
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");
24                break;
26
        case 3:
                if(dan < 21)
                    printf("Zima\n");
                else
                    printf("Prolece\n");
28                break;
30
        case 4: printf("Prolece\n");
32                break;
34
        case 5: printf("Prolece\n");
36                break;
38
        case 6:
                if(dan < 21)
                    printf("Prolece\n");
                else
                    printf("Leto\n");
40
42    }
```

```

        break;
44
    case 7: printf("Leto\n");
46         break;

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

    case 9:
50         if(dan < 23)
52             printf("Leto\n");
54         else
56             printf("Jesen\n");
58         break;

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

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

    case 12:
64         if(dan < 22)
66             printf("Jesen\n");
68         else
69             printf("Zima\n");
70         break;

    default: printf("Lose unet redni broj meseca!\n");
72 }

74 return 0;
}
```

2.3 Petlje

2.3.1 Ispis podataka

REDOSLED: Petlje se sustinski koriste za tri stvari: map, filter i reduce, kao i za kombinaciju te tri stvari.

Map — preslikavanje, dakle ceo niz necega se preslikava na neki nacin u neki novi niz (dupliranje vrednosti svih elemenata niza, dupliranje svake cifre broja, dodavanje prefiksa svim recima...)

Filter — iz niza necega biraju se neki koji zadovoljavaju neki kriterijum (svi parni brojevi, svi koji sadrže karakter "a", svi prosti brojevi, svi savršeni brojevi...)

Reduce — ceo niz se svodi na jednu vrednost (zbir svih vrednosti, proizvod svih vrednosti, nadovezane sve vrednosti...)

Kombinacija — dve tehnike od prethodne tri (npr filter-reduce: zbir svih parnih

brojeva) ili od svake po malo (zbir svih dupliranih brojeva koji su savršeni) Sustinski, studenti treba da usvoje najpre ove tri tehnike, pa onda da idu ne njihove kombinacije, i to najpre na kombinacije dve od tri, pa na kraju na zadatke koje kombinuju sve to. Ove tehnike nisu vezane za nizove, mogu se primeniti i na prirodne brojeve posmatrane kao niz brojeva ili na prirodni broj posmatran kao niz cifara...

Danijela:: ok neka bude redosled koji si predložila

Zadatak 2.40 Napisati program koji 5 puta ispisuje tekst Mi volimo da programiramo.

Primer 1

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

[Rešenje A.28]

Zadatak 2.41 Napisati program koji učitava ceo broj n i ispisuje n puta tekst Mi volimo da programiramo.

Primer 1

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

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite ceo pozitivan broj 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 .

Primer 1

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

Primer 2

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

[Rešenje 2.42]

2 Kontrola toka

Zadatak 2.43 Napisati program koji učitava dva cela broja n i m ispisuje sve cele brojeve iz intervala $[n, m]$.

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

Primer 1

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

Primer 2

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

[Rešenje 2.43]

Zadatak 2.44 Uskladiti formulaciju zadatka sa odgovarajućom formulacijom kod nizova. Fibonačijev niz počinje ciframa 1 i 1, a svaki član se dobija zbirom prethodna dva. Napisati program koji učitava ceo neoznačen broj n i određuje i na standardni izlaz ispisuje n -ti član Fibonačijevog niza.

Primer 1

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

Primer 2

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

[Rešenje 2.121]

* **Zadatak 2.45** Niz prirodnih brojeva formira se prema sledećem pravilu:

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

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

Primer 1

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

Primer 2

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

[Rešenje 2.45]

* **Zadatak 2.46** Papir A_0 ima površinu $1m^2$ i odnos stranica $1 : \sqrt{2}$. Papir A_1 dobija se podelom papira A_0 po dužoj ivici. Papir A_2 dobija se podelom A_1 papira po dužoj ivici itd. Napisati program koji za uneti 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 2.101]

2.3.2 Obrada celih brojeva, rad sa ciframa broja

Zadatak 2.47 Napisati program koji učitava ceo broj i ispisuje njegove cifre u obrnutom poretaku.

[Rešenje 2.47]

Zadatak 2.48 Pravi delioci celog broja su svi delioci sem jedinice i samog tog broja. Napisati program koji učitava ceo pozitivan broj n i ispisuje sve prave delioce unetog broja. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

[Rešenje 2.48]

Zadatak 2.49 Sa standardnog ulaza unosi se ceo neoznačen broj. Napisati program koji proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu.

Primer 1

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

Primer 2

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

[Rešenje 2.49]

2 Kontrola toka

Zadatak 2.50 Sa standardnog ulaza unosi se ceo broj. Napisati program koji na standardni izlaz ispisuje odgovor da li je uneti prirodan broj deljiv sumom svojih cifara.

[Rešenje [2.101](#)]

Zadatak 2.51 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 [2.51](#)]

Zadatak 2.52 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 standardni 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 [2.52](#)]

Zadatak 2.53 Sa standardnog ulaza unosi se neoznačen ceo broj. Napisati program koji formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre polaznog broja, počevši od krajnje desne cifre.

Primer 1

```

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

```

Primer 2

```

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

```

Primer 3

```

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

```

[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 cifara koje su jednake zbiru svojih suseda.

Primer 1

```

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

```

Primer 2

```

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

```

Primer 3

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 242
|| 22

```

[Rešenje 2.54]

* **Zadatak 2.55** Broj je *palindrom* ukoliko 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 2.55]

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

Zadatak 2.56 Napisati program koji učitava pozitivan ceo broj n , a zatim učitava n celih brojeva i na standardni izlaz ispisuje sumu pozitivnih i sumu negativnih unetih brojeva.

[Rešenje 2.56]

Zadatak 2.57 Sa standardnog ulaza unosi se ceo pozitivan broj n , a potom i n celih brojeva. Izračunati i ispisati zbir onih brojeva koji su neparni i negativni.

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 2.57]

Zadatak 2.58 Napisati program koji učitava 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 standardni izlaz ispisuje najmanju cenu.

[Rešenje 2.59]

Zadatak 2.60 Sa standardnog ulaza se unose realni brojevi sve do unosa broja nula 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 2.61]

Zadatak 2.61 U prodavnici se nalaze artikala čije cene su realni pozitivni brojevi. Cene artikala se unose sa standardnog unosa sve do unosa broja nula 0. Napisati program koji izračunava i ispisuje prosečnu vrednost cena u radnji.

I ovo bi moglo da se preformulise u cene, tj da se sracuna prosečna vrednost cena u radnji. Čak mislim da bi mogli da stavimo dva zadatka, ovaj i jedan sa cenama, a u rešenju da se pozovemo samo na rešenje ovog zadatka, tako da se vidi da je to u sustini isti problem.

Danijela: dodat još jedan zadatak, u rešenju se pozvati na prethodni.

Danijela: obratiti pažnju da cene mogu biti samo pozitivni brojevi, dok u prethodnom zatku nismo imali takav zahtev – da li menjati prethodni zadatak ili dati rešenje i za ovaj?

Primer 1

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

[Rešenje 2.61]

Zadatak 2.62 U narednim zadacima se u tekstu kaže da se unosi ceo pozitivan broj a posle se u rešenju nigde to ne proverava, niti se koristi tip unsigned. Nesto od toga mora, inae rešenje nije dobro.

Danijela: sredicu rešenje u odnosu na ovaj komentar.

Danijela: obratiti pažnju na tekst i Peru – da li nam se sviđa ovako nešto? Ako su cene onda mogu biti samo pozitivni brojevi?

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

Primer 1

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

Primer 2

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

[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 odgovarajuću poruku.

Ukoliko test primer ne moze da stane u midi onda treba da bude maxi, ali mozda bolje skratiti ga u midi.

Danijela: bice promenjeno kad budem sredjivala test primere.

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 2.101]

Zadatak 2.64 Sa standardnog 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 standardni izlaz.

[Rešenje 2.101]

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 cifra najveće težine u zapisu broja. Ukoliko ima više takvih, ispisati prvi.

Primer 1

```

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| Unesite n brojeva: 8 964 32 511 27
|| 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 Vršna su merenja nadmorskih visina na određenom delu teritorije i naučnike zanima razlika između najveće i najmanje nadmorske visine. Napisati program koji učitava n , potom n realnih brojeva koji označavaju nadmorske visine i ispisuje razliku najveće i najmanje nadmorske visine.

Primer 1

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

Primer 2

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

[Rešenje 2.69]

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 sve do kraja ulaza, 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.

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 Prekoracenje se javlja mnoooogo ranije. I ovo je jedan od zadataka koji imamo u funkcijama. Napisati program koji učitava ceo pozitivan broj i i izračunava njegov faktorijel. U slučaju neispravnog unosa ispisati odgovarajuću poruku. 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 n -ti stepen broja x , tj. x^n .

Primer 1

```

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

```

Primer 2

```

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

```

Primer 3

```

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

```

[Rešenje 2.75]

Zadatak 2.76 Sa standradnog ulaza unose se realan broj x i ceo broj n . Napisati program koji izračunava n -ti stepen broja x .

Primer 1

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

Primer 2

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

[Rešenje ??]

Zadatak 2.77 Napisati program koji učitava ceo pozitivan broj n i ispisuje vrednost sume kubova brojeva od 1 do n , odnosno $s = 1 + 2^3 + 3^3 + \dots + n^3$. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

[Rešenje 2.78]

Zadatak 2.78 Napisati program koji učitava ceo pozitivan broj n i ispisuje sumu kubova, $s = 1 + 2^3 + 3^3 + \dots + k^3$, za svaku vrednost $k = 1, \dots, n$. U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

[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 standardni izlaz ispisuje sumu $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \dots + n \cdot x^n$.

Primer 1

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

Primer 2

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

[Rešenje 2.79]

Zadatak 2.80 Sa standardnog ulaza unose se realan broj x i ceo neoznačen broj n . Napisati program koji izračunava i na standardni izlaz ispisuje sumu $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \dots + \frac{1}{x^n}$.

Primer 1

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

Primer 2

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

[Rešenje 2.80]

* **Zadatak 2.81** Mislila sam da se tačnost ϵ ps odnosi na to da je razlika dva uzastopna člana manja od ϵ ps a ne da je sam član manji od ϵ ps? Nisam sigurna, ali možda treba proveriti ili preformulisati zadatak tako da se ne definiše ovaj pojam. Napisati program koji učitava realane brojeve x i ϵ ps i sa zadatom tačnošću ϵ ps izračunava i na standardni izlaz ispisuje sumu $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$. Izračunati sumu u odnosu na tačnost ϵ ps znači uporediti poslednji član sume sa ϵ ps i ukoliko je taj poslednji član manji od ϵ ps prekinuti dalja izračunavanja. UPUTSTVO: Prilikom računanja sume koristiti prethodni izračunati član sume u računanju sledećeg člana sume. Naime, ako je izračunat član sume $\frac{x^n}{n!}$ na osnovu njega se lako može dobiti član $\frac{x^{n+1}}{(n+1)!}$. Nikako ne računati stepen i faktorijel odvojeno zbog neefikasnosti takvog rešenja i zbog mogućnosti prekoračenja.

Primer 1

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

Primer 2

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

[Rešenje 2.81]

* **Zadatak 2.82** Napisati program koji učitava realane brojeve x i ϵ ps i sa zadatom tačnošću ϵ ps izračunava i na standardni izlaz ispisuje sumu $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} \dots$. NAPOMENA: Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.

Primer 1

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

Primer 2

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

[Rešenje 2.82]

Zadatak 2.83 Napisati program koji učitava realan broj x i prirodan broj n izračunava sumu $S = (1 + \cos(x)) \cdot (1 + \cos(x^2)) \cdot \dots \cdot (1 + \cos(x^n))$. NAPOMENA: Voditi računa o efikasnosti rešenja.

[Rešenje 2.101]

* **Zadatak 2.84** Napisati program koji učitava ceo neoznačen broj n , a na standardni izlaz ispisuje vrednost razlomka

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

[Rešenje 2.101]

* **Zadatak 2.85** Napisati program koji računa sumu

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

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

[Rešenje 2.101]

* **Zadatak 2.86** Sa standardnog ulaza unosi se ceo pozitivan broj n veći od 0. Napisati program koji računa proizvod

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 2.101]

* **Zadatak 2.87** Sa standardnog ulaza unosi se ceo pozitivan neparan broj n . Napisati program koji za uneto n izračunava:

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

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 2.101]

Zadatak 2.88 Sa standardnog ulaza unose se realni brojevi x i a i ceo pozitivan broj n veći od 0. Napisati program koji izračunava:

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

U slučaju greške pri unosu podataka ispisati odgovarajuću poruku.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite 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 2.101]

2.3.6 Dvostruka petlja i ispisivanje slike

Zadatak 2.89 Sa standardnog ulaza unosi se neoznačen broj n . Napisati program koji za uneto n zvezdicama iscertava

2 Kontrola toka

a) kvadrat stranice n sastavljen od zvezdica.

Primer 1

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

b) rub kvadrata dimenzije n .

Primer 1

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

Primer 2

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

c) rub kvadrata dimenzije n koji i na glavnoj dijagonali ima zvezdice.

Primer 1

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

[Rešenje 2.101]

* **Zadatak 2.90** Napisati program koji za uneti ceo broj n zvezdicama iscrtava slovo X dimenzije n .

Primer 1

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

Primer 2

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

[Rešenje 2.91]

* **Zadatak 2.91** Napisati program koji za uneti ceo broj n korišćenjem znaka $+$ iscrtava veliko $+$ dimenzije n .

Primer 1

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

Primer 2

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

[Rešenje 2.91]

Zadatak 2.92 Napisati program koji učitava ceo neoznačen broj n , a potom iscrtava

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

Primer 1

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

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

Primer 1

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

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

Primer 1

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

- d) rub jednakokrakog pravouglog trougla čije su katete dužine n . Program učitava karakter c i taj karakter koristi za iscrtavanje ruba trougla.

Primer 1

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

Primer 2

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

[Rešenje 2.101]

Zadatak 2.93 Napisati program koji učitava ceo broj n , a potom iscrtava

- a) jednakostranični trougao stranice n koji je sastavljen od zvezdica.

Primer 1

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

- b) trougao koji se dobija spajanjem dva jednakostranični trougla stranice n koji su sastavljeni od zvezdica.

Primer 1

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

c) rub jednakostraničnog trougla čija stranica je dužine n .

Primer 1

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

d) sliku koja se dobija spajanjem dva jednakostranična trougla čija stranica je dužine n . Iscrtavati samo rub trouglova.

Primer 1

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

[Rešenje 2.101]

* **Zadatak 2.94** 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 2.94]

* **Zadatak 2.95** Napisati program koji učitava ceo broj n , i iscrtava sliku koja se dobija na sledeći način: u prvom redu je jedna zvezdica, u drugom redu su dve zvezdice razdvojene razmakom, treći red je sastavljen od zvezdica i iste je dužine kao i drugi red, četvrti red se sastoji od tri zvezdice razdvojene razmakom, a peti red je sastavljen od zvezdica i iste je dužine kao i četvrti red itd. Ukupna visina slike je n .

Primer 1

```

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

```

[Rešenje 2.101]

**** Zadatak 2.96** Sa standardnog ulaza unose se neoznačeni celi brojevi m i n . Napisati program koji iscrtava jedan do drugog stranice n kvadrata čija je svaka strana sastavljena od m zvezdica razdvojenih prazninom.

Tekst nije u skladu sa slikom jer nije jasno da se crtaju samo rubovi a ne popunjeni kvadrati.

Danijela: Da li je sada jasnije?

Primer 1

```

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

```

Primer 2

```

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

```

[Rešenje 2.101]

*** Zadatak 2.97** Sa standardnog ulaza unosi se ceo neoznačen broj n . Napisati program koji štampa romb sastavljen od minusa u pravougaoniku sastavljenom od zvezdica.

Primer 1

```

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

```

Primer 2

```

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

```

[Rešenje 2.101]

Zadatak 2.98 Napisati program koji učitava ceo broj n ($n \geq 2$) i koji na standardni izlaz iscrtava 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 2.101]

Zadatak 2.99 Sa standardnog ulaza učitava se ceo neoznačen broj n . Napisati program koji za uneto n iscrtava pravougli „trougao” sačinjen od „koordinata” svojih tačaka. „Koordinata” tačke je oblika (i, j) pri čemu $i, j = 0, \dots, n$. Pravougao se nalazi u gornjem levom uglu slike i njegova koordinata je $(0, 0)$. Koordinata i se uvećava po vrsti, a koordinata j po koloni, pa je zato koordinata tačke koja je ispod tačke $(0, 0)$ jednaka $(1, 0)$, a koordinata tačke koja je desno od tačke $(0, 0)$ jednaka $(0, 1)$.

Ovo treba preformulisati jer je bez test primera skroz nejasno. U test primerima negde ima blanko posle zareza, negde nema, i to treba ujednačiti.

Mene ovaj zadatak zbunjuje i ne sviđa mi se. Problem su mi koordinate koje se broje nekako cudno i to od broja 1 a ne od nule. Nije mi jasno zasto u temenu pravog ugla ne bi bila koordinata $(0,0)$?

Danijela: Izmenila sam test primere i tekst, ali se slazem da zadatak nije nesto, mozemo ga izbrisati.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 2.101]

* **Zadatak 2.100** 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 odgovarajuću poruku.

Primer 1

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

Primer 2

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

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 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 2.101]

* **Zadatak 2.101** Napisati program koji učitava ceo pozitivan broj n i ispisuje sve brojeve od 1 do n , zatim svaki drugi broj od 1 do n , zatim svaki treći broj od 1 do n itd., završavajući sa svakim n -tim (tj. samo sa 1). U slučaju greške pri unosu podataka odštampati odgovarajuću poruku.

Primer 1

```

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

```

Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 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 2.101]

2.4 Rešenja

Rešenje A.28

```

#include <stdio.h>

2
int main()
4 {
    /* Promenljiva i kontrolise koliko puta ce se petlja izvesti.
6     Njenu pocetnu vrednost postavljamo na 0 jer se u pocetku petlja
       nije ni jednom izvela. */
    int i = 0;

8
    /* Pre ulaska u telo petlje proverava se da li je
10     ispunjen uslov petlje.
    */
12    while(i < 5)
    {
14        /* Ukoliko uslov petlje jeste ispunjen ulazimo u telo petlje. */
        /* Ispisujemo trazeni tekst. */
16        printf("Mi volimo da programiramo.\n");

18        /* Uvecavamo promenljivu za jedan jer smo jednom prosli kroz
        petlju. */
        i++;

20
        /* Nakon poslednje naredbe tela petlje ponovo se vracamo na
        ispitivanje uslova petlje.

```

2 Kontrola toka

```
22     Ako ovu vrednost ne menjamo dobicemo petlju koja se izvsava
    beskonacno. */
    }
24     return 0;
26 }
```

Rešenje 2.41

```
#include<stdio.h>
2
int main()
4 {
    /* Promenljiva i kontrolise koliko puta ce se petlja izvsiti.
       Najcesce ovakvu promenljivu nazivamo "brojac". */
6     int i=0;
    /* Promenljiva koja oznacava koliko puta cemo ispisati trazeni
       tekst. */
8     int n;

10     printf("Unesite ceo broj: ");
    scanf("%d", &n);
12

    /* Pre ulaska u telo petlje proverava se da li je ispunjen uslov
       petlje. */
14     while (i<n)
    {
16         printf("Mi volimo da programiramo.\n");
        i++;
18     }
    return 0;
20 }
```

Rešenje 2.42

```
1 #include <stdio.h>
  #include <stdlib.h>
3
int main()
5 {
    /* Promenljivu x koristimo u dve svrhe. Prvo, ova promenljiva
       kontrolise koliko puta se petlja izvsila.
7       Drugo, ovu promenljivu koristimo za ispis potrebnih vrednosti.
       */
    int x;
9    /* Promenljiva n se unosi i odredjuje koliko brojeva ispisujemo.
       */
    int n;
11 }
```

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

15 /* U slucaju neispravnih podataka ispisujemo odgovarajucu poruku
i izlazimo iz programa. */
17 if (n < 0)
{
19     printf("Neispravan unos. Promenljiva mora biti pozitivna!\n");
exit(EXIT_FAILURE);
21 }

23 /* Ispis pocinjemo od 0, zato promenljivu x postavljamo na 0. */
x=0;
25 while (x<= n)
{
27     /* Ispisujemo broj. */
printf("%d\n", x);
29     /* Uvecavamo promenljivu za jedan jer smo broj ispisali i sada
zelimo da ispisemo sledeci broj. */
x++;
31 }
return 0;
33 }
```

Rešenje 2.43

```
1 /* Resenje pod a). */
3 #include <stdio.h>
#include <stdlib.h>
5
7 int main()
{
9     /* Promenljive koje oznacavaju granice intervala. */
int n,m;
/* Promenljiva koja oznacava trenutno ispisani broj intervala. */
11 int i;

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

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

21
23 /* Na pocetku ispisujemo prvi broj intervala, a to je n. */
i=n;
/* uslov petlje se proverava pre ulaska u telo petlje */
```

2 Kontrola toka

```
25     while (i<=m)
26     {
27         printf("%d ", i);
28         i++;
29     }
30
31     printf("\n");
32
33     return 0;
34 }
35 }
```

```
1  /* Resenje pod b). */
2
3  #include <stdio.h>
4  #include <stdlib.h>
5
6  int main()
7  {
8
9      /* Promenljive koje oznacavaju granice intervala. */
10     int n,m;
11     /* Promenljiva koja oznacava trenutno ispisani broj intervala. */
12     int i;
13
14     printf("Unesi dva cela broja: ");
15     scanf("%d%d",&n,&m);
16
17     if (m < n)
18     {
19         printf("Neispravan unos. Nisu dobro zadate granice intervala!\n");
20         exit(EXIT_FAILURE);
21     }
22
23     /* naredba i=n se izvorsava jednom, pre prve
24        iteracije */
25     for(i=n;i<=m;i++) /* uslov petlje i<=m se proverava pre svake
26        iteracije */
27         printf("%d ", i); /* naredba i++ se izvorsava nakon svake
28        iteracije */
29
30     printf("\n");
31
32     return 0;
33 }
34 }
```

```
1  /* Resenje pod c). */
2
```

```

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

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

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

    if (m < n)
    {
        printf("Neispravan unos. Nisu dobro zadate granice intervala!\n");
        exit(EXIT_FAILURE);
    }

    /* Uslov petlje se proverava na kraju svake iteracije. */
    /* Zbog toga se do while petlja izvršava bar jednom, čak i u
       slučaju
       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 završava i program
                           se nastavlja od prve naredbe koja sledi za petljom. */
    printf("\n");

    return 0;
}

```

Rešenje 2.121

```

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

int main()
{
    /* Pamtimo uzastopna dva Fibonacijeva broja i na osnovu njih
       racunamo sledeci. */

```

```
/* Promenljive prvi i drugi su brojevi koje pamtimo i na osnovu
   njih racunamo treci. */
8 /* Na osnovu teksta zadatka, promenljive prvi i drugi postavljamo
   na 1. */
int prvi = 1;
10 int drugi = 1;
int treci;
12 /* Promenljiva pozicija je podatak koji ucitavamo i odnosi se na
   poziciju u Fibonaccijevom nizu
   za koju treba izracunati vrednost. */
14 int pozicija;
/* Promenljiva i oznacava do koje pozicije smo izracunali vrednosti
   . Kako imamo prve dve
16 vrednosti, ovu promenljivo postavljamo na 2. */
int i = 2;
18
printf("Unesite poziciju u Fibonaccijevom nizu: ");
20 scanf("%d", &pozicija);

/* Pozicija ne moze biti 0 i ne moze biti negativan broj. */
22 if (pozicija < 1)
24 {
    printf("Neispravan unos. Pozicija u Fibonaccijevom nizu mora biti
    pozitivan broj koji nije 0!\n");
    exit(EXIT_FAILURE);
}

28 while(i < pozicija)
30 {
    /* Na osnovu dva uzastopna racunamo treci. */
    32 treci = prvi + drugi;

    /* Potom razmenjujemo vrednosti. Uzastopna dva koja pamtimo
    postaju
    34 sledeca uzastopna dva broja Fibonaccijevog niza. */
    prvi = drugi;
    drugi = treci;

    38 /* Prelazimo na racunanje sledeceg broja na sledecoj poziciji. */
    i++;
    40 }

42 printf("Trazeni broj je: %d\n", drugi);
44
return 0;
46 }
```

Rešenje 2.45

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



```

3 {
4     int a0;
5     int an,an1;

6
7     printf("Unesi pocetni clan niza brojeva:");
8     scanf("%d",&a0);
9
10    if (a0>0)
11    {
12        printf("%d\n", a0);
13
14        an=a0;
15        while(an!=1)
16        {
17            if (an%2) /* Ukoliko je vrednost izraza an%2 razlicita od nule,
18                */
19            {
20                /* izraz se tumaci kao tacan i izvrsavaju se naredbe
21                iz if grane. */
22                an1=(3*an+1)/2;
23            }
24            else /* U suprotnom, ukoliko je vrednost izraza an%2 jednaka
25                nuli, izraz */
26            {
27                /* se tumaci kao netacan i izvrsavaju se naredbe iz else
28                grane. */
29                an1=an/2;
30            }
31            printf("%d\n",an1);
32            an=an1;
33        }
34    }
35    else
36    {
37        printf("Nekorektan unos. Broj mora biti pozitivan.\n");
38    }
39
40    return 0;
41 }

```

Rešenje 2.101

Rešenje 2.47

```

1  /*
2     Napisati program koji za uneti ceo broj ispisuje njegove cifre
3     u obrnutom poretku.
4  */
5
6  #include<stdio.h>
7  #include<stdlib.h>
8  int main()
9  {
10     int x;

```

2 Kontrola toka

```
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      biti pozitivne
17                               */
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 2.48

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

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

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

21
       /* 1. nacin */
23      printf("----- 1. nacin ----- \n");
       for(i=2;i<x;i++)
```

```

25 {
    printf("proveravam za %d...\n",i);
27     if (x%i==0)
        printf("\t delilac:%d \n",i);
29 }
/* 2. nacin (brzi) */
31 printf("----- 2. nacin ----- \n");
for(i=2;i<=sqrt(x);i++)
33 {
    printf("proveravam za %d...\n",i);
35     if (x%i==0)
        if (i==x/i) /* u slucaju kada je delilac koren broja, npr 4
za 16, ispisujemo ga jednom */
37         printf("\t delilac:%d \n",i);
        else /* u suprotnom, npr 2 za 16, ispisujemo i 2 i 8
*/
39         printf("\t delioci:%d %d \n",i,x/i);
    }
41 return 0;
}

```

Rešenje 2.49

```

1  /* Sa standardnog ulaza unosi se ceo neoznaceni broj. Napisati program
    koji
    proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu ili ne
    . */
3
#include <stdio.h>
5
int main(){
7     int n, cifra;
    int indikator=0;
9
    /* Ucitavamo broj */
11    printf("Unesite broj: ");
    scanf("%d", &n);
13
    /* Sve dok imamo cifara u zapisu broja */
15    while(n>0){
17
        /* Izdvajamo poslednju cifru broja */
        cifra=n%10;
19
        /* Proveravamo da li je bas ona jednaka broju 5 */
21        if(cifra==5){
            /* Ako jeste postavljamo indikator na vrednost 1 tako da
            znamo da smo
23             * pronasli peticu i prekidamo sa izvršavanjem petlje */
            indikator=1;
25            break;
        }
    }
}

```

```
    }
27     /* Ako izdvojena cifra nije jednaka broju 5, broj delimo sa 10
    kako bi
        mogli da izdvojimo i preostale cifre broja na isti nacin */
29     n=n/10;
    }

31     /* Ispisujemo rezultat */
33     if(indikator==0){
        printf("Cifra 5 se ne nalazi u zapisu!\n");
35     }
    else{
37         printf("Cifra 5 se nalazi u zapisu!\n");
    }

39     return 0;
41 }
```

Rešenje 2.101

Rešenje 2.51

```
1  /* Napisati program koji unetom broju uklanja nule sa desne strane.
    Novodobijeni
    broj ispisati na standardni izlaz. */
3
#include <stdio.h>

5
int main(){
7     int n;

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

13    if(n==0){
        printf("0\n");
15    }
    else{
17        /* Sve dok je poslednja cifra u zapisu broja n nula */
        while(n%10==0){
19            /* Broj delimo sa 10 tj. uklanjamo mu nulu sa kraja */
            n=n/10;
21        }

23        /* Ispisujemo rezultat */
        printf("%d\n", n);
25    }

27 }
```

```
    return 0;
29 }
```

Rešenje 2.52

```
1  #include <stdio.h>
3  int main() {
5      unsigned int x;
6      int pozicija; // da li se radi o cifri jedinici, desetici,
7      stotini itd...
8      int cifra; // trenutna izdvojena cifra iz broja x
9      unsigned int y; // broj dobijen nakon transformacije
10
11     printf("Unesite broj: ");
12     scanf("%d", &x);
13
14     if(x > 0) {
15
16         /* Posto pocinjemo sa izdvajanjem cifara od cifre jedinica,
17            postavljamo tezinu (stepen) pozicije na 1 */
18         pozicija = 1;
19         y = 0;
20
21         /* Sve dok imamo cifara u zapisu broja */
22         while(x > 0) {
23
24             /* Izdvajamo poslednju cifru iz zapisa */
25             cifra = x % 10;
26
27             /* Proveravamo da li je cifra parna */
28             if(cifra % 2 == 0){
29                 /* I ako jeste, uvecavamo je */
30                 cifra++;
31             }
32
33             /* Novi broj formiramo tako sto izdvojenu cifru pomnozimo
34                odgovarajucom
35                tezinom (stepenom) pozicije */
36             y += cifra*pozicija;
37
38             /* Pripremamo broj za izdvajanje naredne cifre */
39             x /= 10;
40
41             /* I uvecavamo tezinu (stepen) pozicije */
42             pozicija *= 10;
43         }
44     }
```

```
45     /* Ispisujemo izracunatu vrednost */
    printf("%d\n", y);
47 }
    else
49     printf("Nekorektan unos.\n");
51     return 0;
}
```

Rešenje 2.53

```
1  /* Sa standardnog ulaza unosi se neoznaceni 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...
13      int cifra; // trenutna izdvojena cifra iz broja x
       int rbr; // redni broj cifre koju trenutno obradjujemo, gledano s
       desna na levo
15      unsigned int y; // broj dobijen nakon transformacije
17
       /* Ucitavamo broj */
       printf("Unesite broj: ");
19      scanf("%d", &x);
21
       if(x > 0) {
           /* Postavljamo vrednost stepena na 0 - to znaci da cemo prvo
              mnoziti sa
23             * 10^0=1 */
           stepen_deset = 0;
25
           /* Postavljamo vrednost broja koji se formira na 0 */
           y = 0;
           /* Postavljamo redni broj pozicije na 0 */
27           rbr = 0;
29
           /* Sve dok imamo cifara u zapisu broja */
           while(x > 0) {
31
33               /* Izdvajamo cifru */
35               cifra = x%10;
```

```

37     /* Proveravamo da li je pozicija izdvojene cifre parna -
38        * cifre na parnim pozicijama zadržavamo
39        */
40     if(rbr % 2 == 0) {
41         /* I ako jeste */
42
43         /* Dodajemo izdvojenu cifru novom broju */
44         /* Neophodno je izvršiti "kastovanje" tipova, jer je double
45            povratni tip
46            * funkcije pow */
47         y += cifra * ((int) pow(10, stepen_deset));
48
49         /* Uvecavamo stepen zbog naredne cifre */
50         stepen_deset++;
51     }
52
53     /* Azuriramo redni broj cifre */
54     rbr++;
55     /* I pripremamo broj za naredno izdvajanje */
56     x /= 10;
57 }
58
59 /* Ispisujemo rezultat */
60 printf("%d\n", y);
61 }
62 else
63     printf("Nekorektan unos.\n");
64
65 return 0;
66 }

```

Rešenje 2.54

```

1  /* Sa standardnog ulaza unosi se neoznaceni ceo broj. Napisati program
2     koji formira i ispisuje broj koji se dobija
3     izbacivanjem cifara koje su jednake zbiru svojih suseda. Cifre se
4     posmatraju sa desna na levo. */
5
6  #include <stdio.h>
7
8  int main(){
9      unsigned n, novo_n;
10     int stepen;
11     int cifra_levo, cifra_sredina, cifra_desno;
12
13     /* Ucitavamo broj sa ulaza */
14     printf("Unesite broj: ");
15     scanf("%u", &n);

```

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

19  /* Nova vrednost broja */
    novo_n=0;

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

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

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

35  /* Ako je srednja cifra jednaka zbiru leve i desne cifre */
37  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, zadržavamo 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 2.55


```
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
4  #include <stdio.h>
5  #include <math.h>
6
7  int main() {
8
9      int x;
10     int broj_cifara;
11     int min_stepen, max_stepen;
12     int pom;
13     int leva_cifra, desna_cifra;
14     int indikator;
15
16     printf("Unesite broj: ");
17     scanf("%d", &x);
18
19     /* Ako je korisnik uneo negativan broj, analiziramo njegovu
       apsolutnu
       * vrednost
       */
21     if(x < 0)
22         x=-x;
23
24
25     /* Odredjujemo broj cifara u zapisu broja x
       kako bismo mogli da izdvajamo istovremeno cifre i sa leve i sa
       desne
       strane
       */
29     broj_cifara = 0;
30     pom = x;
31     while(pom > 0) {
32         pom /= 10;
33         broj_cifara++;
34     }
35
36     /* Odredjujemo stepen koji stoji uz krajnju levu cifru broja */
37     max_stepen = (int) pow (10, broj_cifara-1);
38
39     /* Indikator je promenljiva koja ce nam ukazivati da li je broj
       * palindrom ili ne
       */
41     indikator=1;
42     while(x!=0 && indikator==1){
43         /* Izdvajamo levu cifru */
44         leva_cifra=x/max_stepen;
45         /* Izdvajamo desnu cifru */
46         desna_cifra=x%10;
```

2 Kontrola toka

```
49      /* Ako su cifre razlicite, odmah mozemo da zakljucimo da
   * broj nije palindrom i da prekinemo izvršavanje 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 2.56

```
1  /*
   Napisati program koji poziva korisnika da unese pozitivan ceo broj
   n,
3  a zatim za unetih n celih brojeva ispisuje sumu pozitivnih i sumu
   negativnih brojeva.
5
   */
7
   #include<stdio.h>
9
   int main()
11  {
       int n;
13      int x;
       int suma_poz;
15      int suma_neg;
       int i;
17
       printf("Unesi pozitivan ceo broj:");
19      scanf("%d",&n);

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

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

Rešenje 2.57

```
/* 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(){
7     int n, i, x;
8     int zbir=0;
10
11     printf("Unesite broj n: ");
12     scanf("%d", &n);
13
14     printf("Unesite n brojeva: ");
15
16     /* Inicijalizujemo brojac kojim kontrolisemo broj ucitavanja -
17        * treba da ih bude tacno n
18        */
19     i=0;
20     while(i<n){
21         /* Ucitavamo broj */
22         scanf("%d", &x);
23
24         /* Proveravamo da li broj negativan i neparan */
25         if(x<0 && x%2!=0){
26             /* Ako jeste, dodajemo ga na zbir */
27             zbir=zbir+x;
28         }
29     }
```

2 Kontrola toka

```
        /* Uvecavamo brojac iteracija */
30     i++;
    }

32     /* Ispisujemo rezultat */
34     printf("%d\n", zbir);

36     return 0;
}
```

Rešenje 2.58

```
1  /*
   * Napisati program koji omogućava korisniku da unosi cele brojeve dok
   * ne unese nulu. Nakon toga ispisati proizvod onih unetih brojeva
   * koji
   * su pozitivni.
   */
7  #include <stdio.h>
   int main()
9  {
   int x;
11  int p;

13  p=1;
   while (1) /* izraz 1 je konstantan; razlicit je od nule sto znaci
   da ga tumacimo kao tacnog */
15  {
   printf("Unesi jedan ceo broj:");
17   scanf("%d", &x);
   if (x==0) /* ukoliko je uneta nula */
19   break; /* break prekidamo petlju; izvršavanje se nastavlja
   od prve naredbe nakon petlje */

21   if (x<0) /* ukoliko je unet negativan broj, tu vrednost ne
   zelimo da pomnozimo sa ukupnim proizvodom p; zato moramo
   nastaviti dalje */
       continue; /* sa izvršavanjem petlje; continue prekida
   trenutnu iteraciju petlje tako sto preskace sve naredbe
23   koje nakon njega slede; izvršavanje se
   nastavlja od provere uslova petlje */
   p=p*x;
25  }

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

29  return 0;
}
```

Rešenje 2.59

```

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

10

12     printf("Unesi n>0:");
     scanf("%d", &n);
14     if (n<=0)                                /* ako je unos neispravan */
     {
16         printf("Neispravan unos\n");
         return -1;                                /* prekidamo izvršavanje
   programa pomocu naredbe return */
18     }                                           /* u slucaju greske kao sto je
   neispravan unos vratamo 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 2.61

```

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>

```

2 Kontrola toka

```
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

19     printf("Unesite brojeve: ");

21     /* U petlji */
    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
41     return 0;
}
```

Rešenje 2.61

```
1 /* Sa standardnog ulaza se unose realni brojevi sve do unosa broja 0.
   Napisati program koji izracunava i ispisuje
   aritmeticku sredinu unetih brojeva. */
3
4 #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 2.62

```
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
```

```
17     printf("Unesite broj n: ");
    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 2.101

Rešenje 2.101

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

8   int main(){

10       int n;
       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

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

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

```

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 najvećim brojem cifara. Ukoliko ima više takvih, ispisati
prvi. */

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

6
8 int main(){
10     int n;
    int x, x_kopija, br_cifara;
    int max_br_cifara, broj;

```

2 Kontrola toka

```
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 */
printf("Unesite n brojeva: ");
22  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;
11     int x, x_kopija;
12     int broj;
13     int vodeca_cifra, max_vodeca_cifra;
14     int i;

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

20     /* Postavljam maksimalnu vodecu cifru na 0 - cifre broja su vece
        ili jednake od 0 pa je ova vrednost neutralna */
21     max_vodeca_cifra=0;

22     /* Ucitavamo broj po broj */
23     printf("Unesite n brojeva: ");
24     for(i=0; i<n; i++){
25         scanf("%d", &x);

26         /* Odredjujemo vodecu cifru broja */
27         x_kopija=abs(x);
28         while(x_kopija>10){
29             x_kopija=x_kopija/10;
30         }
31         vodeca_cifra=x_kopija;

32         /* Ako je izdvojena cifra veca od maksimalne vodece cifre */
33         if(vodeca_cifra>max_vodeca_cifra){
34             /* Cuvamo je */
35             max_vodeca_cifra=vodeca_cifra;
36             /* I zbog ispisa, cuvamo i broj u kojem se ona pojavljuje */
37             /* Zbog ovoga smo morali i da racunamo vodecu cifru nad kopijom
38                broja x kako ne bismo promenili njegovu vrednost */
39             broj=x;
40         }
41     }

42     /* Ispisujemo rezultat */
43     printf("%d\n", broj);

44     return 0;
45 }

```

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
6  #include <stdio.h>
   #include <math.h>
8
10 int main(){
12     int n;
13     int d;
14     int x, y;
15     int broj_parova;
16     int i;
17
18     /* Ucitavamo vrednosti sa ulaza */
19     printf("Unesite brojeve n i d: ");
20     scanf("%d %d", &n, &d);
21
22     /* Inicijalizujemo broj parova */
23     broj_parova=0;
24
25     printf("Unesite n brojeva: ");
26
27     /* Ucitavamo prvi broj */
28     scanf("%d", &x);
29
30     for(i=1; i<n; i++){
31         /* Ucitavamo naredni broj */
32         scanf("%d", &y);
33
34         /* Ako su brojevi na rastojanju d */
35         if(abs(y-x)==d)
36             /* Treba uvecati broj parova */
37             broj_parova++;
38
39         /* Cuvamo broj iz tekuce iteracije kako bismo mogli da ga
40            upotrebimo u narednoj iteraciji */
41         x=y;
42     }
43
44     /* Ispisujemo rezultat */
45     printf("Broj parova: %d\n", broj_parova);
46
47     return 0;
48 }
```

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  
5  #include <stdio.h>  
   #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()
10 {
11     int c;
13     /* funkcija getchar ucitava jedan karakter.
        naredbom dodele (c=getchar()) promenljivoj c bice dodeljena
        vrednost
        ascii koda unetog karaktera
        obratiti paznju na zagrade!
17    */
18    while((c=getchar())!='.')
19    {
20        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')
24            putchar(c-'a'+'A');
25        else
26            putchar(c);
27    }
28    return 0;
29 }
```

Rešenje 2.71

```
2  /*
3     Napisati program koji omogucava korisniku da unosi karaktere dok
4     ne zada EOF a potom ispisuje broj velikih slova, broj malih slova
5     ,
6     broj cifara, broj belina i zbir cifara.
7  */
8
9  #include <stdio.h>
10
11 int main()
12 {
13     /* promenljivoj c dodelicemo povratnu vrednost funkcije getchar()
        funkcija getchar() ucitava jedan karakter sa standardnog ulaza
        i vraca njegov ascii kod; povratna vrednost funkcije getchar je
        int, pa i promenljiva c mora biti tipa int
14    */
15
16    int c;
18
19    /* brojac moraju biti inicijalizovani na 0 */
```

```

20  int br_v=0;
22  int br_m=0;
24  int br_c=0;
26  int br_b=0;
28  int br_k=0;
30  int suma=0;

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

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

```

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 */
25     for(i=0; i<n; i++){
        printf("Unestite %d. karakter: ", i+1);
27         /*
            Prvo citamo znak za novi red koji je ostao neprocitan nakon
            pritiska Enter tastera
29             posle prethodnog unosa, pa tek onda citamo karakter koji treba
            obradjivati
            */
31         scanf("%c%c", &novi_red, &c);

33         /* Analiziramo karakter */
        switch(c){
35             case 'Z':
                broj_Z++;
37                 break;
            case 'i':
                broj_i++;
39                 break;
            case 'm':
                broj_m++;
41                 break;
            case 'a':
                broj_a++;
43                 break;
45         }
47     }
49 }

/* Ako imamo barem jedno veliko slovo z i barem po jedno malo slovo
    i, m i a */
```



```

51  if(broj_Z && broj_i && broj_m && broj_a){
    /* Zakljucujemo da se rec moze napisati */
53      printf("Moze se napisati rec Zima.\n");
    }
55  else{
    /* Inace, obavestavamo korisnika da je to nemoguce */
57      printf("Ne moze se napisati rec Zima.\n");
    }
59
    return 0;
61 }

```

Rešenje 2.74

```

/*
2   Napisati program koji za uneti pozitivan ceo broj
   izracunava njegov faktorijel. Testirati program
4   za razlicite vrednosti promenljive x. Obratiti paznju
   da pocev od 23! dolazi do prekoračenja.
6 */

8  #include<stdio.h>

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

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

20     original=x;
     f=1;
22     if (x<0)
         printf("Nekorektan unos\n");
24     else
     {
26         while (x>1)
         {
28             f=f*x; /* vrednost izraza sa desne strane naredbe dodele
                        dodeljujemo promenljivoj sa leve strane naredbe
30             */
                 x--; /* operator -- umanjuje vrednost promenljive x za 1
32                     naredba x--; ima isti efekat kao x-=1;
                        ili x=x-1;
34                     */
         }
     }

```

2 Kontrola toka

```
36     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 */
38 }
40
42     return 0;
}
```

Rešenje 2.75

```
1  /* Sa standradnog ulaza unose se realan broj x i ceo neoznacen broj n
   . Napisati
   program koji izracunava x^n */
3
4  #include <stdio.h>
5
6  int main(){
7
8      int n;
9      float x;
10     float vrednost;
11     unsigned exp;
12
13     /* Ucitavaju se brojevi x i n */
14     printf("Unesite redom brojeve x i n: ");
15     scanf("%f %d", &x, &n );
16
17     /* Pocetna vrednost stepena koji se racuna */
18     vrednost=1;
19
20     for(exp=1; exp<=n; exp++)
21         vrednost=vrednost*x;
22
23     /* Stampamo rezultat */
24     printf("%f\n",vrednost);
25
26     return 0;
27 }
```

Rešenje 2.76

```
1  /* Sa standradnog ulaza unose se realan broj x i ceo broj n. Napisati
   program koji izracunava x^n */
3
4  #include <stdio.h>
5
```

```

7  int main(void){
9      int n, n_abs;
10     float x;
11     float vrednost;
12     unsigned exp;

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

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

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

23     /* Stampamo rezultat */
24     if(n<0){
25         printf("%.3f\n",1/vrednost);
26     }
27     else{
28         printf("%.3f\n",vrednost);
29     }

30     return 0;
31 }

```

Rešenje 2.78

```

14 /*
2     a) Napisati program za uneti pozitivan ceo broj n ispisuje zbir
3        s = 1+2^3+3^3+...+n^3. Na primer, za n=4, izlaz iz programa
4        treba da bude:
5        Suma kubova od 1 do 4 je 100
6     b) Modifikovati program tako da ispisuje zbir s = 1+2^3+3^3+...+k
7        ^3
8        za svako i od 1 do n. Na primer, za n=4, izlaz iz programa
9        treba da
10    bude:
11    i=1, n=1
12    i=2, n=9
13    i=3, n=36
14    i=4, n=100
15 */
16 #include <stdio.h>

```

2 Kontrola toka

```
18 int main()
19 {
20     int n;
21     int i;
22     int s;
23
24     printf("Unesite jedan pozitivan ceo broj:");
25     scanf("%d", &n);
26
27     if (n<0)
28         return -1;
29
30     i=1;
31     s=0; /* inicijalizacija promenljive u kojoj se cuva suma kubova */
32
33     for(i=1;i<=n;i++)
34     {
35         s+=i*i*i;
36         /* b) */
37         printf("i=%d, s=%d\n", i, s);
38     }
39     /* a) */
40     printf("Suma kubova od 1 do %d: %d\n", n, s);
41     return 0;
42 }
```

Rešenje 2.78

```
1  /*
2      a) Napisati program za uneti pozitivan ceo broj n ispisuje zbir
3          s = 1+2^3+3^3+...+n^3. Na primer, za n=4, izlaz iz programa
4          treba da bude:
5          Suma kubova od 1 do 4 je 100
6          b) Modifikovati program tako da ispisuje zbir s = 1+2^3+3^3+...+k
7              ^3
8              za svako i od 1 do n. Na primer, za n=4, izlaz iz programa
9              treba da
10             bude:
11             i=1, n=1
12             i=2, n=9
13             i=3, n=36
14             i=4, n=100
15
16             */
17
18     #include <stdio.h>
19
20     int main()
21     {
22         int n;
```

```

21  int i;
22  int s;
23
24
25  printf("Unesite jedan pozitivan ceo broj:");
26  scanf("%d", &n);
27
28  if (n<0)
29      return -1;
30
31  i=1;
32  s=0; /* inicijalizacija promenljive u kojoj se cuva suma kubova */
33
34  for(i=1;i<=n;i++)
35  {
36      s+=i*i*i;
37      /* b) */
38      printf("i=%d, s=%d\n", i, s);
39  }
40  /* a) */
41  printf("Suma kubova od 1 do %d: %d\n", n, s);
42  return 0;
43 }

```

Rešenje 2.79

```

1  /* Sa standardnog ulaza unose se realan broj x i ceo neoznaceni broj n
   . Napisati
   program koji izracunava sumu  $S = x + 2x^2 + 3x^3 + \dots + nx^n$  */
2
3  #include <stdio.h>
4
5  int main(){
6      unsigned n, i;
7      float x, S, stepen;
8
9      printf("Unesite redom brojeve x i n: ");
10     scanf("%f %u", &x, &n);
11
12     /* Inicijalizujemo sumu koju racunamo */
13     S=0;
14
15     /* Stepen promenljiva ce sadrzati vrednosti stepena  $x^n$  -
      * pocetna vrednost joj je 1 */
16     stepen=1;
17
18     for(i=1; i<=n; i++){
19         stepen=stepen*x;
20         S=S+i*stepen;
21     }
22 }
23

```

2 Kontrola toka

```
25     printf("S=%f\n", S);
27     return 0;
}
```

Rešenje 2.80

```
1  /* Sa standardnog ulaza unose se realan broj x i ceo neoznaceni broj n
   *
   * Napisati program koji izracunava sumu  $S=1+1/x+1/x^2+1/x^3+\dots+1/x^n$ 
   */
3
4  #include <stdio.h>
5  int main(){
6      unsigned n, i;
7      float x, S, stepen;
8
9      printf("Unesite redom brojeve x i n: ");
10     scanf("%f %u", &x, &n);
11
12     S=1;
13     stepen=1;
14     for(i=1; i<=n; i++){
15         stepen=stepen*x;
16         S=S+1/stepen;
17     }
18
19     printf("S=%f\n", S);
20
21     return 0;
22 }
```

Rešenje 2.81

```
1  /* Napisati program koji sa zadatom tacnoscu izracunava sumu
2   $S=1+x+x^2/2!+x^3/3!+\dots+x^n/n! + \dots$ 
3  /* Napomena: ovo je razvoj funkcije  $e^x$  */
4
5  #include <stdio.h>
6  #include <math.h>
7  int main(){
8      int n, i, faktoriyel;
9      float S;
10     float x, eps, stepen;
11
12     printf("Unesite x: ");
13     scanf("%f", &x);
14
15     printf("Unesite tacnost eps: ");
```

```

16     scanf("%f", &eps);

18

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

25
26     S=1;
27     faktorijel=1;
28     stepen=x;
29     i=2;
30     while(fabs(stepen/faktorijel)>eps){
31         S=S+stepen/faktorijel;
32         stepen=stepen*x;
33         faktorijel=faktorijel*i;
34         i++;
35     }

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

38     return 0;

40 }

```

Rešenje 2.82

```

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

4

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

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

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

16

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

```

```
24     S=1;
26     faktorijel=1;
27     stepen=x;
28     i=2;
29     znak=-1;
30     while(fabs(stepen/faktorijel)>eps){
31         S=S+znak*stepen/faktorijel;
32         stepen=stepen*x;
33         faktorijel=faktorijel*i;
34         znak=-znak;
35         i++;
36     }
37
38     printf("S=%f\n", S);
39
40     return 0;
41 }
```

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.91](#)

```
1 #include <stdio.h>
2
3 int main(){
4
5     int n, i, j;
6
7     printf("Unesite broj n: ");
8     scanf("%d", &n);
9 }
```



```

11  /* Krstice koje iscrtavamo mozemo posmatrati kao dijagonale
    kvadrata dimenzije n */
13  /* Prolazimo kroz sve vrste kvadrata */
    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 2.91

```

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

```

```
25     putchar(' ');
27     /* Nakon uspesno iscrtane vrste, stampamo znak za novi red */
29     putchar('\n');
31     return 0;
}
```

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.94](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

Rešenje [2.101](#)

2.5 Funkcije

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

TODO U nekim zadacima pise samo Napisati program koji testira rad ove funkcije, dok u nekim zadacima je detaljnije opisan sam program, npr "Napisati program koji sa standardnog ulaza ucitava tri cela broja i ispisuje rezultat poziva funkcije."Nekako bi trebalo to ujednaciti. Meni je lepsi ovaj drugi stil, jer je precizniji.

Ne postoji glavni i sporedni program, vec samo jedan program, izbaciti rec glavni, ja sam izbacila sa puno mesta, mozda mi je negde promaklo.

TODO Smisliti odgovarajuci redosled za ove zadatke

Zadatak 2.102 Ovaj zadatak bih razbila na dva zadatka jer u resenju ima dve poenteod kojih je prvi resen, a drugi neresen. Poentu o vidljivosti promenljivih bih ostavila za kasnije, za neki drugi zadatak, jer mi je ovde to mnogo rano — prvih par zadataka ne bi trebalo dodatno time opterecivati. 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 2.102]

Zadatak 2.103 Napisati funkciju `float stepen(float x, int n)` koja računa vrednost n -tog stepena realnog broja x . Napisati program koji testira rad ove funkcije.

[Rešenje 2.103]

Zadatak 2.104 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 2.104]

Zadatak 2.105 Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato n vraća zbir recipročnih vrednosti brojeva od 1 do n . Napisati program koji testira rad ove funkcije. Rezultat zaokružiti na dve decimale.

[Rešenje 2.105]

Zadatak 2.106 Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji testira rad ove 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 2.106]

Zadatak 2.107 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 sinusne funkcije u n ravnomerno raspoređenih tačaka intervala $[x, y]$. Napisati program koji testira rad ove funkcije.

[Rešenje 2.107]

Zadatak 2.108 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 2.108]

Zadatak 2.109 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 2.109]

Zadatak 2.110 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 2.110]

Zadatak 2.111 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 2.111]

Zadatak 2.112 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 2.112]

Zadatak 2.113 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 2.113]

Zadatak 2.114 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 2.114]

Zadatak 2.115 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 [2.115](#)]

Zadatak 2.116 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 [2.121](#)]

Zadatak 2.117 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 [2.121](#)]

Zadatak 2.118 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 [2.121](#)]

Zadatak 2.119 *preimenovala sam funkciju, preimenovati i u resenju* Napisati funkciju `int od_nove_godine(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do datog datuma. Napisati program koji testira napisanu funkciju.

[Rešenje [2.121](#)]

Zadatak 2.120 *Grupisati sve zadatke sa datumima preimenovala sam funkciju, preimenovati i u resenju* Napisati funkciju `int do_kraja_godine(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati program koji testira napisanu funkciju.

[Rešenje [2.121](#)]

Zadatak 2.121 preimenovala sam funkciju, preimenovati i u resenju Napisati funkciju `int broj_dana_između(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati program koji testira napisanu funkciju.

[Rešenje 2.121]

Zadatak 2.122 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 2.122]

Zadatak 2.123 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 2.123]

Zadatak 2.124 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 2.124]

Zadatak 2.125 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 2.125]

Zadatak 2.126 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 2.126]

Zadatak 2.127 Ovo je jedini zadatak gde nije dat potpis funkcije. Ili i ovde dodati potpis, ili razmotriti da se jos negde ukloni potpis funkcije! Napisati program za ispitivanje svojstava cifara datog celog broja.

- 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.
- 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.

Testirati napisane funkcije na unetom celom broju i ispisati odgovarajuće poruke.

[Rešenje 2.127]

Zadatak 2.128 Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva uneta neoznačena broja x i n utvrđuje da li je x neki stepen broja n . Ukoliko jeste, funkcija vraća izložilac stepena, a u suprotnom vraća -1. Napisati program koji testira rad ove funkcije.

[Rešenje [2.128](#)]

Zadatak 2.129 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 [2.129](#)]

Zadatak 2.130 Za dati broj može se formirati niz tako da je svaki sledeći član niza dobijen kao suma cifara prethodnog člana niza. Broj je srećan ako se dati niz završava jedinicom. Napisati funkciju `int srecan(int x)` koja vraća 1 ako je broj srećan, a 0 u suprotnom. Napisati program koji za uneti prirodan broj n ispisuje sve srećne brojeve od 1 do n .

[Rešenje [2.130](#)]

Zadatak 2.131 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 [2.131](#)]

Zadatak 2.132 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 2.132]

Zadatak 2.133 Napisati funkciju *int faktorijel(int n)* koja računa faktorijel broja *n*. Napisati i program koji učitava dva cela broja *x* i *y* iz intervala $[0, 12]$ i ispisuje vrednost zbira $x! + y!$.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 2.133]

Zadatak 2.134 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 2.134]

Zadatak 2.135 Broj je Armstrongov ako je jednak sumi nekog stepena svojih cifara.

2 Kontrola toka

- (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.

Napisati program koji za ceo broj koji se unosi sa standardnog ulaza proverava da li je Armstrongov.

Primer 1

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

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 1634  
|| Broj je Armstrongov!
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 118  
|| Broj nije Armstrongov!
```

[Rešenje 2.135]

Zadatak 2.136 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 2.136]

Zadatak 2.137 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 2.137]

Zadatak 2.138 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 funkciju *long int fibonaci(int n)* koja računa n -ti element Fibonačijevog niza. Napisati i program koji učitava ceo broj n ($0 \leq n \leq 50$) i ispisuje traženi Fibonačijev broj.

Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 7
21

```

Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 65
Greska: nedozvoljena vrednost!

```

[Rešenje 2.138]

Zadatak 2.139 Napisati funkciju *char sifra(char c, int k)* koja za dati karakter c određuje šifru na sledeći način: ukoliko je c slovo, šifra je karakter koji se nalazi k pozicija ispred njega u abecedi. Karakteri koji nisu slova se ne šifruju. Šifrovanje treba da bude kružno, što znači da je, na primer, šifra za karakter b i pomeraj 2 karakter z . Napisati program koji učitava karakter po karakter do kraja ulaza i ispisuje šifrovani tekst.

Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj k: 2
Unesite tekst (CTRL+D za prekid):
c
a
8
8
+
+
Z
X

```

[Rešenje 2.139]

2.6 Rešenja

Rešenje 2.102

```
#include <stdio.h>

2
int kvadrat(int x)
4 {
    /* promenljive u listi argumenata funkcije, kao i one
6     deklarisanе u samoj funkciji, lokalne su za tu funkciju
    sto znaci da se promenljive x i y neće "videti" nigde izvan
8     funkcije kvadrat (ni u funkciji main ni u funkciji kub)
    */

10
    int y;
12    y = x*x;
    return y;
14 }

16 int kub(int a)
{
18     /*
        u listi argumenata funkcije mozemo, a ne moramo, imati
        promenljivu
20         istog naziva kao promenljiva koja je deklarisanа u main
        funkciji
        (u ovom slucaju promenljiva a); ova promenljiva se razlikuje
22         od promenljive a deklarisanе u main funkciji i vidljiva je
        samo unutar funkcije kub
24     */
    return a*a*a;
26 }

28 int main()
{
30     int a, kv, kb;
    printf("Unesi ceo broj:");
32     scanf("%d", &a);

34     kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
        funkcije kvadrat */
    kb = kub(a);      /* promenljivoj kb dodeljujemo povratnu vrednost
        funkcije kub */
36
    printf("Kvadrat broja %d je %d, a njegov kub je %d\n", a, kv, kb);
38     return 0;
}
```

Rešenje 2.103

```
1  /*
2  Napisati program koji za uneti realan broj x i ceo broj n ispisuje
3  vrednost stepena x^n. Unosenje promenljivih, racunanje stepena i
4  ispis promenljivih realizovati u posebnim funkcijama.
5  */
6
7  #include <stdio.h>
8  #include <stdlib.h>
9
10 float stepen(float a, int b)
11 {
12     float s=1;
13     int i;
14
15     for(i=0;i<abs(b);i++)
16         s=s*a;
17
18     return b>0 ? s : 1/s; /* ukoliko je izlozilac b negativan,
19                            izracunamo a^|b| i vracamo reciprocnu vrednost
20                            izracunatog stepena */
21 }
22
23 int main()
24 {
25     int n;
26     float x;
27     float s;
28
29     printf("Unesi jedan realan i jedan ceo broj:");
30     scanf("%f%d",&x,&n);
31
32     s = stepen(x,n);
33
34     printf("%f^%d=%f\n",x,n,s);
35
36     return 0;
37 }
38
39 }
```

Rešenje 2.104

```
1  /*
2  Napisati funkciju koja za dva data cela broja odredjuje
3  najveći zajednicki delilac. Napisati potom glavni program
4  koji testira ovu funkciju.
5  */
6
7  #include <stdio.h>
```

2 Kontrola toka

```
9 int euklid(int x, int y)
{
11     int r;
    /* Euklidov algoritam */
13     while(y) /* algoritam se zaustavlja kada vrednost */
    {          /* promenljive y postane nula */
15         r=x%y;
        x=y;
17         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 2.105

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

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
19  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
26  naredbe printf i umesto specifikatora %.2f bice ispisana
       povratna vrednost funkcije
       zbir_reciprocnih zaokruzena na dve decimale */
28  return 0;
    }

```

Rešenje 2.106

```

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

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

17     if (x==0) /* u slucaju da je uneta 0 */
         return 0; /* aritmeticka sredina cifara iznosi 0 i tu vrednost
            vratamo */
19

21     x=abs(x); /* uzimamo apsolutnu vrednost broja za slucaj da je
        negativan */

23     while(x)
        {

```

2 Kontrola toka

```
25     cifra=x%10;

27     broj_cifara++;
    zbir_cifara+=cifra;

29     x/=10;
31 }

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

37 int main()
38 {
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 2.107

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

7
9  #include <stdio.h>
    #include <math.h>

11 void ispis(float x, float y, int n) /* funkcija nema povratnu
    vrednost; zbog toga je povratni tip void */
    {
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()
22 {
23     float a,b;
24     int n;
25     float t;
26     printf("Unesi dva realna broja:");
27     scanf("%f%f",&a,&b);
28     printf("Unesi jedan ceo broj > 1:");
29     scanf("%u",&n);
30
31     if (n<=1 || a==b)
32     {
33         printf("Nekorektan unos\n");
34         return -1;
35     }
36     if (b<a) /* u slucaju da je desni kraj intervala manji od levog */
37     {       /* zamenimo im mesta */
38         t=a;
39         a=b;
40         b=t;
41     }
42
43
44
45     ispis(a,b,n);
46
47     return 0;
48 }

```

Rešenje 2.108

```

/*
2  Napisati funkciju koja broji neparne cifre u zapisu datog celog broja
   . Napisati
   potom glavni program koji unosi cele brojeve dok se ne unese nula, i
   ispisuje
4  broj neparnih cifara svakog unetog broja koriscenjem napisane
   funkcije.
*/
6
8  #include<stdio.h>
9  #include<stdlib.h>
10
11 int broj_ncifara(int x)
12 {
13     int s=0;
14     char cifra;
15     x = abs(x);
16
17     while(x)

```

2 Kontrola toka

```
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 }

28 int main()
29 {
30     int x;
31     do
32     {
        scanf("%d",&x);
        printf("Broj neparnih cifara u zapisu broja %d: %d\n", x,
        broj_ncifara(x));
34     } while(x!=0);

36     return 0;
38 }
```

Rešenje 2.109

```
1  #include <stdio.h>

3  /*
4   Funkcija koja racuna minimum tri cela broja
5  */
6  int min(int x, int y, int z){
7      int min;

9      min=x;

11     if(min>y)
        min=y;

13     if(min>z)
        min=z;

15     return min;
17 }

19 int main(){
21     int x,y,z;

23     /* Ucitavamo brojeve */
    printf("Unesite brojeve: ");
25     scanf("%d%d%d", &x, &y, &z);
```

```

27  /* Pozivamo funkciju i ispisujemo rezultat */
    printf("Minimum je: %d\n", min(x,y,z));
29
    return 0;
31 }

```

Rešenje 2.110

```

1  #include <stdio.h>

3  /* Funkcija koja racuna apsolutnu vrednost */
unsigned int apsolutna_vrednost(int x){
5      /* Kako funkcija vraca unsigned, a x je tipa int, vrsimo kastovanje
    rezultata u tip unsigned */
    return (unsigned)(x<0?-x:x);
7  }

9  int main(){
    int n;
11
    /* Ucitavamo broj */
13    printf("Unesite broj: ");
    scanf("%d", &n);
15
    /* Ispisujemo njegovu apsolutnu vrednost */
17    printf("Apsolutna vrednost: %u\n", apsolutna_vrednost(n));
19
    return 0;
}

```

Rešenje 2.111

```

1  #include<stdio.h>
    #include<math.h>
3
    /* Funkcija koja vraca razlomljeni deo prosledjenog broja */
5  float razlomljeni_deo(float x){
7
    /* Funkcija fabs vraca apsolutnu vrednost realnog broja
     * NAPOMENA: funkcija fabs se nalazi u zaglavlju math.h
9     * NAPOMENA2: funkcija abs se nalazi u zaglavlju stdlib.h, ali se
    koristi samo za cele brojeve!
    */
11    x = fabs(x);

13    /* Razlomljeni deo broja dobijamo tako sto od samog broja oduzmemo
    njegov ceo deo*/
    return x - (int)x;
}

```

```
15 }  
  
17 int main(){  
    float n;  
  
19     /* Ucitavamo broj */  
21     printf("Unesite broj:");  
    scanf("%f", &n);  
  
23     /* Ispisujemo rezultat */  
25     printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));  
  
27     return 0;  
}
```

Rešenje 2.112

```
1 #include<stdio.h>  
  
3 /* Funkcija koja iscrtava romb */  
void romb(int n){  
5     int i, j;  
  
7     /* U svakoj liniji */  
    for(i=0; i<n; i++){  
9  
11        /* Prvo ispisujemo n-i-1 razmaka */  
        for(j=0; j<n-i-1; j++)  
            printf(" ");  
  
13        /* Zatim ispisujemo n zvezdica */  
15        for(j=0; j<n; j++)  
            printf("*");  
  
17        /* Na kraju svake linije stoji oznaka za novi red */  
19        printf("\n");  
    }  
  
21 }  
  
23 int main(){  
25     int n;  
  
27     /* Ucitavamo broj n */  
    printf("Unesite broj n: ");  
29    scanf("%d", &n);  
  
31    /* Proveravamo korektnost ulaza i ispisujemo rezultat */  
    if(n<=0)  
33        printf("Greska: pogresna dimenzija!\n");  
    else
```

```
35     romb(n);
37     return 0;
}
```

Rešenje 2.113

```
#include<stdio.h>

2
/* Funkcija koja stampa n zvezdica za kojima sledi znak za novi red
   */
4 void stampaj_zvezdice(int n){
    int i;
6    for(i=0; i<n; i++)
        printf("*");

8    printf("\n");
10 }

12 /* Funkcija koja crta grafikon */
void grafikon_h(int a, int b, int c, int d)
14 {
    int i;

16    /* Prvo ispisujemo a zvezdica */
18    stampaj_zvezdice(a);

20    /* Zatim u sledecem redu b zvezdica */
    stampaj_zvezdice(b);

22    /* Zatim u sledecem redu c zvezdica */
24    stampaj_zvezdice(c);

26    /* Zatim u poslednjem redu d zvezdica */
    stampaj_zvezdice(d);

28 }

30
32 int main(){
    int a,b,c,d;

34    /* Ucitavamo vrednosti a,b,c,d */
    printf("Unesite vrednosti: ");
36    scanf("%d%d%d%d", &a, &b, &c, &d);

38    /* Proveravamo korektnost ulaza i ispisujemo rezultat */
    if(a <0 || b<0 || c<0 || d<0){
40        printf("Greska: pogresan unos!\n");
    }else{
42        grafikon_h(a,b,c,d);
    }
}
```

```
44     return 0;
46 }
```

Rešenje 2.114

```
1  #include<stdio.h>
3  int maksimum(int a, int b, int c, int d){
4      int max;
5
6      max=a;
7      if(b>max)
8          max=b;
9      if(c>max)
10         max=c;
11     if(d>max)
12         max=d;
13
14     return max;
15 }
17 /* Funkcija koja iscrtava vertikalni grafikon */
18 void grafikon_v(int a, int b, int c, int d){
19     int i, max;
20
21     /* Na pocetku je potrebno pronaci najveću od ove četiri vrednosti
22      */
23     max=maksimum(a, b, c, d);
24
25     /* Grafikon ukupno ima max horizontalnih linija */
26     for(i=0; i<max; i++){
27
28         /* U svakoj od horizontalnih linija se nalazi po 4 polja:
29          polje za a,b,c i d uspravnu liniju.
30          U svako od polja treba da se upise ili * ili belina,
31          u zavisnosti od vrednosti a i toga u kojoj liniji se trenutno
32          nalazimo
33          */
34
35         /* Proveravamo uslov za polje a */
36         if(i<max-a)
37             printf(" ");
38         else
39             printf("*");
40
41         /* Proveravamo uslov za polje b */
42         if(i<max-b)
43             printf(" ");
44         else
45             printf("*");
```



```

45  /* Proveravamo uslov za polje c */
46  if(i<max-c)
47      printf(" ");
48  else
49      printf("*");
50
51  /* Proveravamo uslov za polje d */
52  if(i<max-d)
53      printf(" ");
54  else
55      printf("*");
56
57  /* Na kraju svake horizontalne linije stampamo novi red */
58  printf("\n");
59  }
60 }
61
62 int main(){
63     int a,b,c,d;
64
65     /* Ucitavamo vrednosti a,b,c,d */
66     printf("Unesite vrednosti: ");
67     scanf("%d%d%d%d", &a, &b, &c, &d);
68
69     /* Proveravamo korektnost ulaza i stampamo grafikon */
70     if(a <0 || b<0 || c<0 || d<0)
71         printf("Greska: pogresan unos!\n");
72     else
73         grafikon_v(a,b,c,d);
74
75     return 0;
76 }

```

Rešenje 2.115

```

1  #include<stdio.h>
2
3  /* Funkcija koja proverava da li je godina prestupna */
4  int prestupna(int godina){
5      if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
6          return 1;
7      else
8          return 0;
9  }
10
11 /* Funkcija koja proverava da li postoji prestupna godina u datom
    intervalu */
12 int postoji_prestupna(int g1, int g2){
13     for(; g1<=g2; g1++){
14         if(prestupna(g1))

```

```
15     return 1;
16 }
17 return 0;
18 }
19
20 int main(){
21
22     int g1, g2;
23
24     /* Ucitavamo godine */
25     printf("Unesite dve godine: ");
26     scanf("%d%d", &g1, &g2);
27
28     /* Proveravamo korektnost ulaza */
29     if(g1 < 0 || g2 < 0 || g1>g2){
30         printf("Greska: pogresan unos!\n");
31     }
32     else{
33
34         /* Proveravamo da li uopste postoji prestupna godina u datom
35            intervalu */
36         if(postoji_prestupna(g1,g2)){
37             /* Ako postoje, ispisujemo ih */
38             printf("Prestupne godine su: ");
39             for(; g1<=g2; g1++){
40                 if(prestupna(g1))
41                     printf("%d ", g1);
42             }
43             printf("\n");
44         }else{
45             /* U suprotnom, stampamo odgovarajucu poruku */
46             printf("Nema prestupnih godina u ovom intervalu!\n");
47         }
48         return 0;
49     }
```

Rešenje 2.121

```
1 #include <stdio.h>
2
3 /* Funkcija koja racuna zbir delilaca broja x */
4 int zbir_delilaca(int x){
5     int i=0;
6
7     /* Na pocetku zbir inicijalizujemo na 0 */
8     int zbir = 0;
9
10    /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
11    for(i=1; i<=x; i++){
12        if(x % i == 0)
```

```
13     zbir += i;
14 }
15
16 /* Vracamo dobijeni zbir */
17 return zbir;
18 }
19
20 int main(){
21
22     int k, i;
23
24     /* Ucitavamo broj k */
25     printf("Unesite broj k:");
26     scanf("%d", &k);
27
28     /* Proveravamo korektnost ulaza */
29     if(k <= 0)
30         printf("Greska: pogresan unos!\n");
31     else{
32
33         /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
34         for(i=1; i<=k; i++)
35             printf("%d ", zbir_delilaca(i));
36
37         printf("\n");
38     }
39
40
41     return 0;
42 }
```

Rešenje 2.121

Rešenje 2.121

Rešenje 2.121

Rešenje 2.121

Rešenje 2.121

Rešenje 2.122

```
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");
   else{

31       /*Za svaki broj od 1 do k ispisujemo zbir delilaca*/
       for(i=1; i<=k; i++)
35         printf("%d ", zbir_delilaca(i));

       printf("\n");
37   }

39

41   return 0;
}
```

Rešenje 2.123

```
#include <stdio.h>

2
/* Funkcija koja uklanja broj stotina iz broja n */
4 int ukloni_stotine(int n){

6   /* Ako je broj izmedju -100 i 100 nema cifru desetica pa onda
       vracamo isti taj broj */
   if(n>-100 && n<100)
8     return n;
```

```

10     else
11     {
12         /* U suprotnom vracamo broj sa uklonjenom cifrom stotina */
13
14         /* Odredjujemo znak broja */
15         int znak=(n<0)? -1 : 1;
16
17         /* I nadalje radimo sa apsolutnom vrednoscu broja */
18         n=abs(n);
19
20         return znak*((n/1000)*100 + n%100);
21     }
22 }
23
24 /* Funkcija koja vraca znak broja */
25 int znak(int broj){
26     return broj<0?-1:1;
27 }
28
29 int main(){
30
31     int broj;
32
33     while(1){
34
35         /* Ucitavamo broj sa standardnog ulaza */
36         printf("Unesite broj: ");
37         scanf("%d", &broj);
38
39         /* Broj 0 oznacava kraj rada */
40         if(broj == 0)
41             break;
42
43         /* Ispisujemo rezultat, vodeci racuna da program treba da radi
44            ispravno i za negativne brojeve */
45         printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
46     }
47
48     return 0;
49 }

```

Rešenje 2.124

```

1 #include<stdio.h>
2 #include<math.h>
3
4 int rotacija(int n){
5
6     /* U promenljivoj broj pamtim originalnu vrednost n */
7     int broj, br = 0, znak;
8
9     while(n!=0){
10         broj=n%10;
11         n=n/10;
12         if(broj!=0){
13             br=br*10+broj;
14             if(n<0) znak=-1;
15             else znak=1;
16         }
17     }
18     return znak*br;
19 }

```

```
8
/* Odredjujemo znak broja */
10 znak=(n<0) ? -1: 1;

12 /* I nadalje radimo sa apsolutnom vrednoscu broja */
n=abs(n);

14
/* U promenljivoj broj cuvamo kopiju broja n */
16 broj=n;

18 /* Ako je broj jednocifren, nema potrebe da ga rotiramo. */
if(n>-10 && n < 10)
20     return n;

22 /* 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,
24   zatim da "pomerimo" 234 u levo i da na kraj nalepimo 1 = 2341 */

26 /* 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 */
28 while(n >=10){
    n/=10;
    br++;
30 }

32
/*
34   Levi deo (234) dobijamo kao n%(10^broj_cifara)
   Zatim levi deo pomnozimo sa 10, da bi dobili 2340
36   Zatim na levi deo dodamo desni deo (1) koja se nalazi u
   promenljivoj n
   */
38
return znak* ((broj%(int)pow(10, br))*10 + n);
40 }

42 int main(){

44     int n;
    while(1){

46
/* Ucitavamo broj */
48     printf("Unesite broj: ");
    scanf("%d", &n);

50
/* Ako je uneta 0, izlazimo iz petlje */
52     if(n == 0)
        break;

54
/* Stampamo broj rotiran za jedno mesto u levo */
```

```

56     printf("%d\n", rotacija(n));
57     }
58
60     return 0;
61 }

```

Rešenje 2.125

```

1  /*
2  Napisati funkciju koja ispituje da li je dati ceo broj prost.
3  Funkcija treba
4  da vrati 1 ako je broj prost i 0 u suprotnom. Napisati potom glavni
5  program
6  koji za uneti ceo broj n ispisuje prvih n prostih brojeva.
7  */
8
9  #include <stdio.h>
10 #include <math.h>
11
12 int prost (int x) /* 1-broj je prost, 0-broj nije prost */
13 {
14     int i;
15
16     if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
17         return 1;
18
19     if (x%2==0) /* parni brojevi nisu prosti */
20         return 0;
21
22     for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
23         if (x%i==0) /* ako je pronadjen, to znaci da broj nije prost */
24             return 0; /* završavamo funkciju */
25
26     /* ukoliko izvršavanje funkcije dodje do poslednje naredbe return,
27     to znaci da broj nije ispunio nijedan od prethodnih uslova
28     (nije ni 2, ni 3, ni paran, niti ima ijednog delioca), odakle
29     sledi da je prost i zbog toga vracamo 1
30     */
31     return 1;
32 }
33
34 int main()
35 {
36     int n;
37     scanf("%d",&n);
38     int i,j;
39
40     i=1; /* kandidat za prost broj */
41     j=0; /* brojac prostih brojeva */
42     while(j<n)

```

2 Kontrola toka

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

Rešenje 2.126

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

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

9

11 int sadrzi(int x, int c)
   {
13     char cifra;
       x=abs(x);
       while(x)
15     {
           cifra = x%10;
17         if (cifra==c)
             return 1;
19         x/=10;
       }
21     return 0;
   }

23 int main()
   {
25     int x;
       int c;
27     printf("Unesi jedan ceo broj i jednu cifru:");
       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 2.127

```
2  /*
3
4  a) Napisati funkciju sve_parne_cifre koja ispituje da li se dati ceo
      broj sastoji isključivo iz parnih cifara. Funkcija treba
5  da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.
6
7  b) Napisati funkciju sve_cifre_jednake koja ispituje da li su sve
      cifre datog celog broja jednake. Funkcija treba
8  da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.
9
10 c) Napisati potom glavni program koji na uneti ceo broj primenjuje
      napisane funkcije i ispisuje odgovarajuće poruke.
11
12 Na primer, za uneti broj 222, program treba da ispise:
13 Sve cifre broja su parne.
14 Sve cifre broja su jednake.
15
16 A za uneti broj -284:
17 Sve cifre broja su parne.
18 Broj sadrži različite cifre
19
20 */
21 #include <stdio.h>
22 #include <stdlib.h>
23
24 int sve_parne_cifre(int x) /* funkcija vraća 1 ako su sve cifre broja
      parne i 0 u suprotnom */
25 {
26     char d;
27     x=abs(x);          /* uzimamo apsolutnu vrednost broja za slučaj da je
      broj negativan */
28     while (x>0)
29     {
30         d=x%10;        /* izdvajamo cifru broja */
31
32         if (d%2==1)    /* u slučaju da je neparna, to znači da nisu sve
      cifre broja parne */
33             return 0;  /* vraćamo 0 */
34
35         x/=10;          /* "uklanjamo" poslednju cifru broja celobrojnim
      deljenjem sa 10 */
36     }
37
38     return 1;          /* ukoliko se while petlja završila, to znači da
      uslov d%2==1 nije
39                          nijednom bio ispunjen i da su sve cifre broja
      parne; zbog toga
```

2 Kontrola toka

```

40             /*          vratamo 1
41             */
42     }
43
44     int sve_cifre_jednake(int x) /* funkcija vraca 1 ako su sve cifre
45         broja jednake i 0 u suprotnom */
46     {
47         char d;
48         char prva_cifra;
49         x=abs(x);
50         prva_cifra = x%10; /* izdvajamo prvu cifru broja */
51         x/=10;             /* broj delimo sa 10 jer smo prvu cifru vec
52             izdvojili */
53
54         while(x)
55         {
56             d = x%10;
57
58             if (d!=prva_cifra)
59                 return 0;
60
61             x/=10;
62         }
63
64         return 1;
65     }
66
67     main()
68     {
69         int x;
70         int d;
71
72         printf("unesi ceo broj:");
73         scanf("%d", &x);
74
75         if (sve_parne_cifre(x))
76             printf("Sve cifre broja su parne\n");
77         else
78             printf("Broj sadrzi bar jednu neparnu cifru\n");
79
80         if (sve_cifre_jednake(x))
81             printf("Sve cifre broja su jednake\n");
82         else
83             printf("Broj sadrzi razlicite cifre \n");
84     }

```

Rešenje 2.128

```
/*
```

```

2  Napisati funkciju koja za dva uneta neoznacena broja x i n utvrđuje
   da li je x neki stepen
   broja n. Ukoliko jeste, funkcija vraća izlozilac stepena, a u
   suprotnom vraća -1. Napisati
4  potom glavni program koji testira ovu funkciju.
   */
6
8  #include <stdio.h>
10
12  int je_stepen(unsigned x, unsigned n) /* funkcija vraća izlozilac
   stepena ukoliko broj x jeste neki stepen broja n */
14  {
16     int i=1;
18     int s=n;
20
22     while(s<x)
24     {
26         s=s*n;
28         i++;
30     }
32
34     if (s==x)
36         return i;
38
40     return -1;
42 }

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 2.129

```

/*
2
   Napisati funkciju
4

```

```
double e_na_x(double x, double eps)

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

14
   */

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

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

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

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

34
       prvi sabirak (kome odgovara n=0) iznosi 1; zbog toga promenljive
36      s i clan inicijalizujemo na vrednost 1

38
       sumiranje se sprovodi dogod je sabirak po apsolutnoj vrednosti
       veci od trazene tacnosti eps
40      */

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

48      return s;
50  }

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

```

58     printf("eps=");
    scanf("%lf", &eps);

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

```

Rešenje 2.130

```

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

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

14 Napisati program koji omogucava korisniku da unese prirodan broj,
    poziva funkciju
    i ispisuje da li je dati broj srecan. Potom traziti od korisnika da
    unese prirodan
16 broj n i ispisati sve srecne brojeve od 1 do n.
*/

18 #include<stdio.h>

20 int zbir_cifara(int x)
22 {
    int s=0;
24     char cifra;
    while(x)
26     {
        cifra = x%10;
28         s+=cifra;
        x/=10;
30     }
    return s;
32 }

```

2 Kontrola toka

```
34 int srecan(int x)
35 {
36     int s; /* promenljiva s sadrzi sumu cifara */
37
38     do
39     {
40         s=zbir_cifara(x);
41         x=s; /* kada izracunamo sumu cifara, dodeljujemo je promenljivoj
42             x jer iz te promenljive izdvajamo cifre u funkciji zbir_cifara
43             */
44     } while(x>=10);
45
46     return (x==1);
47 }
48
49 int main()
50 {
51     unsigned n;
52     int i;
53     printf("Unesi jedan neoznaceni broj:");
54     scanf("%u",&n);
55
56     for(i=1;i<=n;i++)
57         if (srecan(i))
58             printf("%d je srecan\n", i);
59
60     return 0;
61 }
```

Rešenje 2.131

```
/*
2  . a) Napisati funkciju
3
4     int konverzija (int c)
5
6     koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.
7
8     b) Napisati program koji omogućava korisniku da unese niz karaktera
9     sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
10    Na primer, za uneti tekst "Kolokvijum iz Progi je 1.12." program
11    treba da ispise "kOLOVKIJUM IZ pROGI JE 1.12."
12
13    */
14    #include <stdio.h>
15
16    int konverzija(int c)
17    {
18        /* ključna rec return vraća povratnu vrednost funkcije (ako je ima)
19        */
20    }
```

```

20  /* i završava izvršavanje funkcije */
22  if (c>='A' && c<='Z')
    return c+'a'-'A';
24  if (c>='a' && c<='z')
    return c-'a'+'A';
26  return c;
28 }
30 int main()
31 {
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;
39 }

```

Rešenje 2.132

```

1  #include <stdio.h>
3  /* Funkcija int zapis(int x, int y) proverava da li su dva cela broja
    napisana
    * pomocu istih cifara, kao i da li se te cifre pojavljuju
5   * isti broj puta.
    * Ideja je sledeca:
7   * iz broja x izdvajaju se redom cifra po cifra s kraja,
    * a zatim se svaka takva cifra trazi i u broju y.
9   * Ukoliko postoji, eliminise se prvi put kada se pojavi (dakle,
    samo jednom).
    * Ukoliko su sve cifre iste (**redosled nije bitan**),
11  * na kraju ce i iz x i iz y biti sve cifre eliminisane",
    * te ostaju nule u oba broja.
13  *
    * Broj novo_y formira se, zbog jednostavnosti, pomocu Heronovog
    obrasca.
15  * Ovaj postupak obradjen je u okviru funkcije int izbaci_cifru(int
    y, int cifra).
    */
17  int izbaci_cifru(int y, int cifra) {
19      int novo_y = 0;
21      int indikator = 0;
      int izdvojena_cifra;

```

```
23 while(y) {
25     izdvojena_cifra = y % 10;
27     /* U slucaju da se cifra razlikuje od one koju treba eliminisati,
    * ili ukoliko je jedna cifra vec eliminisana =>
29     * tekucu cifru ukljuciti prilikom formiranja novog y
    * */
31     if(izdvojena_cifra != cifra || indikator)

33         /* Heronov obrazac.
    * Menja poredak cifara, ali on u ovom slucaju i nije bitan.
35         */
        novo_y = novo_y*10 + izdvojena_cifra;
37     else

39         /* U slucaju da je cifra vec eliminisana,
    * ne treba je opet eliminisati.
41         * Za svaku pojavu cifre iz x,
    * eliminiše se jedna odgovarajuća pojava
43         * te cifre iz y.
    */
45         indikator = 1;

47     y /= 10;
49 }

51 return novo_y;
52 }

53 int zapis(int x, int y) {

55     /* Cifra koja se izdvaja iz x, a onda eliminiše iz y */
56     int cifra;

57     /* U slucaju da su prosledjeni brojevi negativni */
59     x = abs(x);
60     y = abs(y);

61     while(x) {

63         cifra = x % 10;
65         x /= 10;

67         y = izbaci_cifru(y, cifra);

69         /* otkomentarisi 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);
74 }
75
76 int main() {
77     int x, y;
78     printf("Unesite dva cela broja: ");
79     scanf("%d%d", &x, &y);
80
81     if(zapis(x, y))
82         printf("Uslov je ispunjen!\n");
83     else
84         printf("Uslov nije ispunjen!\n");
85
86     return 0;
87 }
```

Rešenje 2.133

```
1  #include <stdio.h>
2
3  /* Funkcija racuna faktorijel broja.
4   * Faktorijel formiramo mnozenjem sa trenutnom vrednoscu broja x,
5   * a zatim smanjujuci tu vrednost za 1.
6   * Ukoliko je x = 5, f = 5 * 4 * 3 * 2 * 1
7   */
8  int faktorijel(int x) {
9
10     int f = 1;
11     while(x) {
12         f *= x;
13         x--;
14     }
15     return f;
16 }
17
18 int main() {
19     int x, y;
20
21     printf("Unesite dva broja: ");
22     scanf("%d%d", &x, &y);
23
24     /* Provera uslova.
25     *
26     * Faktorijel je veoma brza funkcija, tj.
27     * s povecanjem broja x, drasticno brzo uvecava se i vrednost x!.
28     * Tip podatka int ima ogranicenje u velicini broja koji moze da
29     * sadrzi.
30     * Za 13! i vece, int ne bi mogao da sacuva sve cifre potrebne za
31     * zapis tako velikog broja,
```

2 Kontrola toka

```
31      * te bi doslo do prekoracenja.
32      *
33      * Slicno, faktorijel nije definisan nad skupom negativnih celih
34      * brojeva.
35      */
36      if(x < 0 || y < 0 || x > 12 || y > 12) {
37          printf("Greska: pogresan unos!\n");
38      }
39      else{
40          printf("%d\n", faktorijel(x) + faktorijel(y));
41      }
42      return 0;
43  }
```

Rešenje 2.134

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  /* Funkcija proverava da li se
5   * cifre u zapisu broja nalaze u rastucem poretku.
6   *
7   * Situacija od interesa je kada za dve uzastopne cifre to nije
8   * slucaj.
9   * Tada ne treba proveravati i za ostale cifre,
10  * vec odmah prekinuti izvršavanje funkcije.
11  *
12  * Ukoliko funkcija nije ranije prekinuta,
13  * to znaci da cifre jesu u rastucem poretku
14  * (odnosno, kako izdvajamo cifre od nazad, u stvari proveravamo
15  * opadajuci poredak),
16  * te treba vratiti 1.
17  */
18
19  int rastuce(int n) {
20
21      int tekuca_cifra;
22      int prethodna_cifra;
23
24      n = abs(n);
25
26      /* Prvu cifru (odnosno, poslednju u zapisu broja)
27       * izdvajamo izvan petlje
28       * kako bismo mogli da je poredimo sa narednom
29       */
30      tekuca_cifra = n % 10;
31      n /= 10;
32
33      while(n) {
34
35          /* Cifra koja je bila tekuca u prethodnoj iteraciji petlje,
```

```

35     * u novoj iteraciji postaje prethodna.
36     *
37     * Novoizdvojena cifra je tekuca.
38     */
39     prethodna_cifra = tekuca_cifra;
40     tekuca_cifra = n % 10;
41
42     /* Ukoliko smo naisli na cifre koje kvare rastuci poredak,
43     * prekidamo izvršavanje funkcije sa odgovarajucom povratnom
44     * vrednoscu 0.
45     */
46     if(prethodna_cifra < tekuca_cifra)
47         return 0;
48
49     /* Inace, nastavljamo sa izdvajanjem cifara */
50     n /= 10;
51 }
52
53 return 1;
54 }
55
56 int main() {
57
58     int x;
59     printf("Unesite broj: ");
60     scanf("%d", &x);
61
62     if(rastuce(x))
63         printf("Cifre su u rastucem poretku!\n");
64     else
65         printf("Cifre nisu u rastucem poretku!\n");
66
67     return 0;
68 }

```

Rešenje 2.135

```

1  #include <stdio.h>
2
3  /* Funkcija racuna broj x na n-ti stepen */
4  int stepen(int x, int n) {
5
6      int i;
7      /* Promenljiva u kojoj se cuva proizvod broja x sa samim sobom, n
8      puta */
9      int st = 1;
10
11     for(i = 1; i <= n; i++)
12         st *= x;
13
14     return st;
15 }

```

```

15 }
16
17 /* Funkcija proverava da li je broj Armstrongov. */
18 int armstrong(int x) {
19     /* u y se cuva zbir i-tih stepena cifara */
20     int y;
21     /* stepen za koji se proverava */
22     int i = 1;
23     /* prilikom izdvajanja cifara, broj x se menja,
24      * te treba imati promenlju koja cuva pravu vrednost x
25      */
26     int original = x;
27
28     do {
29
30         y = 0;
31         /* Racunamo i-te stepene za svaku cifru,
32          * i istovremeno te stepen sabiramo.
33          * Rezultat pamtimo u promenljivoj y.
34          */
35         while(x) {
36
37             y += stepen(x % 10, i);
38             x /= 10;
39         }
40
41         /* x je sada promenjen, pa ga treba vratiti na pravu vrednost. */
42         x = original;
43         i++;
44
45     } while(y < x); /* Petlju vrtimo sve dok je zbir stepena cifara
46                    manji od datog broja. */
47
48     /* Ukoliko smo nasli i, takvo da je zbir i-tih stepena cifara
49      * jednak upravo broju x, takav broj je Armstrongov,
50      * te izraz x == y vraca 1.
51      *
52      * Inace, vraca 0, tj. broj nije Armstrongov.
53      */
54     return x == y;
55 }
56
57 int main() {
58
59     int x;
60     printf("Unesite broj: ");
61     scanf("%d", &x);
62
63     if(armstrong(x))
64         printf("Broj je Armstrongov!\n");
65     else

```

```

65     printf("Broj nije Armstrongov!\n");
67     return 0;
}

```

Rešenje 2.136

```

1  #include <stdio.h>
   #include <stdlib.h>
3
4  /* Funkcija proverava da li su dve uzastopne cifre
5   * razlicite parnosti.
6   *
7   * Interesantna situacija je ukoliko su dve uzastopne cifre
8   * obe parne, odnosno obe neparne.
9   * Ovaj uslov svodimo na poredjenje njihovih ostataka pri deljenju sa
10  * 2:
11  * ukoliko su ostaci isti, cifre su iste parnosti,
12  * te ne treba dalje proverati da li je uslov zadovoljen,
13  * vec odmah prekinuti sa izvršavanjem funkcije.
14  *
15  * Ukoliko dve uzastopne cifre ni u jednom slucaju nisu bile iste
16  * parnosti,
17  * a izdvojene su sve cifre iz broja x,
18  * uslov je ispunjen, pa funkcija vraca 1.
19  */
20  int par_nepar(int x) {
21
22     int prethodna_cifra;
23     int tekuca_cifra;
24
25     /* u slucaju da je uneti broj negativan */
26     x = abs(x);
27
28     /* jednu cifru izdvajamo van petlje
29     * kako bismo mogli da je odmah u petlji poredimo sa narednom
30     */
31     prethodna_cifra = x % 10;
32     x /= 10;
33
34     while(x) {
35
36         tekuca_cifra = x % 10;
37
38         if(tekuca_cifra % 2 == prethodna_cifra % 2)
39             return 0;
40
41         /* tekuca cifra postaje prethodna cifra za narednu iteraciju */
42         prethodna_cifra = tekuca_cifra;
43         x /= 10;
44     }
45 }

```

2 Kontrola toka

```
43     return 1;
44 }
45
46
47 int main() {
48
49     int x;
50     printf("Unesite broj: ");
51     scanf("%d", &x);
52
53     if(par_nepar(x))
54         printf("Broj ispunjava uslov!\n");
55     else
56         printf("Broj ne ispunjava uslov!\n");
57
58     return 0;
59 }
```

Rešenje 2.137

```
#include <stdio.h>
2
3 /* Funkcija broji koliko puta se realan broj x
4  * javlja u nizu unetih brojeva sa tastature.
5  *
6  * Brojevi se unose sve do pojave 0,
7  * pa treba koristiti do..while petlju,
8  * kako bi bar jedan broj bio unet (makar bio i 0).
9  */
10 int prebrojavanje(float x) {
11
12     /* y prihvata uneti broj sa tastature */
13     float y;
14     /* br_pojavljivanja je brojac koji broji koliko puta se broj x
15      * javlja u unetom nizu brojeva */
16     int br_pojavljivanja = 0;
17
18     printf("Unesite brojeve: ");
19     do {
20
21         /* Unosimo broj. */
22         scanf("%f", &y);
23
24         /* Poredimo uneti broj sa datim brojem.
25          * Ukoliko je unet bas trazeni broj,
26          * uvecavamo brojac.
27          */
28         if(x == y)
29             br_pojavljivanja++;
30     } while(y); /* Sve dok nije uneta 0 */
```

```

32     return br_pojavljivanja;
33 }
34
35 int main() {
36     float x;
37     int br_pojavljivanja;
38
39     printf("Unesite broj x: ");
40     scanf("%f", &x);
41
42     br_pojavljivanja = prebrojavanje(x);
43     printf("Broj pojavljivanja broja %.2f je: %d\n", x,
44           br_pojavljivanja);
45
46     return 0;
47 }

```

Rešenje 2.138

```

1  #include <stdio.h>
2
3  /* Funkcija racuna n-ti clan Fibonacijevog niza.
4   * Clanovi ovog niza zadaju se rekursivno tj. u zavisnosti od
5   * prethodnih clanova.
6   * Fibonacijevi brojevi od 0. do 47. se mogu smestiti u tip int, a
7   * kako n moze uzimati vrednosti
8   * od 1 do 50, povratni tip funkcije je long int.
9   */
10 long int fibonaci(int n) {
11
12     int i;
13
14     /* f0 i f1 su prva dva clana niza */
15     int f0 = 1;
16     int f1 = 1;
17     /* promenljiva u kojoj se cuvaju opsti clanovi: n+2, n+1. i n-ti
18     clan */
19     long int fn2, fn1, fn;
20
21     /* ukoliko treba vratiti nulti ili prvi clan,
22     * njih ne treba racunati
23     * jer su vec dati.
24     */
25     if(n == 0 || n == 1)
26         return 1;
27
28     /* postavljamo prethodne clanove niza */
29     fn = f0;
30     fn1 = f1;

```

2 Kontrola toka

```
29  /* racunamo od drugog clana, pa dok ne dodjemo do n-tog */
    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 2.139

```
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
9      /* Ukoliko je uneto malo slovo ... */
        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             *
            
```



```
19      * Karakter iz prvog dela abecede, koji je preskocen, je 'a'.
      * Broj preskocenih karaktera iz prvog dela abecede
      * racunamo tako sto izracunamo c - 'a' (rastojanje od datog
      * karaktera do malog slova a)
21      * sto je u ovom slucaju 'b' - 'a' = 1.
      *
23      * Ostatak karaktera do k ispisujemo, ali gledavsi unazad od z.
      * Zato racunamo k - (c - 'a') - 1.
25      *
      * Od k oduzimamo rastojanje izmedju c i 'a',
27      * kako bismo dobili preostali broj karaktera koji treba
      * preskociti.
      */
29      return 'z' - (k - (c - 'a') - 1);
      else
31      /* U suprotnom, karakter ne ispada iz opsega malih slova, te je
      * dovoljno bas njega i vratiti */
      return c-k;
33
/* Ukoliko je uneto veliko slovo ... */
35 else if(c >= 'A' && c <= 'Z')
      if(c-k < 'A')
37          return 'Z' - (k - (c - 'A') - 1);
      else
39          return c-k;
41
      return c;
43 }

45 int main() {
47     int k;
      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 $a = (a_1, \dots, a_n)$ i $b = (b_1, \dots, b_n)$ je suma $a_1 \cdot b_1 + \dots + a_n \cdot b_n$. Napisati program koji računa skalarni proizvod dva vektora. Svaki vektor je zadat kao celobrojni niz sa najviše 100 elemenata. Program treba da učitava dimenziju nizova (oba niza su iste dimenzije), zatim jedan po jedan element niza i da ispiše 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 veći 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]

3 Predstavljanje podataka

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 velikih 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: $a_0, b_0, a_1, b_1, \dots, a_{n-1}, b_{n-1}$. 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 sadrži 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.

- Napisati funkciju `void ucitaj(int niz[], int dimenzija)` koja učitava sadržaj niza.
- Napisati funkciju `void stampaj(int niz[], int dimenzija)` koja štampa sadržaj niza.
- Napisati funkciju koja računa sumu elemenata niza.
- Napisati funkciju koja računa prosečnu vrednost elemenata niza.
- Napisati funkciju koja izračunava minimum elemenata niza.
- 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.

- Napisati funkciju koja proverava da li niz sadrži neku vrednost m .
- Napisati funkciju koja vraća vrednost prve pozicije na kojoj se nalazi element koji ima vrednost m , ili -1 ukoliko element nije u nizu.
- Napisati funkciju koja vraća vrednost poslednje pozicije na kojoj se nalazi element koji ima vrednost m , ili -1 ukoliko element nije u nizu.
- Napisati funkciju koja proverava da li elementi niza čine palindrom.
- Napisati funkciju koja proverava da li su elementi niza uređeni neopadajuće.
- 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 .

3 Predstavljanje podataka

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

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 3.18]

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]

3 Predstavljanje podataka

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]

3 Predstavljanje podataka

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:

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

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

- izbacuje poslednji element niza;

3 Predstavljanje podataka

- (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 dužine 10 elemenata.

[Rešenje 3.37]

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

[Rešenje 3.37]

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

Program testirati pozivom funkcije iz main programa i ispisom rezultata na standardni izlaz, pri čemu korisnik sa standardnog ulaza unosi broj `n`, a zatim niz od

3 Predstavljanje podataka

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 između 0 i $N - 1$. Funkcija izračunava elemente niza `brojac` tako da je i -ti element `brojac[i]` jednak broju pojavljivanja broja i u nizu `a`.
- (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

```

2  /*
   Napisati program koji racuna skalarni proizvod dva vektora. Svaki
   vektor
   je zadat kao celobrojni niz sa najvise 100 elemenata. Program treba
   da
4  ucita dimenziju nizova (oba niza su iste dimenzije), zatim jedan po
   jedan element niza i da ispise njihov skalarni proizvod na
   standardni
6  izlaz.
   */
8
   #include <stdio.h>
10 #define MAX 100
12
14 /*
   Pretprocesorskom direktivom define uvode se simbolicka imena (u ovom
   slucaju
   MAX) kojima se pridruzuje nekakav tekst (u ovom slucaju 100). Pre
   kompilacije,
16 sva pojavljivanja simbolickog imena MAX bice zamenjena pridruzenim
   tekstom
   100. MAX nije promenljiva i za nju se tokom izvršavanja programa ne
   izdvaja
18 memorijski prostor.

20 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
22 sto bi bilo neophodno uraditi je da izmenimo tekst sa 100 na 200. Sa
   druge

```

3 Predstavljanje podataka

```
strane, da nismo koristili pretprocesorsku direktivu i da smo svaki
    put
24 umesto MAX direktno navodili vrednost 100, morali bismo da je
    izmenimo na svakom
    mestu u kodu.
26
27 /*
28 int main()
29 {
30     int a[MAX];
31     int b[MAX];
32     int n;
33     int i;
34     int s;
35
36     printf("Unesi dimenziju niza:");
37     scanf("%d", &n);
38
39     if (n<1 || n>100)
40     {
41         printf("Neispravan unos\n");
42         return -1;
43     }
44
45     /*
46         prvi element niza ima indeks 0, a poslednji n-1,
47         gde je n broj elemenata niza; elementima niza pristupamo
48         preko indeksa; na primer, ako niz a ima 5 elemenata, mozemo
49         im pristupiti pomocu
50         a[0], a[1], a[2], a[3], a[4]
51     */
52
53     /*
54     for (i=0; i<n; i++)
55     {
56         printf("a[%d]=",i);
57         scanf("%d", &a[i]);
58     }
59
60     for (i=0; i<n; i++)
61     {
62         printf("b[%d]=",i);
63         scanf("%d", &b[i]);
64     }
65
66     s=0;
67
68     for (i=0; i<n; i++)
69         s = s + a[i]*b[i];
70
71     printf("Skalarni proizvod: %d\n",s);
72
```

```
    return 0;
74 }
```

Rešenje 3.2

```
/*
2   Napisati program koji ucitava broj elemenata niza (n<=100),
   zatim ucitava elemente niza i ispisuje:
4   a) elemente niza koji se nalaze na parnim indeksima
   b) parne elemente niza
6
7   */
8
9   #include <stdio.h>
10  #define MAX 100
11
12  int main()
13  {
14      int a[MAX];
15      int n;
16      int i;
17
18      printf("Unesi dimenziju niza:");
19      scanf("%d", &n);
20
21      if (n<1 || n>MAX)
22      {
23          printf("Nekorektan unos\n");
24          return -1;
25      }
26
27
28
29
30      for (i=0; i<n; i++)
31      {
32          printf("a[%d]=",i);
33          scanf("%d", &a[i]); /* ucitavamo jedan po jedan element niza */
34      }
35
36      printf("Elementi sa parnim indeksima:\n");
37      for (i=0; i<n; i+=2)
38          printf("a[%d]=%d\n",i,a[i]);
39
40      printf("Parni elementi:\n");
41      for (i=0; i<n; i++)
42          if (a[i]%2==0)
43              printf("a[%d]=%d\n",i,a[i]);
44
45      return 0;
46  }
```

```
}
```

Rešenje 3.3

```
1  /*
   Napisati program koji učitava jedan ceo broj a zatim ispisuje
   koliko puta koja cifra ucestvuje
3  u zapisu tog broja. Nije potrebno ispisivati da se neka cifra
   pojavila 0 puta.

5  Na primer, za uneti broj 4611, izlaz treba da bude:

7  U zapisu broja 4611, cifra 1 se pojaviljuje 2 puta
   U zapisu broja 4611, cifra 4 se pojaviljuje 1 puta
9  U zapisu broja 4611, cifra 6 se pojaviljuje 1 puta

11 A za uneti broj -252

13 U zapisu broja -252, cifra 2 se pojaviljuje 2 puta
   U zapisu broja -252, cifra 5 se pojaviljuje 1 puta

15 */
17 #include<stdio.h>
19 #include<stdlib.h>
21 #define MAX 100
23 int main()
24 {
25     int x;
26     int brojac[10];
27     char cifra;
28     int original;
29     int i;

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

33     /*
34        svaki element niza brojac predstavlja
35        brojac za jednu cifru:
36        brojac[0] sadrzi broj nula
37        brojac[1] sadrzi broj jedinica
38        ...
39        brojac[9] sadrzi broj devetki

41        brojac se inicijalizuju na vrednost 0
42     */
43
44     for(i=0;i<10;i++)
45         brojac[i]=0;
```

```

47  /*
48     vrednost promenljive x ce biti unistena
49     u while petlji jer je u svakom koraku delimo
50     sa 10; njenu vrednost cuvamo u promenljivoj
51     original kako bismo mogli da je iskoristimo
52     na kraju prilikom ispisa
53  */
54  original = x;
55
56  /*
57     Uzimamo apsolutnu vrednost broja za slucaj
58     da je uneti broj negativan
59  */
60  x=abs(x);
61
62  /* Izdvajanje cifara broja */
63  do
64  {
65      cifra = x%10;
66      brojaci[cifra]++; /* Uvecavamo brojac odgovarajuce cifre */
67      x/=10;
68  } while(x);
69
70  /* Ispis brojaca koji su razliciti od nule */
71  for(i=0;i<10;i++)
72      if(brojaci[i])
73          printf("U zapisu broja %d, cifra %d se pojaviljuje %d puta\n",
74                original, i, brojaci[i]);
75
76  return 0;
77  }

```

Rešenje 3.4

```

1  /* Napisati program koji ucitava karakter po karakter do EOF i
2     ispisuje koliko se puta
3     u unetom tekstu pojavila svaka cifra, svako malo slovo i svako
4     veliko slovo. Ispisati
5     broj pojavljivanja samo za ona mala slova, velika slova i cifre
6     koji su se u unetom
7     tekstu pojavili >0 puta.
8  */
9
10 #include <stdio.h>
11
12 int main()
13 {
14     /* Za svaku dekadnu cifru definisemo jedan brojac (tj. imamo niz

```

3 Predstavljanje podataka

```
13     od 10 brojaca): brojaci[0] broji koliko se puta pojavio karakter
15     '0', brojaci[1] broji koliko se puta pojavio karakter '1' i tako
        dalje. Svi brojaci se inicijalizuju nulama.
16
17     */
18     int cifre[10];
19     int mala[26];
20     int velika[26];
21
22     int c, i;
23
24     for(i=0; i<10; i++)
25         cifre[i]=0;
26
27     for(i=0; i<26; i++)
28     {
29         mala[i]=0;
30         velika[i]=0;
31     }
32
33     while((c = getchar()) != EOF)
34     {
35         if (c>='A' && c<='Z')
36             velika[c-'A']++;
37         else if (c>='a' && c<='z')
38             mala[c-'a']++;
39         else if (c >='0' && c <= '9') /* Ako je karakter c dekadna cifra
        ... */
40             cifre[c-'0']++;          /* Uvecavamo odgovarajuci brojac za
        1 */
41
42         /*
43         Izraz c - '0' ce u slucaju da je c dekadna cifra imati
        upravo
44         vrednost 0, 1, ..., 9 za karaktere '0', '1', ..., '9' respektivno,
45         a to su upravo indeksi u nizu brojaci (jer niz ima 10 elemenata,
46         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.
        */
48     }
49
50     /* Prikazujemo elemente niza, tj. vrednosti brojaca: */
51     for(i = 0; i < 10; i++)
52     {
53         if (cifre[i]!=0)
54             printf("Karakter %c se pojavljuje %d puta\n", '0' + i,
55                 cifre[i]);
56
57         for(i = 0; i < 26; i++)
58         {
59             if (mala[i]!=0)
60                 printf("Karakter %c se pojavljuje %d puta\n", 'a' + i,
61                     mala[i]);
```

```

    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     }
}

```

3 Predstavljanje podataka

```
38 printf("\nUnesi elemente niza b:\n");
39 for(i=0;i<n;i++)
40 {
41     printf("b[%d]=",i);
42     scanf("%d", &b[i]);
43 }
44
45 /*
46  Koristimo dva indeksa:
47  1. i, sa kojim pristupamo
48     elementima niza a i b, i koji uvecavamo za 1
49     nakon svake iteracije,
50  2. j, sa kojim pristupamo
51     elementima niza c; s obzirom da u svakoj
52     iteraciji dodeljujemo vrednost za dva
53     elementa niza c (c[j] i c[j+1]), indeks
54     j uvecavamo za 2 nakon svake iteracije
55 */
56 for(i=0,j=0;i<n;i++,j+=2)
57 {
58     c[j]=a[i];
59     c[j+1]=b[i];
60 }
61
62 printf("\nNiz c:\n");
63 for(i=0;i<2*n;i++)
64     printf("c[%d]=%d\n",i,c[i]);
65
66 return 0;
67 }
```

Rešenje 3.6

```
1  /*
2     Napisati program koji ucitava dimenziju n celobrojnog niza a i
3     njegove elemente, a zatim iz niza a izbacuje sve elemente
4     koji nisu deljivi svojom poslednjom cifrom, izuzev elemenata
5     cija je poslednja cifra 0 koji treba zadržati. Program treba da
6     ispise
7     izmenjeni niz na standardni izlaz. Mozemo pretpostaviti da niz a
8     sadrzi najvise 100 elemenata.
9  */
10
11 #include <stdio.h>
12 #define MAX 100
13
14 int main()
15 {
16     int a[MAX];
17 }
```



```
19  int n;
20  int i,j;
21  char poslednja_cifra;
22  int novo_n;

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

25
26  if (n<1 || n>MAX)
27  {
28      printf("Neispravan unos\n");
29      return -1;
30  }

31
32  printf("\nUnesi elemente niza a:\n");
33  for(i=0;i<n;i++)
34  {
35      printf("a[%d]=",i);
36      scanf("%d", &a[i]);
37  }

38
39
40
41  /*
42   Dodadni indeks j se uvecava u slucaju da element na indeksu
43   i treba da ostane u nizu, tj da je deljiv svojim
44   indeksom i; u suprotnom, j se nece uvecati i
45   element i ce u narednoj iteraciji biti zamenjen elementom koji
46   jeste deljiv svojim indeksom
47  */

48
49  for(i=0,j=0;i<n;i++)
50  {
51      poslednja_cifra = a[i]%10;

52
53      /*
54       zbog lenjog izracunavanja, ako je prvi uslov
55       u disjunkciji tacan, drugi se nece ispitivati
56       (jer ce tada disjunkcija biti tacna bez obzira
57       da li je drugi uslov tacan ili ne)
58      */
59      if (poslednja_cifra==0 || a[i]%poslednja_cifra==0)
60      {
61          a[j]=a[i];
62          j++;
63      }
64  }

65  /*
66   Izbacivanjem elemenata dimenzija niza se menja, odnosno
67   smanjuje se za broj izbacenih elemenata
68  */
69  novo_n=j;
```

3 Predstavljanje podataka

```
71     printf("Nakon izmena:\n");
72     for(i=0;i<novo_n;i++)
73         printf("a[%d]=%d\n",i,a[i]);
74
75     return 0;
76 }
```

Rešenje 3.7

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

```
        s+=a[i];
42     return s;
43 }
44
45 /* d) */
46 float prosek(int a[], int n)
47 {
48     int i;
49     int s = suma(a,n);
50     return (float) s/n;
51 }
52
53 /* e) */
54 int minimum (int a[],int n)
55 {
56     int m;
57     int i;
58     m = a[0];
59
60     /*
61      * minimum inicijalizujemo na prvi element niza (a[0])
62      * u svakom koraku poredimo vrednost minimuma
63      * sa jednim elementom niza, iduci redom; s obzirom
64      * da je minimum inicijalizovan na a[0], nema potrebe
65      * porediti a[0] sa a[0] i zbog toga indeksiranje kreće
66      * od 1
67      */
68
69     for(i=1;i<n;i++)
70         if (m>a[i])
71             m = a[i];
72
73     return m;
74 }
75
76 /* f) */
77 int max_pozicija (int a[],int n)
78 {
79     int m;
80     int m_poz;
81     int i;
82     m = a[0];
83     m_poz=0;
84
85     for(i=1;i<n;i++)
86         if (m<a[i])
87         {
88             m = a[i];
89             m_poz=i;
90         }
91     return m_poz;
92 }
```

3 Predstavljanje podataka

```

    m_poz=i;
94     }

96     return m_poz;
}

98

100
102 int main()
{
    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.
```

```

10      g) Napisati funkciju koja proverava da li su elementi niza
      uredjeni
      neopadajuće.
12      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
14      treba da vrati 3.
      i) Napisati program koji testira prethodne funkcije.
16  */
17  #include <stdio.h>
18  #define MAX 100

20  /* a) */
21  void ucitaj(int a[], int n)
22  {
23      int i;
24      for(i=0;i<n;i++)
25      {
26          printf("Unesi element na poziciji %d:",i);
27          scanf("%d",&a[i]);
28      }
29  }

30  /* b) */
31  void stampaj(int a[], int n)
32  {
33      int i;
34      for(i=0;i<n;i++)
35          printf("%d\t",a[i]);
36      printf("\n");
37  }

38  /* c) */
39  int sadrzi(int a[], int n, int m)
40  {
41      int i;
42      /*
43       * poredimo jedan po jedan element niza a sa datim m; ukoliko
44       * ustanovimo jednakost, to znaci da niz sadrzi element jednak
45       * m i vracamo 1
46       */
47      for(i=0;i<n;i++)
48          if (a[i]==m)
49              return 1;
50  }

51  /*
52   * ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
53   * ispunjen,
54   * to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
55   * vrati 0
56   */

```

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

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

92     return -1;
}

94
/* f) */
96 int palindrom(int a[], int n)
{
98     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
    */
}
```

```
108     */
109     for(i=0,j=n-1;i<j;i++,j--)
110         if(a[i]!=a[j])
111             return 0;
112     return 1;
113 }
114
115 /* g) */
116 int neopadajuci(int a[], int n)
117 {
118     int i;
119
120     /*
121     Funkcija neopadajuci proverava da li je dati niz sortiran
122     neopadajuce i vraca
123     1 ako jeste, a 0 u suprotnom
124
125     Sortiranost proveravamo na sledeci nacin: za svaki par susednih
126     elemenata
127     a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
128     proveravamo
129     da li vazii da je drugi clan para manji od prvog. Ako naidjemo na
130     par za koji
131     to ne vazii, niz nije sortiran i funkcija vraca 0. Ukoliko se
132     petlja zavrsii
133     a da pritom uslov a[i]<a[i-1] nije nijednom bio ispunjen, to znaci
134     da je
135     niz sortiran i funkcija vraca 1
136
137     */
138     for(i=1; i<n; i++)
139         if (a[i]<a[i-1])
140             return 0;
141     return 1;
142 }
143
144 /* h) */
145 int najduza_konstanta(int a[], int n)
146 {
147     int i; /* indeks niza */
148     int j; /* duzina intervala */
149     int duzina;
150     int max_duzina=0;
151
152     for(i=0,j=0;i<n-1;i++)
153     {
154         if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
155         {
156             j++;          /* uvecavamo duzinu konstantnog intervala */
157         }
158     }
159     return j;
160 }
```

```
154      /*
      ako se niz završava konstantnim intervalom (nalazimo se u
poslednjoj
156      iteraciji petlje i tada je i==n-2), ispitujemo da li je
taj konstantni
      interval maksimalne dužine
158      */
      if(i==n-2)
160      {
          j++;
162          if(j>max_duzina)
              max_duzina=j;
164      }
      else
166      {
168          /*
              izasli smo iz konstantnog intervala
170
              ukoliko smo imali bar dva elementa u konstantnom
intervalu,
172          vrednost promenljive j ce biti 1, a dužina 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          dužina tog intervala je 0
          */
178
          if (j>0)
180              j++;

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

192      }

194  }

196  return max_duzina;
198 }
200
```



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

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

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

    učitaj(a,n);
218     printf("Učitani niz:");
    stampaj(a,n);

220     printf("Unesi jedan ceo broj:");
222     scanf("%d",&m);

224     if(sadrzi(a,n,m))
226         printf("Niz sadrzi element cija je vrednost %d\n", m);
    else
228         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

    i = prvo_pojavljivanje(a,n,m);
230     if(i!=-1)
232         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
            prvog pojavljivanja u nizu je %d\n", m,i);
    else
234         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

236     i = poslednje_pojavljivanje(a,n,m);
238     if(i!=-1)
        printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
            poslednjeg pojavljivanja u nizu je %d\n", m,i);
240     else
        printf("Niz ne sadrzi element cija je vrednost %d\n", m);

242     if(palindrom(a,n))
244         printf("Elementi niza cine palindrom\n");
    else
246         printf("Elementi niza ne cine palindrom\n");

248     if(neopadajuci(a,n))
        printf("Niz je sortiran neopadajuće\n");
250     else
```

3 Predstavljanje podataka

```
252     printf("Niz nije sortiran neopadajuće\n");  
254     printf("Duzina najduzeg konstantnog intervala: %d\n",  
           najduza_konstanta(a,n));  
256     return 0;  
}
```

Rešenje 3.9

```
1  /*  
   a) Napisati funkciju koja učitava sadržaj niza.  
3  b) Napisati funkciju koja stampa sadržaj niza.  
   c) Napisati funkciju koja proverava da li niz sadrži neku vrednost  
     m.  
5  d) Napisati funkciju koja vraća vrednost prve pozicije na kojoj se  
     nalazi element koji ima vrednost m, ili -1 ukoliko element nije u  
     nizu.  
7  e) Napisati funkciju koja vraća vrednost poslednje pozicije na  
     kojoj se  
     nalazi element koji ima vrednost m, ili -1 ukoliko element nije u  
     nizu.  
9  f) Napisati funkciju koja proverava da li elementi niza čine  
     palindrom.  
   g) Napisati funkciju koja proverava da li su elementi niza  
     uređeni  
11 neopadajuće.  
   h) Napisati funkciju koja izračunava najdužu uzastopnu seriju  
     jednakih  
13 elemenata u nizu. Na primer, za uneti niz 1 2 3 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)
```

```
33 {
34     int i;
35     for(i=0;i<n;i++)
36         printf("%d\t",a[i]);
37     printf("\n");
38 }
39
40
41 /* c) */
42 int sadrzi(int a[], int n, int m)
43 {
44     int i;
45     /*
46      * poredimo jedan po jedan element niza a sa datim m; ukoliko
47      * ustanovimo jednakost, to znaci da niz sadrzi element jednak
48      * m i vracamo 1
49      */
50     for(i=0;i<n;i++)
51         if (a[i]==m)
52             return 1;
53
54     /*
55      * ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
56      * ispunjen,
57      * to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
58      * vrati 0
59      */
60     return 0;
61 }
62
63 /* d) */
64 int prvo_pojavljivanje(int a[], int n, int m)
65 {
66     int i;
67     /*
68      * poredimo jedan po jedan element niza a sa datim m; ukoliko
69      * ustanovimo jednakost, vracamo indeks elementa niza a koji
70      * je jednak sa m
71      */
72     for(i=0;i<n;i++)
73         if (a[i]==m)
74             return i;
75
76     /*
77      * ukoliko se petlja završi a uslov a[i]==m nijednom nije bio
78      * ispunjen,
79      * to znaci da se broj m ne nalazi u nizu a i da funkcija treba da
80      * vrati -1
81      */
82     return -1;
83 }
```

3 Predstavljanje podataka

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

95  /* f) */
    int palindrom(int a[], int n)
97  {
99      int i,j;

101     /*
            uporedjujemo element na poziciji 0 sa elementom na poziciji n-1
103     uporedjujemo element na poziciji 1 sa elementom na poziciji n-2
            .
105     .
            i tako redom dok je prva pozicija manja od druge
107     */
        for(i=0, j=n-1; i<j; i++, j--)
109            if (a[i]!=a[j])
                return 0;
111
        return 1;
113  }

115  /* g) */
    int neopadajuci(int a[], int n)
117  {
        int i;
119
        /*
121     Funkcija neopadajuci proverava da li je dati niz sortiran
            neopadajuće i vraća
            1 ako jeste, a 0 u suprotnom
123
            Sortiranost proveravamo na sledeci nacin: za svaki par susednih
            elemenata
125     a[0] i a[1], a[1] i a[2], a[2] i a[3], ..., a[n-2] i a[n-1]
            proveravamo
            da li vazi da je drugi clan para manji od prvog. Ako naidjemo na
            par za koji
127     to ne vazi, niz nije sortiran i funkcija vraća 0. Ukoliko se
            petlja završi
```

```

129     a da pritom uslov  $a[i] < a[i-1]$  nije nijednom bio ispunjen, to znaci
        da je
niz sortiran i funkcija vraca 1

131     */
133     for(i=1; i<n; i++)
        if (a[i]<a[i-1])
            return 0;
135
        return 1;
137 }

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

147     for(i=0,j=0;i<n-1;i++)
148     {
149         if(a[i]==a[i+1]) /* nalazimo se unutar konstantnog intervala */
150         {
151             j++;          /* uvecavamo duzinu konstantnog intervala */
153
154             /*
155              ako se niz zavrшава konstantnim intervalom (nalazimo se u
poslednjoj
156              iteraciji petlje i tada je i==n-2), ispitujemo da li je
taj konstantni
157              interval maksimalne duzine
*/
158             if(i==n-2)
159             {
160                 j++;
161                 if(j>max_duzina)
162                     max_duzina=j;
163             }
164         }
165         else
166         {
167             /*
168              izasli smo iz konstantnog intervala
169
170              ukoliko smo imali bar dva elementa u konstantnom
intervalu,
171              vrednost promenljive j ce biti 1, a duzina tog intervala
je 2;
172              zbog toga je neophodno takve (pozitivne) j uvecati za 1;
173

```

3 Predstavljanje podataka

```
175         sa druge strane, ako su a[i] i a[i+1] razliciti,
176         duzina tog intervala je 0
177     */
178
179     if (j>0)
180         j++;
181
182     /* azuriramo maksimalnu duzinu uspona */
183     if(j>max_duzina)
184         max_duzina=j;
185     /*
186         duzina uspona se postavlja na nulu
187         kako bi mogli da je iskoristimo
188         za naredni uspon
189     */
190     j=0;
191
192 }
193
194 }
195
196 return max_duzina;
197 }
198
199
200
201 int main()
202 {
203     int a[MAX];
204     int n;
205     int m;
206     int i;
207
208     printf("Unesi dimenziju niza:");
209     scanf("%d",&n);
210
211     if (n<1 || n>MAX)
212     {
213         printf("Nekorektan unos\n");
214         return -1;
215     }
216
217     ucitaj(a,n);
218     printf("Ucitani niz:");
219     stampaj(a,n);
220
221     printf("Unesi jedan ceo broj:");
222     scanf("%d",&m);
223
224
225     if(sadrzi(a,n,m))
226         printf("Niz sadrzi element cija je vrednost %d\n", m);
```

```

227     else
229         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

231     i = prvo_pojavljivanje(a,n,m);
231     if(i!=-1)
233         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
233         prvog pojavljivanja u nizu je %d\n", m,i);
235     else
235         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

237     i = poslednje_pojavljivanje(a,n,m);
237     if(i!=-1)
239         printf("Niz sadrzi element cija je vrednost %d. Indeks njegovog
239         poslednjeg pojavljivanja u nizu je %d\n", m,i);
241     else
241         printf("Niz ne sadrzi element cija je vrednost %d\n", m);

243     if(palindrom(a,n))
243         printf("Elementi niza cine palindrom\n");
245     else
245         printf("Elementi niza ne cine palindrom\n");

247     if(neopadajuci(a,n))
247         printf("Niz je sortiran neopadajuće\n");
249     else
249         printf("Niz nije sortiran neopadajuće\n");

251     printf("Duzina najduzeg konstantnog intervala: %d\n",
251     najduza_konstanta(a,n));

253     return 0;
255 }
257

```

Rešenje 3.10

```

1  /*
3  a) Napisati funkciju koja sve vrednosti niza uvecava za vrednost m.
3  b) Napisati funkciju koja obrce vrednosti elementima niza.
3  c) Napisati funkciju koja rotira niz ciklicno za jedno mesto u levo
3  .
5  d) Napisati funkciju koja rotira niz ciklicno za k mesta u levo.
5  e) Napisati program koji testira prethodne funkcije.
7
7  Napisati potom glavni program koji testira ovu funkciju.
9  */
11 #include<stdio.h>
11 #define MAX 100
13

```

```
void ucitaj(int a[], int n)
15 {
    int i;
17     for(i=0;i<n;i++)
    {
19         printf("Unesi element na poziciji %d:",i);
        scanf("%d",&a[i]);
21     }
}

23 void stampaj(int a[], int n)
25 {
    int i;
27     for(i=0;i<n;i++)
        printf("%d\t",a[i]);
29     printf("\n");
}

31

33 void uvecaj(int a[], int n, int m)
35 {
    int i;
    for(i=0;i<n;i++)
37         a[i]+=m;
}

39

41 void obrni(int a[], int n)
43 {
    int t;
45     int i,j;
    /*
47     Niz obrcemo tako sto razmenimo vrednosti elemenata na pozicijama
        0 i n-1,
        zatim 1 i n-2, 2 i n-3 i tako redom dok je prva pozicija manja od
        druge
49     */

51     for(i=0,j=n-1;i<j;i++, j--)
    {
53         t = a[i];
        a[i] = a[j];
55         a[j] = t;
    }

57 }

59 void rotiraj1(int a[], int n)
61 {
    int i;
63     int tmp;
```



```
65     tmp=a[0]; /* izdvajamo prvi element */
66     for(i=0;i<n-1;i++)
67         a[i]=a[i+1]; /* pomeramo preostale elemente */
68     a[n-1] = tmp; /* poslednjem elementu dodeljujemo
69                     sacuvanu vrednost prvog elementa */
70 }
71 void rotirajk(int a[], int n, int k)
72 {
73     int i;
74     /*
75      * k puta rotiramo niz za jednu poziciju
76      * ulevo
77      */
78     for(i=0;i<k;i++)
79         rotiraj1(a,n);
80 }
81
82 int main()
83 {
84     int a[MAX];
85     int n;
86     int i;
87     int k;
88     int m;
89
90     printf("Unesi dimenziju niza:");
91     scanf("%d",&n);
92
93     if (n<1 || n>MAX)
94     {
95         printf("Nekorektan unos\n");
96         return -1;
97     }
98
99     ucitaj(a,n);
100
101     printf("Unesi jedan ceo broj:");
102     scanf("%d", &m);
103
104     uvecaj(a,n,m);
105     printf("Elementi niza nakon uvecanja za %d:\n",m);
106     stampaj(a,n);
107
108     obrni(a,n);
109     printf("Elementi niza nakon obrtanja:\n");
110     stampaj(a,n);
111
112     printf("Unesi jedan pozitivan ceo broj:");
113     scanf("%d",&k);
114
115     if (k<=0)
```

3 Predstavljanje podataka

```
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)  
                brojevi[i] *= brojevi[i];  
  
31        /* Ispisujemo sve elemente niza. */  
  
33        for(i=0;i<n;i++)
```

```
    printf("%g ", brojevi[i]);
35 }
    return 0;
37 }
```

Rešenje 3.115

```
1  #include <stdio.h>
3  #define MAX 100
5  int main()
6  {
7      int brojevi[MAX];
8      int n, i, k, indikator;
9
10     printf("Unesite dimenziju niza: ");
11     scanf("%d", &n);
12
13     if(n<1 || n>100)
14     {
15         printf("Greska: pogresan unos!\n");
16     }
17     else{
18         printf("Unesite elemente niza: ");
19
20         for(i=0;i<n;i++)
21             scanf("%d", &brojevi[i]);
22
23         printf("Unesite broj k: ");
24         scanf("%d", &k);
25
26         if(k == 0)
27         {
28             printf("Greska: pogresan unos!\n");
29         }
30         else{
31             /*
32              Promenljiva koja nam cuva informaciju o tome
33              da li je u nizu postojao element koji je deljiv brojem k.
34              Inicijalno je postavimo na nulu.
35             */
36
37             indikator = 0;
38
39             /*
40              Ukoliko je element niza deljiv brojem k, postavljamo indikator na
41              1
42              i ispisujemo indeks tog elementa.
43             */
44         }
```

3 Predstavljanje podataka

```
45     for(i=0;i<n;i++)
        if(brojevi[i]%k == 0)
        {
47             indikator = 1;
            printf("%d ",i);
49         }

51     /*
        Ukoliko je indikator jednak nuli to znaci da ne postoji element u
        nizu koji je deljiv brojem k.
53     */

55     if(indikator == 0)
        printf("U nizu nema elemenata koji su deljivi brojem %d!\n",k);
57     }
    }
59     return 0;
}
```

Rešenje 3.116

```
1  #include <stdio.h>

3  #define MAX 100

5  int main()
{
7     int brojevi[MAX];
    int n, i, poz_max, poz_min, max, min, tmp;

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

13    if(n<1 || n>100)
    {
15        printf("Greska: pogresan unos!\n");
        return 0;
17    }

19    printf("Unesite elemente niza:\n");
    for(i=0;i<n;i++)
21        scanf("%d", &brojevi[i]);

23    /*
        Maksimum tj. minimum pre ulaska u petlju postavimo da budu prvi
        element niza.
25        Pozicije maksimuma tj. minimuma postavimo na 0.
    */
27    max = brojevi[0];
    min = brojevi[0];
29    poz_max = 0;
```

```

31     poz_min = 0;
32
33     /*
34      Pronadjemo maksimalni tj. minimalni element tako sto u petlji
35      prodjemo kroz sve elemente i ukoliko naletimo na element veci od
36      maksimuma
37      tj. manji od minimuma, promenimo tako da sada maksimum tj.
38      minimum budu taj element
39      i promenimo njihove pozicije.
40     */
41     for(i=1;i<n;i++)
42     {
43         if(brojevi[i] > max)
44         {
45             max = brojevi[i];
46             poz_max = i;
47         }
48
49         if(brojevi[i] < min)
50         {
51             min = brojevi[i];
52             poz_min = i;
53         }
54     }
55
56     /*
57      Zamenimo minimalni i maksimalni element na pozicijama poz_min i
58      poz_max.
59      Koristimo pomocnu promenljivu tmp kako bismo sacuvali vrednost
60      maksimalnog elementa.
61     */
62     tmp = max;
63     brojevi[poz_max] = min;
64     brojevi[poz_min] = tmp;
65
66     for(i=0;i<n;i++)
67         printf("%d ", brojevi[i]);
68
69     return 0;
70 }

```

Rešenje 3.117

```

1  #include <stdio.h>
2
3  #define MAX 100
4
5  int main()
6  {
7      char karakteri[MAX];
8      char c;
9
10

```

3 Predstavljanje podataka

```
9  int i, n;

11

13  for(i=0; i<MAX; i++)
14  {
15      /*
16       Ucitavamo karakter po karakter dok ne unesemo * ili ne
17       prekoracimo 100 karaktera
18       i upisujemo ih u niz.
19      */
20      printf("Unesite karakter: ");
21      scanf("%c", &c);

22      /*
23       Citamo belinu nakon unesenog karaktera.
24      */
25      getchar();

26      /*
27       Ukoliko smo uneli * izlazimo iz petlje
28      */
29      if(c == '*')
30          break;

31      /*
32       Stavljamo karakter u niz.
33      */
34      karakteri[i] = c;
35  }

36

37  /*
38   Broj unetih karaktera je nakon prolaska kroz petlju i-1.
39  */
40  n = i-1;

41

42  /*
43   Ispisujemo karaktere u obrnutom redosledu.
44  */
45  for(i=n; i>=0; i--)
46  {
47      printf("%c ", karakteri[i]);
48  }

49

51  return 0;
}
```

Rešenje 3.118

```
1  #include <stdio.h>

3  int main()
```

```
{
5  char c;
   int cifrex[10], cifrey[10];
7  int x, y, i, indikator;

9  printf("Unesite dva broja: ");
   scanf("%d%d", &x, &y);

11

   /*
13    Uzmemo apsolutnu vrednost brojeva za slucaj da su negativni.
   */
15  x=abs(x);
   y=abs(y);

17

   /*
19    Niz cifrex nam predstavlja brojeve za cifre broja x, na pocetku
       ga inicijalizujemo na 0.
       Analogno za cifrey.
   */
21  /*
   for(i=0;i<10;i++)
23  {
       cifrex[i] = 0;
25     cifrey[i] = 0;
   }

27

   /*
29    Skidamo jednu po jednu cifru broja x i povecavamo njen broj u
       nizu cifrex.
   */
31  while(x)
   {
33     c = x%10;
       cifrex[c]++;
35     x /= 10;
   }

37

   /*
39    Isto radimo i za broj y.
   */
41  while(y)
   {
43     c = y%10;
       cifrey[c]++;
45     y /= 10;
   }

47

   /*
49    Promenljiva koja nam služi za proveru da li su oba broja
       sastavljena od istih cifara.
       Pretpostavice da jesu i postaviti indikator na 1.
       Nakon toga u petlji prolazimo kroz nizove cifrex i cifrey u
51    kojima se nalaze
```

3 Predstavljanje podataka

```
    brojevi pojavljivanja svih cifri 0-9 u broju x i y, i prvi put
    kada naletimo na
53    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)
55    i izlazimo iz petlje.
    */
57    indikator = 1;
    for(i=0;i<10;i++)
59        if(cifrey[i] != cifrex[i])
        {
61            indikator = 0;
            break;
63        }

65    /*
    Ako je promenljiva indikator ostala 1, to znaci da u petlji nismo
    pronasli cifru
67    koja se ne pojavljuje isti broj puta u brojevima x i y, sto znaci
    da se oni zapisuju istim ciframa.
    */
69    if(indikator)
        printf("Brojevi se zapisuju istim ciframa!\n");
71    else
        printf("Brojevi se ne zapisuju istim ciframa!\n");
73
75    return 0;
}
```

Rešenje 3.119

```
1  #include <stdio.h>

3  #define MAX 100

5  int main()
{
7      int a[MAX], b[MAX], c[2*MAX];
      int i, n;

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

13     if(n<1 || n>100)
    {
15         printf("Greska: pogresan unos!\n");
        return 0;
17     }

19     printf("Unesite elemente niza a: ");
```



```

21     for(i=0;i<n;i++)
        scanf("%d", &a[i]);

23     printf("Unesite elemente niza b: ");
    for(i=0;i<n;i++)
25         scanf("%d", &b[i]);

27     /*
        Niz c ima 2*n elemenata. Prvih n elemenata niza b, i nakon toga n
        elemenata niza a.
29     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],
31     a na poziciju n+i element niza a - a[i].
    */
33     for(i=0;i<n;i++)
    {
35         c[i] = b[i];
        c[n+i] = a[i];
37     }

39     for(i=0;i<2*n;i++)
        printf("%d ", c[i]);

41     return 0;
43 }

```

Rešenje 3.17

```

1  #include <stdio.h>

3  #define MAX 100

5  /*
6   Funkcija koja vraca broj pojavljivanja broja x u nizu.
7  */

9  int broj_pojavljivanja(int niz[], int n, int x)
10 {
11     int i, rezultat = 0;

13     /*
        Kada naidjemo na element niza koji je jednak broju x, povecamo
        brojac rezultat.
15     */
    for(i=0;i<n;i++)
17         if(niz[i] == x)
            rezultat++;

19     return rezultat;

```

```
21 }
23 int main()
24 {
25     int a[MAX], b[MAX];
26     int i, j, n, n_b;
27
28     printf("Unesite broj n: ");
29     scanf("%d", &n);
30
31     if(n<1 || n>100)
32     {
33         printf("Greska: pogresan unos!\n");
34         return -1;
35     }
36
37     printf("Unesite elemente niza a: ");
38     for(i=0; i<n; i++)
39         scanf("%d", &a[i]);
40
41     /*
42      Brojac elemenata rezultujucega niza b.
43     */
44     j = 0;
45     for(i=0; i<n; i++)
46     {
47         /*
48          Ukoliko se element niza pojavljuje tacno tri puta i ne postoji
49          u nizu b koji trenutno ima j elemenata
50          (nismo ga jos uvek dodali) dodajemo ga u niz b i povecavamo
51          brojac j.
52         */
53         if(broj_pojavljivanja(a, n, a[i])==3 && broj_pojavljivanja(b, j,
54             a[i])==0)
55         {
56             b[j] = a[i];
57             j++;
58         }
59     }
60     /*
61      Broj elemenata u nizu b je j.
62     */
63     n_b = j;
64
65     for(i=0; i<n_b; i++)
66         printf("%d ", b[i]);
67
68     return 0;
69 }
```

Rešenje 3.18

```
1  #include <stdio.h>
2
3  #define MAX 100
4
5  /*
6   Funkcija koja vraca 1 ukoliko broj x postoji u nizu, 0 inace.
7   */
8
9  int postoji(int niz[], int n, int x)
10 {
11     int i;
12
13     for(i=0;i<n;i++)
14         if(niz[i] == x)
15             return 1;
16
17     return 0;
18 }
19
20 int main()
21 {
22     int a[MAX], b[MAX], unija[2*MAX], presek[MAX], razlika[MAX];
23     int i, j, n_a, n_b, n_u, n_p, n_r, indikator;
24
25     printf("Unesite broj elemenata niza a: ");
26     scanf("%d", &n_a);
27
28     if(n_a<1 || n_a>100)
29     {
30         printf("Greska: pogresan unos!\n");
31         return -1;
32     }
33
34     printf("Unesite elemente niza a: ");
35     for(i=0;i<n_a;i++)
36         scanf("%d", &a[i]);
37
38     printf("Unesite broj elemenata niza b: ");
39     scanf("%d", &n_b);
40
41     if(n_b<1 || n_b>100)
42     {
43         printf("Greska: pogresan unos!\n");
44         return -1;
45     }
46
47     printf("Unesite elemente niza b: ");
48     for(i=0;i<n_b;i++)
49         scanf("%d", &b[i]);
50
51     /*
```

```
52     Brojaci elemenata u nizovima unija, presek i razlika.
53     */
54     n_u = 0;
55     n_p = 0;
56     n_r = 0;
57
58     for(i=0;i<n_a;i++)
59     {
60         /*
61          *   Ukoliko se element a[i] ne nalazi u uniji, dodajemo ga u uniju
62          *   i povecamo brojac elemenata u nizu unija.
63          */
64         if(postoji(unija,n_u,a[i]) == 0)
65         {
66             unija[n_u] = a[i];
67             n_u++;
68         }
69
70         /*
71          *   Ukoliko se element a[i] postoji u nizu b i ne postoji u nizu
72          *   presek, dodajemo ga u presek i povecavamo brojac elemenata u nizu
73          *   presek.
74          */
75         if(postoji(b, n_b, a[i])==1 && postoji(presek, n_p, a[i])==0)
76         {
77             presek[n_p] = a[i];
78             n_p++;
79         }
80
81         /*
82          *   Ukoliko element a[i] ne postoji u nizu b i ne postoji u nizu
83          *   razlika, dodajemo ga u razliku i povecavamo brojac elemenata u
84          *   nizu razlika.
85          */
86         if(postoji(b, n_b, a[i])==0 && postoji(razlika, n_r, a[i])==0)
87         {
88             razlika[n_r] = a[i];
89             n_r++;
90         }
91     }
92
93     /*
94     *   Elemente niza b koji ne postoje u uniji dodajemo u uniju.
95     */
96     for(i=0;i<n_b;i++)
97     {
98         if(postoji(unija, n_u, b[i]))
99         {
100             unija[n_u] = b[i];
101             n_u++;
102         }
103     }
104
105     printf("Unija: ");
```

```

100     for(i=0;i<n_u;i++)
        printf("%d ", uniija[i]);

102     printf("\nPresek: ");
    for(i=0;i<n_p;i++)
104         printf("%d ", presek[i]);

106     printf("\nRazlika: ");
    for(i=0;i<n_r;i++)
108         printf("%d ", razlika[i]);

110     return 0;
}

```

Rešenje 3.19

```

#include <stdio.h>
2
#define MAX 100
4
int main()
6 {
    int a[MAX], b[MAX];
8     int i, j, n_a, n_b;

10     printf("Unesite broj elemenata niza: ");
    scanf("%d", &n_a);

12     if(n_a<1 || n_a>100)
14     {
        printf("Greska: pogresan unos!\n");
16         return -1;
    }

18     printf("Unesite elemente niza: ");
20     for(i=0;i<n_a;i++)
        scanf("%d", &a[i]);

22
    /*
24     1. nacin

26     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.
28     Ukoliko naletimo na element koji je neparan, njega samo preskocimo
        .
    */
30
    for(i=0, j=0;i<n_a;i++)
32     {

```

```

    if(a[i]%2 == 0)
34     {
        a[j] = a[i];
36         j++;
    }
38 }

40 /*
    Na pozicijama od 0...j-1 se sada nalaze elementi koji su parni,
    te je nova dimenzija niza sada j.
42 */
    n_a=j;

44     for(i=0;i<n_a;i++)
46         printf("%d ", a[i]);

48     /*
        2. nacin
50
        Kada naletimo na element niza koji je paran, kopiramo ga u niz b
        i povecamo j - brojac elemenata niza b.
52
        for(i=0, j=0;i<n_a;i++)
54         if(a[i]%2 == 0)
        {
            b[j] = a[i];
56             j++;
58         }

60     n_b = j;

62     for(i=0;i<n_b;i++)
        printf("%d ", b[i]);

64
66     */
    return 0;
}
```

Rešenje 3.20

```

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

4  #define MAX 100

6  /*
    Funkcija koja proverava da li je broj prost.
8  Vraca 1 ukoliko broj jeste prost, inace 0.
    */
10 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         }
    }
}
```

3 Predstavljanje podataka

```
62     n_a=j;
64
66     for(i=0;i<n_a;i++)
        printf("%d ", a[i]);
68
69     /*
70      2. nacin
71
72      Prolazimo kroz niz a i svaki broj koji nije prost kopiramo u niz
73      b i povecamo j - brojac elemenata u nizu b.
74
75      for(i=0, j=0;i<n_a;i++)
76          if(prost(a[i]) == 0)
77          {
78              b[j] = a[i];
79              j++;
80          }
81
82      n_b = j;
83
84      for(i=0;i<n_b;i++)
85          printf("%d ", b[i]);
86      */
87      return 0;
88  }
```

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 elementa niza. Napisati i program koji
   testira rad funkcije. Pretpostaviti da duzina
4  niza nece biti veca od 100.
6
7  */
8  #include <stdio.h>
9  #define MAX 100
10
11  /*
12  * Funkcija prebrojavanje vraca broj elemenata niza koji su manji od
   poslednjeg
13  * NAPOMENA: Poslednji element niza se nalazi na poziciji n-1
14  */
15  int prebrojavanje(int a[], int n)
16  {
17      int i;
18      /*Inicijalizujemo brojac na 0*/
19      int br=0;
```



```

20
21  /*
22   * Petljom prolazimo kroz sve clanove niza,
23   * poredimo ih sa poslednjim elementom i
24   * ukoliko su manji, uvecavamo brojac
25   */
26  for(i=0; i<n-1; i++){
27      if(a[i]<a[n-1]){
28          br++;
29      }
30  }
31
32  /*Vracamo izracunatu vrednost*/
33  return br;
34 }
35
36 int main()
37 {
38     int a[MAX];
39     int n;
40     int i;
41
42     printf("Unesite broj elemenata niza:");
43     scanf("%d", &n);
44
45     /*Provera korektnosti ulaznih podataka*/
46     if(n<=0 || n>100)
47     {
48         printf("Greska: pogresan unos!\n");
49         return 0;
50     }
51
52     /*Ucitavanje niza*/
53     printf("Unesite elemente niza:");
54     for(i=0; i<n; i++)
55         scanf("%d", &a[i]);
56
57     /*Ispis rezultata*/
58     printf("%d\n", prebrojavanje(a,n));
59
60     return 0;
61 }

```

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.

```

3 Predstavljanje podataka

```
6  */
8  #include <stdio.h>
8  #define MAX 100

10 /*Funkcija koja vraca broj parnih elemenata niza koji se nalaze
    ispred najveceg elementa u nizu
    * Ideja je da prvo pronadjemo najveći element niza i njegovu
    poziciju, a zatim da jos jednom
12  * prodjemo kroz niz sa ciljem da nadjemo sve parne brojeve koje
    prethode maksimalnom.
    */
14 int prebrojavanje(int a[], int n)
{
16     int i;
16     int max;
18     int max_ind;
18     int br = 0;

20     /*Na pocetku postavljamo da je maksimalni element a[0] i da je
        odgovarajuca pozicija 0*/
22     max = a[0];
22     max_ind=0;

24     /*Pronalazimo maksimum niza i pamtimo i vrednost i poziciju*/
26     for(i=1;i<n-1;i++)
        if(a[i]>max)
28         {
            max = a[i];
30             max_ind = i;
        }

32     /* Krecemo od pocetka niza i idemo do pozicije na kojoj se nalazi
        najveći element
34     * i pronalazimo sve parne brojeve
        */
36     for(i=0;i<max_ind;i++)
        if(a[i]%2==0)
38         br++;

40     return br;
}

42
44 int main()
{
46     int a[MAX];
46     int n;
46     int i;

48     printf("Unesite broj elemenata niza:");
50     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++)
        if(isdigit(a[i]))
21         br++;

23     return br;
25 }

27 int main()

```

3 Predstavljanje podataka

```
{
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 učitamo 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*/
    printf("Unesite elemente niza:");
39     for(i=0; i<n; i++)
        scanf("%d", &a[i]);

41     /*Ucitavamo interval [i,j]*/
    printf("Unesite vrednosti za i i j: ");
43     scanf("%d%d", &i, &j);

45     /*Proveravamo korektnost zadatog intervala */
    if(i > n-1 || j > n-1 || i > j){
47     printf("Greska: nekorektne vrednosti granica!\n");
49     return 0;
    }

51     /*Ispisujemo rezultat*/
    printf("Zbir je: %d", zbir(a,n,i,j));

53     return 0;
55 }
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.

```

3 Predstavljanje podataka

```

    Napisati i program koji testira rad funkcije. Pretpostaviti da
    duzina niza nece biti veca od 100.
4  */

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

8

10 /*Funkcija racuna zbir prvih k pozitivnih clanova niza a*/
   float zbir_pozitivnih(float a[], int n, int k){
       int i;

12

14   /*Na pocetku inicijalizujemo sumu na 0*/
       float s=0;

16   /*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++){
18       if(a[i] >= 0){
           /*Kada naidjemo na pozitivan element, uvecavamo sumu i smanjujemo
           k*/
20           s+=a[i];
           k--;
22       }
       }

24

26   /*Na kraju vracamo izracunatu sumu */
       return s;
   }

28

29 int main(){
30     int n, i, k;
31     float a[MAX];

32

33     printf("Unesite broj elemenata niza: ");
34     scanf("%d", &n);

36     /*Proveravamo korektnost ulaza*/
37     if(n<=0 || n> MAX){
38         printf("Greska: pogresan unos!\n");
39         return 0;
40     }

42     /*Ucitavamo elemente niza*/
43     printf("Unesite elemente niza: ");
44     for(i=0; i<n; i++){
45         scanf("%f", &a[i]);
46     }

47     /*Ucitavamo k*/
48     printf("Unesite vrednost za k: ");
49     scanf("%d", &k);

50     /*Proveravamo korektnost za k*/

```

```

52     if(k<0){
53         printf("Greska: pogresan unos!");
54         return 0;
55     }
56
57     /*Ispisujemo rezultat*/
58     printf("Zbir je: %.2f\n", zbir_pozitivnih(a,n,k));
59     return 0;
60 }

```

Rešenje 3.26

```

1  /*
2     Napisati funkciju void kvadriranje(float a[], int n) koja kvadrira
3     elemente realnog niza a duzine n koji se nalaze na parnim
4     pozicijama.
5
6     Napisati i program koji testira rad funkcije. Pretpostaviti da
7     duzina niza nece biti veca od 100.
8
9     */
10
11 #include<stdio.h>
12 #define MAX 100
13
14 /*Funkcija kvadirraj menja niz a tako sto kvadrira sve elemente na
15 parnim pozicijama. */
16 void kvadriraj(float a[], int n){
17     int i;
18
19     /*Petljom prolazimo kroz niz i ukoliko je pozicija parna, a[i]
20     postaje a[i]*a[i]*/
21     for(i=0; i<n; i++){
22         if(i%2 ==0)
23             a[i]*=a[i]; //skraceno od a[i] = a[i]*a[i]
24     }
25 }
26
27 int main(){
28
29     int n, i, j;
30     float a[MAX];
31
32     printf("Unesite broj elemenata niza: ");
33     scanf("%d", &n);
34
35     /*Proveravamo korektnost ulaza*/
36     if(n <=0 || n>100)
37     {
38         printf("Greska: pogresan unos!\n");
39         return 0;
40     }

```

3 Predstavljanje podataka

```
35 }  
  
37 /*Ucitavamo elemente niza*/  
printf("Unesite elemente niza:");  
39 for(i=0; i<n; i++)  
scanf("%f", &a[i]);  
  
41  
43 /*Pozivamo funkciju koja kvadrira odgovarajuce elemente*/  
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 koristiti funkcija *atoi*.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 3.44]

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 efg
|| 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 zamenjuje velikim. Broj izvršenih modifikacija se čuva u okviru argumenta *br_modifikacija*. Pretpostaviti da niska *s* neće biti duža od 20 karaktera. Napisati i program koji testira rad napisane funkcije.

3 Predstavljanje podataka

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

Primer 1

```

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

```

Primer 2

```

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

```

Primer 3

```

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

```

[Rešenje 3.52]

Zadatak 3.53 Tekst

[Rešenje 3.56]

3 Predstavljanje podataka

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 3.56]

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

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 3.56]

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

svojstva ispisati poruku o grešci.

[Rešenje 3.56]

3.4 Rešenja

Rešenje 3.56

```
2  /*
4  Napisati funkciju uredi koja uredjuje svoja dva
6  celobrojna argumenta tako da se u prvom nalazi manji
8  a u drugom veci. Napisati potom glavni program koji
10 učitava dva cela broja i uredjuje njihove vrednosti
12 primenom napisane funkcije. Na primer, ako su učitane
14 promenljive x=5 i y=2, njihove vrednosti nakon
16 primene funkcije uredi treba da budu x=2 i y=5.
18 */
20 #include <stdio.h>
22
24 /*
26 Argumenti funkcije uredi_pogresno, promenljive a i b,
28 predstavljaju lokalne promenljive za ovu funkciju
30 i prestaju da postoje po završetku funkcije. Zbog toga
32 se efekti razmene vrednosti promenljivih a i b u slučaju
34 da je a>b vide u funkciji, ali se ne vide u glavnom programu.
36 */
38 void uredi_pogresno(int a, int b)
39 {
40     int t;
41
42     if (a>b)
43     {
44         t = a;
45         a = b;
46         b = t;
47     }
48     printf("uredi_pogresno :: a=%d, b=%d\n", a, b);
49     printf("uredi_pogresno :: &a=%p, &b=%p\n", &a, &b);
50 }
51
52 /*
54 Argumenti funkcije uredi_tacno, promenljive pa i pb,
56 takodje su lokalne promenljive za ovu funkciju i
58 prestaju da postoje kada se funkcija završi.
60 Njima prosledjujemo adrese promenljivih a i b koje zelimo
62 da razmenimo u slučaju da je a>b.
64 */
```

3 Predstavljanje podataka

```
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
46     i vrednosti bilo koje dve celobrojne promenljive.

48     */
49 void uredi_tacno(int * pa, int * pb)
50 {
51     int t;
52     if (*pa>*pb)
53     {
54         t = *pa;
55         *pa = *pb;
56         *pb = t;
57     }
58     printf("uredi_tacno :: *pa=%d, *pb=%d\n ", *pa, *pb);
59     printf("uredi_tacno :: pa=%p, pb=%p\n ", pa, pb);
60 }
61 int main()
62 {
63     int a,b;
64
65     printf("Unesi dve celobrojne promenljive:");
66     scanf("%d%d",&a,&b);
67
68     printf("main :: a=%d, b=%d\n", a,b);
69     printf("main :: &a=%p, &b=%p\n", &a, &b);
70     uredi_pogresno(a,b);
71     printf("main :: nakon uredi_pogresno, a=%d, b=%d\n", a, b);
72
73     /*
74     Funkcija uredi_tacno kao argument ima dve pokazivacke
75     promenljive
76     (int*,int*). Zbog toga joj je u pozivu funkcije neophodno
77     proslediti
78     adrese promenljivih koje zelimo da uredimo rastuce, &a i &b.
79     */
80     uredi_tacno(&a, &b);
81     printf("main :: nakon uredi_tacno, a=%d, b=%d\n", a, b);
82
83     return 0;
84 }
```

Rešenje 3.39

```
1 /*
   Napisati funkciju koja za boju datu u rgb formatu
```



```
3   racuna cmy format po formulama:
4   C = 1 - ( R / 255 )
5   M = 1 - ( G / 255 )
6   Y = 1 - ( B / 255 )
7
8   Napisati program koji ucitava boju u rgb formatu,
9   primenjuje odgovarajucu funkciju i ispisuje boju u cmy formatu.
10
11  */
12
13  #include <stdio.h>
14  #include <math.h>
15
16  void rgb_to_cmy(float* a, float* b, float* c)
17  {
18      /* Zagrade su neophodne jer aritmeticke operacije
19       imaju veci prioritet od operatora dereferenciranja (*).
20      */
21      *a=1-(*a)/255;
22      *b=1-(*b)/255;
23      *c=1-(*c)/255;
24
25      /*
26      Pomocu return ne mozemo vratiti vise od jedne vrednosti.
27
28      Ceste greske:
29      return a,b,c;          return vraca samo jednu vrednost
30      return a; return b; return c; return ce vratiti samo a
31
32      Zato je neophodno da promenljive ciju vrednost
33      zelimo da promenimo prenesemo preko pokazivaca.
34      */
35  }
36
37  int rgb_korektno(float a)
38  {
39      if(a<0 || a>255)
40          return 0;
41      return 1;
42  }
43
44
45  int main()
46  {
47      float a,b,c;
48
49      /*
50      Argumenti funkcije rgb_to_cmy su
51      pokazivaci na float. Njima prosledjujemo
52      adrese promenljivih a, b i c.
53      */
```

3 Predstavljanje podataka

```
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 presečne tacke (ukoliko prave nisu paralelne)
16   3. drugu koordinatu presečne tacke (ukoliko prave nisu paralelne)

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

20   Koordinate presečne 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)
33         return 0;
34
35     *px = -(n1-n2)/(k1-k2);
36     *py = k1*(*px)+n1;
37     return 1;
38 }
39
40 int main()
41 {
42     float k1,k2,n1,n2;
43     float x,y;
44
45     printf("Unesi k i n za prvu pravu:");
46     scanf("%f%f",&k1,&n1);
47
48     printf("Unesi k i n za drugu pravu:");
49     scanf("%f%f",&k2,&n2);
50
51     if(presek(k1,n1,k2,n2,&x,&y))
52         printf("Prave se seku u tacki (%.2f,%.2f)\n", x,y);
53     else
54         printf("Prave su paralelne\n");
55
56     return 0;
57 }

```

Rešenje 3.41

```

1  /*
2     Napisati program koji ispisuje broj navedenih argumenata komandne
3     linije,
4     a zatim i same argumenate i njihove redne brojeve.
5  */
6
7  #include <stdio.h>
8
9  /*
10     Argumenti komandne linije cuvaju se u nizu niski pod nazivom
11     argv. Svaki element tog niza odgovara jednom argumentu komandne
12     linije pri cemu prvi element predstavlja naziv programa koji
13     pokrecemo. Celobrojna promenljiva argc predstavlja ukupan
14     broj argumenata komandne linije ukljucujuci i argument koji
15     odgovara nazivu programa.
16  */
17
18 int main(int argc, char *argv[])
19 {
20     int i;
21
22     printf("Broj argumenata je: %d\n",argc);

```

3 Predstavljanje podataka

```
23     for(i=0; i<argc; i++)
        printf("%d: %s\n",i,argv[i]);
25
        return 0;
27 }
```

Rešenje 3.42

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

```

39     strspn(s1,s2)
40
41  */
42
43  int strspn_klon(char str[], char accept[])
44  {
45      int br=0;
46      int i;
47
48      for(i=0; str[i];i++)
49          if(strchr(accept, str[i])!=NULL)
50              br++;
51      else /* ako pronadjemo karakter u stringu str koji nije */
52          break; /* u stringu accept, prekidamo petlju */
53
54      return br;
55  }
56
57  int main(int argc, char* argv[])
58  {
59
60      int br;
61
62      if(argc<3)
63      {
64          printf("Nekorektan poziv\nProgram treba pozvati sa ./a.out arg1
65              arg2\n");
66          return -1;
67      }
68
69      br = strspn_klon(argv[1],argv[2]);
70      printf("Broj karaktera stringa %s koji se nalaze u stringu %s,
71          pocev od pocetka stringa %s: %d\n", argv[1],argv[2],argv[1],br);
72      return 0;
73  }

```

Rešenje 3.43

```

2  /*
3      Napisati funkciju void sifruj(char s[], char c, int k) koja
4      sifruje
5      string s na sledeci nacin: svako malo i veliko slovo stringa s
6      konvertuje u
7      slovo koje je u abecedi od njega udaljeno k pozicija, i to
8      k pozicija ulevo, ako je karakter c jednak karakteru 'L' ili
9      udesno
10     ako je karakter c jednak karakteru 'D'. Sifrovanje treba da bude
11     kruzno. Ako string
12     s sadrzi karakter koji nije alfanumericki, ostaviti ga
13     nesifriranog.
14
15     */

```

3 Predstavljanje podataka

```
10      Napisati potom glavni program koji testira napisanu funkciju za
      string i prirodan
12      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
14      navedena, podrazumevani pravac je udesno.

      Mozemo podrazumevati da string sadrzi najvise 30 karaktera.

16      Primeri upotrebe:

18      1:
      ./a.out abcd 2
20      cdef

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

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

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

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

38      6:
      ./a.out ab12cd 2 -p D
40      cd12ef

42      */

44      #include <stdio.h>
      #include <string.h>
46      #include <stdlib.h>
      #define MAX 31

48      void sifruj(char s[], char c, int k)
50      {
      int i;
52      int znak;
      char t;

54      /*
56      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 služi 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
78                (26),
                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        {

```

3 Predstavljanje podataka

```
108     printf("Nekorektan unos: broj argumenata moze biti 3 ili 5\n");
109     return -1;
110 }
111
112 /*
113  Argumenti komandne linije su stringovi. Ako program pokrecemo
114  na sledeci nacin:
115  ./a.out abcd 2 -p D
116  to znaci da je argument koji odgovara dvojci u stvari
117  string "2". Da bismo string konvertovali u ceo broj,
118  koristimo ugradjenu funkciju atoi iz biblioteke stdlib.h.
119 */
120
121 k = atoi(argv[2]);
122
123 /*
124  Ispitujemo korektnost datih podataka:
125 */
126 if (k<=0)
127 {
128     printf("Nekorektan unos: broj pozicija mora biti pozitivan ceo
129     broj\n");
130     return -1;
131 }
132
133 /* Korektnost unosa je ispitana, sto znaci da
134 argc moze biti 3 ili 5 */
135
136 if (argc==3) /* Ako je argc 3: */
137     pravac='D';
138 else /* Ako argc nije 3, tada je sigurno 5, jer je */
139 { /* korektnost unosa ispitana, a unos je korektan
140     jedino za argc==3 ili argc==5 */
141     /*
142     Ispitujemo korektnost pretposlednjeg argumenta koji mora da
143     bude u formatu "-p".
144     Ovaj argument je string argv[3]. Njegovom prvom karakteru (
145     koji treba
146     da bude '-' ) pristupamo sa argv[3][0] a drugom sa argv
147     [3][1].
148     */
149     if (argv[3][0] != '-')
150     {
151         printf("Nekorektan unos: pri zadavanju opcija prvi karakter
152         mora biti '-' \n");
153         return -1;
154     }
155
156     if (argv[3][1]!='p')
157     {
158         printf("Nekorektan unos: nedozvoljena opcija\n");
159         return -1;
160     }
161 }
```



```

154     }
155     /*
156     Nakon argumenta -p sledi argument koji zadaje vrednost ove
opcije. To je
157     poslednji argument kome pristupamo sa argv[4]. Ovaj argument
158     treba
159     da sadrzi samo jedan karakter - 'L' ili 'D' i njemu
pristupamo sa
160     argv[4][0].
161     */
162     if(argv[4][0]=='L' || argv[4][0]=='D')
163         pravac=argv[4][0];
164     else
165     {
166         printf("Nekorektan unos: pravac moze biti L ili D\n");
167         return -1;
168     }
169 }
170 strcpy(rec, argv[1]);
171 sifruj(rec, pravac, k);
172
173 printf("Sifrovana rec: %s\n", rec);
174
175 return 0;
176 }

```

Rešenje 3.44

```

1  #include <stdio.h>
2
3  int main(int argc, char* argv[]) {
4
5      int i;
6      int s = 0;
7
8      /* char *argv[] <--- niz niski koje predstavljaju argumente
9       navedene iza poziva programa
10       int argc      <--- ukupan broj niski (sa sve nazivom programa)
11       navedenih prilikom pozivanja
12
13       Ukoliko je program pozvan sa ./a.out 12 abc 6 5 3ab
14
15       argv[0] = "./a.out".
16       argv[1] = "12"
17       argv[2] = "abc"
18       argv[3] = "6"
19       argv[4] = "5"
20       argv[5] = "3ab"

```

3 Predstavljanje podataka

```
21     argc iznosi 6

23     Kako je argv[] po prirodi niz,
24     koristimo tzv. brojacku odnosno
25     for petlju
26     i obradjujemo svaki od argumenata.
27     */

29     /* Funkcija atoi() prihvata nisku,
30     i racuna dekadnu vrednost prosledjene niske,
31     dokle god se ona moze racunati.
32     Na primer, ukoliko je niska "-123",
33     atoi() vraca broj -123.
34     Ako je, pak, niska "123abc",
35     atoi() ce vratiti 123
36     (prilikom prve pojave karaktera koji nije cifra, funkcija prekida
37     izracunavanje).

38     To za posledicu ima da, ukoliko je funkciji
39     prosledjeno nesto
40     sto se ne moze pretvoriti u broj,
41     na primer niska "abcd",
42     funkcija atoi() vraca 0.
43     */

45     for(i = 1; i < argc; i++)
46         s += atoi(argv[i]); /* Zbog nacina rada funkcije atoi(), mozemo
47         je pozvati nad svim argumentima
48         komandne linije, i sabrati odgovarajuce dekadne
49         vrednosti.
50         Ukoliko neki argument i nije broj, to ne predstavlja
51         problem
52         jer ce u tom slucaju odgovarajuci sabirak biti 0
53         */

54     printf("Zbir numerickih argumenata: %d\n", s);

55     return 0;
56 }
```

Rešenje 3.45

```
1 #include <stdio.h>

3 int main(int argc, char* argv[]) {

5     int i;

7     /* Prolazimo for petljom kroz niz argumenata,
```

```

9      i trazimo one niske ciji je prvi karakter bas 'z'.
      Ukoliko je trenutni argument koji se ispituje
      argv[i],
11     kako je on sam po sebi niska,
      do prvog karaktera dolazimo kao i pri dosadasnjem
13     radu sa niskama --> argv[i][0]
      ^
15     |
      index prvog karaktera u niski argv[i]
17 */
19     for(i = 1; i < argc; i++)
        if(argv[i][0] == 'z')
21         printf("%s ", argv[i]);
23     putchar('\n');
25     return 0;
}

```

Rešenje 3.46

```

1  #include <stdio.h>
   #include <string.h>
3
   int main(int argc, char* argv[]) {
5
       int i;
       int br = 0;
7
9       /* Da bismo proverili da li se karakter 'z' (tj. 'Z')
       nalazi u niski argv[i],
       to mozemo uciniti koriscenjem funkcije
11      strchr() koja se nalazi u string.h.
13
       Ukoliko je karakter sadržan u okviru niske,
15      strchr() vraća pokazivac na taj karakter
       unutar same niske.
17      Inace, ukoliko se karakter ne nalazi u niski,
       funkcija vraća NULL.
19      */
21
       for(i = 1; i < argc; i++)
           if(strchr(argv[i], 'z') != NULL || strchr(argv[i], 'Z') != NULL)
23               br++;
25
       printf("%d\n", br);
27
       return 0;
   }

```

Rešenje 3.47

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(int argc, char *argv[])
5 {
6     int n,i;
7
8     /*
9      Ispisujemo gresku ukoliko nema dovoljno argumenata komandne
10     linije.
11     */
12     if(argc != 2)
13     {
14         printf("Greska: nedostaje argument komandne linije!\n");
15         return -1;
16     }
17
18     /*
19     Pretvaramo argument komandne linije koji je string u ceo broj
20     koriscenjem funkcije atoi
21     */
22     n = atoi(argv[1]);
23     n = abs(n);
24
25     for(i=(-1)*n;i<=n;i++)
26         printf("%d ",i);
27
28     return 0;
29 }
```

Rešenje 3.48

```
1 #include <stdio.h>
2 #include <string.h>
3
4 int main(int argc, char *argv[])
5 {
6     int indikator = 0;
7     int i,j;
8
9     /*
10     Ukoliko imamo samo jedan argument komandne linije,
11     ispisujemo da nema istih i završavamo program.
12     */
13     if(argc < 2)
14     {
15         printf("Medju argumentima nema istih.\n");
16         return -1;
17     }
18 }
```

```

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 završavamo program.
    Ako smo izašli iz prve petlje to znaci da nismo pronašli 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;
    for(i=0;s[i];i++)
        if(s[i]>='a' && s[i]<='z')
10        {
            t[i] = toupper(s[i]);
            (*br_modifikacija)++;
12        }
        else
14        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;  
14     char c;  
15  
16     while((c=getchar())!=EOF){  
17         if(c=='.')  
18             tacke++;  
19  
20         if(c==',')  
21             zarezi++;  
22     }  
23  
24     *br_tacaka=tacke;  
25     *br_zareza=zarezi;  
26  
27 }  
28  
29 int main(){  
30     int br_tacaka, br_zareza;  
31  
32     printf("Unesite tekst: \n");  
33  
34     interpunkcija(&br_tacaka, &br_zareza);  
35  
36     printf("Broj tacaka: %d\n", br_tacaka);  
37     printf("Broj zareza: %d\n", br_zareza);  
38  
39     return 0;  
40 }
```

Rešenje 3.51

```

/*
2  Napisati funkciju
    void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
        int* nn)
4  koja razbija niz a na niz parnih i niz neparnih brojeva. Pokazivaci
    pn i nn
    redom treba da sadrže broj elemenata niza parnih tj. niza neparnih
        elemenata.
6  Pretpostaviti da dužina niza a neće biti veća od 50. Napisati program
    koji
    testira napisanu funkciju.
8  */

10 #include <stdio.h>
    #define MAX 50

12 void par_nepar(int a[], int n, int parni[], int* pn, int neparni[],
    int* nn){
14
16     int i, j, k;

18     /* i - brojac niza a */
    /* j - brojac niza parnih brojeva */
    /* k - brojac niza neparnih brojeva */

20     for(i=0, j=0, k=0; i<n; i++){
22         /* Ako je element niza paran */
        if(a[i]%2==0){
24             /* Smestamo ga u niz parnih brojeva i uvecavamo indeks niza
                j */
                parni[j]=a[i];
26                 j++;
            }
            else{
28                 /* Inace, smestamo ga u niz neparnih brojeva i uvecavamo
                    indeks niza k */
                    neparni[k]=a[i];
30                     k++;
                }
            }
32     }

34     *pn=j;
    *nn=k;
36
38 }

40 int main(){

```

3 Predstavljanje podataka

```
42  int n, i, j, pn, nn;
    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 */
    printf("Unesite elemente niza: ");
54  for(i=0; i<n; i++){
        scanf("%d", &a[i]);
56  }

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 */
    printf("Niz parnih brojeva: ");
64  for(i=0; i<pn; i++)
        printf("%d ", parni[i]);
66  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 funkciju
    void min_max(float a[], int n, float* min, float* max)
4  koja izracunava minimalni i maksimalni element niza a duzine n.
    Napisati zatim i program koji ucitava niz realnih brojeva
        maksimalne
6  duzine 50 i ispisuje vrednosti minimuma i maksimuma na tri decimale
    .

8  */

10 #include<stdio.h>
```



```
12 #define MAX 50
13
14 void min_max(float a[], int n, float* min, float* max){
15
16     int i;
17
18     /* Inicijalizujemo vrednosti minimuma i maksimuma */
19     *min=a[0];
20     *max=a[0];
21
22     /* Obilazimo preostale elemente niza */
23     for(i=1; i<n; i++){
24
25         /* Ako je tekuca vrednost veca od maksimalne, azuriramo maksimum */
26         if(a[i]>*max){
27             *max=a[i];
28         }
29
30         /* Ako je tekuca vrednost manja od minimalne, azuriramo minimum */
31         if(a[i]<*min){
32             *min=a[i];
33         }
34     }
35
36 int main(){
37     int i, n;
38     float a[MAX], min, max;
39
40     /* Ucitavamo dimenziju niza */
41     printf("Unesite broj elemenata niza: ");
42     scanf("%d", &n);
43
44     if(n<0 || n>MAX){
45         printf("Greska: pogresna dimenzija niza!\n");
46         return 0;
47     }
48
49     /* Ucitavamo elemente niza */
50     printf("Unesite elemente niza:\n");
51     for(i=0; i<n; i++){
52         scanf("%f", &a[i]);
53     }
54
55     /* Pozivamo funkciju za racunanje maksimuma i minimuma */
56     min_max(a, n, &min, &max);
57
58     /* Ispisujemo rezultat */
59     printf("Minimum: %.3f\n", min);
60     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
6  int main()
7  {
8      int a,b,s;
9
10     scanf("%d%d",&a,&b);
11
12     suma(a,b,&s);
13
14     printf("suma: %d\n",s);
15
16     return 0;
17 }
19 void suma(int a, int b, int *s)
20 {
21     *s = a + b;
22 }
```

Rešenje 3.56

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

```
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
6  int main()
7  {
8      int a,b,s;
9
10     scanf("%d%d",&a,&b);
11
12     suma(a,b,&s);
13
14     printf("suma: %d\n",s);
15
16     return 0;
17 }
19 void suma(int a, int b, int *s)
21 {
    *s = a + b;
}
```

Rešenje 3.56

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

```
18 }  
20 void suma(int a, int b, int *s)  
21 {  
22     *s = a + b;  
23 }
```

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 završava samoglasnikom!
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: AaBb+cCdD
|| Niska se ne završava samoglasnikom!
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite nisku: pRograMiranjE
|| Niska se završava samoglasnikom!
```

[Rešenje 3.65]

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.

3 Predstavljanje podataka

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]

3 Predstavljanje podataka

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 između svojih samoglasnika. (Reč je niz karaktera između dve praznine; razmak između samoglasnika reči `dan`as i `jut`ro je 2, a razmak između `sut`rk i `mno`zenje je 5). Testirati pozivom u `main`-u. Maksimalna dužina niske je 20 karaktera.

[Rešenje 3.91]

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

Primer 1

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

[Rešenje 3.91]

3 Predstavljanje podataka

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

Primer 1

```

|| INTERAKCIJA SA PROGRAMOM:
|| novi red*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 ključa određuje koliki je (alfabetski) razmak između njih i označimo taj broj sa `k`. Potom to slovo `reci` zamenjuje `k`-tim slovom alfabeta. Podrazumeva se da je ključ duži od reči.

Primer 1

```

|| INTERAKCIJA SA PROGRAMOM:
|| bac
|| dfge
|| bed

```

[Rešenje 3.91]

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

Primer 1

```

|| INTERAKCIJA SA PROGRAMOM:
|| sveska
|| 2
|| eskasv

```

[Rešenje 3.91]

Zadatak 3.83 Napisati 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 najviše 20 znakova) i, koristeći ove dve funkcije, ispisuje da li su one jednake ako se sva

3 Predstavljanje podataka

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 stringa šifrira tako što se svako slovo zamenjuje sledećim ASCII slovom, a znakovi 'z' i 'Z' zamenjuju redom sa 'a' i 'A'. Uneta reč nije duža od 20 karaktera.

[Rešenje 3.91]

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

[Rešenje 3.91]

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

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

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

[Rešenje 3.91]

Zadatak 3.87 Sa ulaza se unosi reč koja nije duža od 20 znakova. Napisati funkciju koja svako pojavljivanje znaka koji se zadaje kao prvi argument funkcije 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 takođ zadaju sa standardnog ulaza. Na primer, ako se unese reč **ana** i znakovi **a** i **n**, tada funkcija treba da izmeni reč tako da ona postane **aaaa**, ako se unese reč **abrakadabra** i znakovi **a** i **b**, tada funkcija treba reč da izmeni tako da ona postane **aaraakaadkkraa**.

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

[Rešenje 3.91]

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

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| zdRaVo 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

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

[Rešenje 3.91]

Zadatak 3.90

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

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

```
8
9
10 #include <stdio.h>
11 #include <ctype.h>
12
13 /*
14  Kada je niz argument funkcije, dodatni argument je obavezno
15  njegova dimenzija. Kod stringova to nije slucaj jer svaki string
16  ima isti poslednji element - terminirajucu nulu - i to je oznaka
17  kraja stringa.
18 */
19 void konvertuj(char s[])
20 {
21     int i;
22
23     for(i=0; s[i]!='\0'; i++)
24         if (s[i]>='a' && s[i]<='z')
25             s[i] = toupper(s[i]); /* toupper - konvertuje malo slovo u
26             odgovarajuce veliko */
27         else if (s[i]>='A' && s[i]<='Z')
28             s[i] = tolower(s[i]); /* tolower - konvertuje veliko slovo
29             u odgovarajuce malo */
30     /*
31      Funkcije toupper i tolower se nalaze u zaglavlju ctype.h.
32
33      Konverzija malog slova u veliko bez upotrebe funkcije toupper:
34      s[i] = s[i]-'a'+'A';
35      Konverzija velikog slova u malo bez upotrebe funkcije tolower:
36      s[i] = s[i]+'a'-'A';
37
38      */
39 }
40
41 int main()
42 {
43     /*
44      Poslednji karakter svakog stringa je terminirajuca
45      nula '\0', specijalni karakter ciji je ASCII kod 0.
46
47      Ukoliko pretpostavljamo da string sadrzi najvise 30
48      karaktera, neophodno je deklarirati niz od 31 karaktera,
49      pri cemu se dodatni izdvaja za terminirajucu nulu.
50
51      */
52     char s[31];
53     printf("Unesi string:");
54
55     /*
56      Za razliku od nizova koji se ucitavaju i stampaju
57      element po element, stringovi se mogu ucitati i
58      odstampati pomocu jedne scanf/printf naredbe koriscenjem
59      specifikatora %s.
60     */
61 }
```

3 Predstavljanje podataka

```
58     Funkcija scanf ucitava string do prvog pojavljivanja razmaka.
59     */
60     scanf("%s", s);
61
62     konvertuj(s);
63
64     printf("Konvertovani string: %s\n", s);
65
66     return 0;
67
68 }
```

Rešenje 3.58

```
/*
2  Napisati funkciju skрати koja uklanja beline sa
   kraja datog stringa.
4
   Napisati glavni program koji testira napisanu
6  funkciju na stringu "rep belina"
   ".
8  */
10 #include <stdio.h>
11 #include <ctype.h>
12
13 /*
14  Funkcija koja racuna duzinu niza
   ne racunajuci '\0'.
16
17  U biblioteci string.h definisan je veliki
18  broj funkcija za rad sa stringovima,
   ukljucujuci i funkciju strlen koja racunana
20  duzinu stringa.
21
22  Funkcija strlen_klon predstavlja jednu
   implementaciju funkcije strlen.
24
25  U zadacima cemo uvek koristiti ugradjenu
26  funkciju strlen osim ako u tekstu zadatka
   nije naglaseno da se ona ne sme koristiti.
28  Funkcija strlen_klon služi da pokaze na koji
   nacin radi ugradjena funkcija strlen.
30
31  Ugradjena funkcija strlen poziva se na
32  isti nacin kao funkcija strlen_klon:
   strlen(s1)
34
35  */
36 int strlen_klon(char s[])
37 {
```



```

38     int i=0;
39     while(s[i]) /* ASCII kod karaktera '\0' je jednak nuli. */
40         i++;
41
42     return i;
43 }
44
45 void skрати(char s[])
46 {
47     /*
48      Poslednji karakter stringa s(ne racunajuci '\0') ima
49      indeks strlen_klon(s)-1; ideja je da pocnemo od poslednjeg
50      karaktera stringa i da smanjujemo indeks dokle god
51      je karakter na poziciji i blanko znak.
52
53      */
54     int i;
55     for(i=strlen_klon(s)-1;i>=0;i--)
56         if (!isspace (s[i])) /* Ako s[i] nije blanko znak, prekidamo
57                                petlju. */
58             break;
59
60     s[i+1]='\0'; /* D0dajemo terminirajucu nulu iza indeksa i (prvi
61                  neblanko karakter gledano sdesna nalevo).*/
62
63     /*
64      Ugradjena funkcija isspace nalazi se u biblioteci ctype.h i
65      vraca 1 ako
66      je dati karakter blanko znak a 0 u suprotnom.
67
68      Unarni logicki operator ! oznacava negaciju.
69
70      */
71 }
72
73 int main()
74 {
75     /*
76      Ukoliko string ne zelimo da ucitavamo po pokretanju programa
77      vec da ga unapred zadamo, to mozemo uraditi na sledeci nacin:
78
79      */
80     char s[]="rep belina";
81     /* U ovom slucaju nije neophodno navoditi dimenziju stringa vec
82        ce ona biti automatski postavljena na broj karaktera u stringu +
83        1 za
84        terminirajucu nulu. */
85
86     printf("Pre skracivanja: %s\n", s);
87     skрати(s);
88     printf("Posle skracivanja: %s\n", s);

```

```
86     return 0;
88 }
```

Rešenje 3.59

```
/*
2   Napisati program koji učitava string src i formira string dst
   trostrukim nadovezivanjem stringa src. Program treba da ispise
4   string dst. Na primer, za uneti string "dan", string dst treba
   da bude "dandandan". Pretpostaviti da string src nije duzi od
6   30 karaktera.
   */
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

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

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

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

42   */
44 void strcpy_klon(char kopija[], char original[])
   {
```

```

46  int i;
    for(i=0; original[i]; i++)
48      kopija[i]=original[i];

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

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

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

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

68  strcpy_klon(dst,src);

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

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

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

82  return 0;
}

```

Rešenje 3.60

```

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

7  Napisati potom glavni program koji ucitava linije
    do EOF i ispisuje najduzu liniju i njenu duzinu. Ukoliko

```

```
9      ima vise linija maksimalne duzine, ispisati prvu. Mozemo
10     pretpostviti da svaka linija sadrzi najvise 80 karaktera,
11     zajedno sa \n.
12
13 */
14
15 #include<stdio.h>
16 #include<string.h>
17 #define MAX 81
18
19 /*
20     Ukoliko zelimo da učitamo string koji sadrzi beline
21     (npr liniju teksta), ne mozemo koristiti funkciju
22     scanf jer ona učitava string do prvog blanko znaka.
23
24     Zbog toga je neophodno napisati funkciju koja učitava
25     string karakter po karakter.
26
27     Ova funkcija ne dopusta unosenje vise karaktera od
28     unapred odredjene granice (argument n).
29
30     U standardnoj biblioteci stdio.h postoji definisana
31     funkcija char *gets(char *s) koja učitava karaktere
32     dok se ne pojavi novi red ili EOF. Ova funkcija
33     dopusta unosenje vise karaktera nego sto string
34     s sadrzi, sto moze dovesti do neocekivanog ponasanja
35     programa.
36
37     Pored funkcije gets, koja vrsi učitavanje sa standardnog
38     ulaza, u standardnoj biblioteci stdio.h postoji
39     i ugradjena funkcija fgets koja vrsi učitavanje iz
40     datoteke. Nju cemo koristiti za nekoliko casova
41     kada budemo radili datoteke. Prototim funkcije fgets je
42     ovakav:
43
44     char *fgets(char *s, int size, FILE *stream);
45
46     Argumenti funkcije fgets su:
47     s - string u koji vrsimo učitavanje
48     size - maksimalna duzina unetog stringa
49     stream - datoteka iz koje vrsimo učitavanje
50
51     Funkcija fgets, za razliku od funkcije gets, ne dopusta
52     unos vise karaktera od date vrednosti size. Zbog toga
53     je ona sigurnija nego funkcija gets. Funkciju fgets
54     mozemo koristiti i za unos sa standardnog ulaza
55     ukoliko kao treci argument navedemo stdin.
56
57 */
58 int ucitaj_liniju(char s[], int n)
59 {
60     int i=0;
```

```
61     int c;
63     while((c=getchar())!='\n' && i<n-2 && c!=EOF)
64     {
65         s[i] = c;
66         i++;
67     }
69     /* Ucitavamo najvise n-2 karaktera jer na kraju dodajemo jos
70        dva: '\n' i '\0' */
71
72     s[i]='\n';
73     s[i+1]='\0';
75     return i;
77 }
79 int main()
80 {
81     char linija[MAX];
82     char najduza_linija[MAX];
83     int max_duzina=0;
84     int duzina;
85
86     /*
87        Petlja se završava ukoliko je promenljiva duzina
88        jednaka nuli, sto cemo postici zadavanjem linije koja ne sadrzi
89        nijedan karakter osim EOF.
90    */
91
92     while ((duzina=ucitaj_liniju(linija, MAX))>0)
93     {
94         /*
95            Proveravamo da li je uneta linija duza od trenutnog
96            maksimuma i azuriramo promenljive max_duzina i najduza_linija
97        */
98         if (max_duzina<duzina)
99         {
100             max_duzina = duzina;
101             strcpy(najduza_linija,linija);
102         }
103     }
105     printf("Najduza linija: %s duzine: %d\n", najduza_linija,
106           max_duzina);
107     return 0;
108 }
```

Rešenje 3.61

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

i=0;
60 if(znak==-1)
62     i=1;

64 /*
Funkcija isdigit proverava da li je broj
66 cifra. Nalazi se u biblioteci ctype.h

68 Proveravamo da li je karakter u zapisu
broja cifra kako bismo se osigurali
70 od nekorektnog unosa, npr ako korisnik
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
```

```
104     return 0;
}
```

Rešenje 3.62

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

6  #include <stdio.h>
   #include <string.h>
8  #define MAX 10
   /*
10
12   Funkcija transformisi_negativan vraca
   1 ako je broj negativan i 0 u suprotnom, a
   uz to, ako broj jeste negativan, funkcija
14   treba da ga konvertuje u njegovu apsolutnu
   vrednost. S obzirom da funkcija treba da vrati dve
16   vrednosti, to realizujemo na sledeci nacin:
   1. indikator da li je broj negativan
18   ce vratiti kao povratnu vrednost
   2. apsolutnu vrednost broja ce vratiti
20   preko liste argumenata, zbog cega broj
   prenosimo preko pokazivaca
22
23  */
24  int transformisi_negativan(int* pn)
   {
26     if(*pn<0)
       {
28         *pn = -(*pn);
         return 1;
30     }
     return 0;
32  }

34  int formiraj_niz_cifara(int n, char b[], int neg)
   {
36     int i=0;
     char cifra;
38
40     do
       {
42         cifra = n%10;

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



```

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

50
        Greska bi bila navesti b[i]=2
52         jer 2 nije ASCII kod koji odgovara karakteru
        '2'.

54     */
    b[i]=cifra+'0';

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

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

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

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

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

88 }

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

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

```

3 Predstavljanje podataka

```
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()
107 {
108     int n;
109     char broj[MAX];
110     int negativan;

112     /* Ucitavamo broj: */
113     scanf("%d", &n);

114
115     /* Kreiramo broj na osnovu niza cifara: */
116     broj_u_niz_cifara(n, broj);

117
118     /* Ispisujemo rezultat: */
119     printf("Broj zapisan kao string: %s\n", broj);
120
121     return 0;
122 }
```

Rešenje 3.63

```
2      /*
3      Napisati program koji ucitava dva stringa i ispituje najpre da li
4      su jednaki. Ako jesu, program
5      treba da izda odgovarajucu poruku, a ako nisu, treba da ispita da
6      li je drugi podstring
7      prvog. Ukoliko jeste, program treba da ispise pocev od kog indeksa
8      prvog
9      stringa pocinje drugi string. U suprotnom, ispisati odgovarajucu
10     poruku. Mozemo
11     pretpostaviti da stringovi ne sadrze vise od 20 karaktera.
12     */
13
14     #include <stdio.h>
15     #include <string.h>

16     /*
17     Funkcija strcmp(s,t) je ugradjena funkcija koja utvrdjuje da
18     li su strinovi
19     s i t jednaki. Ukoliko jesu, vraca 0, a u suprotnom vraca
20     razliku
21     ASCII kodova prva dva razlicita karaktera na istim pozicijama
22     */
```

```

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 služi da pokaze na koji
24     način radi ugradjena funkcija strcmp.
26     Ugradjena funkcija strcmp poziva se na isti način kao funkcija
    strcmp_klon:
    strcmp(s1,s2)
28     gde poredimo stringove s1 i s2.
30 */
32 int strcmp_klon(char s1[], char s2[])
33 {
34     int i;
35     for(i=0; s1[i]==s2[i];i++)
36         if (s1[i]=='\0')
37             return 0;
38     return s1[i] - s2[i];
39 }
40
42 int main()
43 {
44     char s1[21];
45     char s2[21];
46     char* p;
47
48     printf("Unesi dva stringa:");
49     scanf("%s%s",s1,s2);
50
51     /*
52     Funkcija strstr(s,t) je ugradjena funkcija koja utvrđuje da
    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:
    ako je string deklarisan sa s1[21], tada je njegov naziv s1
60

```

3 Predstavljanje podataka

```

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

74     */

76     p = strstr(s1,s2);

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

84     return 0;
86 }
```

Rešenje 3.64

```

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

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

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

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

```

22  /*
24  Da smo 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 zavrшава samoglasnikom
5  (koristiti funkciju iz tacke a)).
    c) Napisati program koji ucitava nisku maksimalne duzine 20 karaktera
        i ispisuje da li zavrшава samoglasnikom ili ne.
7  */

9  #include <stdio.h>

```

3 Predstavljanje podataka

```
11 #include <ctype.h>
12 #include <string.h>
13 #define MAX_DUZINA 20
14
15 /* Funkcija proverava da li je karakter c samoglasnik */
16 int samoglasnik(char c){
17     char C;
18
19     /* Konvertujemo karakter u veliko slovo kako bismo smanjili broj
20        provera */
21     C=toupper(c);
22
23     /* Samoglasnici su slova a, e, i, o i u */
24     if(C=='A' || C=='E' || C=='I' || C=='O' || C=='U')
25         return 1;
26
27     return 0;
28 }
29
30 /* Funkcija koja proverava da li se niska s zavrшава samoglasnikom */
31 int samoglasnik_na_kraju(char s[]){
32     int duzina;
33
34     /* Odredjujemo duzinu niske */
35     duzina=strlen(s);
36
37     /* Proveravamo da li je niska prazna */
38     if(duzina==0)
39         return 0;
40
41     /* Ako niska nije prazna, proveravamo da li se samoglasnik nalazi
42        na kraju */
43     /* Numeracija karaktera u niski pocinje nulom pa zato proveravamo
44        poziciju duzina -1 */
45     return samoglasnik(s[duzina-1]);
46 }
47
48 int main(){
49     char s[MAX_DUZINA+1];
50
51     /* Ucitavamo nisku */
52     printf("Unesite nisku: ");
53     scanf("%s", s);
54
55     /* Proveravamo da li se zavrшава samoglasnikom i ispisujemo
56        odgovarajucu poruku */
57     if(samoglasnik_na_kraju(s))
58         printf("Niska se zavrшава samoglasnikom!\n");
59     else
60         printf("Niska se ne zavrшава samoglasnikom!\n");
61 }
```

```

59     return 0;
}

```

Rešenje 3.66

```

/*
2  Napisati funkciju void kopiraj_n(char t[], char s[], int n) koja
   kopira najviše n karaktera niske s u nisku t.
   Napisati i program koji učitava nisku maksimalne dužine 20 karaktera
   i jedan ceo broj i testira rad napisane funkcije.
4  */

6  #include <stdio.h>
   #define MAX_DUZINA 20

8
void kopiraj_n(char t[], char s[], int n){
10     int i;

12     /* Brojac i oznacava tekucu poziciju u niski */
   /* Uslov i<n je neophodan zbog kopiranja najviše n karaktera */
14     /* Uslov s[i]!='\0' (ili skraceno samo s[i]) je neophodan kako bi
       bili sigurni da na poziciji i postoji karakter u niski s */
   for(i=0; i<n && s[i]!='\0'; i++){
16         t[i]=s[i];
   }

18     /* Upisujemo terminirajucu nulu u novodobijenu nisku */
20     t[i]='\0';
}

22

24 int main(){
   int n;
26     char s[MAX_DUZINA+1], t[MAX_DUZINA+1];

   /* Ucitavamo nisku */
   printf("Unesite nisku: ");
30     scanf("%s", s);

   /* Ucitavamo broj n i proveravamo korektnost unosa */
   printf("Unesite broj n: ");
34     scanf("%d", &n);
   if(n<0 || n>MAX){
36         printf("Greska: pogresan unos!\n");
       return 0;
   }

38

40     /* Pozivamo funkciju za kopiranje */
   kopiraj_n(t, s, n);
42

```

3 Predstavljanje podataka

```
44  /* Ispisujemo dobijenu nisku */
    printf("Rezultujuca niska: %s\n", t);
46  return 0;
}
```

Rešenje 3.67

```
1  /*
   Napisati funkciju void dupliranje(char t[], char s[]) koja na osnovu
   niske s formira nisku t tako sto duplira svaki
3  karakter niske s. Napisati i program koji ucitava nisku maksimalne
   duzine 20 karaktera i testira rad napisane funkcije.
   */
5
7  #include <stdio.h>
   #define MAX_DUZINA 20
9
   void dupliranje(char t[], char s[]){
       int i, j;
11
       /* Brojac i oznacava tekucu poziciju u niski s */
       /* Brojac j oznacava tekucu poziciju u niski t */
13       for(i=0, j=0; s[i]!='\0'; i++, j+=2){
15           t[j]=s[i];
           t[j+1]=s[i];
17       }
19
       /* Upisujemo terminirajucu nulu u novodobijenu nisku */
       t[j]='\0';
21   }
23
25   int main(){
       int n;
       char s[MAX_DUZINA+1], t[2*MAX_DUZINA+1];
27
       /* Ucitavamo nisku */
29       printf("Unesite nisku: ");
       scanf("%s", s);
31
       /* Pozivamo funkciju za dupliranje */
33       dupliranje(t, s);
35
       /* Ispisujemo dobijenu nisku */
       printf("%s\n", t);
37
       return 0;
39   }
```


Rešenje 3.68

```

1  /*
   Napisati funkciju int heksa_broj(char s[]) koja proverava da li je
   niskom s zadat korektan heksadekadni broj.
3  Heksadekadni broj je korektno zadat ako pocinje prefiksom 0x ili 0X
   i ako sadrzi samo cifre i mala ili velika slova A, B, C, D, E i F
   .
   Funkcija treba da vrati vrednost 1 ako je niska korektan
   heksadekadni broj, odnosno 0 ako nije.
5  Napisati i program koji ucitava nisku maksimalne duzine 7 karaktera
   i ispisuje rezultat rada funkcije.
   */

7  #include<stdio.h>
9  #define MAX 8

11 /*
13  Funkcija koja proverava da li je prosledjeni karakter ispravna
   heksadekadna cifra (broj ili slovo a,b,c,d,e,f)
   Ukoliko jeste, funkcija vraca 1, u suprotnom 0.
15  */
int heksa_cifra(char c){
17     /*Pretvaramo karakter c u veliko slovo*/
    c = toupper(c);

19     /*Proveravamo da li je u pitanju cifra ili slovo A,B,C,D,E,F i
       ukoliko jeste, vracamo 1*/
21     if(isdigit(c) || (c >= 'A' && c <= 'F'))
        return 1;

23     /*Ukoliko nije, vracamo 0*/
25     return 0;
}

27 /*Funkcija koja proverava da li prosledjena niska s predstavlja
   ispravan heksadekadni broj */
29 int heksa_broj(char s[]){
    int i;

31     /*Kako heksadekadni brojevi pocinju sa 0x ili 0X, prvo proveravamo
       da li je taj uslov ispunjen,
33     tj. da li je s[0] jednak 0 i da li je s[1] jednak X i ako taj uslov
       nije ispunjen, onda
       niska s ne predstavlja korektan heksadekadni broj */
35     if(s[0] != '0' || toupper(s[1]) != 'X')
        return 0;

37     /*Prolazimo kroz nisku, pocev od pozicije 2 (jer su prve dve
       pozicije 0x) i za svaki karakter do kraja
39     niske proveravamo da li je ispravna heksadekadna cifra.

```

3 Predstavljanje podataka

```

    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 funkciju i stampamo odgovarajucu poruku*/
59 if(heksa_broj(s))
    printf("Korektan heksadekadni broj!\n");
61 else
    printf("Nekorektan heksadekadni broj!\n");
63
    return 0;
65 }
```

Rešenje 3.69

```

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", hekza_broj(s));

61  return 0;
}

```

Rešenje 3.70

```
/*
2  Napisati funkciju int podniska(char s[], char t[]) koja proverava da
   li je niska t podniska niske s.
   Napisati i program koji ucitava dve niske maksimalne duzine 10
   karaktera i testira rad napisane funkcije.
4
   Napomena: u biblioteci string.h postoji funkcija strstr
6  char* strstr(const char* s, const char* t)
   koja vraca adresu pocetka prvog pojavljivanja niske t u niski s ili
   NULL ako se niska t ne javlja
8  u niski s.
   */
10
12 #include<stdio.h>
13 #define MAX 11
14
15 /*Funkcija koja proverava da li je t podniska od s*/
16 int podniska(char s[], char t[]){
17     int i, j, k;
18
19     /*Spoljna petlja ide redom po niski s*/
20     for(i=0; s[i] != '\0'; i++){
21
22         /*Unutrasnja petlja ide redom po niski t*/
23         /*Promenljiva k pamti vrednost i, služi za poredjenje s[k] i t[j] i
24          *zatim se vrsi pomeranje i po niski s i po niski t (k++, j++) */
25         /*Cim naidjemo na situaciju da se karakteri ne poklapaju, izlazimo
26          iz unutrasnje petlje*/
27         for(j=0, k = i; t[j] != '\0'; j++, k++){
28             if(s[k] != t[j])
29                 break;
30
31         /*Ako smo prosli celu unutrasnju petlju (do kraja niske t), to
32          znaci da su se svi karakteri iz t poklopili
33          sa karakterima iz s i onda vracamo 1*/
34         if(t[j] == '\0')
35             return 1;
36     }
37
38     /*Na kraju vracamo 0*/
39     return 0;
40 }
41
42 int main(){
43     char s[MAX], t[MAX];
44
45     /*Ucitavamo prvu nisku*/
46     printf("Unesite nisku s: ");
47     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 reflektivna(int a[][MAX], int n)
```
- kojom se za relaciju zadatom matricom **a** (matruca je kvadratna) ispitije da li je reflektivna.
- b) Napisati funkciju
- ```
int simetricna(int a[][MAX], int n)
```
- kojom se za relaciju zadatom matricom **a** (matruca je kvadratna) ispitije da li je simetricna.
- c) Napisati funkciju
- ```
int tranzitivna(int a[][MAX], int n)
```
- kojom se za relaciju zadatom matricom **a** (matruca je kvadratna) ispitije da li je tranzitivna.

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

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

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

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

Funkcija postavlja na 1 odgovarajuci indikator.

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

[Rešenje 3.108]

**Zadatak 3.93** Napisati funkciju `float sumD(float a[][max], int n)` koja određuje sumu elemenata iznad glavne dijagonale. Potom napisati funkciju `float sumd(float a[][max], int n)` koja određuje sumu elemenata ispod glavne dijagonale. Funkciju testirati pozivom u main-u. Matrica je maksimalne dimenzije 50x50. Matrica je kvadratna.

[Rešenje 3.108]

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

[Rešenje 3.108]

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

[Rešenje 3.108]

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

[Rešenje 3.108]

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

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

### 3 Predstavljanje podataka

---

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

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
3 4
1 2 3 4
5 6 7 8
9 10 11 12
3 3
54 63
```

#### Primer 2

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

#### Primer 3

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

#### Primer 4

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

[Rešenje 3.108]

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

- (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 standardnog ulaza se unosi broj  $n$  ( $0 < n \leq 10$ ), a potom i elementi kvadratne matrice dimenzije  $n \times n$ . Elementi matrice su celi brojevi. Proveriti da li važi da su zbrovi elemenata kolona matrice 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 standardnog ulaza unosi se broj  $n$  ( $0 < n \leq 200$ ), a potom i elementi kvadratne matrice dimenzije  $n \times n$ . Elementi matrice su celi brojevi. Proveriti da li je uneta matrica ortonormirana i na standardni izlaz ispisati **da** ako jeste ili **ne** ako nije ortonormirana. Matrica je ortonormirana ako je skalarni proizvod svakog para različitih vrsta jednak 0, a skalarni proizvod vrste sa samom sobom 1. U slučaju greške ispisati -1.

### 3 Predstavljanje podataka

---

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

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

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
4
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
da
```

#### Primer 2

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

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
3
2 -2 1
1 2 2
2 1 -2
ne
```

#### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
5
-1 0 2 0 20
0 0 0 10 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
da
```

[Rešenje 3.108]

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

[Rešenje 3.108]

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

[Rešenje 3.108]

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

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
```

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

[Rešenje 3.108]

**Zadatak 3.104** Napisati program koji sa standardnog ulaza učitava prvo dimenzije matrice ( $n$  i  $m$ ) a zatim redom i elemente matrice (ne postoje pretpostavke o dimenziji matrice). Nakon toga na standardni izlaz, zapisati indekse ( $i$  i  $j$ ) onih elemenata matrice koji su jednaki zbiru svih svojih susednih elemenata (pod susednim elementima podrazumevamo okolnih 8 polja matrice ako postoje).

*Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
```

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

[Rešenje 3.108]

**Zadatak 3.105** Sa standardnog 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]

### 3 Predstavljanje podataka

---

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

*Primer 1*

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

[Rešenje 3.108]

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

*Primer 1*

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

*Primer 2*

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

*Primer 3*

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

[Rešenje 3.108]

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

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 3.108]

3.8 Rešenja

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje 3.108

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

Rešenje [3.108](#)

## 3.9 Strukture

**Zadatak 3.109** Tekst

[Rešenje [3.109](#)]

**Zadatak 3.110** Tekst

[Rešenje [3.110](#)]

**Zadatak 3.111** Tekst

[Rešenje [3.111](#)]

**Zadatak 3.112** Tekst

[Rešenje [3.112](#)]

**Zadatak 3.113** Tekst

[Rešenje [3.113](#)]

**Zadatak 3.114** Definirati strukturu kojom se predstavlja kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja, a zatim i program koji učitava dva kompleksna broja i ispisuje vrednost

zbira, razlike, proizvoda i količnika.

#### Primer 1

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

[Rešenje 3.114]

**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 *Vočka* sadrži ime vočke (nisku maksimalne dužine 20 karaktera) i količinu vitamina C u miligramima (realan broj). Napisati program koji sa standardnog ulaza učitava podatke o vočkama sve do unosa reči KRAJ i ispisuje ime vočke sa najviše vitamina C. Pretpostaviti da broj vočki neće biti veći od 50. *Napomena: probati sa testiranjem zadataka pomoću preusmeravanja.*

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime vočke i njenu količinu vitamina C: jabuka 4.6
Unesite ime vočke i njenu količinu vitamina C: limun 51
Unesite ime vočke i njenu količinu vitamina C: kivi 92.7
Unesite ime vočke i njenu količinu vitamina C: banana 8.7
Unesite ime vočke i njenu količinu vitamina C: pomorandza 53.2
Unesite ime vočke i njenu količinu vitamina C: KRAJ
Voce sa najvise C vitamina je: kivi
```

[Rešenje 3.116]

**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 (niske 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 standardnog ulaza (najviše 100). Ispisati njihov zbir, razliku i proizvod na standardni izlaz.

[Rešenje 3.125]

### 3 Predstavljanje podataka

---

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

[Rešenje 3.125]

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

```
apple jabuka
pineapple ananas
orange narandza
pear kruska
grape grozdje
```

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

[Rešenje 3.125]

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

Primer rada programa:

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

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

[Rešenje 3.125]

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

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

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

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

[Rešenje 3.125]

### Zadatak 3.125

- Definirati tip podataka **TACKA** pogodan za predstavljanje tačke Dekartovske ravni (čije su  $x$  i  $y$  koordinate podaci tipa **double**).
- Definirati funkciju **double rastojanje(TACKA a, TACKA b)** koja izračunava rastojanje između dve tačke.
- Definirati 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.
- Definirati 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).
- Definirati 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).
- Definirati funkciju **main** u kojoj se sa standardnog ulaza učitava celobrojna nenegativna vrednost  $N$  ( $0 < N \leq 100$ ).

Inače, poziva se funkcija **ucitaj\_poligon**. Ukoliko je uspešno učitano  $m$  tačka ( $N$  ne mora da bude jednako  $m$ ), onda se poziva funkcija **obim** za

m učitanih tačaka i ispisuje njen rezultat na standardni izlaz (ukoliko ova funkcija nije implementirana — ispisati na standardni izlaz simbol ?). Posle toga se poziva funkcija `maksimalna_stranica` za m učitanih tačaka i ispisuje njen rezultat na standardni izlaz (ukoliko ova funkcija nije implementirana — ispisati na standardni izlaz simbol ?).

[Rešenje 3.125]

## 3.10 Rešenja

### Rešenje 3.109

```
1 /*
3 Data je struktura koja opisuje koordinate
 tacke u ravni:
5
7 typedef struct point
 {
9 float x;
 float y;
11 } POINT;
13
 U glavnom programu date su dve tacke: tacka
 A sa fiksiranim koordinatama (1,2) i tacka B
 cije koordinate zadaje korisnik. Napisati
15 funkcije:
 a) za racunanje rastojanja izmedju dve date tacke
17 b) za odredjivanje tacke koja se nalazi na
 sredini duzi odredjene dvema datim tackama
19
 Testirati napisane funkcije u glavnom programu.
21
22 */
23
24 #include <stdio.h>
25 #include <math.h>
26
27 typedef struct point
 {
28 float x;
29 float y;
30 } POINT;
31
32 /*
33 Poljima strukture pristupamo pomocu
34 operatora .
35 */
```

```
37 Ako je promenljiva a tipa POINT,
 njenim koordinatama pristupamo
39 pomocu a.x i a.y
 */
41
 float rastojanje (POINT a, POINT b)
43 {
 return sqrt(pow(a.x-b.x,2)+pow(a.y-b.y,2));
45 }

 POINT sredina (POINT a, POINT b)
47 {
 POINT s;
 s.x = (a.x+b.x)/2;
 s.y = (a.y+b.y)/2;
51 return s;
53 }

55
 int main()
57 {

 POINT a = {1,2};
 POINT b;
 POINT sredina_a_b;

63 /* Ispisujemo koordinate tacke a. */
 printf("Tacka a ima koordinate %.2f,%.2f\n", a.x, a.y);

65
 /* Ucitavamo koordinate tacke b. */
67 printf("Unesi prvu koordinatu tacke: ");
 scanf("%f", &b.x);
69 printf("Unesi drugu koordinatu tacke: ");
 scanf("%f", &b.y);
71 printf("Tacka b ima koordinate %.2f,%.2f\n", b.x, b.y);

73 /* Strukture kao argumenti funkcije - prenos po vrednosti. */
 printf("Rastojanje izmedju tacaka a i b je %.2f\n", rastojanje(a,b)
);

75
 /* Struktura kao povratna vrednost funkcije. */
77 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);

79
 return 0;
81 }
```

## Rešenje 3.110

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

11 I Napisati funkciju koja ucitava sa standardnog ulaza podatke o
 studentu. Mozemo pretpostaviti da
 ime i prezime studenta ne sadrže više od 30 karaktera.
13 II Napisati funkciju koja ispisuje podatke o studentu na standardni
 izlaz.
 III Ucitati niz od n studenata i :
15 a) ispisati imena i prezimena onih koji su na smeru R
 b) ispisati podatke za studenta sa najvećim prosekom; ako ima
 više takvih studenata, ispisati
17 1) sve njih
 2) prvog
19 3) poslednjeg
21 */

23 #include <stdio.h>
25 #define MAXST 100
27 #define MAX 31

29 typedef struct Student
31 {
33 char ime[MAX];
35 char prezime[MAX];
37 char smer;
39 float prosek;
41 } STUDENT;

43 /*
45 I
47
 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 veći prioritet nego operator . .

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

```

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

```

### 3 Predstavljanje podataka

---

```
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.
 */
 s = studenti[i];
 break;
127 }
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 zadatka moze biti izabran bilo koji od opisanih
 nacina rada, osim
 ako neki nacin nije posebno naglasen u tekstu zadatka.

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

169 /* printf("Unesi broj studenata:"); */
 scanf("%d", &n);
171

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

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

### 3 Predstavljanje podataka

```
181 {
 /* printf("%d. student:\n", i); */
 ucitaj(&studenti[i]);
183 }

185 printf("Studenti sa R smer:\n");
 for(i=0;i<n;i++)
187 if(studenti[i].smer == 'R')
 ispisi(&studenti[i]);
189 printf("-----\n");

191
 /* b)1)

193 Stampamo podatke o svim studentima sa
195 maksimalnim prosekom.
 */

197 max_prosek = najveći_prosek(studenti, n);
199 printf("Svi studenti koji imaju maksimalni prosek:");
 for(i=0;i<n;i++)
201 if(studenti[i].prosek==max_prosek)
 ispisi(&studenti[i]);
203

 /* b)2) */
205 student_sa_max_prosekom = prvi_student_sa_najvecim_prosekom(
 studenti,n,max_prosek);

207 printf("Prvi student u nizu sa najvecim prosekom: ");
 ispisi(&student_sa_max_prosekom);
209

 /* b)3) */
211 poslednji_student_sa_najvecim_prosekom(studenti,n,max_prosek,&
 student_sa_max_prosekom);

213 printf("Poslednji student u nizu sa najvecim prosekom: ");
 ispisi(&student_sa_max_prosekom);
215

 return 0;
217 }
```

#### Rešenje 3.111

```
1 /*
3 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
5 na poslednji karakter najduze reci. Koristiti
 strukturu typedef struct rec
```

```

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

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

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

35
36 */
37
38 #include<stdio.h>
39 #include<string.h>
40 #define MAXRECI 100
41 #define MAX 31

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

47
48 void ucitaj_rec(REC* rec)
49 {
50 scanf("%s", rec->s);
51 rec->duzina = strlen(rec->s);
52 }

53
54 /*
55
56 U funkciji ucitaj_niz_rec argument n oznacava broj

```

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

```

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

```

### Rešenje 3.112

```

12 /*
13 Napisati program koji izracunava prosečnu cenu jedne potrosacke
14 korpe. Potrosacka korpa se sastoji od broja kupljenih artikala i
15 niza kupljenih artikala. Svaki artikal odredjen je svojim nazivom,
16 kolicinom i cenom. Program treba da ucita broj potrosaca n (
17 najvise 100),
18 zatim podatke za n potrosackih korpi i da na osnovu ucitanih
19 podataka
20 izracuna prosečnu cenu potrosacke korpe. Ucitavanje se vrši sa
21 standardnog
22 ulaza pri čemu se prvo zadaje broj artikala, a zatim za svaki
23 artikal naziv,
24 kolicina i cena. Mozemo pretpostaviti da nijedan
25 potrosac neće kupiti više od 20 artikala, kao i da naziv svakog
26 artikla
27 sadrži maksimalno 30 karaktera.
28 */

```

### 3 Predstavljanje podataka

---

```
14 */
16 #include <stdio.h>
18 #define MAXART 20
20 #define MAXPOT 100
22 #define MAXNAZIV 31

24 typedef struct artikal
26 {
28 char naziv[MAXNAZIV];
30 int kolicina;
32 float cena;
34 } ARTIKAL;

36 typedef struct korpa
38 {
40 int br_art;
42 ARTIKAL artikli[MAXART];
44 } KORPA;

46 /*
48 Funkcija ucitaj_artikal učitava podatke za jedan
50 artikal i vraća 1 ako je učitavanje bilo uspesno
52 a 0 u suprotnom. Učitavanje je neuspesno ukoliko
54 kolicina nekog artikla ili njegova cena nisu pozitivni
56 brojevi.

58 S obzirom da funkcija ucitaj_artikal treba da vrati
60 dve vrednosti (ucitanu strukturu i indikator uspesnosti),
62 strukturu ARTIKAL prenosimo preko pokazivaca a
64 indikator uspesnosti vraćamo kao povratnu vrednost.
66 */

68 int ucitaj_artikal(ARTIKAL* a)
70 {
72 scanf("%s", a->naziv);
74 scanf("%d", &a->kolicina);

76 if (a->kolicina<=0)
78 {
80 printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina);
82 return 0;
84 }

86 scanf("%f",&a->cena);
88 if (a->cena<0)
90 {
92 printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
94 return 0;
96 }
98 }
```

```

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

```

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



```

168 korpi
169 */
170 float prosek(KORPA korpe[], int n)
171 {
172 int i;
173 float p;
174
175 for(i=0; i<n; i++)
176 p+=izracunaj_racun(&korpe[i]);
177
178 return p/n;
179 }
180
181 int main()
182 {
183 int n;
184 KORPA korpe[MAXPOT];
185
186 printf("Unesi broj potrosackih korpi:");
187 scanf("%d", &n);
188
189 if(n<0 || n>MAXPOT)
190 {
191 printf("Nekorektan unos broja potrosackih korpi: %d\n", n);
192 return -1;
193 }
194
195 if (ucitaj_niz_korpi(korpe, n)==0)
196 return -1;
197
198 stampaj_racune_za_korpe(korpe, n);
199 printf("Prosecna cena potrosacke korpe: %.2f\n", prosek(korpe, n));
200
201 return 0;
202 }

```

### Rešenje 3.113

```

1 /*
2 Struktura IZRAZ opisuje numericki izraz nad celim brojevima koji
3 se sastoji
4 od dva celobrojna operanda, numericke operacije nad celim
5 brojevima i
6 vrednosti izraza:
7
8 typedef struct izraz
9 {
10 char o;
11 int x;
12 int y;

```

### 3 Predstavljanje podataka

---

```
11 } IZRAZ;

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

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

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

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

27

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

36

37

38 */

39 #include <stdio.h>
40 #define MAX 1000

41 typedef struct izraz
42 {
43 char o;
44 int x;
45 int y;
46 } IZRAZ;

47

48 /*
49 Funkcija korektan_izraz vraca 1 ako je izraz korektan a 0
50 u suprotnom. Izraz je korektan ukoliko se sastoji od
51 aritmetickih operacija +,-,* ili /, i ukoliko je u slucaju
52 operacije deljenja drugi operand razlicit od nule.
53 */
54 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
82 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;
94 case '*':
95 v=izraz->x*izraz->y;
 break;
96 case '/':
97 v=izraz->x/izraz->y;
99 break;
 }
101 return v;
}

103
105 /*
 Promenljiva izraz ce se promeniti u funkciji
 ucitaj_izraz tako sto ce njenim neinicijalizovanim
107 poljima o,x,y biti dodeljene vrednosti ucitane
 sa standardnog ulaza. Zbog toga ovu promenljivu
109 funkciji prosledjujemo po adresi, preko pokazivaca.

```

### 3 Predstavljanje podataka

---

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

```

163 printf("Nekorektna vrednost broja n!\n");
164 return -1;
165 }
166
167 /* U petlji učitavamo jedan po jedan izraz: */
168 for(i=0; i<n; i++)
169 if(ucitaj_izraz(&izrazi[i])==0)
170 {
171 printf("Nekorektan unos\n");
172 return -1;
173 }
174
175 printf("Svi izrazi:\n");
176 for(i=0; i<n; i++)
177 stampaj_izraz(&izrazi[i]);
178
179 max = max_vr(izrazi, n);
180 printf("Maksimalna vrednost izraza:%d\n", max);
181
182 printf("Izrazi cija je vrednost manja od polovine maksimalne
183 vrednosti:\n");
184
185 for(i=0; i<n; i++)
186 if(vrednost(&izrazi[i])<max/2)/* Ako je vrednost tekuceg izraza
187 manja od polovine maksimalne, ispisujemo ga. */
188 stampaj_izraz(&izrazi[i]);
189
190 return 0;
191 }

```

### Rešenje 3.114

```

1 #include <stdio.h>
2
3 /* Struktura koja opisuje kompleksni broj obuhvata polje za realni i
4 polje za imaginarni deo broja. */
5 typedef struct Complex {
6
7 float re;
8 float im;
9 } Complex;
10
11 /* Funkcija kojom se izracunava zbir kompleksnih brojeva */
12 Complex saberi(Complex *a, Complex *b) {
13
14 Complex c;
15 c.re = a->re + b->re;
16 c.im = a->im + b->im;
17 return c;
18 }

```

```
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,
 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 nabiranja 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;

```

### 3 Predstavljanje podataka

---

```
27 for(i = 0; i < n; i++)
28 z += zapremina(&lopte[i]);
29
30 return z;
31 }
32
33 /*
34 Funkcija je opstija od trazene i broji sve lopte odredjene boje u
35 nizu lopti.
36 U zavisnosti od prosledjene boje funkciji, funkcija vraca
37 odgovarajuci broj.
38 */
39 int broj_lopti_u_boji(LOPTA lopte[], int n, int boja) {
40
41 int br = 0;
42 int i;
43 for(i = 0; i < n; i++)
44 if(lopte[i].boja == boja)
45 br++;
46 return br;
47 }
48
49 int main() {
50
51 LOPTA lopte[MAX];
52 int n;
53 int i;
54 int boja;
55
56 printf("Unesite broj lopti: ");
57 scanf("%d", &n);
58
59 if(n < 1 || n > MAX) {
60
61 printf("Nekorektan unos.\n");
62 return 0;
63 }
64
65 printf("Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta,
66 3-crvena, 4-zelena):\n");
67 for(i = 0; i < n; i++) {
68
69 printf("%d. lopta: ", i+1);
70 scanf("%d%d", &lopte[i].poluprecnik, &boja);
71
72 /* U zavisnosti od unetog
73 celog broja,
74 bira se boja lopte.
75 */
76 switch(boja) {
77
78 case 1: lopte[i].boja = plava; break;
```



```

77 case 2: lopte[i].boja = zuta; break;
78 case 3: lopte[i].boja = crvena; break;
79 case 4: lopte[i].boja = zelena; break;
80 default:
81 printf("Nekorektan unos.\n");
82 return 0;
83 }
84 }
85
86 printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));
87
88 printf("Ukupno crvenih lopti: %d\n", broj_lopti_u_boji(lopte, n,
89 crvena));
90
91 return 0;
92 }

```

### Rešenje 3.116

```

1 #include <stdio.h>
2 #include <string.h>
3
4 #define MAX_DUZINA 21
5 #define MAX_BR_VOCKI 50
6
7 typedef struct vocka
8 {
9 char ime[MAX_DUZINA];
10 float vitamin;
11 } VOCKA;
12
13
14 int main()
15 {
16 VOCKA vocke[MAX_BR_VOCKI];
17 int i = 0, n, max_vocka;
18 char ime[MAX_DUZINA];
19
20 /*
21 Ucitavamo podatke o vockama i smestamo ih u niz
22 sve dok ne unesemo rec KRAJ ili ucitamo MAX_BR_VOCKI vocki.
23 */
24 do
25 {
26 printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
27 scanf("%s", ime);
28 /*
29 Kada unesemo rec KRAJ prekidamo petlju.
30 */
31 if(strcmp(ime, "KRAJ") == 0)
32 break;

```

### 3 Predstavljanje podataka

---

```
34 /*
35 Inace učitavamo i kolicinu vitamina
36 i tu vrednost smestamo u vocku na poziciji i
37 */
38 strcpy(vocke[i].ime, ime);
39 scanf("%f", &vocke[i].vitamin);
40 i++;
41 }
42 while(i < MAX_BR_VOCKI);
43
44 n = i;
45
46 /*
47 Pretpostavice da prva vocka ima najviše vitamina.
48 Proćicemo kroz niz vocki i ukoliko nađjemo na vocku koja ima
49 više vitamina
50 od one koja trenutno ima najviše, azuriracemo vrednosti
51 maksimalne vocke.
52
53 Sve vreme cuvamo indeks vocke sa najviše vitamina C.
54 */
55
56 max_vocka = 0;
57 for(i=1; i<n; i++)
58 if(vocke[i].vitamin > vocke[max_vocka].vitamin)
59 {
60 max_vocka = i;
61 }
62
63 printf("Voce sa najviše C vitamina je: %s\n", vocke[max_vocka].ime)
64 ;
65
66 return 0;
67 }
```

#### Rešenje 3.117

```
1 #include <stdio.h>
2 #include <string.h>
3
4 #define MAX_IME_PREZIME 51
5 #define MAX_ZELJA 101
6 #define MAX_BR_STUDENATA 100
7
8 typedef struct student
9 {
10 char imeipre[MAX_IME_PREZIME];
11 char zelja[MAX_ZELJA];
12 } STUDENT;
```

```
14 int main()
15 {
16 STUDENT studenti[MAX_BR_STUDENATA];
17 char odgovor[3];
18 char imeiprezime[MAX_IME_PREZIME];
19 int i = 0, n;
20 int broj_pronalazaka;
21
22
23
24 /*
25 Ucitavamo studente i njihove zelje i upisujemo ih u niz
26 sve dok to zelimo ili dok ne unesemo MAX_BR_STUDENATA studenata.
27 */
28 while(i < MAX_BR_STUDENATA)
29 {
30 printf("Ime i prezime studenta: \n");
31 /*
32 Funkcija fgets ucitava jednu liniju i smesta je promenljivu
33 koju zadajemo kao njen prvi argument.
34 Drugi argument je maksimalna duzina linije.
35 Treci argument je kod nas stdin sto predstavlja standardni ulaz
36 .
37 */
38 if(fgets(studenti[i].imeipre, MAX_IME_PREZIME, stdin) == NULL)
39 {
40 printf("Greska: nismo dobro ucitali ime i prezime studenta.");
41 return 0;
42 }
43
44 printf("Njegova zelja: \n");
45
46 if(fgets(studenti[i].zelja, MAX_ZELJA, stdin) == NULL)
47 {
48 printf("Greska: nismo dobro ucitali zelju studenta.");
49 return 0;
50 }
51
52 i++;
53
54 printf("Jos vrednih studenta (da/ne)?\n");
55
56 scanf("%s", odgovor);
57
58 /*
59 Moramo da pokupimo karakter koji unesemo nakon odgovora
60 kako ga ne bismo ucitali u sledecoj iteraciji petlje.
61 */
62 getchar();
63
64 /*
65 Ukoliko je nas odgovor "ne" prekidamo petlju.
66 */
67 }
```

### 3 Predstavljanje podataka

```
64 A ukoliko nije ni "da" ni "ne" onda nismo dobro uneli odgovor.
65 */
66 if(strcmp(odgovor,"ne") == 0)
67 break;
68 else if(strcmp(odgovor,"da") != 0)
69 {
70 printf("Greska: odgovor mora biti u obliku (da/ne)!");
71 return 0;
72 }
73
74 }
75
76 /*
77 Postavljamo dimenziju niza.
78 */
79 n = i;
80 printf("Za podsecanje uneti ime i prezime: \n");
81 if(fgets(imeiprezime, MAX_IME_PREZIME, stdin) == NULL)
82 {
83 printf("Greska: nismo dobro ucitali ime i prezime studenta.");
84 return -1;
85 }
86
87 /* Prolazimo kroz listu studenta i ispisujemo zelje studenta cije
88 ime i prezime smo uneli. */
89 broj_pronalazaka=0;
90 for(i=0;i<n;i++){
91 if(strcmp(imeiprezime, studenti[i].imeipre) == 0){
92 broj_pronalazaka++;
93 printf("Novogodisnja zelja: %s\n",studenti[i].zelja);
94 }
95 }
96
97 /* Za slucaj da nismo pronasli studenta sa trazenim imenom */
98 if(broj_pronalazaka==0){
99 printf("Trazeni student ne postoji - mozda ce mu poklon odneti
100 drugi Deda Mraz\n");
101 }
102
103 return 0;
104 }
```

#### Rešenje 3.118

```
1 /*
2 Definisati strukturu Grad u kojoj se nalazi ime grada (niska duzine
3 20 karaktera) i prosečna temperatura u
4 toku decembra (realan broj). Napisati program koji ucitava imena n
5 (0<n<50) gradova i njihove prosečne
6 temperature, a zatim ispisuje one gradove koji imaju idealnu
7 temperaturu za klizanje: od 3 do 8 stepeni.
```

```

5 Napomena: probati sa testiranjem zadataka pomocu preusmeravanja.
6 */
7
8 #include <stdio.h>
9 #define MAX_DUZINA 20
10 #define MAX_BR_GRADOVA 50
11
12 typedef struct Grad{
13 char ime_grada[MAX_DUZINA+1];
14 float temperatura;
15 }Grad;
16
17
18 int main(){
19 int n, i;
20 Grad grad[MAX_BR_GRADOVA];
21
22 printf("Unesite broj n: ");
23 scanf("%d", &n);
24 if(n<0 || n>MAX_BR_GRADOVA){
25 printf("Greska: pogresan unos!\n");
26 return 0;
27 }
28
29 for(i=0; i<n; i++){
30 printf("Unesite grad i temperaturu: ");
31 scanf("%s %f", grad[i].ime_grada, &grad[i].temperatura);
32 }
33
34 printf("Gradovi sa idealnom temperaturom za klizanje u decembru:\n"
35);
36 for(i=0; i<n; i++){
37 if(grad[i].temperatura>=3 && grad[i].temperatura<=8){
38 printf("%s\n", grad[i].ime_grada);
39 }
40 }
41
42 return 0;
43 }

```

### Rešenje 3.119

```

1 /*
2 Definisati strukturu ParReci koja sadrzi rec na srpskom jeziku i
3 odgovarajuci prevod na engleski jezik. Zatim
4 sa standardnog ulaza sve do kraja ulaza ucitavati parove reci i,
5 posebno, za recenicu koja se zadaje sa ulaza
 ispisati prevod - ako je rec u recenici nepoznata umesto nje
 ispisati odgovarajuci broj zvezdica. Rec i nece biti
 duze od 50 karaktera, a ukupan broj reci nece biti veci od 100.
 Napomena: probati sa testiranjem zadataka

```

```
 pomocu preusmeravanja.
7 */

9 #include <stdio.h>
 #include <string.h>
11 #define MAX_DUZINA 20
 #define MAX_BR_RECII 100
13 #define MAX_DUZINA_RECENICE 100

15 typedef struct ParReci{
 char sr[MAX_DUZINA+1];
17 char en[MAX_DUZINA+1];
 }ParReci;
19

21 /*
 Funkcija koja u recniku koji sadrzi n reci trazi prevod reci rec i
 upisuje ga u prevod.
23 Ukoliko se rec ne nalazi u recniku, prevod se sastoji od zvezdica
 pri cemu broj zvezdica odgovara
 duzini nepoznate reci.
25 */

27 void pronadji_prevod(ParReci recnik[], int n, char rec[], char prevod
 []){
 int i;

29

 /* Pretrazujemo recnik i trazimo zadatu rec */
31 for(i=0; i<n; i++){

33 /* Ukoliko se rec nalazi u recniku */
 if(strcmp(recnik[i].sr, rec)==0){
35 /* Ocitavamo njen prevod */
 strcpy(prevod, recnik[i].en);
37 /* I obustavljamo pretragu */
 return;
39 }
 }

41 /* Ukoliko rec nije pronadjena, formiramo prevod reci koji se
 sastoji od zvezdica */
43 for(i=0; rec[i]; i++){
 prevod[i]='*';
45 }
 prevod[i]='\0';
47 }

49
51 int main(){
 ParReci recnik[MAX_BR_RECII];
 int n;
53 char sr[MAX_DUZINA+1];
```

```
55 char en[MAX_DUZINA+1];
56 int i, j, k;
57 char recenica[MAX_DUZINA_RECENICE+1];
58 char rec[MAX_DUZINA+1];
59 char prevod[MAX_DUZINA+1];
60 int citamo_rec;
61 char* novi_red;
62
63 /* Ucitavamo parove reci sa standardnog ulaza sve do kraja ulaza*/
64 i=0;
65 while(scanf("%s %s", sr, en)!=EOF){
66 if(i==MAX_BR_REC)
67 break;
68
69 strcpy(recnik[i].sr, sr);
70 strcpy(recnik[i].en, en);
71
72 i++;
73 }
74 /* Broj parova reci cuvamo u promenljivoj n */
75 n=i;
76
77 /* Ucitavamo recenicu - nisku karaktera sve do pojave znaka za novi
78 red */
79 printf("Unesite recenicu za prevod:\n");
80 fgets(recenica, MAX_DUZINA_RECENICE, stdin);
81
82 /* Ako postoji, zamenjujemo znak za novi red terminirajucom nulom
83 */
84 novi_red=strchr(recenica, '\n');
85 if(novi_red!=NULL)
86 *novi_red='\0';
87
88 /* Izdvajamo rec po rec unesene recenice */
89 /* j oznacava tekuci karakter recenice koji se obradjuje */
90 j=0;
91 /* citamo_rec sa mogucim vrednostima 1 i 0 ce biti indikator koji
92 pokazuje da li citamo rec ili ne */
93 citamo_rec=0;
94
95 while(1){
96 /* Proveravamo da li smo stigli do kraja recenice */
97 if(recenica[j]=='\0')
98 break;
99
100 /* Ukoliko smo procitali karakter koji je sastavni deo reci (nije
101 belina) */
102 if(recenica[j]!=' ' && recenica[j]!='\n' && recenica[j]!='\t'){
103 /* Smestamo ga u rec */
104 if(citamo_rec==0){
105 citamo_rec=1;
```

### 3 Predstavljanje podataka

---

```

 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';

 /* I trazimo i ispisujemo odgovarajuci prevod reci */
115 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 obradjuvane:
Petlje
Funkcije
Nizovi
Strukture

IZLAZ:
Teme koje su obradjuvane:
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
m114005 8 8 9 8 10
m115112 9 8 8 7 10
mr15007 10 10 10 10 10
mn13208 7 7 9 6 10

IZLAZ:
korisnicko ime: mr15007, prosek ocena: 10.00

```

[Rešenje 4.12]

## 4 Ulaz i izlaz programa

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

#### Primer 1

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

#### Primer 2

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

### Primer 3

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

IZLAZ:
-1
```

[Rešenje 4.33]

**Zadatak 4.17** Sastaviti program koji sa standardnog ulaza prima ime datoteke koju treba otvoriti. Ispisati (na standardnom izlazu) koja cifra (međusvim 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 standardnom ulazu, radi sledeće: za cifru u prvoj datoteci, u drugu datoteku se upisuje 0, za slovo se upisuje 1, a za sve ostale karaktere se upisuje 2. Maksimalna dužina naziva datoteka je 20 karaktera.

#### Primer 2

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

[Rešenje 4.33]

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

[Rešenje 4.33]

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

## 4 Ulaz i izlaz programa

---

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

[Rešenje 4.33]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 4.33]

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

### Primer 1

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

### Primer 2

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

[Rešenje 4.33]

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

(a) ispisuje ga [3],

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

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

*Primer 1*

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

*Primer 2*

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

[Rešenje 4.33]

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

(a) ispisuje ga, [3]

(b) u datoteku `rez.txt` upisuje sve reči koje sadrže prvu reč i podvlatku. [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 standardnom ulazu. U prvoj datoteci navedena je rec `r` i niz linija. Napisati program koji u drugu datoteku upisuje sve linije u kojima se rec `r` pojavljuje bar `n` puta, gde je `n` prirodan broj koji se unosi sa standardnog ulaza. Ispis treba da bude u formatu `broj_pojavljivanja: linija`. Linije brojati počevši od 1. Maksimalna dužina naziva datoteka je 20 karaktera.

[Rešenje 4.33]

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

### Primer 1

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

### Primer 2

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

*Primer 3*

```

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

```

*Primer 2*

```

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

```

[Rešenje 4.33]

**Zadatak 4.28** Definirati strukturu

```

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

```

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

*Primer 1*

```

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

```

*Primer 2*

```

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

```

### Primer 3

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

### Primer 4

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

### Primer 5

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

[Rešenje 4.33]

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 4.33]

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

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

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

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

*Primer 1*

```
|| POKRETANJE: ./a.out
 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**

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

### Primer 1

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

### Primer 2

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

### Primer 3

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

### Primer 4

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

[Rešenje 4.33]

### Zadatak 4.32 Definirati strukturu

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

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

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

U slučaju greške ispisati -1 i prekinuti izvršavanje programa.

### Primer 1

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

### Primer 2

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

### Primer 3

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



*Primer 4*

```

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

```

[Rešenje 4.33]

**Zadatak 4.33** Prvi red datoteke `ulaz.txt` sadrži 2 cela broja manja od 50 koji predstavljaju redom broj vrsta i broj kolona realne matrice  $A$ . Svaki sledeći red sadrži po jednu vrstu matrice. Napisati program koji nalazi i štampa sve četvorke oblika  $(A(i,j), A(i+1,j), A(i,j+1), A(i+1,j+1))$  u kojima su svi elementi međusobno različiti.

[Rešenje 4.33]

## 4.4 Rešenja

## Rešenje 4.1

```

1 /*
 Napisati program koji prepisuje sadrzaj datoteke ulaz.txt u
3 datoteku izlaz.txt karakter po karakter.
 */
5
7 #include <stdio.h>
 #include <stdlib.h>
9
11 int main()
 {
13 int c;
 FILE *ulaz, *izlaz;
15
17 /*
 Promenljive ulaz i izlaz predstavljaju
 pokazivace na ugradjenu strukturu FILE.
 Unutar ove strukture nalaze se polja neophodna
 za rad sa datotekama.
19
 Kada zelimo da radimo sa nekom datotekom,
 moramo je prvo otvoriti. Ugradjena funkcija
21

```

```
23 fopen(dat, mode) otvara datoteku sa nazivom
24 dat. Datoteka moze biti otvorena za citanje,
25 pisanje ili nadovezivanje, sto odredjuje
26 argument mode koji moze imati vrednost "r" (read),
27 "w"(write) ili "a"(append).
28 */
29
30 ulaz=fopen("ulaz.txt", "r");
31
32 /*
33 Do neuspesnog otvaranja datoteke moze doci
34 ukoliko ne postoji datoteka sa datim nazivom
35 ili je putanja do datoteke pogresna. U tom
36 slucaju, funkcija fopen vraca pokazivac na NULL
37 i tada treba prijaviti gresku. Datoteka stderr
38 predstavlja standardnu datoteku u koju se upisuju
39 greske. Stderr je podrazumevano postavljen
40 na standardni izlaz.
41
42 Ugradjena funkcija exit prouzrokuje zavrsetak programa.
43 Argument ove funkcije je jedna od konstanti definisanih
44 u biblioteci stdlib.h koje pokazuju da li se program
45 zavrrio uspesno (EXIT_SUCCESS) ili neuspesno (EXIT_FAILURE).
46
47 */
48 if(ulaz==NULL)
49 {
50 fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke ulaz
51 .txt za citanje.\n");
52 exit(EXIT_FAILURE);
53 }
54
55 izlaz= fopen("izlaz.txt", "w");
56 if(izlaz==NULL)
57 {
58 fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke
59 izlaz.txt za citanje.\n");
60 exit(EXIT_FAILURE);
61 }
62
63 /*
64 Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
65 Povratna vrednost ove funkcije je ascii kod unetog
66 karaktera.
67
68 Funkcija fputc ispisuje karakter c u datoteku izlaz.
69
70 */
71 while((c=fgetc(ulaz))!=EOF)
72 fputc(c, izlaz);
73
74 /*
```

```

73 Nakon zavrsetka rada sa datotekama, neophodno ih je
 zatvoriti pomocu ugradjene funkcije fclose.
75 */
 fclose(ulaz);
 fclose(izlaz);
77 return 0;
}

```

## Rešenje 4.2

```

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

20 FILE *ulaz, *izlaz;

22 /*
 Proveravamo da li poziv programa ima dovoljan broj argumenata.
24 */
 if(argc!=2)
26 {
 fprintf(stderr, "Greska: program se pokrece sa: %s
 ime_ulazne_datoteke\n", argv[0]);
28 exit(EXIT_FAILURE);
 }

30 ulaz=fopen(argv[1], "r");
32 if(ulaz==NULL)
 {
34 fprintf(stderr, "error fopen(): Neuspelo otvaranje datoteke %s
 za citanje.\n", argv[1]);
 exit(EXIT_FAILURE);
36 }

38 /*

```

```

 Funkcija fgets ucitava jednu liniju teksta maksimalne duzine
 MAX_LEN
40 iz datoteke ulaz u string linija. Ukoliko ucitavanje ne uspe (
 na primer,
 zato sto smo dosli do kraja datoteke), povratna vrednost ove
 funkcije
42 bice prazan pokazivac (NULL).
 */
44
 max_duzina=0;
46 while(fgets(linija, MAX_LEN, ulaz)!=NULL)
 {
48 duzina = strlen(linija);
 /*
50 Promenljivu max_duzina inicijalizovali smo na 0 pre ulaska u
 petlju.
 Ovu promenljivu menjamo kada je duzina ucitana linije
52 veca od max_duzina ili kada su jednake, ali je ucitana
 linija
 leksikografski ispred trenutne linije sa maksimalnom duzinom
 .
54
 Setimo se da funkcija strcmp(s1,s2) vraca razliku ascii
 kodova prva dva
56 razlicita karaktera stringova s1 i s2 na istim indeksima,
 ukoliko oni
 postoje, ili 0 ukoliko su jednaki. Ova funkcija je stoga
 osetljiva
58 na mala i velika slova (npr 'D' je leksikografski ispred 'p'
 ').
60
 */
62 if(duzina>max_duzina || (duzina==max_duzina && strcmp(linija,
 max_linija)<0))
 {
64 strcpy(max_linija, linija);
 max_duzina=duzina;
66 }
 }
68
 /*
70 Funkcija fputs ispisuje string koji je njen prvi argument u
 datoteku
 koja je njen drugi argument. Sve funkcije za ucitavanje iz
 datoteka i
72 upis u datoteke (fgetc, fputc, fgets, fputs, ...) mozemo
 koristiti
 i kada radimo sa standardnim ulazom i standardnim izlazom. Kao
 nazive
74 datoteka tada navodimo stdin i stdout.
 */

```

```

76 fputs(max_linija, stdout);
77
78 fclose(ulaz);
79 return 0;
80 }

```

### Rešenje 4.3

```

/*
2 U datoteci cije se ime zadaje kao prvi argument komandne linije
 nalazi se
 prirodan broj n a zatim i n celih brojeva. Napisati program koji
 prebrojava
4 koliko k-tocifrenih brojeva postoji u datoteci, pri cemu se
 prirodan broj k
 zadaje kao drugi argument komandne linije.
6 */

8 #include <stdio.h>
9 #include <stdlib.h>
10 #include <math.h>

12 /*
 Funkcija ucitaj_i_prebroj ucitava brojeve
14 iz datoteke na koju pokazuje f i prebrojava
 koliko je medju njima k-tocifrenih brojeva
16 */
17 int ucitaj_i_prebroj(FILE* f, int k)
18 {
19 int n;
20 int x;
21 int i;
22 int br;

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

26 /* Ako je taj broj negativan ili nula, izdajemo poruku o gresci.
 */
27 if(n <= 0)
28 {
29 fprintf(stderr, "Greska: broj n mora biti prirodan\n");
30 exit(EXIT_FAILURE);
31 }

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

```

```
40 }
42 return br;
43 }
44
45 int broj_cifara(int x)
46 {
47 int br_c;
48
49 br_c=0;
50
51 /*
52 Do while petlja je pogodnija od petlji sa preduslovom
53 jer tacno racuna broj cifara i za broj 0.
54 */
55 do
56 {
57 br_c++;
58 x/=10;
59 } while(x);
60
61 return br_c;
62 }
63
64 int main(int argc, char* argv[])
65 {
66 int n;
67 int k;
68 FILE* f;
69 int br;
70
71 if(argc!=3)
72 {
73 fprintf(stderr, "Greska: program se pokrece sa: %s
74 naziv_datoteke k \n", argv[0]);
75 exit(EXIT_FAILURE);
76 }
77
78 f=fopen(argv[1], "r");
79
80 if(f==NULL)
81 {
82 fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
83 \n", argv[1]);
84 exit(EXIT_FAILURE);
85 }
86
87 /* Argumenti komandne linije su stringovi. Da bismo konvertovali
88 string
89 u ceo broj koristimo ugradjenu funkciju atoi. */
90 k = atoi(argv[2]);
```

```

 if (k<=0)
90 {
 fprintf(stderr, "Greska: broj k mora biti prirodan\n");
92 exit(EXIT_FAILURE);
 }

94 printf("Broj %d-cifrenih brojeva u datoteci: %d\n", k,
 ucitaj_i_prebroj(f,k));

96 fclose(f);
98 return 0;
}

```

### Rešenje 4.4

```

1 /*
 U datoteci cije se ime navodi kao prvi argument komandne
3 linije navedena je rec r i niz linija. Napisati
 program koji u datoteku cije se ime navodi kao
5 drugi argument komandne linije upisuje sve linije
 u kojima se rec r pojavljuje bar n puta, gde je
7 n prirodan broj koji se unosi sa standardnog ulaza. Ispis
 treba da bude u formatu broj_pojavljivanja: linija.
9 */

11 #include <stdio.h>
12 #include <stdlib.h>
13 #define MAXL 81
14 #define MAXR 31
15
16 /*
17 Funkcija broj_pojavljivanja broji koliko
 se puta pojavio string t u stringu s
19 */
20 int broj_pojavljivanja(char s[], char t[])
21 {
 int br;
23 int i,j;
 /*
25 i - indeks karaktera u s
 j - indeks karaktera u t
27 br - brojac koliko se puta t javlja u s
 */
29 br=0;
 for(i=0;s[i];i++)
31 {
 for(j=0;t[j];j++)
33 if(s[i+j]!=t[j]) /* Ako naidjemo na razlicite karaktere, */
 break; /* prekidamo petlju. */
35 }

 /*
 Do prekida petlje moze doci ili zbog toga sto su pronadjeni

```

```
37 razliciti karakteri i usledio je break ili zbog toga sto
39 je prestao da vazi uslov petlje, odnosno karakter t[j] je
 jednak '\0'. Ako vazi drugi slucaj, to znaci da se string
 t nalazi u stringu s pocev od indeksa i i potrebno je
 uvecati
41 brojac br.
 */
43 if (t[j]!='\0')
 br++;
45 }

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

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

63 in= fopen(argv[1], "r");
 if(in==NULL)
65 {
 fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
 .\n", argv[1]);
 exit(EXIT_FAILURE);
67 }

69 out= fopen(argv[2], "w");
 if(out==NULL)
71 {
 fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
 .\n", argv[2]);
 exit(EXIT_FAILURE);
73 }

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

77 if(n<=0)
 {
79 fprintf(stderr, "Greska: n treba da bude prirodan broj.\n");
 exit(EXIT_FAILURE);
81 }
83 }
```



```

85 fscanf(in, "%s", rec);
87
89 while(fgets(linija, MAXL, in) != NULL)
91 {
92 br = broj_pojavljivanja(linija, rec);
93 if (br >= n)
94 fprintf(out, "%d: %s\n", br, linija);
95 }
96 fclose(in);
97 fclose(out);
98 return 0;
99 }

```

### Rešenje 4.5

```

1 /* Program se pokrece tako sto se navedu nazivi dve datoteke(ulazna i
 izlazna) i opcije.
 U datoteci cije se ime navodi kao prvi argument komandne linije
 nalaze se podaci o razlomcima:
3 u prvom redu se nalazi broj razlomaka, a u svakom sledecem redu
 brojilac i imenilac jednog razlomka.
 Potrebno je kreirati strukturu koja opisuje razlomak i ucitati niz
 razlomaka
5 iz datoteke, a potom:
 a) ukoliko je navedena opcija x, upisati u datoteku cije je ime
 drugi argument komandne linije
7 recipročni razlomak za svaki razlomak iz niza (npr. za 2/3
 treba upisati 3/2)
 b) ukoliko je navedena opcija y, upisati u datoteku cije je ime
 drugi argument komandne linije
9 realnu vrednost reciprocnog razlomka svakog razlomka iz niza
 (npr. za 2/3 treba upisati 1.5)
 Mozemo pretpostaviti da se u datoteci sa podacima o razlomcima
 nalazi najviše 100 razlomaka.
11 */
13
14 /*
15 Prilikom pokretanja programa se, pored naziva ulazne i izlazne
 datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
 obe opcije, sto znaci da je minimalni broj argumenata 3.
17
18 Moguci nacini pokretanja:
19 ./a.out ulaz.txt izlaz.txt -x
 ./a.out ulaz.txt izlaz.txt -y
21 ./a.out ulaz.txt izlaz.txt -yx
 ./a.out ulaz.txt izlaz.txt -xy
23
24 */
25
26 #include <stdio.h>

```

```
27 #include <stdlib.h>
 #include <ctype.h>
29
 #define MAX 100
31
 typedef struct razlomak
33 {
 int br;
35 int im;
 } RAZLOMAK;
37
 /*
39 Funkcija ucitaj_razlomke ucitava razlomke iz datoteke
 na koju pokazuje f u niz. Dimenzija niza, na koju
41 pokazuje pokazivac dim, nije poznata. Prva vrednost
 u datoteci je ukupan broj razlomaka i tu vrednost
43 ucitavamo u promenljivu dim.

45 Funkcija fscanf se koristi isto kao i funkcija scanf
 uz dodatni prvi argument koji predstavlja naziv
47 datoteke iz koje se vrsi ucitavanje.

49 */
 int ucitaj_razlomke(RAZLOMAK niz[], int* dim, FILE* f)
51 {
 int i;
53
 fscanf(f, "%d", dim);
55 for (i=0; i<*dim; i++)
 {
 fscanf(f, "%d %d", &niz[i].br, &niz[i].im);
57 if (niz[i].im==0)
59 return 0;
 }
61 return 1;
 }
63
 RAZLOMAK reciprocni(RAZLOMAK* r)
65 {
 RAZLOMAK rec;
67 rec.im = r->br;
 rec.br = r->im;
69 return rec;
 }
71
 float vrednost(RAZLOMAK* r)
73 {
 return 1.0*r->br/r->im;
75 }

77 int main(int argc, char* argv[])
 {
```

```
79 FILE *in, *out;
 char c;
81 int i;
 int j;
83 int xoption=0;
 int yoption=0;
85 int dim;
 RAZLOMAK razlomci[MAX];
87 RAZLOMAK r;

89 /*
 Prilikom pokretanja programa se, pored naziva ulazne i izlazne
91 datoteke, navode i opcije -x i -y. Moguce je navesti jednu ili
 obe opcije, sto znaci da je minimalni broj argumenata 3.

93
 Moguci nacini pokretanja:
95 ./a.out ulaz.txt izlaz.txt -x
 ./a.out ulaz.txt izlaz.txt -y
97 ./a.out ulaz.txt izlaz.txt -yx
 ./a.out ulaz.txt izlaz.txt -xy
99
101 */
 if(argc!=4)
103 {
 fprintf(stderr, "Greska: program se pokrece sa: %s
 ime_ulazne_datoteke ime_izlazne_datoteke [-x | -y]\n", argv[0]);
105 exit(EXIT_FAILURE);
 }

107
109 in= fopen(argv[1], "r");
 if(in==NULL)
111 {
 fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
 .\n", argv[1]);
113 exit(EXIT_FAILURE);
 }

115
 out= fopen(argv[2], "w");
 if(out==NULL)
117 {
 fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
 .\n", argv[2]);
119 exit(EXIT_FAILURE);
 }

121
123 /* Ispitujemo da li je treca opcija u trazenom formatu. Prvi
 karakter mora biti '-' */

125 if (argv[3][0] != '-')
 {
```

```

127 fprintf(stderr, "Greska u zadavanju opcija: program se pokrece
sa: %s ime_ulazne_datoteke ime_izlazne_datoteke [-x | -y]\n",
argv[0]);
 exit(EXIT_FAILURE);
129 }

131 /* Ostali karakteri mogu biti 'x' ili 'y'. U zavisnosti od date
opcije, postavljamo
vrednosti indikatorskih promenljivih xoption i yoption. */
133
135 for(j=1;argv[3][j]!='\0';j++)
 switch(argv[3][j])
 {
137 case 'x': xoption=1;
 break;
139 case 'y': yoption=1;
 break;
141 default:
 fprintf(stderr, "Greska: nedozvoljeni karakter\n"
);
 exit(EXIT_FAILURE);
 }
143
145
147 if(ucitaj_razlomke(razlomci, &dim, in)==0)
 {
149 fprintf(stderr, "Greska pri zadavanju razlomaka\n");
 exit(EXIT_FAILURE);
151 }

153 /*
 U zavisnosti od datih opcija, vrsimo upis reciprocnih
155 razlomaka u trazenom formatu.

 Funkcija fprintf se koristi na isti nacin kao
157 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)
176 fprintf(out, "%f ", vrednost(&r));
177
178 fprintf(out, "\n");
179 }
180
181 fclose(in);
182 fclose(out);
183
184 return 0;
185 }

```

### Rešenje 4.6

```

1 /*
2 Za svaki automobil poznati su marka, model i cena. Iz datoteke cije
3 se ime zadaje sa standardnog ulaza učitava se broj automobila a
4 potom
5 i podaci za svaki automobil. Program treba da:
6 a) izracuna prosečnu cenu po marki kola
7 b) za maksimalnu cenu koju je kupac spreman da plati, a koja se
8 zadaje
9 kao argument komandne linije, da ispise automobile u tom cenovnom
10 rangju zajednu sa prosečnom cenom odgovarajuće marke
11
12 Mozemo pretpostaviti da se model i marka sastoje od jedne reci i
13 da svaka od njih sadrzi najviše 30 karaktera kao i da se u datoteci
14 nalaze podaci za najviše 100 automobila.
15
16 */
17
18 #include <stdio.h>
19 #include <stdlib.h>
20 #include <string.h>
21 #define MAX 31
22 #define MAXA 100
23
24 typedef struct automobil
25 {
26 char marka[MAX];
27 char model[MAX];
28 float cena;
29 } AUTOMOBIL;
30
31 /*
32 Struktura INFO sadrzi naziv
33 marke automobila, prosek cena
34 za tu marku i broj automobila
35 te marke
36 */
37 typedef struct info

```

```
{
37 char marka[MAX];
 float vrednost;
39 int n;
} INFO;

41 int ucitaj_podatke(FILE* f, AUTOMOBIL a[], int* pn, int max)
43 {
 int i;

45 fscanf(f, "%d", pn);
47 if (*pn <= 0 || *pn > max)
 {
49 printf("Nekorektan unos dimenzije niza automobila\n");
 return 0;
51 }
 for(i=0; i<*pn; i++)
53 fscanf(f, "%s %s %f", a[i].marka, a[i].model, &a[i].cena);

55 return 1;
}

57 /*
59 Funkcija sadrzi ispituje da li se u nizu proseka po marki
 nalazi prosek za marku m. Posto podatak o marki automobila
61 predstavlja string, poredjenje vrsimo pomocu funkcije strcmp.

63 Povratna vrednost ove funkcije je indeks pojavljivanja, ukoliko
 se marka m pojavljuje u nizu proseka, ili -1 u suprotnom.
65 */

67 int sadrzi(INFO p[], int n, char m[])
{
69 int i;
 for(i=0; i<n; i++)
71 if(strcmp(p[i].marka, m) == 0)
 return i;

73 return -1;
75 }

77 /*
79 Funkcija informacije_o_markama za niz automobila a dimenzije n
 racuna proseke cena automobila po markama i smesta ih u niz
 p. Na dimenziju niza p pokazuje pokazivac pn.

81 Ideja je da jednim prolaskom kroz niz sa svaku marku izracunamo
83 sumu cena automobila te marke (koju cemo smestiti u polje vrednost
 strukture
 INFO), i broj automobila te marke (koju cemo smestiti u polje
85 n strukture INFO) i da na kraju podelimo ove dve promenljive
 i tako dobijemo prosečnu vrednost cene.
```

```

87 Za svaki automobil a[i] proveravamo da li se njegova marka vec
89 nalazi u nizu p. Ukoliko se nalazi, nadjenom elementu dodajemo
91 vredost cene automobila a[i] i uvecavamo broj automobila sa
93 tom markom. U suprotnom, dodajemo novi element u niz p. Posto
95 ga dodajemo na kraj, njegov indeks odgovarace dimenziji niza p
97 na koju pokazuje pokazivac *pn.
99 */
100 void informacije_o_markama(AUTOMOBIL a[], int n, INFO p[], int* pn1)
101 {
102 int i,j;
103 int ind;
104 for(i=0;i<n;i++)
105 {
106 /* Proveravamo da li se marka automobila a[i] vec nalazi u
107 nizu p (niz proseka po markama) */
108 ind = sadrzi(p,*pn1,a[i].marka);
109 if(ind==-1) /* Ako se ne nalazi, uvodimo novi element niza na
110 kraj, na poziciju *pn. */
111 {
112 strcpy(p[*pn1].marka, a[i].marka);
113 p[*pn1].vrednost = a[i].cena;
114 p[*pn1].n = 1;
115 (*pn1)++; /* Zagrade su neophodne zbog prioriteta operatora.
116 */
117 }
118 else /* Ako se nalazi, azuriramo polja strukture. */
119 {
120 p[ind].vrednost+=a[i].cena;
121 p[ind].n++;
122 }
123 }
124
125 /* Na osnovu sume cena i broja automobila racunamo prosečnu
126 vrednost. */
127 for(i=0;i<*pn1;i++)
128 p[i].vrednost = p[i].vrednost/p[i].n;
129 }
130
131 void stampaj_informacije(INFO p[], int n)
132 {
133 printf("Informacije o broju automobila i prosečnoj ceni po markama
134 :\n");
135 int i;
136 for(i=0;i<n;i++)
137 printf("%s %.2f %d\n", p[i].marka, p[i].vrednost, p[i].n);
138 }
139
140 /*
141 Funkcija stampa automobile cija je cena manja od maksimalne
142 cene koju je korisnik naveo u komandnoj liniji da je spreman

```

## 4 Ulaz i izlaz programa

---

```
135 da plati, zajedno sa prosecom cenom za tu marku automobila
136 */
137 void stampaj_kandidate(AUTOMOBIL a[], int n, float g, INFO p[], int
 n1)
138 {
139 /*
140 S obzirom da je niz p formiran na osnovu niza a, marka svakog
141 automobila iz niza a se sigurno nalazi u nizu p. Zbog toga
142 nije neophodno proveravati da li je povratna vrednost funkcije
143 sadrzi razlicita od -1.
144 */
145 int i;
146 printf("Kola u vasem cenovnom rangu:\n");
147 for(i=0;i<n;i++)
148 if(a[i].cena<g)
149 printf("%s %s %.2f\n", a[i].marka, a[i].model, p[sadrzi(p,n1
 ,a[i].marka)].vrednost);
150 }
151
152 int main(int argc, char* argv[])
153 {
154 AUTOMOBIL kola[MAXA];
155 FILE* f;
156 char dat[MAX]; /* Naziv datoteke koji se unosi sa standardnog
 ulaza. */
157 float granica; /* Maksimalna cena koju je korisnik spreman da
 plati.
158 Zadaje se kao argument komandne linije.
159 */
160 INFO infos[MAXA];
161 int dim_kola,dim_infos;
162 int i;
163
164 if(argc!=2)
165 {
166 fprintf(stderr,"Greska: program se pokrece sa: %s
 gornja_granica_cene \n", argv[0]);
167 exit(EXIT_FAILURE);
168 }
169
170 /* Argumenti komandne linije su stringovi. Da bismo od stringa
 dobili
171 realan broj, koristimo ugradjenu funkciju atof. */
172 granica = atof(argv[1]);
173
174 printf("Unesi naziv datoteke:");
175 scanf("%s", dat);
176
177 f=fopen(dat, "r");
178
179 if(f==NULL)
180 {
```



```

181 fprintf(stderr, "Greska fopen(): Neuspelo otvaranje datoteke %s
 .\n", dat);
 exit(EXIT_FAILURE);
183 }

185 if (ucitaj_podatke(f,kola,&dim_kola,MAXA)==0)
 {
187 fprintf(stderr, "Greska pri učitavanju 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);

```

## 4 Ulaz i izlaz programa

---

```
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;
 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 veka od k. Moze se
 pretpostaviti da duzina linije nece biti veka
 od 80 karaktera */
4
#include<stdio.h>
6 #include<string.h>
8 #define MAXL 81
10 int main(int argc, char* argv[]){
12 FILE* in;
 char linija[MAXL];
14 int k;
16 /*Proveravamo da li je program ispravno pozvan*/
 if(argc!=3){
18 printf("Greska: pogresan broj argumenata!");
 return 0;
20 }
22 /*Otvaramo za citanje datoteku koja se navodi kao prvi argument
 komandne linije*/
 in = fopen(argv[1], "r");
24 if(in == NULL){
 printf("Greska: neuspesno otvaranje datoteke!");
26 return 0;
 }
28
 /*Uzimamo brojevnu vrednost drugog argumenta komandne linije*/
30 k = atoi(argv[2]);
32 /*Citamo liniju po liniju i sve linije duze od k ispisujemo na
 standardni izlaz*/
 while(fgets(linija, MAXL, in) != NULL){
34 if(strlen(linija) > k)
 printf("%s", linija);
36 }
 printf("\n");
38

```

```
/*Zatvaramo datoteku*/
40 fclose(in);
 return 0;
42 }
```

### Rešenje 4.10

```
/* Napisati program koji prebrojava koliko se linija datoteke ulaz.
 txt završava niskom s koja se učitava sa stan-
2 dardnog ulaza. Može se pretpostaviti da dužina linije neće biti veća
 od 80 karaktera, kao i da dužina niske s
 ne će biti veća od 20 karaktera */

4
#include<stdio.h>
6 #include <string.h>
#define MAXL 81
8 #define MAXS 21

10 /*Funkcija brojLinija proverava koliko linija u datoteci in se
 završava niskom s.
 Funkcija radi tako što čita jednu po jednu liniju iz datoteke,
12 i zatim kraj te linije poredi sa niskom s.*/
int brojLinija(FILE* in, char* s){
14
 char linija[MAXL];
16 int broj_linija = 0;
 int dužina_s = strlen(s);
18 int dužina_linije;

20 while(fgets(linija, MAXL, in) != NULL){
 dužina_linije = strlen(linija);
22

 /* Ukoliko je znak za novi red bio indikacija kraja linije,
 uklanjamo ga kako bi mogli da izvršimo
24 *ispravno poredjenje (jer niska s nema novi red na kraju) */
 if(linija[dužina_linije-1]=='\n'){
26 linija[dužina_linije-1] = '\0';
 dužina_linije--;
28 }

30 /*linija + dužina_linije će nas odvesti na kraj tog stringa, a kada
 oduzmemo dužinu stringa s,
 a kada od toga oduzmemo dužinu niske s, dobićemo bas onoliko
 poslednjih karaktera, koliko
32 nam i treba. U primeru uspravna crta (|) označava pokazivač
 s ab
34 dužina_s 2
 Linija: aaabbbdfssab
36 |
 Linija + dužina linije aaabbbdfssab
38 |
```

```

 Linija + duzina linije - 2 aaabbbdfssab
40 |
 kada kazemo strcmp(linija + duzina_linije - duzina_s, s), mi
 cemo u nasem primeru zaista porediti "ab" i "ab".
42 */
 if(strcmp(linija + duzina_linije - duzina_s, s) == 0)
44 broj_linija++;
 }
46 return broj_linija;
 }
48
49 int main(){
50
51 FILE* in;
52 char s[MAXS];
53
54 /*Otvaramo datoteku ulaz.txt za citanje i proveravamo da li smo je
 uspesno otvorili*/
 in = fopen("ulaz.txt", "r");
56 if(in == NULL){
 printf("Greska: neuspesno otvaranje datoteke!\n");
58 return 0;
 }
60
 /*Ucitavamo nisku*/
62 printf("Unesite nisku s: ");
 scanf("%s", s);
64
 /*Ispisujemo koliko linija iz datoteke se zavrшава sa niskom s*/
66 printf("Broj linija: %d\n", brojLinija(in, s));
68
 /*Zatvaramo datoteku*/
 fclose(in);
70
 return 0;
72 }

```

### Rešenje 4.11

```

/* Napisati program koji pronalazi maksimum brojeva zapisanih u
 datoteci brojevi.txt */
2
#include<stdio.h>
4
int main(){
6
 FILE* in;
8 float broj, max_broj;
10
 /*Otvaramo datoteku brojevi.txt za citanje i proveravamo da li smo
 je uspesno otvorili*/

```

## 4 Ulaz i izlaz programa

```
12 in = fopen("brojevi.txt", "r");
13 if(in == NULL){
14 printf("Greska pri otvaranju datoteke!");
15 return 0;
16 }
17
18 /*
19 Kako bismo inicijalizovali promenljivu max_broj,
20 citamo jedan broj iz datoteke i smestamo ga u
21 ovu promenljivu. */
22 fscanf(in, "%f", &max_broj);
23
24 /*U petlji citamo sve ostale brojeve i poredimo ih sa trenutnim
25 maksimumom.*/
26 while(fscanf(in, "%f", &broj) != EOF){
27 if(broj > max_broj)
28 max_broj = broj;
29 }
30
31 /*Ispisujemo rezultat*/
32 printf("Najveci broj je: %.2f\n", max_broj);
33
34 /*Zatvaramo datoteku brojevi.txt*/
35 fclose(in);
36
37 return 0;
38 }
```

### Rešenje 4.12

```
1 /* U datoteci studenti. txt se nalaze informacije o studentima: prvo
2 broj studenata, a zatim u pojedinacnim linijama
3 korisnicko ime i pet poslednjih ocena koje je student dobio. Napisati
4 program koji pronalazi studenta koji je
5 ostvario najbolji uspeh i ispisuje njegove podatke. Pretpostaviti da
6 broj studenata nece biti veci od 100. */
7
8 #include<stdio.h>
9
10 #define MAXS 100
11
12 /*Definisemo strukturu za cuvanje studenata*/
13 typedef struct st{
14 char korisnicko_ime[8];
15 float prosek;
16 }STUDENT;
17
18 int main(){
19
20 FILE *ulaz;
21 STUDENT studenti[MAXS];
```

```

19 int ocena1,ocena2,ocena3,ocena4,ocena5, i=0, i_max_prosek;
21 float max_prosek = 0;

23 /*Otvaramo datoteku studenti.txt za citanje*/
 ulaz = fopen("studenti.txt", "r");
25 if(ulaz == NULL){
 printf("Greska pri otvaranju datoteke!\n");
27 return 0;
 }

29 /*Ucitavamo liniju po liniju iz datoteke, sve dok ne dodjemo do
 kraja.
31 Korisnicko ime smestamo u niz, a ocene ucitavamo u pomocne
 promenljive ocena1,...ocena5.
 Zatim, na osnovu ocena racunamo prosek.

33 Ovdje paralelno sa ucitavanjem pronalazimo i studenta sa najvećim
 prosekom i
35 pamtimo njegov prosek i njegovu poziciju u nizu studenata,
 Nismo morali ovako. Mogli smo i prvu da ucitamo sve studente, a
 zatim da prodjemo
37 jednom kroz niz i da nadjemo onog sa najvećim prosekom.

39 */
 while(fscanf(ulaz, "%s%d%d%d%d", studenti[i].korisnicko_ime, &
 ocena1, &ocena2, &ocena3, &ocena4, &ocena5) != EOF){
41 studenti[i].prosek = (ocena1 + ocena2 + ocena3 + ocena4 + ocena5)
 /5.0;

43 if(studenti[i].prosek > max_prosek){
 max_prosek= studenti[i].prosek;
45 i_max_prosek = i;
 }
47 i++;
 }

49 /*Ispisujemo rezultat*/
51 printf("korisnicko ime: %s, prosek ocena: %.2f\n", studenti[
 i_max_prosek].korisnicko_ime, studenti[i_max_prosek].prosek);

53 /*Zatvaramo datoteku*/
 fclose(ulaz);

55 return 0;
57 }

```

### Rešenje 4.13

### Rešenje 4.14

```
1 /* Napisati program koji za rec s maksimalne duzine 20 karaktera koja
 se zadaje sa standardnog ulaza u datoteku
 rotacije.txt upisuje sve rotacije reci s */
3
4 #include<stdio.h>
5 #include<string.h>
6
7 #define MAXS 21
8
9 /*Funkcija rotira nisku za jedno mesto u desno.
 Duzina niske n nije obavezan argument. Mogli smo
11 i da je racunamo u okviru funkcije, ali kako ce sve niske
 sa kojima radimo biti iste duzine, efikasnije je da jednom
13 izracunamo tu duzinu u glavnom programu,
 pa da je prosledjujemo kao argument.*/
15 void rotiraj_zal(char* s, int n){
 int i;
17 char c = s[0];
 for(i=0; i<n-1; i++){
19 s[i] = s[i+1];
 }
21 s[n-1] = c;
 }
23
24 int main(){
25
26 char s[MAXS];
27 int n, i;
28 FILE * izlaz;
29
30 /*Otvaramo datoteku rotacije.txt za pisanje i proveravamo da li smo
 je uspesno otvorili*/
31 izlaz = fopen("rotacije.txt", "w");
 if(izlaz == NULL){
33 printf("Greska pri otvaranju fajla!");
 return 0;
35 }
36
37 /*Sa standardnog ulaza učitavamo rec koju treba da rotiramo*/
 scanf("%s", s);
39
40 /*Racunamo njenu duzinu*/
41 n = strlen(s);
42
43 /*U petlji, ispisujemo tu rec u datoteku, pa je rotiramo za jedno
 mesto u desno.*/
 for(i=0; i<n; i++){
45 fprintf(izlaz, "%s\n", s);
 rotiraj_zal(s,n);
47 }
```



```

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*/
 if(argc > 2){
25 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*/

```

```
39 if(strcmp(argv[1], "-v") == 0 || strcmp(argv[1], "-V") == 0)
 vind = 1;
41 else if(strcmp(argv[1], "-m") == 0 || strcmp(argv[1], "-M") == 0)
 mind = 1;
43 else{
 printf("Greska pri zadavanju opcije!\n");
 return 0;
45 }
 }

47

49 /*Otvaramo datoteku izlaz.txt za pisanje i proveravamo da li smo je
 uspesno otvorili*/
 izlaz = fopen("izlaz.txt", "w");
51 if(izlaz == NULL){
 printf("Greska pri otvaranju datoteke!\n");
53 return 0;
 }

55

 /*Citamo liniju po liniju sa standardnog ulaza i ispisujemo je u
 datoteku.
57 Liniju ispisujemo ukoliko je ispunjen neki od dva uslova:
 1. Izabrana je opcija za ispis malih slova i linija pocinje malim
 slovom
59 2. Izabrana je opcija za velika slova i linija pocinje velikim
 slovom
 NAPOMENA: Kada dodje do kraja ulaza, funkcija fgets vraca NULL
61 */
 while(fgets(linija, MAXL, stdin) != NULL){
63 if(mind && islower(linija[0]) || vind && isupper(linija[0]) ||
 mind && vind)
 fputs(linija, izlaz);
65 }

67 /*Zatvaramo datoteku izlaz.txt*/
 fclose(izlaz);

69 return 0;
71 }
```

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33

Rešenje 4.33



# 5

## Razni zadaci

### 5.1 Rešenja



# Dodatak A

## Ispitni zadaci

### A.1 Testovi/Kolokvijumi

#### A.1.1 Programiranje 1, i-smer, kolokvijum

##### Grupa I

**Zadatak A.1** Napisati URM program koji izračunava funkciju:

$$f(x, y) = \begin{cases} 2x - y & 2x \geq y \\ 3y & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.2** Sa standardnog ulaza unose se jedan karakter (**p** ili **n**) i dva pozitivna trocifrena broja. Na osnovu vrednosti unetog karaktera izračunati i ispisati na standardni izlaz:

**p** - zbir cifara na parnim pozicijama unetih brojeva

**n** - zbir cifara na neparnim pozicijama unetih brojeva

Cifre se broje sa desne strane, tako da cifri jedinice odgovara pozicija 1.

U slučaju greške ( ukoliko karakter nije p ili n ili nisu uneti pozitivni trocifreni brojevi ) ispisati -1.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| p 235 645
|| 8
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| n 567 101
|| 14
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| A 432 543
|| -1
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| p 102 1234
|| -1
```

[Rešenje A.28]

**Zadatak A.3** Sa standardnog ulaza učitava se pozitivan ceo broj  $i$  i ceo broj  $i$  ( $1 \leq i$ ). Na standardni izlaz ispisati broj koji se dobija kada se ukloni  $i$ -ta cifra broja. Cifre se broje sa desne strane, tako da cifri jedinice odgovara pozicija 1. Neispravan ulaz je kada se unose negativan broj ili negativna vrednost ili nula za  $i$  i u tom slučaju na standardni izlaz ispisati -1. Ukoliko broj nema  $i$ -tu cifru broj ostaje nepromenjen.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| 35243 2
|| 3523
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| -14423 1
|| -1
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| 1234 5
|| 1234
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| 523156 6
|| 23156
```

[Rešenje A.28]

## Grupa II

**Zadatak A.4** Napisati URM program koji izračunava funkciju:

$$f(x, y, z) = 4x + 2y + 3z$$

[Rešenje A.28]

**Zadatak A.5** Korisnik unosi 7 karaktera koji predstavljaju indeks studenta koji je oblika OOGGBBB. OO je oznaka smeru i može biti mi, ma, mr, ms, mm, mv. GG je oznaka godine upisa. BBB je oznaka broja koji može biti jednocifren, trocifren ili dvocifren sa vodećim nulama. Na osnovu ovih podataka na standardni



izlaz ispisati ime smeru kome student pripada i indeks u obliku broj/godina. U slučaju greške ( ukoliko OO kao oznaka smeru nije ispravna ili ostali karakteri nisu brojevi ) ispisati -1. Nazivi smerova su: mi - informatika, ma - astronomija, mr - računarstvo i informatika, ms - statistika, mm - teorijska matematika, mp - primenjena matematika

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| mi11275
|| informatika 275/2011
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| mm98005
|| teorijska matematika 5/1998
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| mo23112
|| -1
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| ms12001
|| statistika 1/2012
```

[Rešenje [A.28](#)]

**Zadatak A.6** Državna lutrija došla je na ideju o novoj igri na sreću. Ova igra na sreću igra se tako što se izvuče jedan broj od 1000 do 9999, Nagrada koja se dobija ako ste pogodili izvučen broj je proizvod njegovih parnih cifara i samog broja. Vaš zadatak je da na osnovu izučenog broja izračunate nagradu koja se dobija. Kao ulaz sigurno ćete dobiti ispravan broj. Ako broj nema parnih cifara, nagrada je sam taj broj. Na standardni izlaz ispišite nagradu.

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| 1321
|| 2642
```

### Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| 3284
|| 210176
```

### Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| 1111
|| 1111
```

### Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:
|| 2222
|| 35552
```

### Primer 5

```
|| INTERAKCIJA SA PROGRAMOM:
|| 6031
|| 0
```

### Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| 4321
|| 34568
```

[Rešenje [A.28](#)]

## Grupa III

**Zadatak A.7** Napisati URM program koji izračunava funkciju:

$$f(x, y, z) = \begin{cases} 2 \cdot x + 2 \cdot y & x \leq z \\ z + 3 & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.8** Napisati C program koji sa standardnog ulaza učitava 4 velika slova abecede i nenegativan ceo broj k. Program na standardni izlaz ispisuje 4 karaktera koji se dobijaju cikličkim pomeranjem (u okviru karakterske tabele) unetih karaktera za k mesta unapred. Na primer, karakter A pomeren za 4 mesta unapred postaje E dok karakter Z pomeren za 3 mesta unapred postaje C. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ako neki od unetih karaktera ne predstavlja veliko slovo abecede ili ako je broj k negativan, pretpostaviti da se na ulazu uvek zadaje tačno četiri karaktera.

*Primer 1*

```
|| INTERAKCIJA SA PROGRAMOM:
|| BABA 3
|| EDED
```

*Primer 2*

```
|| INTERAKCIJA SA PROGRAMOM:
|| DEDA 26
|| DEDA
```

*Primer 3*

```
|| INTERAKCIJA SA PROGRAMOM:
|| ZABC 53
|| ABCD
```

*Primer 4*

```
|| INTERAKCIJA SA PROGRAMOM:
|| PERA -2
|| -1
```

*Primer 1*

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

[Rešenje [A.28](#)]

**Zadatak A.9** Napisati C program koji sa standardnog ulaza učitava dva četvorocifrena, pozitivna, cela broja i proverava da li je broj koji se dobija učešljanjem unetih brojeva palindrom. Ako uneti brojevi imaju cifre a1 a2 a3 a4 i b1 b2 b3 b4 tada su cifre učešljanog broja a1 b1 a2 b2 a3 b3 a4 b4. Broj je palindrom ako se čita isto sa obe strane. Ukoliko je broj palindrom ispisati na standardni izlaz 1, ukoliko nije tada ispisati 0, a u slučaju neispravnog ulaza ispisati -1, neispravnim ulazom smatraju se negativni brojevi i brojevi sa brojem cifara manjim ili većim od 4.

Primer 1:

1234 5678

Primer 2:

1342 2431

Primer 3:

1234 4321

Primer 4:

-1234 1234

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 standardni izlaz ispisati koji karakter se najviše puta pojavio u okviru nje, kao i broj pojavljivanja datog karaktera. Ukoliko postoji više karaktera čiji broj pojavljivanja odgovara maksimalnom broju, ispisati onaj sa najmanjim kodom u ASCII tabeli. Pretpostaviti da se na sistemu koristi ASCII tabela.

|             |           |            |                     |
|-------------|-----------|------------|---------------------|
| Primer 1:   | Primer 2: | Primer 3:  | Primer 4:           |
| abrakadabra | cvrcak    | jorgovan99 | s@rm@ ponek@d v@zno |
| a 5         | c 2       | 9 2        | @ 4                 |

[Rešenje [A.28](#)]

**Zadatak A.13** (7 poena) Igra "Minesweeper" sastoji se od pravougaone table izdvojene na polja koja mogu biti bezbedna ili su na njima rasporedjene mine. Sa standardnog ulaza učitavaju se brojevi *n* i *m* koji označavaju dimenzije table. Nako toga unosi se broj *k* kojim se navodi koliko mina se nalazi na tabli i *k* pozicija (*i*, *j*) koja označavaju pozicije na tabli na kojima se nalaze mine (*i*-ti red, *j*-ta kolona). Korisnik zatim unosi koordinate *l* i *m* za koje se na standardni izlaz ispisuje broj koliko se mina nalazi na poljima susednim tom polju. Proveravaju se susedna polja u svih 8 pravaca. Ukoliko je polje koje se proverava baš mina ispisati na standardni izlaz **MINA**. Maksimalna dimenzija table je 100x100. Ukoliko je neka od koordinata izvan dimenzija table ili su dimenzije table izvan dozvoljenih granica na standardni izlaz ispisati -1.

|                 |                 |                 |                 |
|-----------------|-----------------|-----------------|-----------------|
| Primer 1:       | Primer 2:       | Primer 3:       | Primer 4:       |
| Ulaz:    Izlaz: | Ulaz:    Izlaz: | Ulaz:    Izlaz: | Ulaz:    Izlaz: |
| 4 4       2     | 4 4       MINA  | 2 3       -1    | 101 10    -1    |
| 3               | 2               | 1               | 1               |
| 0 1             | 0 1             | -1 0            | 45 67           |
| 1 2             | 1 2             | 2 2             | 30 31           |
| 2 3             | 2 3             |                 |                 |
| 2 2             | 2 3             |                 |                 |

[Rešenje [A.28](#)]

**Zadatak A.14** (7 poena) Služba gradskog prevoza želi da u svakom trenutku ima evidenciju o opterećenju svojih linija. Na linijama saobraćaju autobusi, trolejbusi i tramvaji. Maksimalni kapacitet autobusa je 25, trolejbusa 20 a tramvaja 30 putnika. Broj linije je pozitivan ceo broj manji od 1000.

- a) (1 poen) Definirati strukturu kojom se opisuje vozilo. Svako vozilo zadato je svojim tipom (autobus, trolejbus, tramvaj), linijom na kojom saobraća i brojem putnika koji se u vozilu nalaze.
- b) (6 poena) Sa standardnog ulaza se učitava broj  $n$  ( $0 \leq n \leq 1000$ ),  **$n$  vozila i broj linije**. Za zadati broj linije na standardni izlaz ispisati ukupan broj slobodnih mesta na toj liniji. Koristiti strukturu definisanu pod a). Neispravnim ulazom smatraju se negativan broj putnika, broj putnika veći od dozvoljenog kapaciteta za navedeni tip vozila, tip vozila sa nazivom različitim od navedena tri ili negativan broj linije. U tim slučajevima na standardni izlaz ispisati -1.

Primer 1:

Ulaz:  
4  
autobus 27 18  
trolejbus 28 15  
tramvaj 7 29  
autobus 27 24  
27

Izlaz:  
8

Primer 2:

Ulaz:  
3  
AutobuS 65 23  
Kombi 1 10  
minibus 6 21  
6

Izlaz:  
-1

Primer 3:

Ulaz  
3  
tramvaj 7 29  
tramvaj 3 15  
tramvaj 12 12  
14

Izlaz:  
0

Primer 4:

Ulaz:  
2  
autobus 26 20  
tramvaj 9 32

Izlaz:  
-1

Primer 5:

Ulaz:  
500  
-1

Izlaz:  
-1

[Rešenje [A.28](#)]

## Grupa II

**Zadatak A.15** (4 poena) Napisati C program koji za uneti niz celobrojnog tipa i neparne dužine  $n$  ispisuje po  $k$  elemenata levo i desno od sredine niza (ne uključujući sredinu). Prvo se unosi  $n$ , zatim niz od  $n$  elemenata, a na kraju i  $k$ .

Neispravnim ulazom se smatra niz parne ili negativne dužine, kao i  $k$  koje je negativno ili veće od polovine dužine niza. U slučaju neispravnog ulaza ispisati -1 na standardni izlaz.

Smatrati da je maksimalna veličina niza 100 elemenata.

Primer 1:

Ulaz:  
5  
1 2 3 4 5

Primer 2:

Ulaz:  
9  
9 8 7 6 5 4 3 2 1

Primer 3:

Ulaz:  
6  
1 2 3 4 5 6

Primer 4:

Ulaz:  
3  
1 2 3

Primer 5:

Ulaz:  
5  
10 9 8 7 6

|         |        |        |        |        |
|---------|--------|--------|--------|--------|
| 2       | 1      | 5      | 10     | -6     |
| Izlaz:  | Izlaz: | Izlaz: | Izlaz: | Izlaz: |
| 1 2 3 4 | 6 4    | -1     | -1     | -1     |

[Rešenje [A.28](#)]

**Zadatak A.16** (7 poena) Barkod kodira broj proizvoda dodajući mu kontrolnu cifru. Kontrolna cifra izračunava se kao poslednja cifra zbira jedinica u zapisu svake cifre broja proizvoda. Npr. broj 86012 kodira se kao 1000 0110 0000 0001 0010 a kontrolna cifra je  $(1 + 1 + 1 + 1 + 1) \bmod 10 = 5$ .

Napisati funkciju

```
void kontrolna(char broj_proizvoda[], int *kont)
```

koja izračunava kontrolnu cifru broja proizvoda, koji se zadaje kao niska, i smešta ga u promenljivu kont. Niska može sadržati beline i druge karaktere, ali ih pri izračunavanju kontrolne cifre treba ignorisati, samo cifre uzeti u obzir.

Napisati program koji sa standardnog ulaza učitava liniju teksta kojom je predstavljen broj proizvoda i testira funkciju kontrolna. Na standardni izlaz ispisati izračunatu kontrolnu cifru. Maksimalna dužina niske je 100 karaktera.

Na sistemu se koristi ASCII tabela. Ukoliko ne postoji ni jedna cifra u bar-kodu, onda je kontrolna cifra 0.

|           |           |             |               |
|-----------|-----------|-------------|---------------|
| Primer 1: | Primer 2: | Primer 3:   | Primer 4:     |
| Ulaz:     | Ulaz:     | Ulaz:       | Ulaz:         |
| 86012     | 001-223-4 | 555 555-555 | AB-- 123 --BA |
| Izlaz:    | Izlaz:    | Izlaz:      | Izlaz:        |
| 5         | 6         | 8           | 4             |

[Rešenje [A.28](#)]

**Zadatak A.17** (7 poena) Napisati program koji ispisuje 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  $100 \times 100$ . Ukoliko je ulaz neispravan (za vrednosti  $m$  i  $n$ ) prekinuti rad programa i ispisati -1.

|                 |           |                   |           |
|-----------------|-----------|-------------------|-----------|
| Primer 1:       | Primer 2: | Primer 3:         | Primer 4: |
| Ulaz:           | Ulaz:     | Ulaz:             | Ulaz:     |
| 4 4             | 3 2       | 2 4               | 3 3       |
| 0.2 0.4 0.7 1.3 | 1.23 4.56 | 0.1 0.2 0.3 0.4   | 1 0 0     |
| 1.5 1.7 2.2 2.5 | 0 1       | 10.98 7.65 4.32 1 | 0 1 0     |

|                  |        |        |        |
|------------------|--------|--------|--------|
| 6.3 -1.2 4.4 5.6 | 7.89 1 | Izlaz: | 0 0 1  |
| 1.6 2.3 2.8 3.5  | Izlaz: | 6.2375 | Izlaz: |
| Izlaz:           | 7.8400 |        | 1.000  |
| 8.9500           |        |        |        |

[Rešenje [A.28](#)]

**Zadatak A.18** (*7 poena*) Profesor na jednom predmetu je uveo pravilo da njegov predmet položio svako ko na ispitu osvoji broj poena koji je veći ili jednak od proseka poena umanjenog za 10.

- a) (*1 poen*) Definirati strukturu kojom se opisuje svaki student sa indeksom (**indeks-u-obliku-alas-naloga**) i brojem poena koji je osvojio (ceo broj od 0 do 100).
- b) (*6 poena*) Na ulazu ćete dobiti  $n$  ( $0 \leq n \leq 300$ ), broj studenata koji su polagali predmet, i onda  $n$  redova oblika

indeks-u-obliku-alas-naloga broj-poena-na-ispitu

Ispisati na standardni izlaz indekse svih studenata koji su položili ovaj predmet. Koristiti strukturu definisanu pod a).

Smatrati da je indeks pravilno zapisan. U slučaju loše vrednosti za  $n$  ili loše vrednosti za broj poena ispisati -1.

| Primer 1:  | Primer 2:  | Primer 3:  | Primer 4:   | Primer 5:   |
|------------|------------|------------|-------------|-------------|
| Ulaz:      | Ulaz:      | Ulaz:      | Ulaz:       | Ulaz:       |
| 4          | 4          | 6          | 4           | 3           |
| mi12123 80 | mr12345 91 | mi00001 20 | mi11110 100 | mi05900 98  |
| mi15512 70 | ml54321 80 | mi00002 32 | mi11111 99  | mi13034 120 |
| mi15555 99 | mv36925 29 | mi00003 96 | mi11112 98  | mi11234 34  |
| mi13333 40 | mi14725 55 | mi00004 52 | mi11113 87  | Izlaz:      |
| Izlaz:     | Izlaz:     | mi00005 41 | Izlaz:      | -1          |
| mi12123    | mr12345    | mi00006 15 | mi11110     |             |
| mi15512    | ml54321    | Izlaz:     | mi11111     |             |
| mi15555    | mi14725    | mi00003    | mi11112     |             |
|            |            | mi00004    | mi11113     |             |
|            |            | mi00005    |             |             |

[Rešenje [A.28](#)]

## A.3.2 Programiranje 1, i–smer, Završni ispit, februar, 11.02.2016.

**Zadatak A.19** Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} 2(x - y) & x \geq y \\ 0 & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.20** Sa standardnog ulaza se unose celi, nenegativni brojevi sve dok se ne unese nula. Na standardni izlaz ispisati kvadrat razlike najvećeg i najmanjeg od unetih brojeva. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ukoliko je unet negativan broj ili ukoliko nije unet ni jedan broj osim nule.

| Primer 1:   | Primer 2:    | Primer 3: | Primer 4: |
|-------------|--------------|-----------|-----------|
| 1 2 3 4 5 0 | 1 2 3 -4 5 0 | 1 1 1 1 0 | 0         |
| 16          | -1           | 0         | -1        |

[Rešenje [A.28](#)]

**Zadatak A.21 a)** Napisati funkciju

```
void mutacije(char s1[], char s2[], int *br)
```

koja za navedene niske **s1** i **s2** iste dužine proverava na koliko mesta se karakteri niski razlikuju i rezultat upisuje u promenljivu **br**. Pri poređenju ignorisati beline.

**b)** Napisati program koji sa standardnog ulaza učitava dve DNK sekvence (niske karaktera A, T, C ili G) iste dužine i testira funkciju **mutacije** ispisujući vrednost promenljive **br** na standardni izlaz. Maksimalna dužina niski je 100 karaktera. U slučaju neispravnog ulaza ispisati -1. Ulaz se smatra neispravnim ukoliko neka od niski sadrži karakter koji ne pripada skupu {A, T, C, G} i nije belina ili je jedna niska duža od druge.

| Primer 1:     | Primer 2:      | Primer 3:      | Primer 4:     |
|---------------|----------------|----------------|---------------|
| Ulaz:         | Ulaz:          | Ulaz:          | Ulaz:         |
| AGTC CGCT AGT | ATCG ATCG ATCG | AGTTGTTGT ATGX | AGGGATGGATGAG |
| AGTCC GC TAGT | ACCG ATGC ATCA | TTGTATGGA GGAT | TTGATGACGT    |
| Izlaz:        | Izlaz:         | Izlaz:         | Izlaz:        |
| 0             | 3              | -1             | -1            |



[Rešenje A.28]

**Zadatak A.22** Krtice su organizovano napale baštu šargarepa. Farmer je napravio pravougaonu mapu bašte dimenzija  $n \times m$ , gde je znakom **X** označio polje na kome se nalazi krtičnjak, dok je netaknuta polja označio znakom **-**. Kako je bašta velika, farmer želi da bez mnogo muke izračuna broj krtičnjaka u proizvoljnom pravougaonom delu svoje bašte. Sa standardnog ulaza unose se dimenzije mape **n** i **m**, zatim mapa bašte sa oznakama krtičnjaka i netaknutih polja. Nakon toga farmer zadaje koordinate (**i1**, **j1**) i (**i2**, **j2**) koje označavaju gornji levi i donji desni ugao pravouganika za koji farmer pita koliko krtičnjaka je obuhvaćeno na mapi tim pravouganikom. Na standardni izlaz ispisati broj krtičnjaka u zadatom pravouganiku. Maksimalna dimenzija mape je  $100 \times 100$ . U slučaju neispravnih koordinata uglova pravouganika, neispravnih dimenzija mape ili oznaka na tabli van skupa { **X**, **-** } na standardni izlaz ispisati -1.

| Primer 1: | Primer 2: | Primer 3: | Primer 4: |
|-----------|-----------|-----------|-----------|
| Ulaz:     | Ulaz:     | Ulaz:     | Ulaz:     |
| 4 4       | 4 4       | 4 4       | 4 4       |
| - - X -   | - - X K   | - X - X   | - X - X   |
| X - - -   | - X - -   | X - X -   | X - X -   |
| - X - X   | X - - -   | - X - X   | - X - X   |
| X - X -   | X X - X   | X - X -   | X - X -   |
| 0 1       | 1 2       | 3 4       | 0 0       |
| 2 2       | 3 4       | 1 2       | 3 3       |
| Izlaz:    | Izlaz:    | Izlaz:    | Izlaz:    |
| 2         | -1        | -1        | 8         |

[Rešenje A.28]

**Zadatak A.23** Vlasnik pekare želi da utvrdi koliko je isplativa prodaja njegovog najskupljeg peciva.

**a)** Definisati strukturu **Pecivo** koja sadrži podatke o imenu peciva (najviše 50 karaktera) i ceni peciva (realan broj tipa double).

**b)** Sa standardnog ulaza se unosi broj **n** a zatim mesečni obračun sa **n** prodatih komada peciva, pri čemu je naziv peciva u jednom redu a cena u narednom. Na standardni izlaz ispisati ukupnu zaradu od prodaje najskupljeg peciva zaokruženu na dva decimalna mesta. U slučaju negativne cene peciva ili u slučaju da je **n** manje ili jednako nuli ispisati -1. Pretpostaviti da će samo jedna vrsta peciva imati maksimalnu cenu.

| Primer 1:           | Primer 2: | Primer 3: | Primer 4:         |
|---------------------|-----------|-----------|-------------------|
| Ulaz:               | Ulaz:     | Ulaz:     | Ulaz:             |
| 5                   | 3         | -1        | 5                 |
| burek sa mesom      | mafin     |           | kroasan sa dzemom |
| 100.50              | -50.03    |           | 49.99             |
| buhtla sa cokoladom | krofna    |           | kroasan sa dzemom |
| 50.00               | 56.00     |           | 49.99             |
| burek sa mesom      | krofna    |           | kroasan sa dzemom |
| 100.50              | 56.00     |           | 49.99             |
| rol virsla          |           |           | kroasan sa dzemom |
| 75.00               |           |           | 49.99             |
| kroasan sa kremom   |           |           | kroasan sa dzemom |
| 60.00               |           |           | 49.99             |
| Izlaz:              | Izlaz     | Izlaz:    | Izlaz:            |
| 201.00              | -1        | -1        | 249.95            |

[Rešenje [A.28](#)]

## A.3.3 1. Grupa, I smer, Programiranje 1 2015/2016, ispit, jun

**Zadatak A.24** Napisati URM program koji izračunava funkciju:

$$f(x) = \begin{cases} x - y + 2 & x + 2 \geq y \\ 0 & \text{inače} \end{cases}$$

[Rešenje [A.28](#)]

**Zadatak A.25** Napisati program koji sa standardnog ulaza učitava prvo pozitivan ceo broj  $n$  ( $0 < n \leq 99$ ), a zatim  $n$  celih brojeva i izračunava zbir parnih. Izračunati zbir ispisati na standardni izlaz. U slučaju greške (za  $n \leq 0$  ili  $n \geq 100$ ) na standardni izlaz ispisati -1.

|       |             |                  |             |          |
|-------|-------------|------------------|-------------|----------|
| Ulaz  | 5 1 2 3 4 5 | 5 -1 -2 -3 -4 -5 | 3 10 -10 10 | -3 1 2 3 |
| Izlaz | 6           | -6               | 10          | -1       |

[Rešenje [A.28](#)]

**Zadatak A.26** Napisati funkciju `void f(char s[], char c, int *prva, int* poslednja)` koja u datoj nisci  $s$  pronalazi indekse prvog i poslednjeg pojavljivanja datog karaktera  $c$  i dobijene vrednosti redom smešta u promenljive  $prva$  i

*poslednja*. Ukoliko se karakter ne pojavljuje u nisci, obe vrednosti postaviti na -1.

Potom napisati program koji sa standardnog ulaza učitava karaktersku nisku (dužine ne veće od 150 karaktera) i jedan karakter i nakon toga poziva funkciju *f*, a potom na standardni izlaz ispisuje indekse prvog i poslednjeg pojavljivanja datog karaktera u datoj nisci. Pretpostaviti da je ulaz u ispravnom formatu.

| Ulaz  | ucionica i | ucionica u | ucionica o | ucionica p |
|-------|------------|------------|------------|------------|
| Izlaz | 2 5        | 0 0        | 3 3        | -1 -1      |

[Rešenje A.28]

**Zadatak A.27** Sa standardnog ulaza se zadaje dimenzija kvadratne matrice  $n$  ( $0 < n \leq 99$ ), a zatim elementi matrice koji su celi brojevi. Na standardni izlaz ispisati redni broj vrste koja ima najveći zbir elemenata. U slučaju greške (za  $n \leq 0$  ili  $n \geq 100$ ) na standardni izlaz ispisati -1.

[Rešenje A.28]

**Zadatak A.28** Definirati strukturu *Tacka* za predstavljanje tačaka u ravni sa koordinatama tipa *double*. Sa standardnog ulaza se učitava broj  $n$  ( $1 < n \leq 99$ ), zatim niz od  $n$  tačaka tako što se unosi prvo  $x$ , pa  $y$  koordinata za svaku tačku. Za zadate tačke ispisati na standardni izlaz dužinu najduže duži koja se može obrazovati od neke dve tačke iz učitano g niza. Rezultat ispisati na dve decimalne. Dužina duži između tačaka  $a(x_1; y_1)$  i  $b(x_2; y_2)$  se računa po formuli

$$\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

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](#)