

PROGRAMIRANJE 1

ZBIRKA ZADATAKA SA REŠENJIMA

MILENA VUJOŠEVIĆ JANIČIĆ

JOVANA KONAČEVIĆ DANIJELA SIMIĆ

ANĐELKA ZEČEVIĆ ALEKSANDRA KOĆIĆ



MATEMATIČKI FAKULTET, BEOGRAD 2019.

PROGRAMIRANJE 1

Elektronsko izdanie (2019)

**Milena Vujošević Janičić, Jovana Kovačević,
Danijela Simić, Andjelka Zečević,
Aleksandra Kocić**

**PROGRAMIRANJE 1
Zbirka zadataka sa rešenjima**

**Beograd
2019.**

Autori:

dr Milena Vujošević Janičić, docent na Matematičkom fakultetu u Beogradu

dr Jovana Kovačević, docent na Matematičkom fakultetu u Beogradu

dr Danijela Simić, asistent na Matematičkom fakultetu u Beogradu

Andelka Zečević, asistent na Matematičkom fakultetu u Beogradu

Aleksandra Kocić, asistent na Matematičkom fakultetu u Beogradu

PROGRAMIRANJE 1

Zbirka zadataka sa rešenjima

Izdavač: Matematički fakultet Univerziteta u Beogradu. Studentski trg 16, Beograd.

Za izdavača: *prof. dr Zoran Rakić*, dekan

Recenzenti:

dr Gordana Pavlović-Lažetić, redovni profesor na Matematičkom fakultetu u Beogradu

dr Dragan Urošević, naučni savetnik na Matematičkom institutu SANU

Obrada teksta, crteži i korice: *autori*.

Sadržaj

1 Osnovni elementi imperativnog programiranja	1
1.1 Izrazi	1
1.2 Rešenja	12
1.3 Granaњa	27
1.4 Rešenja	39
1.5 Petlje	68
1.6 Rešenja	96
1.7 Funkcije	152
1.8 Rešenja	165
2 Napredni tipovi podataka	197
2.1 Nizovi	197
2.2 Rešenja	217
2.3 Pokazivači	266
2.4 Rešenja	270
2.5 Niske	278
2.6 Rešenja	289
2.7 Višedimenzioni nizovi	323
2.8 Rešenja	338
2.9 Strukture	371
2.10 Rešenja	382
3 Ulaz i izlaz programa	409
3.1 Argumenti komandne linije	409
3.2 Rešenja	412
3.3 Datoteke	417
3.4 Rešenja	435

A Ispitni rokovi	471
A.1 Modul Matematika	471
A.1.1 Praktični deo ispita, januar 2019.	471
A.1.2 Praktični deo ispita, februar 2019.	473
A.2 Modul Informatika	475
A.2.1 Praktični deo ispita, januar 2019.	475
A.2.2 Praktični deo ispita, februar 2019.	477
A.3 Rešenja	480

Predgovor

U okviru kursa *Programiranje 1*, koji se drži na prvoj godini na svim smerovima na Matematičkom fakultetu, vežbaju se zadaci koji imaju za cilj da studentima pomognu da nauče osnovne algoritme i strukture podataka koji se sreću u imperativnim programskim jezicima. Ova zbirka predstavlja objedinjen skup zadataka sa vežbi i praktikuma ovog kursa, kao i primere zadataka sa održanih ispita. Sva rešenja su data u programskom jeziku C, ali se većina zadataka može koristiti za vežbanje proizvoljnog imperativnog programskog jezika. Elektronska verzija zbirke i propratna rešenja u elektronskom formatu, dostupna su besplatno u skladu sa navedenom licencom i mogu se naći na adresi <http://www.programiranje1.matf.bg.ac.rs/zbirka>.

Zbirka je podeljena u četiri poglavlja. U prvom poglavlju obrađene su uvodne teme koje obuhvataju osnovne elemente imperativnog programiranja koje se koriste u rešavanju svih ostalih zadataka u zbirci. Uvodne teme uključuju osnovne tipove podataka, elementarnu komunikaciju sa korisnikom, građenje izraza, upotrebu naredbi dodele i naredbi koje regulišu kontrolu toka programa (sekvenca, selekcija i iteracija) uključujući i izdvajanje logičkih celina u funkcije. Drugo poglavlje je posvećeno radu sa naprednjim tipovima podataka: nizovima (uključujući niske i višedimenzione nizove), pokazivačima i strukturama. Treće poglavlje posvećeno je dodatnim tehnikama koje se koriste za komunikaciju sa korisnikom. Obrađen je rad sa argumentima komandne linije, kao i rad sa datotekama. Dodatak sadrži primere dva ispitna roka iz jedne akademske godine. Većina zadataka je rešena, a teži zadaci su obeleženi zvezdicom.

Autori velikog broja zadataka ove zbirke su ujedno i autori same zbirke, ali postoje i zadaci za koje se ne može tačno utvrditi ko je originalni autor jer su zadacima davali svoje doprinose različiti asistenti koji su držali vežbe iz ovog kursa tokom prethodnih godina. Zbog toga smatramo da je naš osnovni doprinos što smo objedinili, precizno formulisali, rešili i detaljno iskomentarisali sve najvažnije zadatke koji su potrebni za uspešno savlađivanje koncepata koji se obrađuju u okviru kursa. Takođe, formulacije zadataka smo obogatili prime-rima koji upotpunjaju razumevanje zahteva zadataka i koji omogućavaju čitaocu

zbirke da proveri sopstvena rešenja. Primeri su dati u obliku testova i interakcija sa programom. Testovi su svedene prirode i obuhvataju samo jednostavne ulaze i izlaze iz programa. Interakcija sa programom obuhvata naizmeničnu interakciju čovek-računar u kojoj su ulazi i izlazi isprepletani. U zadacima koji zahtevaju rad sa argumentima komandne linije, navedeni su i primeri poziva programa, a u zadacima koji demonstriraju rad sa datotekama, i primeri ulaznih ili izlaznih datoteka. Test primeri koji su navedeni uz ispitne zadatke u dodatku su oni koji su korišćeni u okviru testiranja i ocenjivanja studentskih rada na ispitima.

Neizmerno zahvaljujemo recenzentima, Gordani Pavlović Lažetić i Draganu Uroševiću, na veoma pažljivom čitanju rukopisa i na brojnim korisnim sugestijama koje su unapredile kvalitet zbirke. Takođe, zahvaljujemo studentima koji su svojim aktivnim učešćem u nastavi pomogli i doprineli uobličavanju ovog materijala kao i svim kolegama koje su držale vežbe na ovom kursu.

Svi komentari i sugestije na sadržaj zbirke su dobrodošli i osećajte se slobodnim da ih pošaljete elektronskom poštom bilo kom autoru¹.

Autori

¹Adrese autora su: milena, jovana, danijela, andjelkaz, aleksandra_kocic, sa nastavkom @matf.bg.ac.rs

1

Osnovni elementi imperativnog programiranja

1.1 Izrazi

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

Test 1

```
|| IZLAZ:  
||   Zdravo svima!
```

[Rešenje 1.1.1]

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

Primer 1

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

Primer 2

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

[Rešenje 1.1.2]

1 Osnovni elementi imperativnog programiranja

Zadatak 1.1.3 Napisati program koji za uneta dva cela broja x i y ispisuje njihov zbir, razliku, proizvod, ceo deo pri deljenju prvog broja drugim brojem i ostatak pri deljenju prvog broja drugim brojem. NAPOMENA: *Prepostaviti da je unos ispravan.*¹

Primer 1

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

Primer 2

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

[Rešenje 1.1.3]

Zadatak 1.1.4 Napisati program koji pomaže kasirki da izračuna ukupan račun ako su poznate cene dva kupljena artikla. Cene artikala su pozitivni celi broevi. NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.1.4]

Zadatak 1.1.5 Napisati program koji za unetu količinu jabuka u kilogramima i unetu cenu po kilogramu ispisuje ukupnu cenu date količine jabuka. Obe ulazne vrednosti su pozitivni celi broevi. NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

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

Primer 1

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

[Rešenje 1.1.5]

¹U zadacima sa ovom napomenom treba prepostaviti da se na ulazu zadaje očekivani broj vrednosti očekivanog tipa u, gde je primereno, odgovarajućem formatu.

Zadatak 1.1.6 Napisati program koji pomaže kasirki da obračuna kusur koji treba da vrati kupcu. Za unetu cenu artikla, količinu artikla i iznos koji je kupac dao, program treba da ispiše vrednost kusura. Sve ulazne vrednosti su pozitivni celi brojevi. NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.1.6]

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

Primer 1

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

Primer 2

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

[Rešenje 1.1.7]

Zadatak 1.1.8 Napisati program koji razmenjuje vrednosti dveju promenljivih x i y . Njihove vrednosti, kao dva cela broja, zadaje korisnik.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vrednosti x i y: 5 7  
Pre zamene: x = 5, y = 7  
Posle zamene: x = 7, y = 5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite vrednosti x i y: 237 -592  
Pre zamene: x = 237, y = -592  
Posle zamene: x = -592, y = 237
```

[Rešenje 1.1.8]

Zadatak 1.1.9 Date su dve celobrojne promenljive a i b . Napisati program koji promenljivoj a dodeljuje njihovu sumu, a promenljivoj b njihovu razliku. NAPOMENA: *Ne koristiti pomoćne promenljive.*

1 Osnovni elementi imperativnog programiranja

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite vrednosti a i b: 5 7  
|| Nove vrednosti su: a = 12, b = -2
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite vrednosti a i b: 237 -592  
|| Nove vrednosti su: a = -355, b = 829
```

[Rešenje 1.1.9]

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

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite trocifreni broj: 697  
|| Cifra jedinica: 7  
|| Cifra desetica: 9  
|| Cifra stotina: 6
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite trocifreni broj: 504  
|| Cifra jedinica: 4  
|| Cifra desetica: 0  
|| Cifra stotina: 5
```

[Rešenje 1.1.10]

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

Primer 1

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

Primer 2

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

[Rešenje 1.1.11]

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

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite trocifreni broj: 892
Obrnuto: 298

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite trocifreni broj: 230
Obrnuto: 32

[Rešenje 1.1.12]

Zadatak 1.1.13 Napisati program koji za uneti pozitivan četvorocifreni broj:

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

NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.1.13]

Zadatak 1.1.14 Napisati program koji ispisuje broj koji se dobija izbacivanjem cifre desetica u unetom pozitivnom celom broju. NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 1349
Rezultat: 139

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 825
Rezultat: 85

[Rešenje 1.1.14]

1 Osnovni elementi imperativnog programiranja

Zadatak 1.1.15 Napisati program koji učitava pozitivan ceo broj n i pozitivan dvocifreni broj m i ispisuje broj dobijen umetanjem broja m između cifre stotina i cifre hiljada broja n . NAPOMENA: Za neke ulazne podatke može se dobiti neočekivan rezultat zbog prekoračenja, što ilustruje test primer broj 2.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite pozitivan ceo broj: 12345
Unesite pozitivan dvocifreni broj: 67
Rezultat: 1267345

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite pozitivan ceo broj: 50000000
Unesite pozitivan dvocifreni broj: 12
Rezultat: 705044704

[Rešenje 1.1.15]

Zadatak 1.1.16 Napisati program koji učitava dužinu dijagonale monitora izraženu u inčima, konvertuje je u centimetre i ispisuje zaokruženu na dve decimale. UPUTSTVO: *Jedan inč ima 2,54 centimetra.*

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj inca: 4.69
4.69 in = 11.91 cm

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj inca: 71.426
71.43 in = 181.42 cm

[Rešenje 1.1.16]

Zadatak 1.1.17 Napisati program koji učitava dužinu pređenog puta izraženu u miljama, konvertuje je u kilometre i ispisuje zaokruženu na dve decimale. UPUTSTVO: *Jedna milja ima 1,609344 kilometara.*

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj milja: 50.42
50.42 mi = 81.14 km

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj milja: 327.128
327.128 mi = 526.46 km

[Rešenje 1.1.17]

Zadatak 1.1.18 Napisati program koji učitava težinu avionskog tereta izraženu u funtama, konvertuje je u kilograme i ispisuje zaokruženu na dve decimale. UPUTSTVO: *Jedna funta ima 0,45359237 kilograma.*

Primer 1

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

Primer 2

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

[Rešenje 1.1.18]

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

Primer 1

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

Primer 2

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

[Rešenje 1.1.19]

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

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

Pri ispisu vrednost zaokružiti na četiri decimale.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 1.1.20]

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

1 Osnovni elementi imperativnog programiranja

Primer 1

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

Primer 2

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

[Rešenje 1.1.21]

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

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite poluprecnik: 4.2
|| Obim: 26.39
|| Povrsina: 55.42

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite poluprecnik: 14.932
|| Obim: 93.82
|| Povrsina: 700.46

[Rešenje 1.1.22]

Zadatak 1.1.23 Napisati program koji za unetu realnu vrednost dužine stranice jednakostraničnog trougla ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Prepostaviti da je unos ispravan.* UPUTSTVO: Za računanje površine jednakostraničnog trougla može se iskoristiti obrazac $P = \frac{a^2 \cdot \sqrt{3}}{4}$ pri čemu je a dužina stranice. Za računanje korena broja koristiti funkciju $\sqrt{ }$ čija se deklaracija nalazi u zaglavlju $math.h$.

Primer 1

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

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite duzinu stranice trougla: 2
|| Obim: 6.00
|| Povrsina: 1.73

[Rešenje 1.1.23]

Zadatak 1.1.24 Napisati program koji za unete realne vrednosti dužina stranica trougla ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Prepostaviti da je unos ispravan.* UPUTSTVO: Za računanje površine trougla može se koristiti Heronov obrazac $P = \sqrt{S \cdot (S - a) \cdot (S - b) \cdot (S - c)}$, pri čemu su a , b i c dužine stranica, a S je poluobim.

Primer 1

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

Primer 2

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

[Rešenje 1.1.24]

Zadatak 1.1.25 Pravougaonik čije su stranice paralelne koordinatnim osama zadan je svojim realnim koordinatama naspramnih temena (gornje levo i donje desno teme). Napisati program koji ispisuje njegov obim i površinu zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.1.25]

Zadatak 1.1.26 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: 6.67
```

Primer 2

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

[Rešenje 1.1.26]

Zadatak 1.1.27 Napisati program koji pomaže moleru da izračuna površinu zidova prostorije koju treba da okreći. Za unete celobrojne vrednosti dimenzije sobe u metrima (dužinu, širinu i visinu), program treba da ispiše površinu zidova za krečenje pod pretpostavkom da na vrata i prozore otpada oko 20%. Omogućiti i da na osnovu unete celobrojne cene usluge po kvadratnom metru program

1 Osnovni elementi imperativnog programiranja

izračuna ukupnu cenu krečenja. Sve realne vrednosti ispisati zaokružene na dve decimale. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenzije sobe: 4 4 3  
|| Unesite cenu po m2: 500  
|| Moler treba da okreci 51.20 m2  
|| Cena krecenja: 25600.00
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenzije sobe: 13 17 3  
|| Unesite cenu po m2: 475  
|| Moler treba da okreci 320.80 m2  
|| Cena krecenja: 152380.00
```

[Rešenje 1.1.27]

Zadatak 1.1.28 Napisati program koji za unete pozitivne cele brojeve x , p i c ispisuje broj koji se dobija ubacivanjem cifre c u broj x na poziciju p . Pretpostaviti da numeracija cifara počinje od nule, odnosno da se cifra najmanje težine nalazi se na nultoj poziciji. NAPOMENA: *Pretpostaviti da je unos ispravan.* UPUTSTVO: Koristiti funkciju pow čija se deklaracija nalazi u zagлавljumu $math.h$.

Primer 1

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

Primer 1

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

[Rešenje 1.1.28]

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

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

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

Primer 1

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

Primer 2

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

[Rešenje 1.1.29]

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

Primer 1

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

Primer 2

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

[Rešenje 1.1.30]

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

Primer 1

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

Primer 2

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

[Rešenje 1.1.31]

Zadatak 1.1.32 Napisati program koji za zadate realne vrednosti x i y ispisuje vrednost sledećeg izraza

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

zaokruženu na dve decimale.

Primer 1

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

Primer 1

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

[Rešenje 1.1.32]

1.2 Rešenja

Rešenje 1.1.1

```
#include <stdio.h>
2
3  /* Prevodjenje programa program.c na Linux operativnom sistemu
4   koriscenjem kompjajlera gcc vrsti se iz terminala komandom
5   gcc program.c
6   Ukoliko program.c ne sadrzi greske, kompjajler ce napraviti
7   izvrsnu verziju programa koja ce se zvati a.out
8   Ovaj program se moze pokrenuti iz terminala navodjenjem komande
9   ./a.out
10  Ukoliko se program prevede na sledeci nacin
11    gcc program.c -o program
12    onda ce izvrsna verzija biti imenovana nazivom koji je dat nakon
13    -o opcije (u ovom slucaju je to rec program) tako da se pokretanje
14    iz terminala moze uraditi navodjenjem komande
15    ./program
16 */
17
18 int main() {
19   /* Ispis trazene poruke. Na kraju poruke se ispisuje novi red. */
20   printf("Zdravo svima!\n");
21
22   /* Povratna vrednost 0 se obicno koristi da oznaci da je prilikom
23   izvrsavanja programa sve proslo u redu. */
24   return 0;
25 }
```

Rešenje 1.1.2

```
#include <stdio.h>
2
3  int main() {
4    /* Deklaracija celobrojne promenljive. */
5    int n;
6
7    /* Ucitavanje vrednosti celog broja. */
8    printf("Unesite ceo broj: ");
9    scanf("%d", &n);
10
11   /* Ispis kvadratne vrednosti unetog broja. */
12   printf("Kvadrat: %d\n", n * n);
13
14   /* Ispis kubne vrednosti unetog broja. */
15   printf("Kub: %d\n", n * n * n);
16
17   return 0;
18 }
```

Rešenje 1.1.3

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int x, y, rezultat;
6
7     /* Ucitavanje vrednosti broja x. */
8     printf("Unesite vrednost broja x: ");
9     scanf("%d", &x);
10
11    /* Ucitavanje vrednosti broja y. */
12    printf("Unesite vrednost broja y: ");
13    scanf("%d", &y);
14
15    /* I nacin ispisa: Dodela zbira x+y promenljivoj rezultat i
16       ispis vrednosti promenljive rezultat. */
17    rezultat = x + y;
18    printf("%d + %d = %d\n", x, y, rezultat);
19
20    /* II nacin ispisa: Direktni ispis vrednosti izraza, bez njegovog
21       dodeljivanja posebnoj promenljivoj. */
22    printf("%d - %d = %d\n", x, y, x - y);
23    printf("%d * %d = %d\n", x, y, x * y);
24
25    /* Kada se operator / primeni na dva celobrojna operanda x i y,
26       kao rezultat se dobije ceo deo pri deljenju broja x brojem y,
27       a ne kolicnik. Na primer, rezultat primene operatora / na 7 i 2
28       je 3, a ne 3.5. */
29    printf("%d / %d = %d\n", x, y, x / y);
30
31    /* Operator % izracunava ostatak pri celobrojnem deljenju dve
32       celobrojne promenljive ili izraza.
33       Da bi se odstampaо karakter %, u pozivu funkcije printf se pise
34       %%.. */
35    printf("%d %% %d = %d\n", x, y, x % y);
36
37    return 0;
38}

```

Rešenje 1.1.4 Pogledajte zadatak 1.1.3. Zbog prepostavke da su cene artikala pozitivni celi brojevi, tip promenljivih za cene treba da bude `unsigned int`.

Rešenje 1.1.5 Pogledajte zadatak 1.1.3. Zbog prepostavke da su cene artikala pozitivni celi brojevi, tip promenljivih za cene treba da bude `unsigned int`.

Rešenje 1.1.6

```
#include <stdio.h>
2
int main() {
4     /* Deklaracija promenljivih cija je vrednost neoznacen ceo broj. */
    unsigned int cena, kolicina, iznos, kusur;
6
8     /* Ucitavanje vrednosti cene, kolicine i iznosa. */
    printf("Unesite cenu, kolicinu i iznos:\n");
    scanf("%u%u%u", &cena, &kolicina, &iznos);
10
12    /* Racunanje kusura. */
    kusur = iznos - kolicina * cena;
14
16    /* Ispis vrednosti kusura. */
    printf("Kusur: %u dinara\n", kusur);
18
}
19
```

Rešenje 1.1.7

```
#include <stdio.h>
2
int main() {
4     /* Deklaracije potrebnih promenljivih. */
    unsigned int poletanje, poletanje_sat, poletanje_minut;
    unsigned int sletanje, sletanje_sat, sletanje_minut;
    unsigned int duzina, duzina_sat, duzina_minut;
8
10    /* Ucitavanje vremena poletanja. */
    printf("Unesite vreme poletanja: ");
    scanf("%u%u", &poletanje_sat, &poletanje_minut);
12
14    /* Ucitavanje vremena sletanja. */
    printf("Unesite vreme sletanja: ");
    scanf("%u%u", &sletanje_sat, &sletanje_minut);
16
18    /* Pretvaranje oba vremena u minute radi lakseg racunanja
       razlike. */
    poletanje = poletanje_sat * 60 + poletanje_minut;
    sletanje = sletanje_sat * 60 + sletanje_minut;
22
24    /* Racunanje razlike u minutima izmedju sletanja i poletanja. */
    duzina = sletanje - poletanje;
26
28    /* Pretvaranje razlike u minutama u razliku u satima i minutama.
       Razlika u satima se dobija celobrojnim deljenjem broja minuta
       sa 60. Preostali broj minuta se dobija kao ostatak pri
       deljenju sa 60. */
}
29
```

```

30     duzina_sat = duzina / 60;
31     duzina_minut = duzina % 60;

32     /* II nacin: duzina_minut = duzina - duzina * 60; */

33     /* Ispis rezultata. */
34     printf("Duzina trajanja leta: %u h i %u min\n", duzina_sat,
35            duzina_minut);

36     return 0;
37 }
```

Rešenje 1.1.8

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int x, y, p;
6
7     /* Ucitavanje vrednosti x i y. */
8     printf("Unesite vrednosti x i y: ");
9     scanf("%d%d", &x, &y);
10
11    /* Ispis vrednosti promenljivih pre zamene. */
12    printf("Pre zamene: x=%d, y=%d\n", x, y);
13
14    /* Pomocna promenljiva p je potrebna da sacuva vrednost
15       promenljive x pre nego sto se ona izmeni i dobije vrednost
16       promenljive y. */
17    p = x;
18    x = y;
19    y = p;
20
21    /* Ispis vrednosti promenljivih nakon zamene. */
22    printf("Posle zamene: x=%d, y=%d\n", x, y);
23
24    return 0;
25 }
```

Rešenje 1.1.9

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int a, b;
6
7     /* Ucitavanje vrednosti a i b. */
8     printf("Unesite vrednosti a i b: ");
```

1 Osnovni elementi imperativnog programiranja

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

11   /* Smestanje sume a + b u promenljivu a. */
12   a = a + b;

13   /* Smestanje izraza a - 2*b u promenljivu b. Uzimajuci u obzir
14      promenu vrednosti promenljive a, u odnosu na pocetne vrednosti
15      promenljivih a i b, vrednost ovog izraza je jednaka
16      a + b - 2*b = a - b. */
17   b = a - 2 * b;

18   /* Ispis rezultata. */
19   printf ("Nove vrednosti su: a=%d, b=%d\n", a, b);

20   return 0;
21 }
```

Rešenje 1.1.10

```
1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     unsigned int x;

7     /* Promenljive koje cuvaju cifre treba da budu najmanjeg
8        celobrojnog tipa jer nece sadrzati druge vrednosti osim
9        jednocifrenih celih brojeva. Zbog toga se koristi tip char. */
10    char cifra_jedinica, cifra_desetica, cifra_stotina;

11    /* Ucitavanje trocifrenog broja. */
12    printf("Unesite trocifreni broj: ");
13    scanf("%u", &x);

15    /* Izdvajanje cifara jedinice, desetice i stotine. */
16    cifra_jedinica = x % 10;
17    cifra_desetica = (x / 10) % 10;
18    cifra_stotina = x / 100;

21    /* Ispis rezultata.
22       Napomena: Kada se stampa numericka vrednost promenljive tipa
23       char koristi se %d. Kada se stampa karakter ciji je ASCII
24       kod jednak vrednosti te promenljive, tada se koristi %c.
25       U ovom slucaju je potrebno stampati numericku vrednost. */
26    printf("Cifra jedinica: %d\n", cifra_jedinica);
27    printf("Cifra desetica: %d\n", cifra_desetica);
28    printf("Cifra stotina: %d\n", cifra_stotina);

31    /* II nacin: Ispis rezultata bez uvodjenja dodatnih promenljivih
32       cifra_jedinica, cifra_desetica i cifra_stotina:
33       printf("Cifre unetog broja su %d,%d,%d\n", x%10, (x/10)%10,
```

```

33         x/100); */
35
36     return 0;
}

```

Rešenje 1.1.11

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     unsigned int cena;
6
7     /* Ucitavanje cene proizvoda. */
8     printf("Unesite cenu proizvoda: ");
9     scanf("%u", &cena);
10
11    /* Vrednost cena/5000 predstavlja maksimalan broj novcanica od 5000
12       dinara koje je moguce iskoristiti za placanje racuna.
13       Na primer, neka je uneta cena 8367 dinara, vrednost izraza
14       8367/5000 je jednaka 1. */
15    printf("%u = %u*5000 + ", cena, cena / 5000);
16
17    /* Da bi se isti postupak primenio i na ostale novcanice, potrebno
18       je izracunati preostali iznos. Jedan nacin da se to uradi je
19       racunanje ostatka pri deljenju unete vrednosti cena
20       (u primeru 8367) sa 5000. On iznosi 3367. Ova vrednost se
21       dodeljuje promenljivoj cena. */
22    cena = cena % 5000;
23
24    /* Ponavljanje postupka za ostale novcanice. */
25    printf("%u*2000 + ", cena / 2000);
26    cena = cena % 2000;
27    printf("%u*1000 + ", cena / 1000);
28    cena = cena % 1000;
29    printf("%u*500 + ", cena / 500);
30    cena = cena % 500;
31    printf("%u*200 + ", cena / 200);
32    cena = cena % 200;
33    printf("%u*100 + ", cena / 100);
34    cena = cena % 100;
35    printf("%u*50 + ", cena / 50);
36    cena = cena % 50;
37    printf("%u*20 + ", cena / 20);
38    cena = cena % 20;
39    printf("%u*10 + ", cena / 10);
40    cena = cena % 10;
41    printf("%u*1\n", cena);
42
43
44     return 0;
45 }

```

Rešenje 1.1.12

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int x, obrnuto_x;
6     char cifra_jedinice, cifra_desetice, cifra_stotine;
7
8     /* Ucitavanje neoznacenog trocifrenog broja. */
9     printf("Unesite trocifreni broj: ");
10    scanf("%u", &x);
11
12    /* Izdvajanje pojedinačnih cifara broja. */
13    cifra_jedinice = x % 10;
14    cifra_desetice = (x / 10) % 10;
15    cifra_stotine = x / 100;
16
17    /* Formiranje rezultujuceg broja. */
18    obrnuto_x = cifra_jedinice * 100 + cifra_desetice * 10 +
19                cifra_stotine;
20
21    /* Ispis rezultata. */
22    printf("Obrnuto: %u\n", obrnuto_x);
23
24    return 0;
25 }
```

Rešenje 1.1.13 Pogledajte zadatak 1.1.12.

Rešenje 1.1.14

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int broj, novi_broj;
6     unsigned int levo, desno;
7
8     /* Ucitavanje neoznacenog celog broja. */
9     printf("Unesite broj: ");
10    scanf("%u", &broj);
11
12    /* Desni deo rezultata je cifra jedinice unetog broja.
13       Na primer, za broj 1234, desni deo je cifra 4. */
14    desno = broj%10;
15
16    /* Levi deo rezultata su sve cifre levo od cifre desetice.
17       Na primer, za broj 1234,levi deo je broj 12 i dobija se
18       deljenjem unetog broja sa 100. */
19    levo = broj/100;
```

```

21  /* Rezultat se dobija spajanjem levog i desnog dela.
   U datom primeru: 12*10 + 4 = 124. */
23  novi_broj = levo*10 + desno;
25  /* Ispis rezultata. */
26  printf("Rezultat: %u\n", novi_broj);
27
28  return 0;
29 }
```

Rešenje 1.1.15

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n, novi_broj;
6     unsigned int levi, desni, m;
7
8     /* Ucitavanje brojeva n i m. */
9     printf("Unesite pozitivan ceo broj: ");
10    scanf("%u", &n);
11    printf("Unesite pozitivan dvocifreni broj: ");
12    scanf("%u", &m);
13
14    /* Levi deo rezultata su sve cifre levo od cifre stotina.
15       Na primer, ako je n=12345, levi deo rezultata je 12.
16       On se dobija deljenjem unetog broja sa 1000. */
17    levi = n / 1000;
18
19    /* Desni deo rezultata su sve cifre desno od cifre hiljada.
20       Za n=12345, desni deo rezultata je 345. */
21    desni = n % 1000;
22
23    /* Srednji deo rezultata je broj m.
24       U navedenom primeru, rezultat se dobija nadovezivanjem
25       brojeva 12, 67 i 345. Ovo se radi mnozenjem delova
26       odgovarajucim stepenom broja 10 i njihovim sabiranjem. */
27    novi_broj = levi * 100000 + m * 1000 + desni;
28
29    /* Ispis rezultata. */
30    printf("Rezultat: %u\n", novi_broj);
31
32    return 0;
33 }
```

Rešenje 1.1.16

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     float in, cm;
6
7     /* Ucitavanje realnog broja koji predstavlja broj inca. */
8     printf("Unesite broj inca: ");
9     scanf("%f", &in);
10
11    /* Racunanje rezultata (1 in = 2.54 cm) */
12    cm = in * 2.54;
13
14    /* Ispis rezultata (na dve decimale). */
15    printf("%.2f in = %.2f cm\n", in, cm);
16
17    return 0;
}
```

Rešenje 1.1.17 Pogledajte zadatak 1.1.16.

Rešenje 1.1.18 Pogledajte zadatak 1.1.16.

Rešenje 1.1.19 Pogledajte zadatak 1.1.16.

Rešenje 1.1.20

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     float a11, a12, a21, a22, determinanta;
6
7     /* Ucitavanje elemenata matrice. */
8     printf("Unesite brojeve: ");
9     scanf("%f%f%f%f", &a11, &a12, &a21, &a22);
10
11    /* Racunanje determinante matrice. */
12    determinanta = a11*a22 - a12*a21;
13
14    /* Ispis rezultata na cetiri decimale. */
15    printf("Determinanta: %.4f\n", determinanta);
16
17    return 0;
}
```

Rešenje 1.1.21

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     float a, b, obim, povrsina;
6
7     /* Ucitavanje duzina stranica pravougaonika. */
8     printf("Unesite duzine stranica pravougaonika: ");
9     scanf("%f%f", &a, &b);
10
11    /* Racunanje obima pravougaonika. */
12    obim = 2 * (a + b);
13
14    /* Racunanje povrsine pravougaonika. */
15    povrsina = a * b;
16
17    /* Ispis rezultata na dve decimale. */
18    printf("Obim: %.2f\n", obim);
19    printf("Povrsina: %.2f\n", povrsina);
20
21    return 0;
22 }
```

Rešenje 1.1.22

```

1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     /* Deklaracija potrebnih promenljivih. */
6     float r, obim, povrsina;
7
8     /* Ucitavanje poluprecnika kruga. */
9     printf("Unesite poluprecnik: ");
10    scanf("%f", &r);
11
12    /* Racunanje obima i povrsine.
13       M_PI je konstanta koja se nalazi u zaglavlju math.h
14       i njena vrednost odgovara pribлизnoj vrednosti broja pi. */
15    obim = 2 * r * M_PI;
16    povrsina = r * r * M_PI;
17
18    /* Ispis rezultata na dve decimale. */
19    printf("Obim: %.2f\nPovrsina: %.2f\n", obim, povrsina);
20
21    return 0;
22 }
```

Rešenje 1.1.23

```
1 #include <stdio.h>
2 #include <math.h>
3
4 /* Uvek kada se koristi neka funkcija iz matematicke biblioteke
5  (na primer , funkcije sqrt , pow , sin , cos)
6  potrebno je prilikom prevodjenja dodati i opciju -lm kojom se
7  kompjajler upucuje da je za pravljenje izvrsne verzije programa
8  potrebno da se program poveze sa matematickom bibliotekom.
9  Ukoliko se ova opcija ne navede, kompjajler prijavljuje gresku:
10 collect2: error: ld returned 1 exit status
11 */
12
13 int main() {
14     /* Deklaracija potrebnih promenljivih. */
15     float a, povrsina, obim;
16
17     /* Ucitavanje duzina stranice. */
18     printf("Unesite duzinu stranice trougla: ");
19     scanf("%f", &a);
20
21     /* Racunanje obima i povrsine. */
22     obim = 3 * a;
23     povrsina = (a * a * sqrt(3)) / 4;
24
25     /* Ispis rezultata na dve decimale. */
26     printf("Obim: %.2f\n", obim);
27     printf("Povrsina: %.2f\n", povrsina);
28
29     return 0;
30 }
```

Rešenje 1.1.24 Pogledajte zadatke 1.1.21 i 1.1.22.

Rešenje 1.1.25 Pogledajte zadatak 1.1.21.

Rešenje 1.1.26

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int a, b, c;
6     float aritmeticka_sredina;
7
8     /* Ucitavanje vrednosti tri cela broja. */
9     printf("Unesite tri cela broja:");
10    scanf("%d%d%d", &a, &b, &c);
11
12    /* Pogresan nacin: aritmeticka_sredina = (a+b+c)/3;
```

```

13     Kada se operacija / koristi nad celim brojevima,
14     deljenje je celobrojno.
15     Na primer, (1+1+3)/3 ima vrednost 1.*/

17 /* Ispravan nacin je da se bar jedan operand
18    pretvori u realan broj. */
19 aritmeticka_sredina = (a + b + c) / 3.0;

21 /* Drugi ispravni nacini:
22    aritmeticka_sredina=1.0*(a+b+c)/3;
23    aritmeticka_sredina=(0.0+a+b+c)/3;
24    aritmeticka_sredina=((float)(a+b+c))/3; */

25 /* Ispis rezultata.*/
26 printf("Aritmeticka sredina: %.2f\n", aritmeticka_sredina);

27
28 return 0;
29 }

```

Rešenje 1.1.27

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int duzina, sirina, visina, cena;
6     float povrsina_za_krecenje, ukupna_cena;
7
8     /* Ucitavanje vrednosti duzine, sirine i visine sobe. */
9     printf("Unesite dimenzije sobe: ");
10    scanf("%u%u%u", &duzina, &sirina, &visina);
11
12    /* Ucitavanje vrednosti cene krecenja. */
13    printf("Unesite cenu po m2: ");
14    scanf("%u", &cena);
15
16    /* Povrsina za krecenje odgovara povrsini kvadra
17       umanjena za povrsinu poda jer se on ne kreci. */
18    povrsina_za_krecenje = 0.8 * (duzina * sirina +
19                                2 * duzina * visina + 2 * sirina * visina);
20
21    /* Racunanje ukupne cene. */
22    ukupna_cena = povrsina_za_krecenje * cena;
23
24    /* Ispis rezultata.*/
25    printf("Moler treba da okreci %.2f m2\n", povrsina_za_krecenje);
26    printf("Cena krecenja: %.2f\n", ukupna_cena);
27
28 return 0;
29 }

```

Rešenje 1.1.28

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     unsigned int x, p, c;
7     unsigned int levo, desno, novo_x;
8
9     /* Ucitavanje broja, pozicije i cifre. */
10    printf("Unesite redom x, p i c: ");
11    scanf("%u%u%u", &x, &p, &c);
12
13    /* Racunanje dela broja koji se nalazi desno od pozicije p.
14       Funkcija pow kao povratnu vrednost vraca realan broj dvostrukih
15       tacnosti, a operacija % ocekuje celobrojne operande. Iz tog
16       razloga je neophodno izvrsiti pretvaranje povratne vrednosti
17       u tip unsigned int. */
18    desno = x % (unsigned int) pow(10, p);
19
20    /* Racunanje dela broja koji se nalazi levo od pozicije p. */
21    levo = x / (unsigned int) pow(10, p);
22
23    /* Racunanje rezultata nadovezivanjem levog dela, cifre c
24       i desnog dela. */
25    novo_x = levo * (unsigned int) pow(10, p + 1) +
26             c * (unsigned int) pow(10, p) + desno;
27
28    /* Ispis rezultata. */
29    printf("Rezultat: %u\n", novo_x);
30
31    return 0;
32}
```

Rešenje 1.1.29

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int a, b;
6     int rezultat_a, rezultat_b, rezultat_c;
7
8     /* Ucitavanje dva cela broja. */
9     printf("Unesite dva cela broja: ");
10    scanf("%d%d", &a, &b);
11
12    /* Izraz a != b ima vrednost 1 ako je ova relacija tacna, a 0 ako
13       je netacna. */
14    rezultat_a = a != b;
```

```

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

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

27  /* Ispis rezultata. */
28  printf("a) Rezultat: %d\n", rezultat_a);
29  printf("b) Rezultat: %d\n", rezultat_b);
30  printf("c) Rezultat: %d\n", rezultat_c);

31  return 0;
32 }

```

Rešenje 1.1.30

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int a, b, max;

7     /* Ucitavanje dve celobrojne vrednosti. */
8     printf("Unesite dva cela broja: ");
9     scanf("%d%d", &a, &b);

11    /* Racunanje maksimuma koriscenjem ternarnog operatara. */
12    max = (a > b) ? a : b;

14    /* Ispis rezultata. */
15    printf("Maksimum: %d\n", max);

17    return 0;
}

```

Rešenje 1.1.31 Pogledajte zadatak 1.1.30**Rešenje 1.1.32**

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     float a, b, rezultat;

```

```
float min, max;
7
/* Ucitavanje vrednosti dva realna broja. */
9 printf("Unesite dva realna broja: ");
scanf("%f%f", &a, &b);
11
/* Racunanje minimalne i maksimalne vrednost unetih brojeva. */
13 min = (a < b) ? a : b;
max = (a > b) ? a : b;
15
/* Racunanje rezultata. */
17 rezultat = (min + 0.5) / (1 + max * max);
19
/* Ispis rezultata. */
printf("Rezultat: %.2f\n", rezultat);
21
return 0;
23 }
```

1.3 Grananja

Zadatak 1.3.1 Napisati program koji ispisuje najmanji od tri uneta cela broja.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite tri cela broja:
5 18 -1
Najmanji: -1

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite tri cela broja:
0 43 16
Najmanji: 0

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite tri cela broja:
-5 -5 -5
Najmanji: -5

[Rešenje 1.3.1]

Zadatak 1.3.2 Napisati program koji za uneti realan broj ispisuje njegovu apsolutnu vrednost zaokruženu na dve decimale.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite jedan realan broj:
7.42
Apsolutna vrednost: 7.42

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite jedan realan broj:
-562.428
Apsolutna vrednost: 562.43

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite jedan realan broj:
0
Apsolutna vrednost: 0.00

[Rešenje 1.3.2]

Zadatak 1.3.3 Napisati program koji za uneti ceo broj ispisuje njegovu recipročnu vrednost zaokruženu na četiri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite jedan ceo broj: 22
Recipročna vrednost: 0.0455

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite jedan ceo broj: -9
Recipročna vrednost: -0.1111

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite jedan ceo broj: 0
Greska: nedozvoljeno je deljenje nulom.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite jedan ceo broj: 57298
Recipročna vrednost: 0.0000

[Rešenje 1.3.3]

1 Osnovni elementi imperativnog programiranja

Zadatak 1.3.4 Napisati program koji učitava tri cela broja i ispisuje zbir pozitivnih.

Primer 1

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

Primer 2

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

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cela broja:  
|| 16 2 576  
|| Zbir pozitivnih: 594
```

[Rešenje 1.3.4]

Zadatak 1.3.5 U prodavnici je organizovana akcija da svaki kupac dobije najjeftiniji od tri artikla za jedan dinar. Napisati program koji za unete cene tri artikla izračunava ukupnu cenu, kao i koliko dinara se uštedi zahvaljujući popustu. Cene artikala su pozitivni celi brojevi. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite tri cene: 247 -133 126  
|| Greska: neispravan unos cene.
```

[Rešenje 1.3.5]

Zadatak 1.3.6 Napisati program koji za uneto vreme u formatu *sat:minut* ispisuje koliko je sati i minuta ostalo do ponoći. Broj sati treba da bude iz intervala [0, 24), a broj minuta iz intervala [0, 60). U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite vreme: 18:19  
|| Do ponoci: 5 sati i 41 minuta
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite vreme: 23:7  
|| Do ponoci: 0 sati i 53 minuta
```

1.3 Grananja

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite vreme: 24:20  
|| Greska: neispravan unos vremena.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite vreme: 14:0  
|| Do ponoci: 10 sati i 0 minuta
```

[Rešenje 1.3.6]

Zadatak 1.3.7 Napisati program koji za unetu godinu ispisuje da li je prestupna. Godina je neoznačen ceo broj.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 1.3.7]

Zadatak 1.3.8 Napisati program koji za učitani karakter ispisuje uneti karakter i njegov ASCII kod. Ukoliko je uneti karakter malo (veliko) slovo, ispisati i odgovarajuće veliko (malo) slovo i njegov ASCII kod.

Primer 1

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

Primer 2

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

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: A  
|| Uneti karakter: A  
|| ASCII kod: 65  
|| Odgovarajuce malo slovo: a  
|| ASCII kod: 97
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karakter: v  
|| Uneti karakter: v  
|| ASCII kod: 118  
|| Odgovarajuce veliko slovo: V  
|| ASCII kod: 86
```

[Rešenje 1.3.8]

Zadatak 1.3.9 Napisati program koji učitava tri karaktera. Ispitati da li među unetim karakterima ima cifara i ako je tako odrediti proizvod tih cifara. Ukoliko među unetim karakterima nema cifara, program treba da ispiše

1 Osnovni elementi imperativnog programiranja

odgovarajuću poruku. NAPOMENA: Karakteri koji se unose su razdvojeni blanko znacima.

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: A 5 3
|| Proizvod cifara: 15

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: k ! m
|| Među unetim karakterima nema cifara.

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: 9 9 9
|| Proizvod cifara: 729

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: a 8 0
|| Proizvod cifara: 0

[Rešenje 1.3.9]

Zadatak 1.3.10 Kasirka unosi šifru artikla koja se zadaje u formi tri spojena karaktera koji mogu biti mala slova, velika slova ili cifre. U kasi su sve šifre zapisane malim slovima i ciframa. Napisati program koji kasirkin unos konvertuje u unos koji je odgovarajući za kasu, tj. koji sva velika slova pretvara u odgovarajuća mala, a ostale karaktere ne menja. U slučaju neispravnog unosa šifre, ispisati odgovarajuću poruku o grešci.

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite sifru: aBc
|| Rezultat: abc

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite sifru: a?!

Greška: ? je neispravan karakter.

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: 5A5
|| Rezultat: 5a5

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite karaktere: 123
|| Rezultat: 123

[Rešenje 1.3.10]

Zadatak 1.3.11 Napisati program koji za uneti četvorocifreni broj ispisuje njegovu najveću cifru. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: 6835
|| Najveća cifra je: 8

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite cetvorocifreni broj: 7777
|| Najveća cifra je: 7

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite cetvorocifreni broj: 238
Greska: niste uneli cetvorocifreni broj.

Primer 4

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

[Rešenje 1.3.11]

Zadatak 1.3.12 Trocifreni broj je Armstrongov ako je jednak zbiru kubova svojih cifara. Napisati program koji za dati pozitivan trocifreni broj proverava da li je Armstrongov. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite pozitivan trocifreni broj:
153
Broj je Armstrongov.

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite pozitivan trocifreni broj:
111
Broj nije Armstrongov.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite pozitivan trocifreni broj:
84
Greska: niste uneli pozitivan trocifreni broj.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite pozitivan trocifreni broj:
371
Broj je Armstrongov.

[Rešenje 1.3.12]

Zadatak 1.3.13 Napisati program koji ispisuje proizvod parnih cifara unetog četvorocifrenog broja. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite cetvorocifreni broj: 8123
Proizvod parnih cifara: 16

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite cetvorocifreni broj: 3579
Nema parnih cifara.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite cetvorocifreni broj: 288
Greska: niste uneli cetvorocifreni broj.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite cetvorocifreni broj: -1234
Proizvod parnih cifara: 8

[Rešenje 1.3.13]

1 Osnovni elementi imperativnog programiranja

Zadatak 1.3.14 Napisati program koji učitava četvorocifreni broj i ispisuje broj koji se dobija kada se unetom broju razmene najmanja i najveća cifra. U slučaju da se najmanja ili najveća cifra pojavljuju na više pozicija, uzeti prvo pojavljivanje, gledajući sa desna na levo. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite cetvorocifreni broj: 1192  
|| Rezultat: 1912
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite cetvorocifreni broj: 247  
|| Greska: niste uneli cetvorocifreni broj.
```

Primer 4

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

[Rešenje 1.3.14]

Zadatak 1.3.15 Napisati program koji za uneti četvorocifreni broj proverava da li su njegove cifre uređene neopadajuće, nerastuće ili nisu uređene i štampa odgovarajuću poruku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite cetvorocifreni broj: 88  
|| Greska: niste uneli cetvorocifreni broj.
```

Primer 4

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

[Rešenje 1.3.15]

Zadatak 1.3.16 Napisati program koji ispituje da li se tačke $A(x_1, y_1)$ i $B(x_2, y_2)$ nalaze u istom kvadrantu. Koordinate tačaka su realni brojevi jednostrukne tačnosti.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 1.5 6
Unesite koordinate tacke B: 2.33 9.8
Tacke se nalaze u istom kvadrantu.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: -3 6
Unesite koordinate tacke B: 0.33 -5
Tacke se ne nalaze u istom kvadrantu.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 0 -6
Unesite koordinate tacke B: -1 -99.66
Tacke se nalaze u istom kvadrantu.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 3 -6
Unesite koordinate tacke B: -0.33 0
Tacke se ne nalaze u istom kvadrantu.
```

[Rešenje 1.3.16]

Zadatak 1.3.17 Napisati program koji ispituje da li se tačke $A(x_1, y_1)$, $B(x_2, y_2)$ i $C(x_3, y_3)$ nalaze na istoj pravoj.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 1.5 6
Unesite koordinate tacke B: -2.5 -10
Unesite koordinate tacke C: 3 12
Tacke se nalaze na istoj pravoj.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: -1.5 3
Unesite koordinate tacke B: -0.4 9.8
Unesite koordinate tacke C: 2 3
Tacke se ne nalaze na istoj pravoj.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 1.55 6
Unesite koordinate tacke B: -8.4 9.8
Unesite koordinate tacke C: 5 4.682412
Tacke se nalaze na istoj pravoj.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 5.5 3.5
Unesite koordinate tacke B: 5.5 3.5
Unesite koordinate tacke C: 5.5 3.5
Tacke se nalaze na istoj pravoj.
```

Primer 5

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 1 2
Unesite koordinate tacke B: 1 2
Unesite koordinate tacke C: -56 1.3
Tacke se nalaze na istoj pravoj.
```

Primer 6

```
INTERAKCIJA SA PROGRAMOM:
Unesite koordinate tacke A: 3.4 3.5
Unesite koordinate tacke B: -10 -1
Unesite koordinate tacke C: -10 -1
Tacke se nalaze na istoj pravoj.
```

[Rešenje 1.3.17]

Zadatak 1.3.18 Napisati program za rad sa intervalima. Za dva celobrojna intervala $[a_1, b_1]$ i $[a_2, b_2]$ program treba da odredi:

1 Osnovni elementi imperativnog programiranja

- (a) dužinu preseka datih intervala
- (b) presečni interval datih intervala
- (c) dužinu dela prave koju pokrivaju dati intervali
- (d) najmanji interval koji sadrži date intervale.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite a1, b1, a2 i b2: 2 9 4 11  
|| Duzina preseka: 5  
|| Presecni interval: [4,9]  
|| Duzina koju pokrivaju: 9  
|| Najmanji interval: [2, 11]
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite a1, b1, a2 i b2: 1 2 10 13  
|| Duzina preseka: 0  
|| Presecni interval: prazan  
|| Duzina koju pokrivaju: 4  
|| Najmanji interval: [1, 13]
```

[Rešenje 1.3.18]

Zadatak 1.3.19 Napisati program koji za unete koeficijente kvadratne jednačine ispisuje koliko realnih rešenja jednačina ima i ako ih ima, ispisuje ih zaokružene na dve decimale.

Primer 1

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

Primer 2

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

[Rešenje 1.3.19]

Zadatak 1.3.20 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. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite k: 200  
|| Greska: neispravan unos pozicije.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite k: 10  
|| Na 10-toj poziciji je broj 1.
```

[Rešenje 1.3.20]

Zadatak 1.3.21 Data je funkcija $f(x) = 2 \cdot \cos(x) - x^3$. Napisati program koji za učitanu vrednost realne promenljive x i vrednost celobrojne promenljive k koja može biti 1, 2 ili 3 izračunava vrednost funkcije $F(x, k)$ koja se dobija tako što se funkcija f primeni k -puta ($F(x, 1) = f(x), F(x, 2) = f(f(x)), F(x, 3) = f(f(f(x)))$). Dobijenu vrednosti ispisati zaokruženu na dve decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 1.3.21]

Zadatak 1.3.22 Napisati program koji za uneti redni broj dana u nedelji ispisuje ime odgovarajućeg dana. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj: 8  
Greska: neispravan unos  
dana.
```

Primer 3

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

[Rešenje 1.3.22]

Zadatak 1.3.23 Napisati program koji za uneti karakter ispituje da li je samoglasnik ili ne.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 1.3.23]

1 Osnovni elementi imperativnog programiranja

Zadatak 1.3.24 Napisati program koji učitava dva cela broja i jedan od karaktera +, -, *, / ili % i ispisuje vrednost izraza dobijenog primenom date operacije nad učitanim vrednostima. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite izraz: 8 - 11  
|| Rezultat je: -3
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite izraz: 14 / 0  
|| Greska: deljenje nulom.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite izraz: 5 ? 7  
|| Greska: nepoznat operator.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite izraz: 19 / 5  
|| Rezultat je: 3
```

[Rešenje 1.3.24]

Zadatak 1.3.25 Napisati program koji za uneti datum u formatu *dan.mesec.* ispisuje godišnje doba kojem pripadaju. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dan i mesec: 14.10.  
|| jesen
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dan i mesec: 2.8.  
|| letno
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dan i mesec: 27.2.  
|| zima
```

[Rešenje 1.3.25]

Zadatak 1.3.26 Napisati program koji za unetu godinu i mesec ispisuje naziv meseca kao i koliko dana ima u tom mesecu te godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite godinu: 2018  
|| Unesite mesec: 1  
|| Januar, 31 dan
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite godinu: 2000  
|| Unesite mesec: 2  
|| Februar, 29 dana
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite godinu: 2018  
|| Unesite mesec: 13  
|| Greska: neispravan unos  
|| meseca.
```

[Rešenje 1.3.26]

Zadatak 1.3.27 Napisati program koji za uneti datum u formatu *dan.mesec.godina.* proverava da li je korektan.

Primer 1

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

Primer 2

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

[Rešenje 1.3.27]

Zadatak 1.3.28 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.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum:  
|| 1.1.2019.  
|| Prethodni datum:  
|| 31.12.2018.
```

[Rešenje 1.3.28]

Zadatak 1.3.29 Napisati program koji za korektno unet datum u formatu *dan.mesec.godina.* ispisuje datum narednog dana.

Primer 1

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

Primer 2

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

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum:  
|| 31.12.2008.  
|| Naredni datum:  
|| 1.1.2009.
```

[Rešenje 1.3.29]

* **Zadatak 1.3.30** Polje šahovske table se definiše parom celih brojeva (x, y) , $1 \leq x, y \leq 8$, gde je x redni broj reda, a y redni broj kolone. Napisati program koji za unete parove (k, l) i (m, n) proverava

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

1 Osnovni elementi imperativnog programiranja

(c) da li konj sa (k, l) ugrožava polje (m, n)

Prepostaviti da je polje $(1, 1)$ crno i da predstavlja donji levi ugao šahovske table. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite (k,l): 1 1  
Unesite (m,n): 2 2  
Polja su iste boje.  
Kraljica sa (1,1) ugrozava (2,2).  
Konj sa (1,1) ne ugrozava (2,2).
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite (k,l): 1 1  
Unesite (m,n): 3 2  
Polja su razlicite boje.  
Kraljica sa (1,1) ne ugrozava (3,2).  
Konj sa (1,1) ugrozava (3,2).
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite (k,l): 5 4  
Unesite (m,n): 3 3  
Polja su razlicite boje.  
Kraljica sa (5,4) ne ugrozava (3,3).  
Konj sa (5,4) ugrozava (3,3).
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite (k,l): 0 1  
Unesite (m,n): 3 9  
Greska: neispravna pozicija.
```

[Rešenje 1.3.30]

1.4 Rešenja

Rešenje 1.3.1

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int a, b, c, najmanji;
6
7     /* Ucitavanje ulaznih vrednosti. */
8     printf("Unesite tri cela broja: ");
9     scanf("%d%d%d", &a, &b, &c);
10
11    /* Inicijalizovanje najmanjeg broja na vrednost prvog broja. */
12    najmanji = a;
13
14    /* Azuriranje vrednosti minimuma u slucaju da je vrednost drugog
15       broja manja od vrednosti tekuceg minimuma. */
16    if (b < najmanji)
17        najmanji = b;
18
19    /* Ponavljanje postupka za treći broj. */
20    if (c < najmanji)
21        najmanji = c;
22
23    /* Ispis rezultata. */
24    printf("Najmanji: %d\n", najmanji);
25
26    return 0;
27}

```

Rešenje 1.3.2

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     float x, apsolutno_x;
6
7     /* Ucitavanje vrednosti broja. */
8     printf("Unesite jedan realan broj:");
9     scanf("%f", &x);
10
11    /* Racunanje apsolutne vrednosti unetog broja. */
12    apsolutno_x = x;
13    if (x < 0)
14        apsolutno_x = -x;
15
16    /* Ispis rezultata. */
17
18}

```

1 Osnovni elementi imperativnog programiranja

```
1     printf("Apsolutna vrednost: %.2f\n", apsolutno_x);
2
3     /* II nacin: koriscenjem funkcije fabs cija se deklaracija nalazi
4      u zaglavlju math.h: apsolutno_x=fabs(x); */
5
6     return 0;
7 }
```

Rešenje 1.3.3

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int x;
6     float reciprocno_x;
7
8     /* Ucitavanje vrednosti broja x. */
9     printf("Unesite jedan ceo broj:");
10    scanf("%d", &x);
11
12    /* Provera ispravnosti ulaznih podataka. Napomena: Za
13       razliku od izlaza iz programa sa kodom 0 (return 0;) koji
14       sluzi kao indikator da se program zavrsio uspesno, izlaz iz
15       programa sa izlaznim kodom razlicitim od nule sluzi kao
16       indikator da je pri izvrsavanju programa doslo do neke greske.
17 */
18    if (x == 0) {
19        printf("Greska: nedozvoljeno je deljenje nulom.\n");
20        return 1;
21    }
22
23    /* Racunanje reciprocne vrednosti. */
24    reciprocno_x = 1.0 / x;
25
26    /* Ispis rezultata. */
27    printf("Reciprocna vrednost: %.4f\n", reciprocno_x);
28
29    return 0;
30 }
```

Rešenje 1.3.4

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int a, b, c, suma;
6
7     /* Ucitavanje ulaznih vrednosti. */
8 }
```

```

8  printf("Unesite tri cela broja:");
9  scanf("%d%d%d", &a, &b, &c);
10 /* Inicijalizovanje sume na nulu. */
11 suma = 0;
12
13 /* Na sumu se dodaju vrednosti onih brojeva cija je vrednost
14 pozitivna. Uvecavanje je moguce uraditi na dva nacina:
15 I nacin: suma = suma + vrednost;
16 II nacin: suma += vrednost; */
17 if (a > 0)
18     suma = suma + a;
19
20 if (b > 0)
21     suma += b;
22
23 if (c > 0)
24     suma += c;
25
26 /* Ispis rezultata. */
27 printf("Zbir pozitivnih: %d\n", suma);
28
29 return 0;
30 }
```

Rešenje 1.3.5

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int cena1, cena2, cena3, najjeftiniji;
6     int cena_bez_popusta, cena_sa_popustom;
7
8     /* Ucitavanje vrednosti cena. */
9     printf("Unesite tri cene: ");
10    scanf("%d%d%d", &cena1, &cena2, &cena3);
11
12    /* Provera ispravnosti ulaznih podataka. */
13    if (cenai <= 0 || cena2 <= 0 || cena3 <= 0) {
14        printf("Greska: neispravan unos cene.");
15        return 1;
16    }
17
18    /* Racunanje vrednosti najjeftinijeg artikla. */
19    najjeftiniji = cena1;
20
21    if (cena2 < najjeftiniji)
22        najjeftiniji = cena2;
23
24    if (cena3 < najjeftiniji)
```

```
25     najjeftiniji = cena3;
26
27     /* Racunanje cene sa i bez popusta. */
28     cena_bez_popusta = cena1 + cena2 + cena3;
29     cena_sa_popustom = cena_bez_popusta - najjeftiniji + 1;
30
31     /* Ispis rezultata. */
32     printf("Cena sa popustom: %d din\n", cena_sa_popustom);
33     printf("Usteda: %d din\n", cena_bez_popusta - cena_sa_popustom);
34
35     return 0;
36 }
```

Rešenje 1.3.6

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int sati, minuti;
6     int preostali_sati, preostali_minuti;
7
8     /* Ucitavanje podataka o vremenu. Napomena: Vreme se zadaje u
9      formatu sat:minut. Iz tog razloga je i odgovarajuci format u
10     funkciji scanf %d:%d. */
11    printf("Unesite vreme: ");
12    scanf("%d:%d", &sati, &minuti);
13
14    /* Provera ispravnosti ulaznih podataka. */
15    if (sati > 24 || sati < 0 || minuti > 59 || minuti < 0) {
16        printf("Greska: neispravan unos vremena.\n");
17        return 1;
18    }
19
20    /* Racunanje preostalog vremena. */
21    preostali_sati = 24 - sati - 1;
22    preostali_minuti = 60 - minuti;
23
24    if (preostali_minuti == 60) {
25        /* Uvecavanje vrednosti broja za 1 se moze uraditi na vise
26         nacina. Neki od njih su:
27         broj = broj + 1;
28         broj += 1;
29         broj++; */
30        preostali_sati++;
31        preostali_minuti = 0;
32    }
33
34    /* Ispis rezultata. */
35    printf("Do ponoci: %d sati i %d minuta\n",
36           preostali_sati, preostali_minuti);
```

```

37     return 0;
38 }

```

Rešenje 1.3.7

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     unsigned int godina;
6
7     /* Ucitavanje vrednosti godine. */
8     printf("Unesite godinu:");
9     scanf("%u", &godina);
10
11    /* Provera da li je godina prestupna i ispis odgovarajuce poruke.
12       Godina je prestupna ukoliko vazi jedan od narednih uslova:
13       1. da je deljiva sa 4, a nije sa 100
14       2. da je deljiva sa 400. */
15    if ((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
16        printf("Godina je prestupna.\n");
17    else
18        printf("Godina nije prestupna.\n");
19
20    return 0;
21 }

```

Rešenje 1.3.8

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     char c;
6
7     /* Ucitavanje jednog karaktera. */
8     printf("Unesite karakter: ");
9     scanf("%c", &c);
10
11    /* Ispis karaktera i vrednosti njegovog ASCII koda. */
12    printf("Uneti karakter: %c\n", c);
13    printf("ASCII kod: %d\n", c);
14
15    /* Karakteri koji odgovaraju velikim slovima su u ASCII tablici
16       smesteni sekvencijalno. Na primer, ASCII kod karaktera 'A' je
17       65, 'B' je 66, ..., 'Z' je 90. Isto vazi i za mala slova: 'a'
18       je 97, 'b' je 98, ..., 'z' je 122.
19
20    Odavde, ako se vrsti provera da li je neki karakter veliko

```

1 Osnovni elementi imperativnog programiranja

```
22     slovo, dovoljno je proveriti da li se njegov ASCII kod nalazi
23     izmedju ASCII kodova slova 'A' i slova 'Z'.
24
25     Dodatno, moze se primetiti da je razlika izmedju ASCII koda
26     svakog malog i odgovarajuceg velikog slova konstanta koja ima
27     vrednost 'a'-'A', sto je isto sto i 'b'-'B', itd. Zbog toga,
28     ako je potrebno od velikog slova dobiti malo, onda je
29     dovoljno ASCII kodu velikog slova dodati pomenutu konstantu.
30     Za mala slova, vazi obrnuto - da bi se dobilo veliko slovo,
31     ova konstanta se oduzima. */
32
33     if (c >= 'A' && c <= 'Z') {
34         printf("Odgovarajuće malo slovo: %c\n", c + ('a' - 'A'));
35         printf("ASCII kod: %d\n", c + ('a' - 'A'));
36     }
37
38     if (c >= 'a' && c <= 'z') {
39         printf("Odgovarajuće veliko slovo: %c\n", c - ('a' - 'A'));
40         printf("ASCII kod: %d\n", c - ('a' - 'A'));
41     }
42
43     return 0;
44 }
```

Rešenje 1.3.9

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int broj_cifara = 0;
6     unsigned int proizvod_cifara = 1;
7     int c1, c2, c3;
8
9     /* I nacin ucitavanja ulaza: koriscenjem funkcije getchar()
10      Funkcija getchar cita jedan karakter sa ulaza i vraca njegov
11      ASCII kod. Napomena: Razmaci su takodje karakteri i nece
12      automatski biti preskoceni. Iz tog razloga se getchar poziva 5
13      puta u ovom primeru. Posto je poznato da su drugi i cetvrti
14      karakter blanko znaci, nema potrebe da se cuva povratna
15      vrednost tih poziva. */
16     printf("Unesite karaktere: ");
17     c1 = getchar();
18     getchar();
19     c2 = getchar();
20     getchar();
21     c3 = getchar();
22
23     /* II nacin ucitavanja ulaza: koriscenjem funkcije scanf()
24      Blanko znaci se navode kao deo ocekivanog formata ulaza.
25      char c1, c2, c3;
```

```

1    scanf("%c %c %c", &c1, &c2, &c3); /*

27   /* Pogresan nacin ucitavanja ulaza:
28   scanf("%c%c%c", &c1, &c2, &c3);
29   U ovom slucaju ce u c1 biti upisan prvi karakter, u c2
30   blanko i u c3 drugi karakter. */

33   /* Karakteri koji predstavljaju cifre su u ASCII tablici takodje
34   smesteni sekvencijalno. Na primer, '0' ima ASCII kod 48, '1'
35   49, ..., '9' ima ASCII kod 57.

37   Odavde, ako se proverava da li je karakter cifra, dovoljno je
38   proveriti da li se njegov ASCII kod nalazi izmedju '0' i '9'.

39   Dodatno, ako je potrebno izracunati dekadnu vrednost karaktera
40   koji je cifra, dovoljno je od ASCII koda tog karaktera,
41   oduzeti ASCII kod karaktera '0'. Na primer, '4'-'0' = 52 - 48
42   = 4. */

45   /* Racunanje proizvoda onih karaktera koji su cifre. */
46   if (c1 >= '0' && c1 <= '9') {
47       proizvod_cifara *= (c1 - '0');
48       broj_cifara++;
49   }

51   if (c2 >= '0' && c2 <= '9') {
52       proizvod_cifara *= (c2 - '0');
53       broj_cifara++;
54   }

55   if (c3 >= '0' && c3 <= '9') {
56       proizvod_cifara *= (c3 - '0');
57       broj_cifara++;
58   }

61   /* Ispis rezultata. */
62   if (broj_cifara == 0)
63       printf("Medju unetim karakterima nema cifara.\n");
64   else
65       printf("Proizvod cifara: %u\n", proizvod_cifara);

67   return 0;
}

```

Rešenje 1.3.10

```

1 #include <stdio.h>
2 #include <ctype.h>
3
4 int main() {
5     /* Deklaracija potrebnih promenljivih. */

```

1 Osnovni elementi imperativnog programiranja

```
1 int c1, c2, c3;
2
3 /* Ucitavanje sifre artikla. */
4 printf("Unesite sifru: ");
5 c1 = getchar();
6 c2 = getchar();
7 c3 = getchar();
8
9 /* Funkcije islower, isupper i isdigit proveravaju da li je
10 prosledjeni karakter malo slovo, veliko slovo ili cifra.
11 Deklaracije ovih funkcija se nalaze u zaglavlju ctype.h.
12
13 Ukoliko prvi karakter nije ni malo slovo ni veliko slovo, ni
14 cifra, ispisuje se odgovarajuca poruka o gresci i izlazi se
15 iz programa. */
16 if (!islower(c1) && !isupper(c1) && !isdigit(c1)) {
17     printf("Greska: %c je neispravan karakter.\n", c1);
18     return 1;
19 }
20
21 /* Postupak se ponavlja za druga dva karaktera. */
22 if (!islower(c2) && !isupper(c2) && !isdigit(c2)) {
23     printf("Greska: %c je neispravan karakter.\n", c2);
24     return 1;
25 }
26
27 if (!islower(c3) && !isupper(c3) && !isdigit(c3)) {
28     printf("Greska: %c je neispravan karakter.\n", c3);
29     return 1;
30 }
31
32 /* Funkcija tolower(c) radi sledece: ako je c veliko slovo, kao
33 povratnu vrednost vraca odgovarajuce malo slovo, u suprotnom
34 vraca c. Dakle, tolower('A') je 'a', a tolower('6') = '6',...
35
36 Slicno, samo obrnuto, radi i funkcija toupper(c). Deklaracije
37 ovih funkcija se, takodje, nalaze u zaglavlju ctype.h. */
38 c1 = tolower(c1);
39 c2 = tolower(c2);
40 c3 = tolower(c3);
41
42 /* Ispis rezultata. */
43 printf("Rezultat: %c%c%c\n", c1, c2, c3);
44
45 return 0;
46 }
```

Rešenje 1.3.11

```
1 #include <stdio.h>
2 #include <stdlib.h>
```

```

3 int main() {
5  /* Deklaracije potrebnih promenljivih. */
6  int n;
7  char jedinica, desetica, stotina, hiljada, najveca_cifra;
8
9  /* Ucitavanje vrednosti broja n. */
10 printf("Unesite cetvorocifreni broj: ");
11 scanf("%d", &n);
12
13 /* Da bi program radio ispravno i za negativne brojeve, koristi
14    se apsolutna vrednost broja n. */
15 n = abs(n);
16
17 /* Provera ispravnosti ulaznih podataka. */
18 if (n < 1000 || n > 9999) {
19    printf("Greska: niste uneli cetvorocifreni broj.\n");
20    return 1;
21 }
22
23 /* Izdvajanje cifara broja n. */
24 jedinica = n % 10;
25 desetica = (n / 10) % 10;
26 stotina = (n / 100) % 10;
27 hiljada = n / 1000;
28
29 /* Racunanje najvece cifra broja n. */
30 najveca_cifra = jedinica;
31
32 if (desetica > najveca_cifra)
33    najveca_cifra = desetica;
34
35 if (stotina > najveca_cifra)
36    najveca_cifra = stotina;
37
38 if (hiljada > najveca_cifra)
39    najveca_cifra = hiljada;
40
41 /* Ispis rezultata. */
42 printf("Najveca cifra je: %d\n", najveca_cifra);
43
44 return 0;
45 }
```

Rešenje 1.3.12

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5  /* Deklaracije potrebnih promenljivih. */
6 }
```

```
6 int n;
7 char jedinica, desetica, stotina;
8
9 /* Ucitavanje vrednosti broja n. */
10 printf("Unesite pozitivan trocifreni broj: ");
11 scanf("%d", &n);
12
13 /* Provera ispravnosti ulaznih podataka. */
14 if (n < 100 || n > 999) {
15     printf("Greska: niste uneli pozitivan trocifreni broj.\n");
16     return 1;
17 }
18
19 /* Izdvajanje cifara broja n. */
20 jedinica = n % 10;
21 desetica = (n / 10) % 10;
22 stotina = n / 100;
23
24 /* Ispis rezultata. */
25 if (n == jedinica * jedinica * jedinica +
26     desetica * desetica * desetica + stotina * stotina * stotina)
27     printf("Broj je Armstrongov.\n");
28 else
29     printf("Broj nije Armstrongov.\n");
30
31 return 0;
32 }
```

Rešenje 1.3.13

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int n, broj_parnih, proizvod_parnih;
7     char jedinica, desetica, stotina, hiljada;
8
9     /* Ucitavanje vrednosti broja n. */
10    printf("Unesite cetvorocifreni broj: ");
11    scanf("%d", &n);
12
13    /* Da bi program radio ispravno i za negativne vrednosti,
14       koristi se apsolutna vrednost broja n. */
15    n = abs(n);
16
17    /* Provera ispravnosti ulaznih podataka. */
18    if (n < 1000 || n > 9999) {
19        printf("Greska: niste uneli cetvorocifreni broj.\n");
20        return 1;
21    }
```

```

23  /* Izdvajanje cifara broja n. */
24  jedinica = n % 10;
25  desetica = (n / 10) % 10;
26  stotina = (n / 100) % 10;
27  hiljada = n / 1000;

28  /* Inicijalizacija brojaca i rezultata. */
29  broj_parnih = 0;
30  proizvod_parnih = 1;

31  /* Za svaku cifru se vrši provjeru da li je parna i ukoliko jeste
32      tekuci rezultat se mnozi tekucom cifrom. */
33  if (jedinica % 2 == 0) {
34      proizvod_parnih = proizvod_parnih * jedinica;
35      broj_parnih++;
36  }

37  if (desetica % 2 == 0) {
38      proizvod_parnih = proizvod_parnih * desetica;
39      broj_parnih++;
40  }

41  if (stotina % 2 == 0) {
42      proizvod_parnih = proizvod_parnih * stotina;
43      broj_parnih++;
44  }

45  if (hiljada % 2 == 0) {
46      proizvod_parnih = proizvod_parnih * hiljada;
47      broj_parnih++;
48  }

49  /* Ispis rezultata. */
50  if (broj_parnih == 0)
51      printf("Nema parnih cifara.\n");
52  else
53      printf("Proizvod parnih cifara: %d\n", proizvod_parnih);

54
55  return 0;
56 }

```

Rešenje 1.3.14

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int n, n_abs, rezultat;
7     char jedinica, desetica, stotina, hiljada;

```

1 Osnovni elementi imperativnog programiranja

```
int najveca, najmanja, stepen_najvece, stepen_najmanje;
9
10 /* Ucitavanje vrednosti broja n. */
11 printf("Unesite cetvorocifreni broj: ");
12 scanf("%d", &n);
13
14 /* Da bi program radio ispravno i za negativne vrednosti,
15   koristi se apsolutna vrednost broja n. */
16 n_abs = abs(n);
17
18 /* Provera ispravnosti ulaznih podataka. */
19 if (n_abs < 1000 || n_abs > 9999) {
20     printf("Greska: niste uneli cetvorocifreni broj.\n");
21     return 1;
22 }
23
24 /* Izdvajanje cifara broja n. */
25 jedinica = n_abs % 10;
26 desetica = (n_abs / 10) % 10;
27 stotina = (n_abs / 100) % 10;
28 hiljada = n_abs / 1000;
29
30 /* Po algoritmu za trazenje najvece/najmanje cifre (koji je
31   prikazan u zadatu 2.1.11) pronalaze se najveca i najmanja
32   cifra broja n, kao i pozicije na kojoj se one nalaze.
33   Radi lakseg izracunavanja, pozicija se pamti kao stepen broja
34   10. Na primer, pozicija cifre jedinica je 1, cifre desetica
35   10, itd... */
36 najveca = jedinica;
37 stepen_najvece = 1;
38
39 if (desetica > najveca) {
40     najveca = desetica;
41     stepen_najvece = 10;
42 }
43
44 if (stotina > najveca) {
45     najveca = stotina;
46     stepen_najvece = 100;
47 }
48
49 if (hiljada > najveca) {
50     najveca = hiljada;
51     stepen_najvece = 1000;
52 }
53
54 /* Racunanje najmanje cifre. */
55 najmanja = jedinica;
56 stepen_najmanje = 1;
57
58 if (desetica < najmanja) {
59     najmanja = desetica;
```

```

61     stepen_najmanje = 10;
62 }
63
64 if (stotina < najmanja) {
65     najmanja = stotina;
66     stepen_najmanje = 100;
67 }
68
69 if (hiljada < najmanja) {
70     najmanja = hiljada;
71     stepen_najmanje = 1000;
72 }
73
74 /* Ideja: U broju 4179, najmanja cifra je 1 i njen stepen je 100,
75    a najveća cifra je 9 i njen stepen je 1. Zamena mesta se vrši
76    tako što se oduzme 9 i doda 1, a zatim oduzme 100 i doda 900. */
77 rezultat = n_abs - najveća * stepen_najveće
78     + najmanja * stepen_najveće - najmanja * stepen_najmanje
79     + najveća * stepen_najmanje;
80
81 /* Ako je početni broj bio negativan i rezultat treba da bude
82    negativan. */
83 if(n < 0)
84     rezultat = -rezultat;
85
86 /* Ispis rezultata. */
87 printf("Rezultat: %d\n", rezultat);
88
89 return 0;
90 }
```

Rešenje 1.3.15

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int n;
7     char jedinica, desetica, stotina, hiljada;
8
9     /* Ucitavanje vrednosti broja n. */
10    printf("Unesite cetvorocifreni broj: ");
11    scanf("%d", &n);
12
13    /* Da bi program radio ispravno i za negativne vrednosti,
14       koristi se apsolutna vrednost broja n. */
15    n = abs(n);
16
17    /* Provera ispravnosti ulaznih podataka. */
18    if (n < 1000 || n > 9999) {
```

1 Osnovni elementi imperativnog programiranja

```
19     printf("Greska: niste uneli cetvorocifreni broj.\n");
20     return 1;
21 }

23 /* Izdvajanje cifara broja n. */
24 jedinica = n % 10;
25 desetica = (n / 10) % 10;
26 stotina = (n / 100) % 10;
27 hiljada = n / 1000;

29 /* Ispis rezultata. */
30 if (hiljada <= stotina && stotina <= desetica &&
31     desetica <= jedinica)
32     printf("Cifre su uredjene neopadajuce. \n");
33 else if (hiljada >= stotina && stotina >= desetica &&
34         desetica >= jedinica)
35     printf("Cifre su uredjene nerastuce. \n");
36 else
37     printf("Cifre nisu uredjene.\n");

38
39 return 0;
}
```

Rešenje 1.3.16

```
#include <stdio.h>

2 int main() {
3     /* Deklaracija potrebnih promenljivih. */
4     float xa, ya, xb, yb;
5
6     /* Ucitavanje koordinata tacaka A i B. */
7     printf("Unesite koordinate tacke A: ");
8     scanf("%f%f", &xa, &ya);
9
10    printf("Unesite koordinate tacke B: ");
11    scanf("%f%f", &xb, &yb);
12
13    /* Provera da li su obe tacke u istom kvadrantu i ispisi
14       odgovarajuce poruke. */
15    if ((xa >= 0 && ya >= 0 && xb >= 0 && yb >= 0) ||
16        (xa <= 0 && ya >= 0 && xb <= 0 && yb >= 0) ||
17        (xa >= 0 && ya <= 0 && xb >= 0 && yb <= 0) ||
18        (xa <= 0 && ya <= 0 && xb <= 0 && yb <= 0))
19        printf("Tacke se nalaze u istom kvadrantu.\n");
20    else
21        printf("Tacke se ne nalaze u istom kvadrantu.\n");
22
23    return 0;
24 }
```

Rešenje 1.3.17

```

1 #include <stdio.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     float xa, ya, xb, yb, xc, yc;
7     float k, n;
8
9     /* Ucitavanje koordinata tacaka A, B i C. */
10    printf("Unesite koordinate tacke A: ");
11    scanf("%f%f", &xa, &ya);
12
13    printf("Unesite koordinate tacke B: ");
14    scanf("%f%f", &xb, &yb);
15
16    printf("Unesite koordinate tacke C: ");
17    scanf("%f%f", &xc, &yc);
18
19    /* Ako su bilo koje dve tacke jednake, onda se sigurno sve tri
20       nalaze na jednoj pravoj. */
21    if ((xa == xb && ya == yb) ||
22        (xa == xc && ya == yc) || (xb == xc && yb == yc)) {
23        printf("Tacke se nalaze na istoj pravoj.\n");
24        return 0;
25    }
26
27    /* Odredjivanje koeficijenta pravca k i odsecka na y osi n prave
28       y = k*x + n koja prolazi kroz tacke A i B. Napomena: U
29       slucaju kada je xb jednako xa, ova prava je paralelna sa y
30       osom pa k ima vrednost beskonacno, a n vrednost 0, tj.
31       jednacina prave je x = xa (sto je isto sto i x = xb). Da bi se
32       izbeglo deljenje nulom (xb-xa), ovaj slucaj se posebno
33       obradjuje. */
34    if (xb != xa) {
35        k = (yb - ya) / (xb - xa);
36        n = ya - k * xa;
37
38        /* Provera da li tacka C pripada pravoj y=k*x + n na
39           kojoj se vec nalaze tacke A i B. */
40        if (yc == k * xc + n)
41            printf("Tacke se nalaze na istoj pravoj.\n");
42        else
43            printf("Tacke se ne nalaze na istoj pravoj.\n");
44    } else {
45        /* Provera da li se i tacka C nalazi na pravoj x = xb. */
46        if (xc == xb)
47            printf("Tacke se nalaze na istoj pravoj.\n");
48        else
49            printf("Tacke se ne nalaze na istoj pravoj.\n");
50
51    /* II nacin: Tacke su kolinearne ako je
52       (yb - ya)*(xc - xb) == (yb - ya)*(xb - xa)
53       (yb - ya)*(xc - xb) == (yb - ya)*(xc - xb)
54       (yb - ya)*(xc - xb) == (yb - ya)*(xc - xb)
55   */
56 }
```

1 Osnovni elementi imperativnog programiranja

```
51 |xa ya 1 |
|xb yb 1 | = 0
53 |xc yc 1 |
odnosno, ako je
55 xa*yb + ya*xc + xb*yc - ya*xb - xa*yc - yb*xc = 0

57 if(xa*yb + ya*xc + xb*yc - ya*xb - xa*yc - yb*xc == 0)
    printf("Tacke se nalaze na istoj pravoj. \n");
59 else
    printf("Tacke se ne nalaze na istoj pravoj. \n"); */

61 return 0;
63 }
```

Rešenje 1.3.18

```
#include <stdio.h>

2 int main() {
4 /* Deklaracija potrebnih promenljivih. */
5     int a1, a2, b1, b2;
6
8 /* Ucitavanje granica intervala. */
9 printf("Unesite a1, b1, a2 i b2: ");
10 scanf("%d%d%d%d", &a1, &b1, &a2, &b2);
12
13 /* Racunanje i ispis trazenih vrednosti (u zavisnosti od
14 razlicitih polozenja dva intervala). */
15 if (a1 <= a2 && b1 >= a2) {
16     /* I slucaj: Intervali se sekut u [a1,b1] je pre [a2,b2]. */
17     printf("Duzina preseka:: %d\n", b1 - a2);
18     printf("Presecni interval: [%d, %d]\n", a2, b1);
19     printf("Duzina koju pokrivaju: %d\n", b2 - a1);
20     printf("Najmanji interval: [%d, %d]\n", a1, b2);
21 } else if (a2 <= a1 && b2 >= a1) {
22     /* II slucaj: Intervali se sekut u [a2,b2] je pre [a1,b1]. */
23     printf("Duzina preseka:: %d\n", b2 - a1);
24     printf("Presecni interval: [%d, %d]\n", a1, b2);
25     printf("Duzina koju pokrivaju: %d\n", b1 - a2);
26     printf("Najmanji interval: [%d, %d]\n", a2, b1);
27 } else if (a1 >= a2 && b1 <= b2) {
28     /* III slucaj: Interval [a1,b1] se nalazi unutar [a2,b2]. */
29     printf("Duzina preseka:: %d\n", b1 - a1);
30     printf("Presecni interval: [%d, %d]\n", a1, b1);
31     printf("Duzina koju pokrivaju: %d\n", b2 - a2);
32     printf("Najmanji interval: [%d, %d]\n", a2, b2);
33 } else if (a2 >= a1 && b2 <= b1) {
34     /* IV slucaj: Interval [a2,b2] se nalazi unutar [a1,b1]. */
35     printf("Duzina preseka:: %d\n", b2 - a2);
36     printf("Presecni interval: [%d, %d]\n", a2, b2);
37     printf("Duzina koju pokrivaju: %d\n", b1 - a1);
```

```

36     printf("Najmanji interval: [%d, %d]\n", a1, b1);
37 } else {
38     /* V slučaj: Intervali su disjunktni. */
39     printf("Duzina preseka:: 0\n");
40     printf("Presecni interval: prazan\n");
41     printf("Duzina koju pokrivaju: %d\n", b1 - a1 + b2 - a2);
42     if (a1 < a2)
43         printf("Najmanji interval: [%d, %d]\n", a1, b2);
44     else
45         printf("Najmanji interval: [%d, %d]\n", a2, b1);
46 }
47
48 return 0;
}

```

Rešenje 1.3.19

```

1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     /* Deklaracija potrebnih promenljivih. */
6     float a, b, c, D;
7
8     /* Ucitavanje koeficijenata kvadratne jednacine. */
9     printf("Unesite koeficijente A, B i C:");
10    scanf("%f%f%f", &a, &b, &c);
11
12    /* Racunanje resenja jednacine u zavisnosti od vrednosti
13       koeficijenata a, b i c i ispis rezultata. */
14    if (a == 0) {
15        if (b == 0) {
16            if (c == 0) {
17                /* Slučaj a==0 && b==0 && c==0: beskonacno mnogo resenja. */
18                printf("Jednacina ima beskonacno mnogo resenja\n");
19            } else {
20                /* Slučaj a==0 && b==0 && c!=0: nema resenja. */
21                printf("Jednacina nema resenja\n");
22            }
23        } else {
24            /* Slučaj a=0 && b!=0: jedinstveno resenje. */
25            printf("Jednacina ima jedinstveno realno resenje %.2f\n",
26                   -c / b);
27        }
28    } else {
29        /* Slučaj a != 0: Racunanje diskriminante. */
30        D = b * b - 4 * a * c;
31
32        /* U zavisnosti od vrednosti diskriminante, ispisuje se
33           rezultat. */
34        if (D < 0) {

```

1 Osnovni elementi imperativnog programiranja

```
35     printf("Jednacina nema realnih resenja\n");
36 } else if (D > 0) {
37     printf("Jednacina ima dva realna resenja %.2f i %.2f\n",
38           (-b + sqrt(D)) / (2 * a), (-b - sqrt(D)) / (2 * a));
39 } else {
40     printf("Jednacina ima jedinstveno realno resenje %.2f\n",
41           -b / (2 * a));
42 }
43 }

44     return 0;
45 }
```

Rešenje 1.3.20

```
1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int k, broj;

7     /* Ucitavanje trazene pozicije. */
8     printf("Unesite k: ");
9     scanf("%d", &k);

11    /* Provera ispravnosti ulaznih podataka. */
12    if (k < 1 || k > 189) {
13        printf("Greska: neispravan unos pozicije.\n");
14        return 1;
15    }

17    /* Racunanje rezultata. */
18    if (k < 10) {
19        /* I slucaj: Trazi se jednocifreni broj. */
20        printf("Na %d-toj poziciji je broj %d.\n", k, k);
21    } else {
22        /* II slucaj: Trazi se dvocifreni broj. */

23        /* Ideja: izracunati broj na koji pokazuje pozicija k. Zatim,
24         ako je k parno, uzeti cifru desetica tog broja, a ako je k
25         neparno, uzeti cifru jedinica tog broja.

26         Na primer, za k=14 i k=15, broj koji se nalazi na ovim
27         pozicijama je 12, pa u slucaju da je k=14, treba ispisati 1,
28         a u slucaju da je k=15, treba ispisati 2. */

29    }

30    /* Odredjivanje odgovarajuceg broja: Kada bi niz izgledao
31     10111213...9899, za dato k, broj bi se dobio kao 9 + k/2 + 1
32     za neparne vrednosti k, odnosno 9 + k/2 za parne (dodaje se
33     vrednost deset jer je prvi broj u nizu desetka). Na primer:
34     k=1, broj = 9 + 1/2 + 1 = 9 + 0 + 1 = 10 k=2, broj = 9 + 2/2
```

```

37     = 10 k=3, broj = 9 + 3/2 + 1 = 9 + 1 + 1 = 11 k=4, broj = 9
39     + 4/2 = 11 ... Posto ovde postoji i 9 pozicija ispred,
41     potrebno je i njih uzeti u obzir. Odatle je broj = 9 +
42     (k-9)/2 + 1 za neparne vrednosti k, odnosno broj = 9 +
43     (k-9)/2 za parne vrednosti k. */
44     if (k % 2 != 0) {
45         broj = 9 + (k - 9) / 2;
46         printf("Na %d-toj poziciji je broj %d.\n", k, broj % 10);
47     } else {
48         broj = 9 + (k - 9) / 2 + 1;
49         printf("Na %d-toj poziciji je broj %d.\n", k, broj / 10);
50     }
51 }

return 0;
}

```

Rešenje 1.3.21

```

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

4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     float x, Fx;
7     int k;

8     /* Ucitavanje vrednosti x i k. */
9     printf("Unesite redom x i k: ");
10    scanf("%f %d", &x, &k);

12    /* Provera ispravnosti ulaznih podataka. */
13    if (k < 1 || k > 3) {
14        printf("Greska: nedozvoljena vrednost za k.\n");
15        return 0;
16    }

18    /* U zavisnosti od vrednosti k, data funkcija ce se izracunati
19     jednom, dva puta ili tri puta. */
20    Fx = 2 * cos(x) - x * x * x;
21    if (k > 1)
22        Fx = 2 * cos(Fx) - Fx * Fx * Fx;
23    if (k > 2)
24        Fx = 2 * cos(Fx) - Fx * Fx * Fx;

26    /* Ispis rezultata. Napomena: Ispis realnih brojeva sa %g
27     rezultuje ispisom na onaj broj decimala koliko sam broj ima.
28     Dakle, broj 1 ce se ispisati kao 1, broj 2.33 kao 2.33, broj
29     0.9999 kao 0.9999. */
30    printf("F(%g, %d) = %.2f\n", x, k, Fx);
31
32

```

```
    return 0;  
34 }
```

Rešenje 1.3.22

```
1 #include <stdio.h>  
  
3 int main() {  
4     /* Deklaracija potrebnih promenljivih. */  
5     int dan;  
  
7     /* Ucitavanje rednog broja dana u nedelji. */  
8     printf("Unesite broj: ");  
9     scanf("%d", &dan);  
  
11    /* I nacin: Koriscenjem if-else naredbe.  
12    if(dan == 1)  
13        printf("ponedeljak\n");  
14    else if(dan == 2)  
15        printf("utorak\n");  
16    else if(dan == 3)  
17        printf("sreda\n");  
18    else if(dan == 4)  
19        printf("cetvrtak\n");  
20    else if(dan == 5)  
21        printf("petak\n");  
22    else if(dan == 6)  
23        printf("subota\n");  
24    else if(dan == 7)  
25        printf("nedelja\n");  
26    else  
27        printf("Greska: neispravan unos dana.\n"); */  
  
29    /* II nacin: Koriscenjem switch naredbe.*/  
30    switch (dan) {  
31        case 1:  
32            /* Ako dan ima vrednost 1, ispisuje se ponedeljak. */  
33            printf("ponedeljak\n");  
  
34            /* Ako se naredba break ne navede, izvrsice se i sledeca  
35                naredba, tj. ispis ce biti "ponedeljak utorak". */  
36            break;  
37        case 2:  
38            /* Postupak se ponavlja i za ostale dane. */  
39            printf("utorak\n");  
40            break;  
41        case 3:  
42            printf("sreda\n");  
43            break;  
44        case 4:  
45            printf("cetvrtak\n");
```

```

47     break;
48 case 5:
49     printf("petak\n");
50     break;
51 case 6:
52     printf("subota\n");
53     break;
54 case 7:
55     printf("nedelja\n");
56     break;
57 default:
58     /* Ako vrednost promenljive dan nije ni jedna od vrednosti
59      izmedju 1 i 7, onda je uneta vrednost neispravna. */
60     printf("Greska: neispravan unos dana.\n");
61 }
62
63 return 0;
}

```

Rešenje 1.3.23

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     char c;
6
7     /* Ucitavanje jednog karaktera. */
8     printf("Unesite jedan karakter:");
9     scanf("%c", &c);
10
11    /* Proverava se da li je karakter c samoglasnik, tj. da li
12       odgovara nekom od sledećih karaktera: A,E,I,O,U,a,e,i,o,u. */
13    switch (c) {
14        case 'A':
15        case 'E':
16        case 'I':
17        case 'O':
18        case 'U':
19        case 'a':
20        case 'e':
21        case 'i':
22        case 'o':
23        case 'u':
24            printf("Uneti karakter je samoglasnik.\n");
25            break;
26        default:
27            printf("Uneti karakter nije samoglasnik.\n");
28            break;
29    }
}

```

```
31     return 0;  
32 }
```

Rešenje 1.3.24

```
1 #include <stdio.h>  
  
3 int main() {  
4     /* Deklaracije potrebnih promenljivih. */  
5     char op;  
6     int x, y;  
7  
8     /* Ucitavanje izraza. */  
9     printf("Unesite izraz: ");  
10    scanf("%d %c %d", &x, &op, &y);  
11  
12    /* Racunanje vrednosti izraza u zavisnosti od unete operacije. */  
13    switch (op) {  
14        case '+':  
15            printf("Rezultat je: %d\n", x + y);  
16            break;  
17        case '-':  
18            printf("Rezultat je: %d\n", x - y);  
19            break;  
20        case '*':  
21            printf("Rezultat je: %d\n", x * y);  
22            break;  
23        case '/':  
24            if (y == 0)  
25                printf("Greska: deljenje nulom.\n");  
26            else  
27                printf("Rezultat je: %d\n", x / y);  
28            break;  
29        case '%':  
30            printf("Rezultat je: %d\n", x % y);  
31            break;  
32        default:  
33            printf("Greska: nepoznat operator.\n");  
34    }  
35  
36    return 0;  
37 }
```

Rešenje 1.3.25

```
1 #include <stdio.h>  
  
3 int main() {  
4     /* Deklaracija potrebnih promenljivih. */  
5     int dan, mesec;
```

```
7  /* Ucitavanje datuma koji je zadat u formatu: dan.mesec. */
8  printf("Unesite dan i mesec");
9  scanf("%d.%d.", &dan, &mesec);

11 /* Racunanje godisnjeg doba. */
12 switch (mesec) {
13     case 1:
14     case 2:
15         /* Ako je mesec januar ili februar, onda je sigurno u pitanju
16             zima. */
17         printf("zima\n");
18         break;
19     case 3:
20         /* Ako je mesec mart, onda se godisnje doba odredjuje u
21             zavisnosti od dana u mesecu. */
22         if (dan < 21)
23             printf("zima\n");
24         else
25             printf("prolece\n");
26         break;
27     case 4:
28     case 5:
29         /* Ako je mesec april ili maj, onda je sigurno u pitanju
30             prolece. */
31         printf("prolece\n");
32         break;
33     case 6:
34         /* Ako je mesec jun, onda se godisnje doba odredjuje u
35             zavisnosti od dana u mesecu. */
36         if (dan < 21)
37             printf("prolece\n");
38         else
39             printf("leto\n");
40         break;
41     case 7:
42     case 8:
43         /* Ako je mesec jul ili avgust, onda je sigurno u pitanju
44             leto. */
45         printf("leto\n");
46         break;
47     case 9:
48         /* Ako je mesec septembar, onda se godisnje doba odredjuje u
49             zavisnosti od dana u mesecu. */
50         if (dan < 23)
51             printf("leto\n");
52         else
53             printf("jesen\n");
54         break;
55     case 10:
56     case 11:
57         /* Ako je mesec oktobar ili novembar, onda je sigurno u pitanju
```

```
        jesen. */
59     printf("jesen\n");
60     break;
61 case 12:
62     /* Ako je mesec decembar, onda se godisnje doba odreduje u
63      zavisnosti od dana u mesecu. */
64     if (dan < 22)
65         printf("jesen\n");
66     else
67         printf("zima\n");
68 }
69
70 return 0;
71 }
```

Rešenje 1.3.26

```
1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int godina, mesec, prestupna;

7     /* Ucitavanje vrednosti godine. */
8     printf("Unesite godinu: ");
9     scanf("%d", &godina);

11    /* Provera ispravnosti ulaznih podataka. */
12    if (godina < 0) {
13        printf("Greska: neispravan unos godine.\n");
14        return 1;
15    }

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

23    /* Ucitavanje rednog broja meseca. */
24    printf("Unesite redni broj meseca: ");
25    scanf("%d", &mesec);

27    /* Ispis rezultata u zavisnosti od vrednosti meseca. */
28    switch (mesec) {
29    case 1:
30        printf("Januar, 31 dan\n");
31        break;
32    case 2:
33        if (prestupna)
34            printf("Februar, 29 dana\n");
35    }
```

```

35     else
36         printf("Februar, 28 dana\n");
37     break;
38 case 3:
39     printf("Mart, 31 dan\n");
40     break;
41 case 4:
42     printf("April, 30 dana\n");
43     break;
44 case 5:
45     printf("Maj, 31 dan\n");
46     break;
47 case 6:
48     printf("Jun, 30 dana\n");
49     break;
50 case 7:
51     printf("Jul, 31 dan\n");
52     break;
53 case 8:
54     printf("Avgust, 31 dan\n");
55     break;
56 case 9:
57     printf("Septembar, 30 dana\n");
58     break;
59 case 10:
60     printf("Oktobar, 31 dan\n");
61     break;
62 case 11:
63     printf("Novembar, 30 dana\n");
64     break;
65 case 12:
66     printf("Decembar, 31 dan\n");
67     break;
68 default:
69     printf("Greska: neispravan unos meseca.\n");
70     return 1;
71 }

72 return 0;
}

```

Rešenje 1.3.27

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int dan, mesec, godina, dozvoljeni_broj_dana;

7     /* Ucitavanje datuma. */
8     printf("Unesite datum: ");

```

1 Osnovni elementi imperativnog programiranja

```
9    scanf("%d.%d.%d", &dan, &mesec, &godina);

11   /* Provera korektnosti vrednosti unete godine. */
12   if (godina < 0) {
13       printf("Datum nije korektan.\n");
14       return 0;
15   }

17   /* Provera korektnosti vrednosti unetog meseca. */
18   if (mesec < 1 || mesec > 12) {
19       printf("Datum nije korektan.\n");
20       return 0;
21   }

23   /* Provera korektnosti vrednosti unetog dana. */
24   switch (mesec) {
25       case 1:
26       case 3:
27       case 5:
28       case 7:
29       case 8:
30       case 10:
31       case 12:
32           /* Dozvoljeni broj dana za januar, mart, maj, jul, avgust,
33              oktobar i decembar je 31. */
34           dozvoljeni_broj_dana = 31;
35           break;
36       case 2:
37           /* Dozvoljeni broj dana za februar je 28 ili 29 u zavisnosti od
38              toga da li je godina prestupna ili ne. */
39           if ((godina % 4 == 0 && godina % 100 != 0) || godina % 400 == 0)
40               dozvoljeni_broj_dana = 29;
41           else
42               dozvoljeni_broj_dana = 28;
43           break;
44       case 4:
45       case 6:
46       case 9:
47       case 11:
48           /* Dozvoljeni broj dana za april, jun, septembar i novembar je
49              30. */
50           dozvoljeni_broj_dana = 30;
51           break;
52   }

53   if (dan < 0 || dan > dozvoljeni_broj_dana) {
54       printf("Datum nije korektan.\n");
55       return 0;
56   }

57   /* Kako su sve provere korektnosti prosle, datum se smatra
      korektnim. */
```

```

61     printf("Datum je korektan.\n");
63
64 }
```

Rešenje 1.3.28

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int dan, mesec, godina;
6     int prethodni_dan, prethodni_mesec, prethodni_godina;
7
8     /* Ucitavanje datuma. */
9     printf("Unesite datum: ");
10    scanf("%d.%d.%d.", &dan, &mesec, &godina);
11
12    /* Racunanje dana, meseca i godine prethodnog dana. */
13    prethodni_dan = dan - 1;
14    prethodni_mesec = mesec;
15    prethodni_godina = godina;
16
17    /* Ako je potrebno, vrse se korekcije. */
18    if (prethodni_dan == 0) {
19        prethodni_mesec = mesec - 1;
20        if (prethodni_mesec == 0) {
21            prethodni_mesec = 12;
22            prethodni_godina = godina - 1;
23        }
24
25        switch (prethodni_mesec) {
26            case 1:
27            case 3:
28            case 5:
29            case 7:
30            case 8:
31            case 10:
32            case 12:
33                prethodni_dan = 31;
34                break;
35            case 2:
36                if ((prethodni_godina % 4 == 0 && prethodni_godina % 100 != 0)
37                    || prethodni_godina % 400 == 0)
38                    prethodni_dan = 29;
39                else
40                    prethodni_dan = 28;
41                break;
42            case 4:
43            case 6:
44            case 9:
```

```
45     case 11:
46         prethodni_dan = 30;
47     }
48 }
49
50 /* Ispis rezultata.*/
51 printf("Prethodni datum: %d.%d.%d.\n",
52       prethodni_dan, prethodni_mesec, prethodni_godina);
53
54 return 0;
55 }
```

Rešenje 1.3.29 Pogledajte zadatak 1.3.28.

Rešenje 1.3.30

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

int main() {
    /* Deklaracija potrebnih promenljivih. */
    int k, l, m, n;

    /* Ucitavanje vrednosti pozicija na tabli. */
    printf("Unesite (k,l): ");
    scanf("%d%d", &k, &l);

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

    /* Provera ispravnosti ulaznih podataka. */
    if (k < 1 || k > 8 || l < 1 || l > 8 ||
        m < 1 || m > 8 || n < 1 || n > 8) {
        printf("Greska: neispravna pozicija.\n");
        return 1;
    }

    if(k == m && l == n){
        printf("Greska: pozicije moraju biti razlicite.\n");
        return 1;
    }

    /* Provera da li su (k,l) i (m,n) iste boje. Polja su iste
       boje ako su:
       1) oba reda parna i obe kolone parne ILI
       2) oba reda neparna i obe kolone neparne. */
    if (((k % 2 == m % 2) && (l % 2 == n % 2))
        || ((k % 2 != m % 2) && (l % 2 != n % 2)))
        printf("Polja su iste boje.\n");
    else
        printf("Polja su razlicite boje.\n");
}
```

```

36  /* II nacin:
37      if((k+l) % 2 == (m+n) % 2)
38          printf("Polja su iste boje.\n");
39      else
40          printf("Polja su razlicite boje.\n"); */

42  /* Provera da li kraljica sa (k,l) napada polje (m,n).
43     Kraljica napada polje u sledecim situacijama:
44     1) Ako se nalaze u istom redu (k==m)
45     2) Ako se nalaze u istoj koloni (l==n)
46     3) Ako se nalaze na istoj dijagonalni. Dijagonalna moze biti:
47         a) paralelna glavnoj dijagonalni (k-l == m-n)
48         b) paralelna sporednoj dijagonalni (k+l == m+n) */
49
50  if ((k == m) || (l == n) || (k - l == m - n)
51      || (k + l == m + n)) {
52      printf("Kraljica sa (%d, %d) ugrozava (%d, %d).\n",
53             k, l, m, n);
54  }
55  else {
56      printf("Kraljica sa (%d, %d) ne ugrozava (%d, %d).\n",
57             k, l, m, n);
58  }

59  /* Provera da li konj sa (k, l) napada polje (m, n). Postoji
60     8 mogucih vrednosti za polja koja konj napada. Vrsi se
61     provera da li je (m,n) jednako nekom od tih polja. */
62  if ((abs(k-m) == 2 && abs(n-l) == 1) || (abs(n-l) == 2 && abs(m-k)
63      == 1))
64      printf("Konj sa (%d, %d) ugrozava (%d, %d).\n",
65             k, l, m, n);
66  else
67      printf("Konj sa (%d, %d) ne ugrozava (%d, %d).\n",
68             k, l, m, n);

69  return 0;
70 }
```

1.5 Petlje

Zadatak 1.5.1 Napisati program koji pet puta ispisuje tekst **Mi volimo da programiramo.**

Primer 1

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

[Rešenje 1.5.1]

Zadatak 1.5.2 Napisati program koji učitava pozitivan ceo broj n i n puta ispisuje tekst **Mi volimo da programiramo.** U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 0  
|| Greska: pogresan unos broja n.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -5  
|| Greska: pogresan unos broja n.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 1  
|| Mi volimo da programiramo.
```

[Rešenje 1.5.2]

Zadatak 1.5.3 Napisati program koji učitava nenegativan ceo broj n a potom ispisuje sve cele brojeve od 0 do n . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -10  
|| Greska: pogresan unos broja n.
```

[Rešenje 1.5.3]

Zadatak 1.5.4 Napisati program koji učitava dva cela broja n i m ($n \leq m$) i ispisuje sve cele brojeve iz intervala $[n, m]$. Pri rešavanju zadatka:

- (a) koristiti **while** petlju
- (b) koristiti **for** petlju
- (c) koristiti **do-while** petlju

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite granice intervala: -2 4  
|| -2 -1 0 1 2 3 4
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite granice intervala: 10 6  
|| Greska: pogresan unos granica.
```

[Rešenje 1.5.4]

Zadatak 1.5.5 Napisati program koji učitava nenegativan ceo broj n i izračunava njegov faktorijel. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 18  
|| 18! = 6402373705728000
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 8  
|| 8! = 40320
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 40  
|| Pri racunanju 40! ce doci do prekoracenja.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -5  
|| Greska: neispravan unos.
```

[Rešenje 1.5.5]

Zadatak 1.5.6 Napisati program koji učitava realan broj x i ceo nenegativan broj n i izračunava n -ti stepen broja x , tj. x^n . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

1 Osnovni elementi imperativnog programiranja

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite redom brojeve x i n: 11.43 -6
|| Greska: neispravan unos broja n.

Primer 4

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

[Rešenje 1.5.6]

Zadatak 1.5.7 Napisati program koji učitava realan broj x i ceo broj n i izračunava n -ti stepen broja x .

Primer 1

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

Primer 2

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

[Rešenje 1.5.7]

Zadatak 1.5.8 Pravi delioci celog broja su svi delioci osim jedinice i samog tog broja. Napisati program koji za uneti pozitivan ceo broj n ispisuje sve njegove prave delioce. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 100
|| Pravi delioci: 2 4 5 10 20 25 50

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -6
|| Greska: neispravan unos.

[Rešenje 1.5.8]

Zadatak 1.5.9 Napisati program koji za uneti ceo broj ispisuje broj dobijen uklanjanjem svih nula sa desne strane unetog broja.

Primer 1

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

Primer 2

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

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -1400
|| Rezultat: -14

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 147
|| Rezultat: 147

[Rešenje 1.5.9]

Zadatak 1.5.10 Napisati program koji učitava ceo broj i ispisuje njegove cifre u obrnutom poretku.

Primer 1

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

Primer 2

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

[Rešenje 1.5.10]

Zadatak 1.5.11 Napisati program koji za uneti pozitivan ceo broj ispisuje da li je on deljiv sumom svojih cifara. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 12  
|| Broj 12 je deljiv sa 3.
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2564  
|| Broj 2564 nije deljiv sa 17.
```

Primer 3

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

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 0  
|| Greska: neispravan ulaz.
```

[Rešenje 1.5.11]

Zadatak 1.5.12 Knjigovoda vodi evidenciju o transakcijama jedne firme i treba da napiše izveštaj o godišnjem poslovanju te firme. Firma je tokom godine imala t transakcija. Transakcije su predstavljene celim brojevima i u slučaju da je vrednost transakcije pozitivna, ta transakcija označava prihod firme, a u slučaju da je negativna rashod. Napisati program koji učitava nenegativan ceo broj t i podatke o t transakcijama i zatim izračunava i ispisuje ukupan prihod, ukupan rashod i zaradu, odnosno gubitak, koji je firma ostvarila tokom godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj t: 7  
|| Unesite transakcije:  
|| 8 -50 45 2007 -67 -123 14  
|| Prihod: 2074  
|| Rashod: -240  
|| Zarada: 1834
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj t: 5  
|| Unesite transakcije:  
|| -5 -20 -4 -200 -8  
|| Prihod: 0  
|| Rashod: -237  
|| Gubitak: 237
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj t: 2  
|| Unesite transakcije:  
|| 120 -120  
|| Prihod: 120  
|| Rashod: -120  
|| Zarada: 0
```

1 Osnovni elementi imperativnog programiranja

Primer 4

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

Primer 5

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 0  
|| Nema evidentiranih transakcija.
```

[Rešenje 1.5.12]

Zadatak 1.5.13 Napisati program koji učitava pozitivan ceo broj n , a potom i n celih brojeva. Izračunati i ispisati zbir onih brojeva koji su istovremeno neparni i negativni. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 5  
|| Unesite n brojeva:  
|| 1 -5 -6 3 -11  
|| Zbir neparnih i negativnih: -16
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 4  
|| Unesite n brojeva:  
|| 5 8 13 17  
|| Zbir neparnih i negativnih: 0
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -4  
|| Greska: neispravan unos.
```

Primer 4

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

[Rešenje 1.5.13]

Zadatak 1.5.14 Napisati program koji učitava cele brojeve sve do unosa broja nula i ispisuje proizvod onih unetih brojeva koji su pozitivni.

Primer 1

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

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve:  
|| 0  
|| Nije unet nijedan broj.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve:  
|| -5 -200 -43 0  
|| Medju unetim brojevima nema pozitivnih.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve:  
|| 1 0  
|| Proizvod pozitivnih brojeva je 1.
```

[Rešenje 1.5.14]

Zadatak 1.5.15 Napisati program koji za uneti ceo broj proverava i ispisuje da li se cifra 5 nalazi u njegovom zapisu.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 1857
Broj 1857 sadrzi cifru 5.

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 84
Broj 84 ne sadrzi cifru 5.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj: -2515
Broj -2515 sadrzi cifru 5.

[Rešenje 1.5.15]

Zadatak 1.5.16 Napisati program koji učitava cele brojeve sve do unosa broja nula, a zatim izračunava i ispisuje aritmetičku sredinu unetih brojeva na četiri decimale.

Primer 1

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

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve:
0
Nisu uneti brojevi.

Primer 3

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

[Rešenje 1.5.16]

Zadatak 1.5.17 U prodavnici se nalaze artikli čije su cene pozitivni realni brojevi. Napisati program koji učitava cene artikala sve do unosa broja nula i izračunava i ispisuje prosečnu vrednost cena u radnji. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite cene:
8 5.2 6.11 3 0
Prosecna cena: 5.5775

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite cene:
6.32 -9
Greska: neispravan unos
cene.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite cene:
0
Nisu unete cene.

[Rešenje 1.5.17]

Zadatak 1.5.18 Napisati program koji učitava pozitivan ceo broj n , a potom i n realnih brojeva. Program ispisuje koliko puta je prilikom unosa došlo do promene znaka. Do promena znaka dolazi kada se nakon pozitivnog broja

1 Osnovni elementi imperativnog programiranja

uneset negativan broj ili kada se nakon negativnog broja unese pozitivan broj. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 9  
|| Unesite brojeve:  
|| 7.82 4.3 -1.2 56.8 -3.4 -72.1 8.9 11.2 -11.2  
|| Broj promena je 5.
```

Primer 2

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

Primer 3

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

Primer 4

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

[Rešenje 1.5.18]

Zadatak 1.5.19 U prodavnici se nalazi n artikala čije su cene pozitivni realni brojevi. Napisati program koji učitava n , a potom i cenu svakog od n artikala i određuje i ispisuje najmanju cenu. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj artikla: 6  
|| Unesite cene artikala:  
|| 12 3.4 90 100.53 53.2 12.8  
|| Najmanja cena: 3.400000
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj artikla: 3  
|| Unesite cene artikala:  
|| 4 -8 92  
|| Greska: neispravan unos
```

Primer 3

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

[Rešenje 1.5.19]

Zadatak 1.5.20 Nikola želi da obraduje baku i da joj kupi jedan poklon u radnji. On na raspolaganju ima m dinara. U radnji se nalazi n artikala i zanima ga koliko ima artikala u radnji čija cena je manja ili jednaka m . Napisati program koji pomaže Nikoli da odredi broj takvih artikala. Program učitava realan nenegativan broj m , ceo nenegativan broj n i n pozitivnih realnih brojeva. Ispisati koliko artikala ima cenu čija je vrednost manja ili jednaka m . NAPOMENA: Prepostaviti da je unos ispravan.

Primer 1

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite broj budzet: 12.37  
||| Unesite broj artikala: 5  
||| Unesite cene artikala: 11 54.13 6 13 8  
||| Ukupno artikala: 3
```

Primer 2

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite broj budzet: 2  
||| Unesite broj artikala: 4  
||| Unesite cene artikala: 1 11 4.32 3  
||| Ukupno artikala: 1
```

[Rešenje 1.5.20]

Zadatak 1.5.21 Napisati program koji učitava ceo nenegativan broj n , n cenih brojeva i zatim izračunava i ispisuje tražene vrednosti. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

- (a) Broj sa najvećom cifrom desetica. Ukoliko ima više takvih, ispisati prvi.

Primer 1

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

Primer 2

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

Primer 3

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite broj n: 4  
||| Unesite brojeve:  
||| 100 200 300 400  
||| Broj sa najvecom cifrom desetica: 100.
```

Primer 4

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

- (b) Broj sa najvećim brojem cifara. Ukoliko ima više takvih, ispisati prvi.

Primer 1

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite broj n: 5  
||| Unesite n brojeva: 18 -365 251 1 78  
||| Najvise cifara ima broj -365.
```

Primer 2

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite broj n: 7  
||| Unesite n brojeva:  
||| 3 892 18 21 639 742 85  
||| Najvise cifara ima broj 892.
```

Primer 3

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite broj n: 0  
||| Nisu uneti brojevi.
```

Primer 4

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite broj n: -7  
||| Greska: neispravan unos.
```

1 Osnovni elementi imperativnog programiranja

Primer 5

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: 0 1 2 -3 4
Najvise cifara ima broj 0.

Primer 6

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: -5 4 -3 2 1
Najvise cifara ima broj -5.

- (c) Broj sa najvećom vodećom cifrom. Vodeća cifra je cifra najveće težine u zapisu broja. Ukoliko ima više takvih, ispisati prvi.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
Unesite n brojeva: 8 964 -32 511 27
Broj sa najvecom vodecom cifrom je 964.

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
Unesite n brojeva: 0 0 0
Broj sa najvecom vodecom cifrom je 0.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 3
Unesite n brojeva: 41 669 -8
Broj sa najvecom vodecom cifrom je -8.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Nisu uneti brojevi.

[Rešenje 1.5.21]

Zadatak 1.5.22 Vršena su merenja nadmorskih visina na određenom delu teritorije i naučnike zanima razlika između najveće i najmanje nadmorske visine. Napisati program koji učitava cele brojeve koji predstavljaju nadmorske visine sve do unosa broja nula 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

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve:
0
Nisu unete nadmorske visine.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve:
-500 0
Razlika: 0

Primer 5

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve:
-500 -300 -5000 0
Razlika: 4700

[Rešenje 1.5.22]

Zadatak 1.5.23 Napisati program koji učitava ceo broj n ($n > 1$), nenegativan ceo broj d , a zatim i n celih brojeva i izračunava i ispisuje koliko ima parova uzastopnih brojeva među unetim brojevima koji se nalaze na rastojanju d . Rastojanje između brojeva je definisano sa $d(x, y) = |y - x|$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: 5 0  
Unesite n brojeva: 1 1 1 1 1  
Broj parova: 4
```

Primer 3

```
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
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite brojeve n i d: 1 3  
Greska: neispravan unos.
```

[Rešenje 1.5.23]

Zadatak 1.5.24 Napisati program koji uneti pozitivan ceo broj transformiše tako što svaku parnu cifru u zapisu broja uveća za jedan. Ispisati dobijeni broj. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 1.5.24]

Zadatak 1.5.25 Napisati program koji učitava jedan ceo broj i zatim formira i ispisuje broj koji se dobija izbacivanjem svake druge cifre polaznog broja, idući sa desna na levo.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj: -18  
Rezultat: -8
```

Primer 3

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

[Rešenje 1.5.25]

1 Osnovni elementi imperativnog programiranja

* **Zadatak 1.5.26** Napisati program koji na osnovu unetog pozitivnog celog broja formira i ispisuje broj koji se dobija izbacivanjem cifara koje su u polaznom broju jednake zbiru svojih suseda. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 28631
Rezultat: 2631

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 440
Rezultat: 40

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj: -5
Greska: neispravan unos.

[Rešenje 1.5.26]

* **Zadatak 1.5.27** Broj je *palindrom* ukoliko se isto čita i sa leve i sa desne strane. Napisati program koji učitava pozitivan ceo broj i proverava da li je učitani broj palindrom. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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 1.5.27]

Zadatak 1.5.28 Fibonačijev niz počinje članovima 0 i 1, a svaki naredni član se dobija kao zbir prethodna dva. Napisati program koji učitava nenegativan ceo broj n i određuje i ispisuje n -ti član Fibonačijevog niza. Niz se indeksira počevši od nule. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 10
F[10] = 55

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -100
Greska: neispravan unos.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 40
F[40] = 102334155

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 20
F[20] = 6765

[Rešenje 1.5.28]

Zadatak 1.5.29 Niz prirodnih brojeva formira se prema sledećem pravilu:

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

Napisati program koji za uneti početni član niza a_0 (pozitivan ceo broj) štampa niz brojeva sve do onog člana niza koji je jednak 1. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite prvi clan: 56  
Clanovi niza:  
56 28 14 7 11 17 26 13  
20 10 5 8 4 2 1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite prvi clan: -48  
Greska: neispravan unos.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite prvi clan: 67  
Clanovi niza:  
67 101 152 76 38 19 29  
44 22 11 17 26 13 20 10  
5 8 4 2 1
```

[Rešenje 1.5.29]

* **Zadatak 1.5.30** 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 nenegativan broj k ispisuje dimenzije papira A_k u milimetrima. Rezultat ispisati kao celobrojne vrednosti. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite format papira: 0  
Dimenzije papira: 840 1189
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite format papira: -7  
Greska: neispravan unos.
```

Primer 4

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

[Rešenje 1.5.30]

Zadatak 1.5.31 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.

1 Osnovni elementi imperativnog programiranja

Primer 1

INTERAKCIJA SA PROGRAMOM:
Danas je Veoma Lep DAN.
dANAS JE vEOMA 1EP dan

Primer 2

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

[Rešenje 1.5.31]

Zadatak 1.5.32 Napisati program koji učitava karaktere sve do kraja ulaza, a potom ispisuje broj velikih slova, broj malih slova, broj cifara, broj belina i zbir unetih cifara.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
Beograd - Nis 230km
Uzice - Cacak 56.3km
Subotica - Ruma 139km
Velika slova: 6
Mala slova: 32
Cifre: 9
Beline: 12
Suma cifara: 32

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite tekst:
Isli smo u Afriku da sadimo papriku.
Velika slova: 2
Mala slova: 27
Cifre: 0
Beline: 7
Suma cifara: 0

[Rešenje 1.5.32]

Zadatak 1.5.33 Program učitava pozitivan ceo 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. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -7
Greska: neispravan unos.

[Rešenje 1.5.33]

Zadatak 1.5.34 Program učitava pozitivan ceo broj n , a zatim i n karaktera. Napisati program koji proverava da li se od unetih karaktera može napisati reč *Zima*. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Greska: neispravan unos.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 10
Unestite 1. karakter: i
Unestite 2. karakter: g
Unestite 3. karakter: o
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 1.5.34]

Zadatak 1.5.35 Napisati program koji učitava pozitivan ceo broj n i ispisuje vrednost sume kubova brojeva od 1 do n , odnosno $S = 1 + 2^3 + 3^3 + \dots + n^3$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 14
Suma kubova: 11025
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 25
Suma kubova: 105625
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Greska: neispravan unos.
```

[Rešenje 1.5.35]

Zadatak 1.5.36 Napisati program koji učitava pozitivan ceo broj n i ispisuje sumu kubova, $S = 1 + 2^3 + 3^3 + \dots + k^3$, za svaku vrednost $k = 1, \dots, n$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 5  
|| [k=1] Suma kubova: 1  
|| [k=2] Suma kubova: 9  
|| [k=3] Suma kubova: 36  
|| [k=4] Suma kubova: 100  
|| [k=5] Suma kubova: 225
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 8  
|| [k=1] Suma kubova: 1  
|| [k=2] Suma kubova: 9  
|| [k=3] Suma kubova: 36  
|| [k=4] Suma kubova: 100  
|| [k=5] Suma kubova: 225  
|| [k=6] Suma kubova: 441  
|| [k=7] Suma kubova: 784  
|| [k=8] Suma kubova: 1296
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -5  
|| Greska: neispravan unos.
```

[Rešenje 1.5.36]

Zadatak 1.5.37 Napisati program koji učitava realan broj x i pozitivan ceo broj n i izračunava i ispisuje sumu $S = x + 2 \cdot x^2 + 3 \cdot x^3 + \dots + n \cdot x^n$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom brojeve x i n: 5.5 0  
|| Greska: neispravan unos.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom brojeve x i n: -0.5 -5  
|| Greska: neispravan unos.
```

[Rešenje 1.5.37]

Zadatak 1.5.38 Napisati program koji učitava realan broj x i pozitivan ceo broj n i izračunava i ispisuje sumu $S = 1 + \frac{1}{x} + \frac{1}{x^2} + \dots + \frac{1}{x^n}$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom brojeve x i n: 5.5 0  
|| Greska: neispravan unos.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite redom brojeve x i n: -0.5 -5  
|| Greska: neispravan unos.
```

[Rešenje 1.5.38]

*** Zadatak 1.5.39** Napisati program koji učitava realne brojeve x i eps i sa tačnošću eps izračunava i ispisuje sumu $S = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots$. Za sumu S se kaže da je izračunata sa tačnošću eps ako je absolutna vrednost poslednjeg člana sume manja od eps . 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 = 20.07966
```

[Rešenje 1.5.39]

*** Zadatak 1.5.40** Napisati program koji učitava realne brojeve x i eps i sa zadatom tačnošću eps izračunava i ispisuje sumu $S = 1 - x + \frac{x^2}{2!} - \frac{x^3}{3!} + \frac{x^4}{4!} - \frac{x^5}{5!} \dots$.
NAPOMENA: Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite x: 3
Unesite tacnost eps: 0.000001
S = 0.049077
```

Primer 2

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

[Rešenje 1.5.40]

Zadatak 1.5.41 Napisati program koji učitava realan broj x i pozitivan ceo broj n i izračunava proizvod $P = (1 + \cos(x)) \cdot (1 + \cos(x^2)) \cdot \dots \cdot (1 + \cos(x^n))$. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.
NAPOMENA: Voditi računa o efikasnosti rešenja.

Primer 1

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

Primer 2

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

1 Osnovni elementi imperativnog programiranja

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite redom brojeve x i n: 12 0
Greska: neispravan unos.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite redom brojeve x i n: 12 -6
Greska: neispravan unos.

[Rešenje 1.5.41]

* **Zadatak 1.5.42** Napisati program koji učitava pozitivan ceo broj n i ispisuje vrednost razlomka

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 4
R = 0.697674

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 20
R = 0.697775

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Greska: neispravan unos.

[Rešenje 1.5.42]

* **Zadatak 1.5.43** Napisati program koji učitava realan broj x i pozitivan ceo broj n i računa sumu

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA:
Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite x i n: 5.6 8
S = 0.779792

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite x i n: 14.32 11
S = -6714.066406

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite x i n: 2 -6
Greska: neispravan unos.

[Rešenje 1.5.43]

*** Zadatak 1.5.44** Napisati program koji učitava pozitivan ceo broj n i koji računa proizvod

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

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA:
Voditi računa o efikasnosti rešenja i o mogućnosti prekoračenja.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 5
P = 1.838108

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 7
P = 1.841026

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Greska: neispravan unos.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 10
P = 1.841077

[Rešenje 1.5.44]

*** Zadatak 1.5.45** Napisati program koji učitava neparan ceo broj n ($n \geq 5$) i izračunava i ispisuje sumu

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

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
S = 855

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 11
S = -9540

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 20
Greska: neispravan unos.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -3
Greska: neispravan unos.

[Rešenje 1.5.45]

Zadatak 1.5.46 Napisati program koji učitava realne brojeve x i a i pozitivan ceo broj n i zatim izračunava i ispisuje vrednost izraza

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

1 Osnovni elementi imperativnog programiranja

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 5  
|| Izraz = 135380494030332048.000000
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 3  
|| Izraz = 10201.000000
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 3  
|| Izraz = 5800.970129
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -2  
|| Greska: neispravan unos.
```

[Rešenje 1.5.46]

Zadatak 1.5.47 Napisati program koji za unetu pozitivnu celobrojnu vrednost n ispisuje odgovarajuće brojeve. NAPOMENA: Pretpostaviti da je unos ispravan.

- (a) Ispisati tablicu množenja brojeva od 1 do n .

Primer 1

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

Primer 2

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

Primer 3

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

- (b) Ispisati sve brojeve od 1 do n^2 , po n brojeva u jednoj vrsti.

Primer 1

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

Primer 2

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

- (c) Ispisati tablicu brojeva tako da su u prvoj vrsti svi brojevi od 1 do n , a u svakoj narednoj brojevi dobijeni rotiranjem prethodne vrste za jedno mesto u levo.

Primer 1

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

Primer 2

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

- (d) Ispisati pravougli „trougao” sačinjen od „koordinata” svojih tačaka. „Koordinata” tačke je oblika (i, j) pri čemu $i, j = 0, \dots, n$. Prav ugao se nalazi u gornjem levom uglu slike i njegova koordinata je $(0, 0)$. Koordinata i se uvećava po vrsti, a koordinata j po koloni, pa je zato koordinata tačke koja je ispod tačke $(0, 0)$ jednaka $(1, 0)$, a koordinata tačke koja je desno od tačke $(0, 0)$ jednaka $(0, 1)$.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 1.5.47]

Zadatak 1.5.48 Napisati program koji za uneti pozitivan ceo broj n zvezdicama iscrtava odgovarajuću sliku. NAPOMENA: *Prepostaviti da je unos ispravan.*

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

Primer 1

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

Primer 2

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

- (b) Slika predstavlja rub kvadrata dimenzije n .

1 Osnovni elementi imperativnog programiranja

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 1

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

[Rešenje 1.5.48]

- * **Zadatak 1.5.49** Napisati program koji za uneti pozitivan ceo broj n zvezdicama iscrtava slovo X dimenzije n . NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.5.49]

- * **Zadatak 1.5.50** Napisati program koji za uneti neparan pozitivan broj n korišćenjem znaka + iscrtava veliko plus dimenzije n . NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.5.50]

Zadatak 1.5.51 Napisati program koji učitava pozitivan ceo broj n , a potom iscrtava odgovarajuću sliku. NAPOMENA: *Pretpostaviti da je unos ispravan.*

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

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

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

1 Osnovni elementi imperativnog programiranja

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

- (f) Slika predstavlja 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 1.5.51]

Zadatak 1.5.52 Napisati program koji učitava pozitivan ceo broj n , a potom iscrtava odgovarajuću sliku. NAPOMENA: *Prepostaviti da je unos ispravan.*

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

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 2

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

- (c) Slika predstavlja trougao koji se dobija spajanjem dva jednakostranična trougla stranice n koji su sastavljeni od zvezdica.

1 Osnovni elementi imperativnog programiranja

Primer 1

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

Primer 2

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

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

Primer 1

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

Primer 1

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

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

Primer 1

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

Primer 2

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

[Rešenje 1.5.52]

* **Zadatak 1.5.53** Napisati program koji za uneti pozitivan ceo broj n isrtava strelice dimenzije n . NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.5.53]

* **Zadatak 1.5.54** Napisati program koji učitava pozitivan 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 . NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

[Rešenje 1.5.54]

* **Zadatak 1.5.55** Napisati program koji učitava pozitivne cele brojeve m i n i iscrtava jedan do drugog n kvadrata čija je svaka stranica sastavljena od m zvezdica razdvojenih praznim. NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite brojeve n i m: 5 3  
* * * * * * * * * * * *  
* * * * * * * * * * * *  
* * * * * * * * * * * *  
* * * * * * * * * * * *  
* * * * * * * * * * * *
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite brojeve n i m: 4 4  
* * * * * * * * * * * *  
* * * * * * * * * * * *  
* * * * * * * * * * * *  
* * * * * * * * * * * *
```

[Rešenje 1.5.55]

* **Zadatak 1.5.56** Napisati program koji učitava pozitivan ceo broj n i iscrtava romb sastavljen od minusa u pravougaoniku sastavljenom od zvezdica.
NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.5.56]

Zadatak 1.5.57 Napisati program koji učitava ceo broj n ($n \geq 2$) i koji iscrtava sliku kuće sa krovom: kuća je kvadrat stranice n , a krov jednakostranični trougao stranice n . NAPOMENA: *Pretpostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

[Rešenje 1.5.57]

* **Zadatak 1.5.58** Napisati program koji učitava pozitivan ceo broj n i 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. NAPOMENA: *Prepostaviti da je unos ispravan.*

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: 6  
1 2 3 4 5 6 2 3 4 5 3 4
```

Primer 3

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

Primer 4

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

[Rešenje 1.5.58]

* **Zadatak 1.5.59** Napisati program koji učitava pozitivan ceo broj n i ispisuje izabrane brojeve u n redova. U prvom redu, program ispisuje sve brojeve iz intervala $[1, n]$. U drugom redu, program ispisuje svaki drugi broj iz ovog intervala. U trećem redu, program ispisuje svaki treći broj iz ovog intervala, i tako redom. Na kraju, u n -tom redu, program ispisuje samo broj 1. NAPOMENA: *Prepostaviti da je unos ispravan.*

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 1.5.59]

1.6 Rešenja

Rešenje 1.5.1

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     int i;
6
7     /* Promenljiva i kontrolise koliko puta ce se petlja izvrsiti i
8      naziva se brojac petlje. Njena pocetna vrednost se postavlja
9      na 0 jer se u pocetku petlja nije ni jednom izvrsila. */
10    i = 0;
11
12    /* Petlja ce se izvrsiti za i=0,1,2,3,4. Kada i dostigne vrednost
13       5 uslov i < 5 nece biti ispunjen i prelazi se na prvu sledecu
14       naredbu nakon tela petlje. */
15    while (i < 5) {
16        /* Ispis poruke. */
17        printf("Mi volimo da programiramo.\n");
18
19        /* Uvecavanje brojaca za 1. */
20        i++;
21    }
22
23    return 0;
24 }
```

Rešenje 1.5.2

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int i, n;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%d", &n);
10
11    /* Provera ispravnosti ulaza. */
12    if (n <= 0) {
13        printf("Greska: pogresan unos broja n.\n");
14        return 1;
15    }
16
17    /* Inicijalizacija brojaca. */
18    i = 0;
19 }
```

```

1  /* Tražena poruka se ispisuje n puta. */
2  while (i < n) {
3      printf("Mi volimo da programiramo.\n");
4      i++;
5  }
6
7  return 0;
8 }
```

Rešenje 1.5.3

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int i, n;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%d", &n);
10
11    /* Provera ispravnosti ulaza. */
12    if (n < 0) {
13        printf("Greska: pogresan unos broja n.\n");
14        return 1;
15    }
16
17    /* Inicijalizacija brojaca. */
18    i = 0;
19
20    /* Posto je potrebno ispisati sve brojeve [0,n], telo petlje se
21       izvrsava za svako i <= n. */
22    while (i <= n) {
23        /* Ispis trenutne vrednosti brojaca. */
24        printf("%d\n", i);
25
26        /* Prelazak na sledeći broj. */
27        i++;
28    }
29
30    return 0;
31 }
```

Rešenje 1.5.4

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int n, m, i;
```

```
7  /* Ucitavanje vrednosti granica intervala. */
8  printf("Unesite granice intervala: ");
9  scanf("%d%d", &n, &m);
10
11 /* Provera ispravnosti ulaznih podataka. */
12 if (m < n) {
13     printf("Greska: pogresan unos granica.\n");
14     return 1;
15 }
16
17 /* a) I nacin: Koriscenjem while petlje. */
18 /* Inicijalizacija brojaca na levu granicu intervala. */
19 i = n;
20
21 /* Ispis svih vrednosti brojaca izmedju leve i desne granice
22    intervala, ukljucujuci i same granice. */
23 while (i <= m) {
24     printf("%d ", i);
25     i++;
26 }
27
28 /* b) II nacin: Koriscenjem for petlje.
29
30 Naredba i=n se izvrsava jednom, pre prve iteracije. Uslov
31 petlje i<=m se proverava pre svake iteracije. Naredba i++ se
32 izvrsava nakon svake iteracije.
33
34 for (i = n; i <= m; i++){
35     printf("%d ", i);
36 } */
37
38 /* c) III nacin: Koriscenjem do while petlje.
39
40 Uslov petlje se proverava na kraju svake iteracije. Zbog toga
41 se do while petlja izvrsava bar jednom, cak i u slucaju da
42 uslov petlje nikada nije ispunjen. U ovom slucaju je to
43 ispravno jer je poznato da ce interval imati bar jedan
44 element. U opstem slucaju to ne mora da vazi.
45
46 i = n;
47 do {
48     printf("%d ", i);
49     i++;
50 } while (i <= m); */
51
52 printf("\n");
53
54 return 0;
55 }
```

Rešenje 1.5.5

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int n, tekuca_vrednost;
6
7     /* Za cuvanje vrednosti faktorijela se koristi tip unsigned long
8      jer izracunata vrednost moze da bude jako veliki broj. */
9     unsigned long faktorijel;
10
11    /* Ucitavanje vrednosti broja n. */
12    printf("Unesite broj n: ");
13    scanf("%d", &n);
14
15    /* Provera ispravnosti ulaza. */
16    if (n < 0) {
17        printf("Greska: neispravan unos..\n");
18        return 1;
19    }
20
21    if (n >= 22) {
22        printf("Pri racunanju %d! ce doci do prekoracenja.\n", n);
23        return 1;
24    }
25
26    /* Tekuca vrednost uzima vrednosti n, n-1, n-2, ..., 2. Na
27     pocetku se inicijalizuje na n, a zatim se u svakoj iteraciji
28     umanjuje za 1. */
29    tekuca_vrednost = n;
30
31    /* Inicijalizacija vrednosti faktorijela. */
32    faktorijel = 1;
33
34    /* Racunanje vrednosti faktorijela mnozenjem trenutnog rezultata
35     promenljivom cija vrednost kreće od n, a zatim se u svakoj
36     iteraciji umanjuje za 1. */
37    while (tekuca_vrednost > 1) {
38        faktorijel = faktorijel * tekuca_vrednost;
39        tekuca_vrednost--;
40    }
41
42    /* Ispis rezultata. */
43    printf("%d! = %lu\n", n, faktorijel);
44
45    return 0;
46}

```

Rešenje 1.5.6

```
1 #include <stdio.h>

3 int main() {
    /* Deklaracije potrebnih promenljivih. */
    int n, i;
    float x, rezultat;

7     /* Ucitavanje vrednosti brojeva x i n. */
9     printf("Unesite redom brojove x i n: ");
    scanf("%f %d", &x, &n);

11    /* Provera ispravnosti ulaza. */
13    if (n < 0) {
        printf("Greska: neispravan unos broja n.\n");
        return 1;
    }

17    /* Inicijalizacija rezultata. */
19    rezultat = 1;

21    /* Vrednost n-tog stepena broja x se dobija tako sto se tekuca
       vrednost rezultata n puta pomnozi brojem x.
       (rezultat = x * x * ... * x) = x^n */
23    for (i = 0; i < n; i++)
        rezultat = rezultat * x;

27    /* Ispis rezultata. */
29    printf("Rezultat: %.5f\n", rezultat);
31}
```

Rešenje 1.5.7

```
1 #include <stdio.h>
2 #include <stdlib.h>

3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, i, znak;
6     float x, rezultat;

7     /* Ucitavanje vrednosti brojeva x i n. */
8     printf("Unesite redom brojove x i n: ");
9     scanf("%f %d", &x, &n);

11    /* Pamti se znak stepena i izracunava se apsolutna vrednost stepena
       . */
12    znak = 1;
```

```

15  if (n < 0) {
16      znak = -1;
17      n = abs(n);
18  }
19
20  /* Inicijalizacija rezultata. */
21  rezultat = 1;
22
23  /* Racunanje vrednosti x^n. */
24  for (i = 0; i < n; i++)
25      rezultat = rezultat * x;
26
27  /* Ako je stepen bio negativan, rezultat je 1/x^n. */
28  if (znak == -1)
29      printf("Rezultat: %.3f\n", 1 / rezultat);
30  else
31      printf("Rezultat: %.3f\n", rezultat);
32
33  return 0;
}

```

Rešenje 1.5.8

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int n, i;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%d", &n);
10
11    /* Provera ispravnosti ulaza. */
12    if (n <= 0) {
13        printf("Greska: neispravan unos.\n");
14        return 1;
15    }
16
17    printf("Pravi delioci:\n");
18    /* I nacin: Za svaki broj iz intervala [2, n-1] se proverava da
19       li deli broj n (tj. da li je ostatak pri deljenju sa n jednak
20       nuli). Ako je uslov ispunjen, taj broj se ispisuje.
21    for (i = 2; i < n; i++)
22        if (n % i == 0)
23            printf("%d ", i);
24
25    printf("\n");
26
27    /* II nacin (brzi): Provera se ne vrsti za sve brojeve iz
28       intervala [2, n-1], vec za brojeve iz intervala [2, sqrt(n)],
29       jer ostatak pri deljenju sa brojem većim od korena n
30       je uvek jednak ili veći od 1.
31
32    for (i = 2; i <= sqrt(n); i++)
33        if (n % i == 0)
34            printf("%d ", i);
35
36    printf("\n");
37
38    /* Treci nacin: Koristi se funkcija ispis_delioca() koja
39       ispisuje delioca niza u intervalu [2, n-1].
40    ispis_delioca(n);
41
42    return 0;
}

```

```
29     tj. za sve brojeve k za koje vazi da je k*k <= n. */
30     for (i = 2; i * i <= n; i++) {
31         /* Ako i deli n, treba razlikovati dva slucaja. */
32         if (n % i == 0) {
33             if (i == n / i) {
34                 /* I slucaj: Kada je i koren broja, ispisuje se samo broj i,
35                  npr. za n = 16, i = 4, ispisuje se samo 4. */
36                 printf("%d ", i);
37             } else {
38                 /* II slucaj: Kada i nije koren broja, ispisuje se i broj
39                  i i broj n/i, npr. za n = 16, i = 2 ispisuju se i 2 i 8.
40                  */
41                 printf("%d %d ", i, n / i);
42             }
43         }
44     }
45     printf("\n");
46
47     return 0;
}
```

Rešenje 1.5.9

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     int n;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj: ");
9     scanf("%d", &n);
10
11    /* Slucaj kada broj n ima vrednost nula se posebno obradjuje.
12       U suprotnom bi se petlja u nastavku beskonacno izvrsavala. */
13    if (n == 0) {
14        printf("0\n");
15        return 0;
16    }
17
18    /* Uklanjanje poslednje cifre broja se vrsti deljenjem broja sa 10.
19       Ovaj postupak se ponavlja sve dok je poslednja cifra nula. */
20    while (n % 10 == 0)
21        n = n / 10;
22
23    /* Ispis rezultata. */
24    printf("Rezultat: %d\n", n);
25
26    return 0;
27}
```

Rešenje 1.5.10

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracija potrebne promenljive. */
6     int x;
7
8     /* Ucitavanje vrednosti broja x. */
9     printf("Unesite ceo broj:");
10    scanf("%d", &x);
11
12    /* Izracunava se apsolutna vrednost broja da bi izdvojene cifre
13       bile pozitivni brojevi. Na primer, 123%10 je 3, a -123%10 je -3. */
14    x = abs(x);
15
16    /* Slučaj kada je uneti broj 0 se posebno obradjuje. */
17    if (x == 0) {
18        printf("0\n");
19        return 0;
20    }
21
22    /* U petlji se obradjuje cifra po cifra broja, dok god ima
23       neobradjenih cifara u broju. */
24    printf("Rezultat: ");
25    while (x != 0) {
26        /* Ispis poslednje cifre broja x. */
27        printf("%d ", x % 10);
28
29        /* Uklanjanje poslednje cifre broja x. */
30        x /= 10;
31    }
32    printf("\n");
33
34    return 0;
35 }
```

Rešenje 1.5.11

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int n, suma, n_kopija;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj: ");
9     scanf("%d", &n);
```

1 Osnovni elementi imperativnog programiranja

```
1  /* Provera ispravnosti ulaza. */
2  if (n <= 0) {
3      printf("Greska: neispravan unos.\n");
4      return 1;
5  }
6
7  /* Pravljenje kopije originalnog broja, da bi originalna vrednost n
8   * ostala nepromenjena. */
9  n_kopija = n;
10
11 /* Inicijalizacija sume cifara. */
12 suma = 0;
13
14 /* Racunanje sume cifara. */
15 while (n_kopija != 0) {
16     /* Dodavanje poslednje cifre na sumu. */
17     suma += n_kopija % 10;
18     /* Uklanjanje poslednje cifre. */
19     n_kopija /= 10;
20 }
21
22 /* Ispis rezultata. */
23 if (n % suma == 0)
24     printf("Broj %d je deljiv sa %d.\n", n, suma);
25 else
26     printf("Broj %d nije deljiv sa %d.\n", n, suma);
27
28 return 0;
29 }
```

Rešenje 1.5.12

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int t, x, i;
7     int ukupan_prihod, ukupan_rashod, ukupan_rashod_abs;
8
9     /* Ucitavanje vrednosti broja t. */
10    printf("Unesite broj t:");
11    scanf("%d", &t);
12
13    /* Provera ispravnosti ulaza. */
14    if (t < 0) {
15        printf("Greska: neispravan unos.\n");
16        return 1;
17    } else if (t == 0) {
18        printf("Nema evidentiranih transakcija.");
19        return 0;
20 }
```

```

1      }
21
21     /* Inicijalizacija suma. */
23     ukupan_prihod = 0;
24     ukupan_rashod = 0;
25
25     /* Ucitavanje transakcija i racunanje suma. */
27     printf("Unesite transakcije: ");
28     i = 0;
29     while (i < t) {
30         /* Ucitavanje jedne transakcije. */
31         scanf("%d", &x);
32
33         /* Dodavanje ucitane vrednosti na odgovarajucu sumu. */
34         if (x < 0)
35             ukupan_rashod += x;
36         else
37             ukupan_prihod += x;
38
39         /* Uvecavanje brojaca. */
40         i++;
41     }
42
43     /* Ispis rezultata. */
44     printf("Prihod: %d\n", ukupan_prihod);
45     printf("Rashod: %d\n", ukupan_rashod);
46
47     ukupan_rashod_abs = abs(ukupan_rashod);
48     if (ukupan_prihod >= ukupan_rashod_abs)
49         printf("Zarada: %d\n", ukupan_prihod - ukupan_rashod_abs);
50     else
51         printf("Gubitak: %d\n", ukupan_rashod_abs - ukupan_prihod);
52
53     return 0;
54 }
```

Rešenje 1.5.13

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, x, i;
6     int zbir = 0;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
```

```
15     printf("Greska: neispravan unos.\n");
16     return 1;
17 }
18
19 /* Ucitavanje brojeva i racunanje trazenog zbir-a.*/
20 printf("Unesite n brojeva: ");
21 i = 0;
22 while (i < n) {
23     /* Ucitavanje jednog broja. */
24     scanf("%d", &x);
25
26     /* Ako je ucitani broj negativan i neparan, dodaje se na
27      zbir. */
28     if (x < 0 && x % 2 != 0)
29         zbir = zbir + x;
30
31     /* Uvecavanje brojaca. */
32     i++;
33 }
34
35 /* Ispis rezultata. */
36 printf("Zbir neparnih i negativnih: %d\n", zbir);
37
38 return 0;
39 }
```

Rešenje 1.5.14

```
#include <stdio.h>
1
2 int main() {
3     /* Deklaracije potrebnih promenljivih. */
4     int x, proizvod;
5
6     /* Indikator koji označava da li je korisnik uneo bar jedan
7      broj. */
8     int unet_bar_jedan = 0;
9
10    /* Indikator koji označava da li je korisnik uneo bar jedan
11       pozitivan broj. */
12    int unet_pozitivan = 0;
13
14    /* Inicijalizacija proizvoda. */
15    proizvod = 1;
16
17
18    printf("Unesite brojeve:");
19    /* Petlja ciji je uslov uvek ispunjen. */
20    while (1) {
21        /* Ucitavanje jednog broja. */
22        scanf("%d", &x);
```

```

24  /* Ako je uneta nula, petlja se prekida naredbom break. */
25  if (x == 0)
26      break;
27
28  /* Ako petlja nije prekinuta, znaci da je unet bar jedan broj.
29      Iz tog razloga se vrednost indikatora za unete brojeve
30      postavlja na 1. */
31  unet_bar_jedan = 1;
32
33  /* Provera da li je broj x pozitivan. */
34  if (x > 0) {
35      /* Ako jeste, znaci da je unet bar jedan pozitivan broj i iz
36          tog razloga se vrednost odgovarajuceg indikatora postavlja
37          na 1. */
38      unet_pozitivan = 1;
39
40      /* Azuriranje vrednosti proizvoda pozitivnih brojeva. */
41      proizvod = proizvod * x;
42  }
43
44  /* Ispis rezultata. */
45  if (unet_bar_jedan == 0)
46      printf("Nije unet nijedan broj.\n");
47  else if (unet_pozitivan == 0)
48      printf("Medju unetim brojevima nema pozitivnih.\n");
49  else
50      printf("Proizvod pozitivnih brojeva je %d.\n", proizvod);
51
52  return 0;
53}

```

Rešenje 1.5.15 Pogledajte zadatak 1.5.10.

Rešenje 1.5.16

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija i inicializacije potrebnih promenljivih. */
5     int x, broj_brojeva = 0, suma = 0;
6
7     /* Ucitavanje brojeva sve do unosa broja 0. */
8     printf("Unesite brojeve: ");
9     while (1) {
10         scanf("%d", &x);
11
12         if (x == 0)
13             break;
14
15         /* Dodavanje ucitanog broja na sumu. */
16         suma += x;
17
18     }
19
20     /* Ispis rezultata. */
21     printf("Suma unetih brojeva je %d.\n", suma);
22
23 }

```

```
16     suma += x;
18
19     /* Uvecavanje broja ucitanih brojeva. */
20     broj_brojeva++;
21 }
22
23 /* Ispis rezultata. Napomena: I suma i broj brojeva su celi
24    brojevi pa je neophodno bar jednu od te dve vrednosti pretvoriti
25    u realan broj kako deljenje ne bi bilo celobrojno. */
26 if (broj_brojeva == 0)
27     printf("Nisu uneti brojevi.\n");
28 else
29     printf("Aritmeticka sredina: %.4f\n",
30            (double) suma / broj_brojeva);
31
32 return 0;
33 }
```

Rešenje 1.5.17

```
#include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     float cena, suma = 0;
6     int broj_artikla = 0;
7
8     /* Ucitavanje cena sve do unosa broja 0. */
9     printf("Unesite cene: ");
10    while (1) {
11        scanf("%f", &cena);
12
13        if (cena == 0)
14            break;
15
16        /* Provera ispravnosti ulaza. */
17        if (cena < 0) {
18            printf("Greska: neispravan unos cene.\n");
19            return 1;
20        }
21
22        /* Uvecavanje sume za vrednost unete cene. */
23        suma += cena;
24
25        /* Uvecavanje broja unetih artikala za 1. */
26        broj_artikla++;
27    }
28
29    /* Ispis rezultata. */
30    if (broj_artikla == 0)
31        printf("Nisu unete cene.\n");
```

```

32     else
33         printf("Prosecna cena: %.4f\n", suma / broj_artikla);
34
35     return 0;
36 }
```

Rešenje 1.5.18

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, i, broj_promena = 0;
6     double prethodni, trenutni;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n ");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    printf("Unesite brojeve: ");
19    /* Provera promene znaka se vrsti za svaka dva susedna uneta
20       broja. Prvi broj se ucitava pre petlje i smesta se u
21       promenljivu prethodni. Zatim se u petlji ucitava drugi i
22       njihov znak se poredi. Postupak se ponavlja za sve parove,
23       tako sto se uvek na kraju petlje poslednji ucitani broj
24       postavi da bude prethodni za sledecu iteraciju. */
25    scanf("%lf", &prethodni);
26
27    /* Kako je vec jedan broj unet, brojac se postavlja na 1, a ne na
28       0. */
29    for (i = 1; i < n; i++) {
30        /* Ucitavanje broja. */
31        scanf("%lf", &trenutni);
32
33        /* Provera da li je doslo do promene znaka izmedju prethodnog
34           i trenutnog broja. Oni su razlicitog znaka ako vazi:
35           1. da im je proizvod negativan ILI
36           2. da im je proizvod nula, a jedan od njih je negativan. */
37        if (prethodni * trenutni < 0)
38            broj_promena++;
39        else if (prethodni * trenutni == 0 &&
40                  (prethodni < 0 || trenutni < 0))
41            broj_promena++;
42
43    /* Trenutni broj postaje prethodni za sledecu iteraciju. */
```

```
45     prethodni = trenutni;
46 }
47 /* Ispis rezultata. */
48 printf("Broj promena je %d.\n", broj_promena);
49
50 return 0;
51 }
```

Rešenje 1.5.19

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, i;
6     float cena, min_cena;
7
8     /* Ucitavanje broja artikala. */
9     printf("Unesite broj artikala:");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    printf("Unesite cene artikala:");
19
20    /* Inicijalizacija minimalne cene na vrednost cenu prvog artikla.
21       Zbog toga se cena prvog artikla ucitava pre petlje. */
22    scanf("%f", &cena);
23    if (cena <= 0) {
24        printf("Greska: neispravan unos cene.\n");
25        return 1;
26    }
27    min_cena = cena;
28
29    /* Ucitavanje i preostalih n-1 cena i racunanje najmanje. */
30    for (i = 1; i < n; i++) {
31        scanf("%f", &cena);
32
33        if (cena <= 0) {
34            printf("Greska: neispravan unos cene.\n");
35            return 1;
36        }
37
38        if (cena < min_cena)
39            min_cena = cena;
40        i++;
41    }
42
43    printf("Minimalna cena je %.2f.\n", min_cena);
44}
```

```

41 }
43 /* Ispis rezultata. */
44 printf("Najmanja cena: %f\n", min_cena);
45
46 return 0;
47 }
```

Rešenje 1.5.20

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     float cena, m;
6     unsigned int n, i, broj_artikala = 0;
7
8     /* Ucitavanje vrednosti budzeta. */
9     printf("Nikolin budzet: ");
10    scanf("%f", &m);
11
12    /* Ucitavanje broja artikala. */
13    printf("Unesite broj artikala: ");
14    scanf("%u", &n);
15
16    /* Ucitavanje cena artikala i racunanje rezultata. */
17    printf("Unesite cene artikala: ");
18    for (i = 0; i < n; i++) {
19        /* Ucitavanje cene artikla. */
20        scanf("%f", &cena);
21
22        /* Provera da li Nikola moze da kupi trenutni artikal. */
23        if (cena <= m)
24            broj_artikala++;
25    }
26
27    /* Ispis rezultata. */
28    printf("Ukupno artikala: %d\n", broj_artikala);
29
30    return 0;
31 }
```

Rešenje 1.5.21*Rešenje (a)*

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
```

```
int main() {
    /* Deklaracije potrebnih promenljivih. */
    int n, i, x, rezultat;
    int x_desetica, najveca_desetica;

    /* Ucitavanje vrednosti broja n. */
    printf("Unesite broj n: ");
    scanf("%d", &n);

    /* Provera ispravnosti ulaza. */
    if (n < 0) {
        printf("Greska: neispravan unos.\n");
        return 1;
    }

    /* Ako nema unetih brojeva, ispisuje se odgovarajuca poruka. */
    if (n == 0) {
        printf("Nisu uneti brojevi.\n");
        return 0;
    }

    printf("Unesite brojeve: ");

    /* Prvi broj se ucitava pre petlje zbog ispravne
       inicijalizacije. */
    scanf("%d", &x);
    /* Promenljiva najveca_desetica se postavlja na cifru desetica
       ucitanog broja. Napomena: Pri racunanju se koristi absolutna
       vrednost broja jer je npr. (-123/10) = -12 i -12 % 10 = -2, a
       cifra desetica treba da bude 2. */
    najveca_desetica = (abs(x) / 10) % 10;
    /* Kako je na kraju potrebno ispisati broj cija je cifra desetica
       najveca, trenutna vrednost rezultata se postavlja na vrednost
       ucitanog broja. */
    rezultat = x;

    /* Ucitavanje preostalih n-1 brojeva i u slucaju nailaska na
       broj cija je cifra desetica veca od trenutno najvece, vrsi se
       azuriranje najvece desetice i rezultata. */
    for (i = 1; i < n; i++) {
        scanf("%d", &x);

        x_desetica = (abs(x) / 10) % 10;

        if (x_desetica > najveca_desetica) {
            najveca_desetica = x_desetica;
            rezultat = x;
        }
    }

    /* II nacin: Inicijalizacija najvece desetice na neku vrednost
       koja je sigurno manja od svih vrednosti koje cifra desetica
```

```

57     moze da uzme (dakle, bilo sta sto je manje od 0 jer cifra
58     desetica moze imati vrednosti izmedju 0 i 9). Zatim se u
59     petlji izracunava rezultat, analogno prvom nacinu.
60
61     najveca_desetica = -1;
62     for(i=0; i<n; i++) {
63         scanf("%d", &x);
64         x_desetica = (abs(x) / 10) % 10;
65         if (x_desetica > najveca_desetica) {
66             najveca_desetica = x_desetica;
67             rezultat = x;
68         }
69     } */
70
71     /* Ispis rezultata. */
72     printf("Broj sa najvecom cifrom desetica: %d\n", rezultat);
73
74     return 0;
75 }
```

Rešenje (b)

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int n, i;
7     int x, x_kopija, broj_cifara;
8     int najveci_broj_cifara, rezultat;
9
10    /* Ucitavanje vrednosti broja n. */
11    printf("Unesite broj n: ");
12    scanf("%d", &n);
13
14    /* Provera ispravnosti ulaza. */
15    if (n < 0) {
16        printf("Greska: neispravan unos.\n");
17        return 1;
18    }
19
20    /* Ako nema unetih brojeva, ispisuje se odgovarajuca poruka. */
21    if (n == 0) {
22        printf("Nisu uneti brojevi.\n");
23        return 0;
24    }
25
26    /* Postavljanje maksimalnog broja cifara na 0. Ovo je ispravno
27     jer svaki broj ima vise od 0 cifara. */
28    najveci_broj_cifara = 0;
```

1 Osnovni elementi imperativnog programiranja

```
30 printf("Unesite n brojeva: ");
31 for (i = 0; i < n; i++) {
32     scanf("%d", &x);
33
34     /* Racunanje broja cifara unetog broja x. */
35     x_kopija = abs(x);
36     broj_cifara = 0;
37     do {
38         broj_cifara++;
39         x_kopija = x_kopija / 10;
40     } while (x_kopija != 0);
41
42     /* Ako je broj cifara unetog broja veci od najveceg broja
43      cifara, azuriraju se vrednosti najveceg broja cifara i
44      tekuceg rezultata. */
45     if (broj_cifara > najveci_broj_cifara) {
46         najveci_broj_cifara = broj_cifara;
47         rezultat = x;
48     }
49 }
50
51 /* Ispis rezultata. */
52 printf("Najvise cifara ima broj %d.\n", rezultat);
53
54 return 0;
}
```

Rešenje (c)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int n, i;
7     int x, x_kopija, vodeca_cifra;
8     int najveca_vodeca_cifra, rezultat;
9
10    /* Ucitavanje vrednosti broja n. */
11    printf("Unesite broj n: ");
12    scanf("%d", &n);
13
14    /* Provera ispravnosti ulaza. */
15    if (n < 0) {
16        printf("Greska: neispravan unos.\n");
17        return 1;
18    }
19
20    /* Ako nema unetih brojeva, ispisuje se odgovarajuca poruka. */
21    if (n == 0) {
22        printf("Nisu uneti brojevi.\n");
23    }
24 }
```

```

23     return 0;
24 }
25
26 /* Inicijalizacija najveće vodeće cifre na -1. */
27 najveca_vodeca_cifra = -1;
28
29 printf("Unesite n brojeva: ");
30 for (i = 0; i < n; i++) {
31     scanf("%d", &x);
32
33     /* Racunanje vodeće cifre ucitanog broja x. */
34     x_kopija = abs(x);
35     while (x_kopija >= 10) {
36         x_kopija = x_kopija / 10;
37     }
38     vodeca_cifra = x_kopija;
39
40     /* Ako je izdvajena cifra veća od najveće vodeće cifre,
41      azuriraju se vrednosti najveće vodeće cifre i rezultata. */
42     if (vodeca_cifra > najveca_vodeca_cifra) {
43         najveca_vodeca_cifra = vodeca_cifra;
44         rezultat = x;
45     }
46 }
47
48 /* Ispis rezultata. */
49 printf("%d\n", rezultat);
50
51 return 0;
52 }
```

Rešenje 1.5.22 Pogledajte zadatak 1.5.19.

Rešenje 1.5.23

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int n, d, i;
7     int x, y, broj_parova = 0;
8
9     /* Ucitavanje vrednosti n i d. */
10    printf("Unesite brojeve n i d: ");
11    scanf("%d %d", &n, &d);
12
13    /* Provera ispravnosti ulaza. */
14    if (n <= 1 || d < 0) {
15        printf("Greska: neispravan unos.\n");
16        return 1;
17    }
18 }
```

```
17    }
18
19    printf("Unesite n brojeva: ");
20
21    /* Prvi broj se ucitava pre petlje. */
22    scanf("%d", &x);
23
24    for (i = 1; i < n; i++) {
25        scanf("%d", &y);
26
27        /* Provera da li su brojevi na rastojanju d. */
28        if (abs(y - x) == d)
29            broj_parova++;
30
31        /* Broj iz tekuce iteracije se cuva kako bi mogao da se
32           upotrebljava u narednoj iteraciji. */
33        x = y;
34    }
35
36    /* Ispis rezultata. */
37    printf("Broj parova: %d\n", broj_parova);
38
39    return 0;
40}
```

Rešenje 1.5.24

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, cifra, pozicija;
6     unsigned int rezultat;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj: ");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    /* Inicijalizacija pozicije i rezultata. Pozicija označava tezinu
19       trenutne cifre i može imati vrednosti 1, 10, 100, 1000, ... */
20    pozicija = 1;
21    rezultat = 0;
22
23    /* Izdvajanje cifre po cifre broja sve dok ima neobradjenih
24       cifara. */
```

```

25 while (n > 0) {
26     /* Izdvajanje poslednje cifre iz zapisa. U slučaju da je njena
27        vrednost paran broj, izdvojena cifra se uvećava za 1. */
28     cifra = n % 10;
29     if (cifra % 2 == 0)
30         cifra++;
31
32     /* Novi broj se formira tako što se izdvojena cifra pomnoži
33        odgovarajućom tezinom (stepenom) pozicije i doda na trenutni
34        rezultat. */
35     rezultat += cifra * pozicija;
36
37     /* Uklanjanje poslednje cifre broja. */
38     n /= 10;
39
40     /* Množenje pozicije sa 10. */
41     pozicija *= 10;
42 }
43
44 /* Ispis izračunate vrednosti. */
45 printf("Rezultat: %d\n", rezultat);
46
47 return 0;
}

```

Rešenje 1.5.25

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int x, pozicija, rezultat, cifra;
7     int znak = 1;
8
9     /* Ucitavanje vrednosti polaznog broja. */
10    printf("Unesite broj: ");
11    scanf("%d", &x);
12
13    /* Ako je broj negativan, koristi se njegova absolutna vrednost
14       i azurira se vrednost znaka broja. */
15    if (x < 0) {
16        x = abs(x);
17        znak = -1;
18    }
19
20    /* Pozicija označava tezinu trenutne cifre rezultata i može imati
21       vrednosti 1, 10, 100, ... */
22    pozicija = 1;
23    rezultat = 0;

```

1 Osnovni elementi imperativnog programiranja

```
25  /* Ideja: u rezultatu se zadrzavaju cifre jedinice, stotine,.. Na
26   primer, x=12345 Pre petlje: pozicija = 1, rezultat = 0 1.
27   iteracija: cifra = 5, rezultat = 0+5*1=5, x = 123, pozicija =
28   10 2. iteracija: cifra = 3, rezultat = 5+3*10 = 35, x = 1,
29   pozicija = 100 3. iteracija: cifra = 1, rezultat = 35+1*100,
30   x = 0, pozicija = 1000 Petlja se zavrsava jer x ima vrednost
31   0. */
32  while (x > 0) {
33      /* Izdvajanje poslednje cifre. */
34      cifra = x % 10;
35
36      /* Rezultat se uvecava za vrednost cifre pomnozene vrednoscu
37       tezine njene pozicije u broju. */
38      rezultat += cifra * pozicija;
39
40      /* Uklanjanje poslednje dve cifre polaznog broja jer u rezultatu
41       treba da ostane svaka druga cifra. */
42      x /= 100;
43
44      /* Mnozenje pozicije sa 10, kako bi imala ispravnu vrednost u
45       sledecoj iteraciji. */
46      pozicija *= 10;
47  }
48
49  /* Ispis rezultata */
50  printf("Rezultat: %d\n", znak * rezultat);
51
52  return 0;
53 }
```

Rešenje 1.5.26

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, c1, c2, c3;
6     int pozicija, rezultat;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj: ");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    /* Ako broj nema bar tri cifre, rezultat ima vrednost unetog
19     broja. */
```

```

11   if (n <= 99) {
21     printf("Rezultat: %d\n", n);
22     return 0;
23   }

25   /* Izdvajanje poslednje tri cifre polaznog broja. */
26   c1 = n % 10;
27   c2 = (n / 10) % 10;
28   c3 = (n / 100) % 10;

29   /* Poslednja cifra se uvek nalazi u rezultatu jer ona nema oba
30    suseda. Zato se rezultat inicijalizuje na poslednju cifru, a
31    pozicija na 10. */
32   rezultat = c1;
33   pozicija = 10;

34   /* Petlja se izvrsava dok god broj ima bar tri cifre. */
35   while (n > 99) {
36     /* Provera da li c2 treba da se nadje u rezultatu. Ako
37      treba, rezultat se uvecava za vrednost cifre pomnozene
38      vrednoscu tezine njene pozicije u rezultatu i tezina
39      pozicije se mnozi sa 10. */
40     if (c2 != c1 + c3) {
41       rezultat += c2 * pozicija;
42       pozicija *= 10;
43     }

44     /* Vrsi se pomeranje na sledece tri cifre polaznog broja. Iz
45      polaznog broja se briše poslednja cifra. Prva i druga cifra
46      su vec izracunate, samo se vrsi njihovo premestanje iz c2 i
47      c3 u c1 i c2. Cifra c3 se racuna. */
48     n = n / 10;
49     c1 = c2;
50     c2 = c3;
51     c3 = (n / 100) % 10;
52   }

53   /* Po zavrsetku petlje, broj n je dvocifren i njegova cifra
54    desetica odgovara vodecoj cifri polaznog broja. Vodeca cifra
55    polaznog broja uvek treba da se nadje u rezultatu jer nema oba
56    suseda i iz tog razloga se dodaje na tekuci rezultat. */
57   rezultat += (n / 10) * pozicija;

58   /* Ispis rezultata. */
59   printf("Rezultat: %d\n", rezultat);

60   return 0;
61 }

```

Rešenje 1.5.27

```
1 #include <stdio.h>

3 int main() {
    /* Deklaracija potrebnih promenljivih. */
    int x, x_kopija, x_obrnuto;

    /* Ucitavanje vrednosti pocetnog broja. */
    printf("Unesite broj: ");
    scanf("%d", &x);

    /* Racunanje absolutne vrednosti unetog broja. */
    if (x < 0)
        x = -x;

    /* Racunanje broja koji se dobije kada se broju x obrnu cifre. Na
       primer, od 12345 treba da se dobije 54321. Broj se obrce tako
       sto se u svakoj iteraciji njegova vrednost pomnozi sa 10 i
       doda mu se sledeca cifra polaznog broja.
       Za x_kopija=12345, x_obrnuto = 0
       1. iteracija: x_obrnuto = 0*10 + 5 = 5, x_kopija = 1234
       2. iteracija: x_obrnuto = 5*10 + 4 = 54, x_kopija = 123,
       3. iteracija: x_obrnuto = 54*10 + 3 = 543, x_kopija = 12,
       itd. */
    x_kopija = x;
    x_obrnuto = 0;
    while (x_kopija != 0) {
        x_obrnuto = x_obrnuto * 10 + x_kopija % 10;
        x_kopija /= 10;
    }

    /* Broj je palindrom ako je jednak broju koji se dobije
       obrtanjem njegovih cifara. Npr. x = 12321, x_obrnuto je
       takodje 12321. */
    if (x == x_obrnuto)
        printf("Broj je palindrom.\n");
    else
        printf("Broj nije palindrom.\n");

    return 0;
}
```

Rešenje 1.5.28

```
1 #include <stdio.h>

3 int main() {
    /* Deklaracije potrebnih promenljivih. */
    int n, i;
    int fib1 = 0, fib2 = 1, fib3;
```

```

7  /* Ucitavanje vrednosti broja n. */
9  printf("Unesite broj n: ");
10 scanf("%d", &n);
11
12 /* Provera ispravnosti ulaza. */
13 if (n < 0) {
14     printf("Greska: neispravan unos.\n");
15     return 1;
16 }
17
18 /* Ako je n=0, F[0] = 0, slicno ako je n=1 F[1] = 1. */
19 if (n < 2) {
20     printf("F[%d] = %d\n", n, n);
21     return 0;
22 }
23
24 fib3 = fib1 + fib2;
25 for (i = 2; i < n; i++) {
26     /* Pomeranje na sledecu trojku. */
27     fib1 = fib2;
28     fib2 = fib3;
29     fib3 = fib1 + fib2;
30 }
31
32 /* Ispis rezultata. */
33 printf("F[%d] = %d\n", n, fib3);
34
35 return 0;
}

```

Rešenje 1.5.29

```

#include <stdio.h>
2
int main() {
4  /* Deklaracija potrebne promenljive. */
5  int a_n;
6
7  /* Ucitavanje vrednosti prvog clana. */
8  printf("Unesite prvi clan:");
9  scanf("%d", &a_n);
10
11 /* Provera ispravnosti ulaza. */
12 if (a_n <= 0) {
13     printf("Greska: neispravan unos.\n");
14     return 1;
15 }
16
17 /* Dok se ne dodje do clana koji je 1, stampa se vrednost
18   trenutnog clana i vrsti se izracunavanje narednog, po zadatoj

```

```
1     formulii. */
20    printf("Clanovi niza su:\n");
21    while (a_n != 1) {
22        printf("%d ", a_n);
23
24        if (a_n % 2 != 0)
25            a_n = (3 * a_n + 1) / 2;
26        else
27            a_n = a_n / 2;
28    }
29
30    /* Ispis jedinice na kraju. */
31    printf("1\n");
32
33    return 0;
34 }
```

Rešenje 1.5.30

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int format, i;
7     double sirina, duzina, nova_duzina;
8
9     /* Ucitavanje formata papira. */
10    printf("Unesite format papira: ");
11    scanf("%d", &format);
12
13    /* Provera ispravnosti ulaza. */
14    if (format < 0) {
15        printf("Greska: neispravan unos.\n");
16        return 1;
17    }
18
19    /* duzina/sirina = 1/sqrt(2)
20       duzina*sirina = 1000mm x 1000mm =>
21
22       duzina = sirina/sqrt(2)
23       duzina*sirina = 1000mm x 1000mm =>
24
25       sirina*sirina/sqrt(2) = 1000*1000
26       sirina*sirina = sqrt(2) * 1000 * 1000
27       sirina = sqrt(sqrt(2) * 1000 * 1000) =>
28
29       duzina = sirina/sqrt(2) */
30    sirina = sqrt(1000 * 1000 * sqrt(2));
31    duzina = sirina / sqrt(2);
```

```

33  /* Racunanje duzine i sirine za uneti format. */
34  for (i = 1; i <= format; i++) {
35      nova_duzina = sirina / 2;
36      sirina = duzina;
37      duzina = nova_duzina;
38  }
39
40  /* Ispis rezultata. Napomena: Duzina i sirina su celi brojevi. */
41  printf("Dimenzije papira: %d %d\n", (int) duzina, (int) sirina);
42
43  return 0;
}

```

Rešenje 1.5.31

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebne promenljive. */
5     char c;
6
7     /* I nacin ucitavanja: U samom uslovu petlje se vrsti ucitavanje
8      jednog karaktera, njegovo smestanje u promenljivu c i provera
9      da li je ucitani karakter tacka. Zgrade oko (c=getchar()) su
10     obavezne jer relacioni operator != ima veci prioritet od
11     dodele i kada ne bi postojale zgrade, redosled operacija bi
12     bio: (c = (getchar() != '.')), sto znaci da bi se u c smestio
13     rezultat poredjenja, odnosno 0 ili 1. */
14     while ((c = getchar()) != '.') {
15         /* Provera uslova i ispis odgovarajuceg karaktera. */
16         if (c >= 'A' && c <= 'Z')
17             putchar(c + 'a' - 'A');
18         else if (c >= 'a' && c <= 'z')
19             putchar(c - 'a' + 'A');
20         else
21             putchar(c);
22     }
23
24     /* II nacin:
25        while(1) {
26            c = getchar();
27            if(c == '.')
28                break;
29
30            if (c >= 'A' && c <= 'Z')
31                putchar(c + 'a' - 'A');
32            else if (c >= 'a' && c <= 'z')
33                putchar(c - 'a' + 'A');
34            else putchar(c);
35        } */
}

```

1 Osnovni elementi imperativnog programiranja

```
37     return 0;  
}
```

Rešenje 1.5.32

```
#include <stdio.h>  
1  
2 int main() {  
3     /* Deklaracije i inicializacije potrebnih promenljivih. */  
4     char c;  
5     int broj_velikih = 0, broj_malih = 0;  
6     int broj_cifara = 0, suma_cifara = 0, broj_belina = 0;  
7  
8     /* Petlja se zavrsava kada korisnik zada konstantu koja označava  
9      kraj ulaza (EOF konstantu). Ova konstanta se zadaje kombinacijom  
10     tastera CTRL+D i ima vrednost -1. */  
11     printf("Unesite tekst:\n");  
12     while ((c = getchar()) != EOF) {  
13         if (c >= 'A' && c <= 'Z')  
14             broj_velikih++;  
15         else if (c >= 'a' && c <= 'z')  
16             broj_malih++;  
17         else if (c >= '0' && c <= '9') {  
18             broj_cifara++;  
19             suma_cifara = suma_cifara + c - '0';  
20         } else if (c == '\t' || c == '\n' || c == ' ')  
21             broj_belina++;  
22     }  
23  
24     /* Ispis rezultata. */  
25     printf("Velika slova: %d\nMala slova: %d\n", broj_velikih,  
26            broj_malih);  
27     printf("Cifre: %d\nBeline: %d\n", broj_cifara, broj_belina);  
28     printf("Suma cifara: %d\n", suma_cifara);  
29  
30     return 0;  
}
```

Rešenje 1.5.33

```
1 #include <stdio.h>  
2  
3 int main() {  
4     /* Deklaracije i inicializacije potrebnih promenljivih. */  
5     int n, i;  
6     int broj_a = 0, broj_e = 0, broj_i = 0, broj_o = 0, broj_u = 0;  
7     char c;  
8  
9     /* Ucitavanje broja karaktera. */  
10    printf("Unesite broj n: ");
```

```

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

13    /* Provera ispravnosti ulaza. */
14    if (n < 0) {
15        printf("Greska: neispravan unos.\n");
16        return 1;
17    }

19    /* Kako je korisnik nakon unosa broja n uneo oznaku za novi red,
20     potrebno je preskociti taj novi red jer bi u suprotnom on bio
21     ucitan kao prvi od n karaktera (oznaka za novi red je
22     regularan karakter kao sto je to 'a' ili ' '). */
23    getchar();

25    /* Ucitavanje karaktera i brojanje samoglasnika. */
26    for (i = 0; i < n; i++) {
27        scanf("%c", &c);

29        switch (c) {
30            case 'a':
31            case 'A':
32                broj_a++;
33                break;
34            case 'e':
35            case 'E':
36                broj_e++;
37                break;
38            case 'i':
39            case 'I':
40                broj_i++;
41                break;
42            case 'o':
43            case 'O':
44                broj_o++;
45                break;
46            case 'u':
47            case 'U':
48                broj_u++;
49                break;
50        }
51    }

53    /* Ispis rezultata. */
54    printf("Samoglasnik a: %d\n", broj_a);
55    printf("Samoglasnik e: %d\n", broj_e);
56    printf("Samoglasnik i: %d\n", broj_i);
57    printf("Samoglasnik o: %d\n", broj_o);
58    printf("Samoglasnik u: %d\n", broj_u);

59
60    return 0;
61 }

```

Rešenje 1.5.34

```
1 #include <stdio.h>

3 int main() {
4     /* Deklaracije i inicializacije potrebnih promenljivih. */
5     int n, i, broj_Z = 0, broj_i = 0, broj_m = 0, broj_a = 0;
6     char novi_red, c;
7
8     /* Ucitavanje broja karaktera. */
9     printf("Unesite broj n: ");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    /* Ucitavanje karaktera. */
19    for (i = 1; i <= n; i++) {
20        printf("Unestite %d. karakter: ", i);
21
22        /* Prvo se cita belina koja se nalazi nakon prethodnog unosa,
23         pa tek posle procitane beline se cita uneti karakter. */
24        scanf("%c%c", &novi_red, &c);
25
26        switch (c) { /* Obrada ucitanog karaktera. */
27            case 'Z':
28                broj_Z++;
29                break;
30            case 'i':
31                broj_i++;
32                break;
33            case 'm':
34                broj_m++;
35                break;
36            case 'a':
37                broj_a++;
38                break;
39        }
40    }
41
42    /* Ako su svi brojac razliciti od nule, rec "Zima" se moze
43     napisati pomocu unetih karaktera. */
44    if (broj_Z && broj_i && broj_m && broj_a)
45        printf("Moze se napisati rec Zima.\n");
46    else
47        printf("Ne moze se napisati rec Zima.\n");
48
49    return 0;
50 }
```

Rešenje 1.5.35

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int n, i, suma_kubova;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n:");
9     scanf("%d", &n);
10
11    /* Provera ispravnosti ulaza. */
12    if (n <= 0) {
13        printf("Greska: neispravan unos.\n");
14        return 1;
15    }
16
17    /* Racunanje sume kubova svih brojeva iz intervala [1,n]. */
18    suma_kubova = 0;
19    for (i = 1; i <= n; i++)
20        suma_kubova += i * i * i;
21
22    /* Ispis rezultata. */
23    printf("Suma kubova: %d\n", suma_kubova);
24
25    return 0;
}

```

Rešenje 1.5.36 Pogledajte zadatak 1.5.35.**Rešenje 1.5.37**

```

1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     int n, i;
6     float x, suma, x_i;
7
8     /* Ucitavanje vrednosti x i n. */
9     printf("Unesite redom brojeve x i n: ");
10    scanf("%f %d", &x, &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0 || x == 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    /* Inicijalizacija vrednosti sume na 0, a vrednosti x^i na x. */
19
20    for (i = 1; i <= n; i++) {
21        suma += x * x;
22        x *= x;
23    }
24
25    printf("Suma: %f\n", suma);
}

```

```
19    suma = 0;
20    x_i = x;
21
22    /* Promenljiva x^i ima vrednosti [x, x^2, ..., x^n]. Vrednost
23       sume se u svakoj iteraciji uvecava za i*x^i. */
24    for (i = 1; i <= n; i++) {
25        suma += i * x_i;
26        x_i *= x;
27    }
28
29    /* Ispis rezultata. */
30    printf("S = %f\n", suma);
31
32    return 0;
33 }
```

Rešenje 1.5.38 Pogledajte zadatak 1.5.37.

Rešenje 1.5.39

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

4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int i;
7     float suma, clan, x, eps;
8
9     /* Ucitavanje vrednosti x i eps. */
10    printf("Unesite x: ");
11    scanf("%f", &x);
12
13    printf("Unesite tacnost eps: ");
14    scanf("%f", &eps);
15
16    /* Inicijalizacija sume, prvog clana i brojaca. */
17    suma = 0;
18    clan = 1;
19    i = 1;
20
21    /* U svakoj iteraciji na sumu se dodaje prethodno izracunati clan
22       sume i zatim se racuna sledeci clan. Petlja se prekida kada
23       vrednost sledeceg clana postane manja ili jednaka eps. */
24    while (clan > eps) {
25        suma += clan;
26        clan = clan * x / i;
27        i++;
28    }
29
30    /* Ispis rezultata. */
31    printf("S = %f\n", suma);
```

```

32     return 0;
33 }
```

Rešenje 1.5.40

```

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

int main() {
    /* Deklaracije potrebnih promenljivih. */
    int i;
    float suma, x, eps, clan;

    /* Ucitavanje vrednosti x i eps. */
    printf("Unesite x: ");
    scanf("%f", &x);

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

    /* Inicijalizacije. */
    suma = 0;
    clan = 1;
    i = 1;

    /* Kako clanovi sume mogu biti negativni, potrebno je posmatrati
       apsolutnu vrednost clana. */
    while (fabs(clan) > eps) {
        suma += clan;

        /* U svakoj iteraciji se racuna novi clan i mnozi se sa -1. Na
           taj nacin se postize da je vrednost clana naizmenicno
           pozitivna i negativna. */
        clan = clan * x / i;
        clan *= -1;

        i++;
    }

    /* Ispis rezultata. */
    printf("S = %f\n", suma);

    return 0;
}
```

Rešenje 1.5.41

```
1 #include <stdio.h>
2 #include <math.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int n, i;
7     double x, x_i, proizvod;
8
9     /* Ucitavanje vrednosti x i n. */
10    printf("Unesite redom brojove x i n: ");
11    scanf("%lf %d", &x, &n);
12
13    /* Provera ispravnosti ulaza. */
14    if (n <= 0) {
15        printf("Greska: neispravan unos.\n");
16        return 1;
17    }
18
19    /* Racunanje trazenog proizvoda. */
20    x_i = 1;
21    proizvod = 1;
22    for (i = 0; i < n; i++) {
23        x_i *= x;
24        proizvod *= 1 + cos(x_i);
25    }
26
27    /* Ispis rezultata. */
28    printf("P = %lf\n", proizvod);
29
30    return 0;
31}
```

Rešenje 1.5.42

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, i;
6     double razlomak;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
```

```

17    }
18
19    /* Razlomak se izracunava "od nazad", odnosno, kreće se od
20     najnizeg razlomka  $1/n$  i od njega se nadalje formira sledeći,
21     "visi" razlomak itd. Završava se kada se stigne do koraka  $0 +$ 
22      $1/R.$  */
23    razlomak = n;
24    for (i = n - 1; i >= 0; i--)
25        razlomak = i + 1 / razlomak;
26
27    /* Ispis rezultata. */
28    printf("R = %lf\n", razlomak);
29
30    return 0;
31}

```

Rešenje 1.5.43

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int i, n;
6     float suma, x, clan;
7
8     /* Ucitavanje vrednosti x i n. */
9     printf("Unesite redom brojeve x i n: ");
10    scanf("%f%d", &x, &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    /* Inicijalizacije. */
19    suma = 1;
20    clan = 1;
21    i = 2;
22
23    /* Racunanje trazene sume. */
24    while (i <= 2 * n) {
25        /* Saki clan sume se od prethodnog clana razlikuje za
26          $x^2/(i*(i-1)).$  */
27        clan = clan * x * x / (i * (i - 1));
28        clan *= -1;
29        suma += clan;
30        i += 2;
31    }
32
33    /* Ispis rezultata. */

```

```
35     printf("S = %f\n", suma);
36
37 }
```

Rešenje 1.5.44

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, i;
6     double clan, proizvod = 1;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%d", &n);
11
12    /* Provera ispravnosti ulaza. */
13    if (n <= 0) {
14        printf("Greska: neispravan unos.\n");
15        return 1;
16    }
17
18    /* Racunanje trazenog proizvoda. */
19    clan = 1;
20    for (i = 2; i <= n; i++) {
21        clan = clan / i;
22        proizvod *= 1 + clan;
23    }
24
25    /* Ispis rezultata. */
26    printf("P = %lf\n", proizvod);
27
28    return 0;
29 }
```

Rešenje 1.5.45

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, i;
6     long int clan, suma = 0;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%d", &n);
11
```

```

13  /* Provera ispravnosti ulaza. */
14  if (n < 5 || n % 2 == 0) {
15      printf("Greska: neispravan unos.\n");
16      return 1;
17  }
18
19  /* Racunanje trazene sume. */
20  clan = -1 * 3;
21  for (i = 5; i <= n; i += 2) {
22      clan = -1 * clan * i;
23      suma += clan;
24  }
25
26  /* Ispis rezultata. */
27  printf("S = %ld\n", suma);
28
29  return 0;
}

```

Rešenje 1.5.46

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     int n, i;
6     double x, a, rezultat;
7
8     /* Ucitavanje vrednosti ulaznih promenljivih. */
9     printf("Unesite brojeve x i a: ");
10    scanf("%lf%lf", &x, &a);
11    printf("Unesite broj n: ");
12    scanf("%d", &n);
13
14    /* Provera ispravnosti ulaza. */
15    if (n <= 0) {
16        printf("Greska: neispravan unos.\n");
17        return 1;
18    }
19
20    /* Racunanje vrednosti zadatog izraza. Kreće se od
21       rezultat = (x + a) ^ 2 i ide se ka spolja.
22       Svaki put vrednost rezultata treba zameniti sa
23       (rezultat + a) ^ 2. */
24    rezultat = x;
25    for (i = 0; i < n; i++)
26        rezultat = (rezultat + a) * (rezultat + a);
27
28    /* Ispis rezultata. */
29    printf("Izraz = %lf\n", rezultat);
30

```

1 Osnovni elementi imperativnog programiranja

```
    return 0;  
32 }
```

Rešenje 1.5.47

Rešenje (a)

```
#include <stdio.h>  
2  
int main() {  
4     /* Deklaracija potrebnih promenljivih. */  
    unsigned int n, i, j;  
6  
    /* Ucitavanje vrednosti broja n. */  
8    printf("Unesite broj n: ");  
    scanf("%u", &n);  
10  
    /* Ispis tablice mnozenja dimenzije n*n. */  
12    for (i = 1; i <= n; i++) {  
        for (j = 1; j <= n; j++) {  
            /* Vrednost svakog polja je proizvod vrste i kolone. */  
            printf("%3d ", i * j);  
        }  
        /* Na kraju svake vrste se ispisuje novi red. */  
18        printf("\n");  
    }  
20  
    return 0;  
22 }
```

Rešenje (b)

```
#include <stdio.h>  
2  
int main() {  
4     /* Deklaracija potrebnih promenljivih. */  
    unsigned int n, i, j;  
6  
    /* Ucitavanje vrednosti broja n. */  
8    printf("Unesite broj n: ");  
    scanf("%u", &n);  
10  
    /* Brojac koji broji koliko brojeva je ispisano u jednom redu. */  
12    j = 0;  
    for (i = 1; i <= n * n; i++) {  
        printf("%3d ", i);  
16  
        j++;  
        /* Kada je ispisano n brojeva u jednom redu, ispisuje se znak
```

```

18     za novi red, da bi ispis krenuo u novom redu i vrednost
19     brojaca j se postavlja na 0 jer u novom redu jos ni jedan
20     broj nije ispisano. */
21     if (j == n) {
22         j = 0;
23         printf("\n");
24     }
25
26     return 0;
27 }
```

Rešenje (c)

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11    /* Ispis trazene tablice. */
12    for (i = 1; i <= n; i++) {
13        for (j = 0; j < n; j++)
14            if ((j + i) % n == 0)
15                printf("%3d", n);
16            else
17                printf("%3d", (j + i) % n);
18
19        printf("\n");
20    }
21
22    return 0;
23 }
```

Rešenje (d)

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
```

```
11  /* Ispis traženog trougla. */
12  for (i = 0; i < n; i++) {
13      for (j = 0; j < n - i; j++)
14          printf("(%d, %d)", i, j);
15
16      printf("\n");
17  }
18
19  return 0;
}
```

Rešenje 1.5.48

Rešenje (a)

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11    /* Kvadrat predstavlja tabelu sa n vrsta gde svaka vrsta sadrzi n
12       polja i u svakom polju je upisana zvezdica. */
13    for (i = 0; i < n; i++) {
14        for (j = 0; j < n; j++)
15            printf("*");
16        printf("\n");
17    }
18
19    return 0;
20 }
```

Rešenje (b)

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
```

```

12  /* Kvadrat predstavlja tabelu sa n vrsta i n kolona u kojoj se
13   na ivicama nalaze zvezdice, a u unutrasnjosti praznine. */
14  for (i = 0; i < n; i++) {
15    for (j = 0; j < n; j++) {
16      /* Provera da li je u pitanju ivica. */
17      if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
18        printf("*");
19      else
20        printf(" ");
21    }
22    printf("\n");
23  }
24
25  return 0;
}

```

Rešenje (c)

```

1 #include <stdio.h>

3 int main() {
4   /* Deklaracija potrebnih promenljivih. */
5   unsigned int n, i, j;
6
7   /* Ucitavanje vrednosti broja n. */
8   printf("Unesite broj n: ");
9   scanf("%u", &n);
10
11  /* Kvadrat predstavlja tabelu sa n vrsta i n kolona u kojoj se
12   na ivicama i glavnoj dijagonali nalaze zvezdice, a na ostalim
13   mestima praznine. */
14  for (i = 0; i < n; i++) {
15    for (j = 0; j < n; j++) {
16      /* Provera da li je ivica ili glavna dijagonala. */
17      if (j == 0 || j == n - 1 || i == 0 || i == n - 1 || i == j)
18        printf("*");
19      else
20        printf(" ");
21    }
22    printf("\n");
23  }
24
25  return 0;
}

```

Rešenje 1.5.49

```

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

```

1 Osnovni elementi imperativnog programiranja

```
/* Deklaracija potrebnih promenljivih. */
5 unsigned int n, i, j;

7 /* Ucitavanje vrednosti broja n. */
printf("Unesite broj n: ");
9 scanf("%u", &n);

11 /* Veliko slovo X se dobija tako sto se na dijagonalama kvadrata
   ispisuju zvezdice, a na ostalim mestima blanko. */
13 for (i = 0; i < n; i++) {
    for (j = 0; j < n; j++) {
        /* Provera da li je mesto glavne ili sporedne dijagonale. */
        if (i == j || i + j == n - 1)
            printf("*");
        else
            printf(" ");
    }
    printf("\n");
}

23 return 0;
25 }
```

Rešenje 1.5.50

```
#include <stdio.h>

3 int main() {
    /* Deklaracija potrebnih promenljivih. */
    5 unsigned int n, i, j;

    7 /* Ucitavanje vrednosti broja n. */
    printf("Unesite broj n: ");
    9 scanf("%u", &n);

    11 /* Isrtavanje znaka plus ispisom karaktera '+' na sredisnjoj
       vrsti i koloni, a blanko karaktera na ostalim mestima. */
    13 for (i = 0; i < n; i++) {
        for (j = 0; j < n; j++)
            15 if (i == n / 2 || j == n / 2)
                printf("+");
            else
                17 printf(" ");
        19 printf("\n");
    }

    21 return 0;
23 }
```

Rešenje 1.5.51*Rešenje (a)*

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11    /* IsCRTavanje traZenog trougla. */
12    for (i = 0; i < n; i++) {
13        for (j = 0; j < n - i; j++)
14            printf("*");
15        printf("\n");
16    }
17
18    return 0;
19 }
```

Rešenje (b)

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11    /* IsCRTavanje traZenog trougla. */
12    for (i = 0; i < n; i++) {
13        for (j = 0; j <= i; j++)
14            printf("*");
15        printf("\n");
16    }
17
18    return 0;
19 }
```

Rešenje (c)

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11    /* Isrtavanje trazenog trougla. */
12    for (i = 0; i < n; i++) {
13        /* Ispis belina koje prethode zvezdicama. */
14        for (j = 0; j < i; j++)
15            printf(" ");
16        /* Ispis potrebnog broja zvezdica. */
17        for (j = 0; j < n - i; j++)
18            printf("*");
19        printf("\n");
20    }
21
22    return 0;
23 }
```

Rešenje (d)

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11    /* Isrtavanje trazenog trougla. */
12    for (i = 0; i < n; i++) {
13        /* Ispis belina koje prethode zvezdicama. */
14        for (j = 0; j < n - i - 1; j++)
15            printf(" ");
16        /* Ispis potrebnog broja zvezdica. */
17        for (j = 0; j <= i; j++)
18            printf("*");
19        printf("\n");
20    }
21
22    return 0;
23 }
```

23 }

Rešenje (e)

```

1 #include <stdio.h>
3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;
6
7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);
10
11    /* IsCRTavanje gornjeg dela traZenog trougla. */
12    for (i = 0; i < n; i++) {
13        /* Ispis belina koje prethode zvezdicama. */
14        for (j = 0; j < n - i - 1; j++)
15            printf(" ");
16        /* Ispis potrebnog broja zvezdica. */
17        for (j = 0; j <= i; j++)
18            printf("*");
19        printf("\n");
20    }
21
22    /* IsCRTavanje donjeg dela traZenog trougla. */
23    for (i = 1; i < n; i++) {
24        /* Ispis belina koje prethode zvezdicama. */
25        for (j = 0; j < i; j++)
26            printf(" ");
27        /* Ispis potrebnog broja zvezdica. */
28        for (j = 0; j < n - i; j++)
29            printf("*");
30        printf("\n");
31    }
32
33    return 0;
}

```

Rešenje (f)

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n, i, j;
6     char c, novi_red;
7
8     /* Ucitavanje vrednosti broja n. */
9
10    /* IsCRTavanje gornjeg dela traZenog trougla. */
11    for (i = 0; i < n; i++) {
12        /* Ispis belina koje prethode zvezdicama. */
13        for (j = 0; j < n - i - 1; j++)
14            printf(" ");
15        /* Ispis potrebnog broja zvezdica. */
16        for (j = 0; j <= i; j++)
17            printf("*");
18        printf("\n");
19    }
20
21    /* IsCRTavanje donjeg dela traZenog trougla. */
22    for (i = 1; i < n; i++) {
23        /* Ispis belina koje prethode zvezdicama. */
24        for (j = 0; j < i; j++)
25            printf(" ");
26        /* Ispis potrebnog broja zvezdica. */
27        for (j = 0; j < n - i; j++)
28            printf("*");
29        printf("\n");
30    }
31
32    return 0;
}

```

1 Osnovni elementi imperativnog programiranja

```
10    printf("Unesite broj n: ");
11    scanf("%u", &n);

12    /* Ucitavanje karaktera koji ce se koristiti za iscrtavanje.
13       Napomena: Voditi racuna da treba preskociti novi red koji
14       korisnik zadaje nakon unosa broja n. */
15    printf("Unesite karakter c: ");
16    scanf("%c%c", &novi_red, &c);

17    /* Iscrtavanje trazenog trougla. Iscrtaju se samo ivice
18       trougla, ostalo se popunjava belinama. */
19    for (i = 0; i < n; i++) {
20        for (j = 0; j <= i; j++)
21            if (i == n - 1 || j == 0 || j == i)
22                printf("%c", c);
23            else
24                printf(" ");
25        printf("\n");
26    }
27
28    return 0;
29}
```

Rešenje 1.5.52

Rešenje (a)

```
1 #include <stdio.h>

3 int main() {
4     /* Deklaracija potrebnih promenljivih. */
5     unsigned int n, i, j;

7     /* Ucitavanje vrednosti broja n. */
8     printf("Unesite broj n: ");
9     scanf("%u", &n);

11    /* Brojac i određuje koji red slike se trenutno ispisuje. */
12    for (i = 0; i < n; i++) {
13        /* Ispis belina koje prethode zvezdicama. */
14        for (j = 0; j < n - i - 1; j++)
15            printf(" ");
16        /* Ispis potrebnog broja zvezdica. */
17        for (j = 0; j < 2 * i + 1; j++)
18            printf("*");
19        printf("\n");
20    }
21
22    return 0;
23}
```

Rešenje (b)

```

1 #include <stdio.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     unsigned int n;
7     int i, j;
8
9     /* Ucitavanje vrednosti broja n. */
10    printf("Unesite broj n: ");
11    scanf("%u", &n);
12
13    /* Brojac i određuje koliko redova se ispisuje. Radi lakseg
14       izracunavanja koliko zvezdica i praznina je potrebno ispisati
15       u svakom redu, i se postavlja na n-1 i smanjuje u svakoj
16       iteraciji petlje. */
17    for (i = n - 1; i >= 0; i--) {
18        /* Ispis belina koje prethode zvezdicama. */
19        for (j = 0; j < n - i - 1; j++)
20            printf(" ");
21        /* Ispis potrebnog broja zvezdica. */
22        for (j = 0; j < 2 * i + 1; j++)
23            printf("*");
24        printf("\n");
25    }
26
27    return 0;
}

```

Rešenje (c)

```

1 #include <stdio.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     unsigned int n;
7     int i, j;
8
9     /* Ucitavanje vrednosti broja n. */
10    printf("Unesite broj n: ");
11    scanf("%u", &n);
12
13    /* Slika se crta iz dva dela. */
14
15    /* Brojac i određuje koji red slike se trenutno ispisuje. */
16    for (i = 0; i < n; i++) {
17        /* Ispis belina koje prethode zvezdicama. */
18        for (j = 0; j < n - i - 1; j++)
19            printf(" ");

```

1 Osnovni elementi imperativnog programiranja

```
19     /* Ispis potrebnog broja zvezdica. */
20     for (j = 0; j < 2 * i + 1; j++)
21         printf("*");
22     printf("\n");
23 }

25 /* II deo: crtanje donjeg trougla. Kako je prvi red donjeg
26    trougla vec ispisan (poslednji red gornjeg trougla), potrebno
27    je naciniti jednu iteraciju manje. */
28 for (i = n - 2; i >= 0; i--) {
29     /* Ispis belina koje prethode zvezdicama. */
30     for (j = 0; j < n - i - 1; j++)
31         printf(" ");
32     /* Ispis potrebnog broja zvezdica. */
33     for (j = 0; j < 2 * i + 1; j++)
34         printf("*");
35     printf("\n");
36 }
37
38 return 0;
39 }
```

Rešenje (d)

```
1 #include <stdio.h>

3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n;
6     int i, j;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Brojac i određuje koji red slike se trenutno ispisuje. */
13    for (i = 0; i < n; i++) {
14        /* Ispis belina koje prethode zvezdicama. */
15        for (j = 0; j < n - i - 1; j++)
16            printf(" ");
17        /* Posle belina se ispisuje sam trougao. Ako je brojac na ivici
18           onda se ispisuje zvezdica, a inace praznina. Takodje,
19           proverava se da li se ispisuje poslednji red (i==n) i u njemu
20           se ispisuje svaki drugi put zvezdica, a inace praznina. */
21        for (j = 0; j < 2 * i + 1; j++)
22            if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
23                printf("*");
24            else
25                printf(" ");
26        printf("\n");
27    }
```

```
29     return 0;
}
```

Rešenje (c)

```
#include <stdio.h>
2
int main() {
4    /* Deklaracije potrebnih promenljivih. */
    unsigned int n;
6    int i, j;
8
/* Ucitavanje vrednosti broja n. */
printf("Unesite broj n: ");
10   scanf("%u", &n);
12
/* Brojac i određuje koji red slike se trenutno ispisuje. */
for (i = 0; i < n; i++) {
14    /* Ispis belina koje prethode zvezdicama. */
    for (j = 0; j < n - i - 1; j++)
        printf(" ");
    /* Ispis trougla. */
18    for (j = 0; j < 2 * i + 1; j++)
        if (j == 0 || j == 2 * i || (i == n - 1 && j % 2 == 0))
            printf("*");
        else
            printf(" ");
        printf("\n");
24}
26
/* II deo: crtanje donjeg trougla. Kako je prvi red donjeg
trougla vec ispisana (poslednji red gornjeg trougla), potrebno
je naciniti jednu iteraciju manje. */
for (i = n - 2; i >= 0; i--) {
30    /* Ispis belina koje prethode zvezdicama. */
    for (j = 0; j < n - i - 1; j++)
        printf(" ");
    /* Ispis potrebnog broja zvezdica. */
34    for (j = 0; j < 2 * i + 1; j++)
        if (j == 0 || j == 2 * i)
            printf("*");
        else
            printf(" ");
        printf("\n");
40}
42
return 0;
}
```

Rešenje 1.5.53

```
1 #include <stdio.h>

3 int main() {
    /* Deklaracije potrebnih promenljivih. */
    unsigned int n;
    int i, j;

7     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);

11     /* Strelica se moze posmatrati kao spojena dva pravouglia trougla
12      kojima se ispisuje hipotenuza i jedna kateta. */

15     /* Brojac i određuje koji red slike se trenutno ispisuje. */
16     for (i = 0; i < n; i++) {
17         for (j = 0; j <= i; j++)
18             /* Provera da li se ispisuje karakter na hipotenuzi (j == i)
19              ili da se ispisuje poslednji red (i == n-1). */
20             if (j == i || i == n - 1)
21                 printf("*");
22             else
23                 printf(" ");
24         printf("\n");
25     }

27     /* II deo: crtanje donjeg dela slike, odnosno donji trougao.
28        Brojac i određuje koji red donjeg trougla se trenutno
29        iscrтava. Kako je prvi red donjeg trougla vec iscrтан (to je
30        poslednji red gornjeg trougla), brojac se postavlja na 1. */
31     for (i = 1; i < n; i++) {
32         for (j = 0; j < n - i; j++)
33             /* Provera da li se ispisuje hipotenuza. */
34             if (j == n - i - 1)
35                 printf("*");
36             else
37                 printf(" ");
38         printf("\n");
39     }

41     return 0;
}
```

Rešenje 1.5.54

```
1 #include <stdio.h>

3 int main() {
    /* Deklaracije potrebnih promenljivih. */
```

```

5     unsigned int n;
6     int i, j, k;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Brojac j određuje koliko se karaktera (praznina i zvezdica)
13       ispisuje u svakom redu.
14       U svakom drugom redu ovaj broj se povecava za 2.
15       Na pocetku je 1 (jer se ispisuje samo jedna zvezdica). */
16    j = 1;
17
18    /* Brojac i određuje koji red slike se trenutno ispisuje. */
19    for (i = 1; i <= n; i++) {
20        /* U svakom drugom redu broj karaktera koji treba da se ispisu
21           se uvecava za 2. */
22        if (i % 2 == 0)
23            j += 2;
24
25        /* Ispisuje se j karaktera. */
26        for (k = 0; k < j; k++)
27            /* U svakom parnom redu se naiatzmenicno ispisuju zvezdica i
28               praznina. */
29            if (i % 2 == 0) {
30                if (k % 2 == 0)
31                    printf("*");
32                else
33                    printf(" ");
34            } else {
35                /* U svakom neparnom redu se ispisuju samo zvezdice. */
36                printf("*");
37            }
38        printf("\n");
39    }
40
41    return 0;
42}

```

Rešenje 1.5.55

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n, m;
6     int i, j;
7
8     /* Ucitavanje dimenzije slike. */
9     printf("Unesite brojve n i m: ");
10    scanf("%u%u", &n, &m);

```

```
12  /* Brojac i određuje koji red slike se trenutno ispisuje. Ukupno
   13   ima m redova. */
14  for (i = 1; i <= m; i++) {
15      /* Brojac j označava koja kolona se trenutno ispisuje. Za
   16       svaki kvadrat se računa dužina bez poslednje ivice. Kvadrat
   17       je sastavljen od (m-1) zvezdice i (m-1) praznine (praznine
   18       se nalaze između zvezdica). Znaci, ukupna dužina je 2*(m-1)
   19       karakter, a kako ima n kvadrata plus jedna kolona za
   20       krajnje desnu ivicu, dužina je n*2*(m-1) + 1. */
21      for (j = 0; j <= n * 2 * (m - 1); j++)
22          /* Provera da li se ispisuje prvi ili poslednji red. */
23          if (i == 1 || i == m)
24              /* Naizmeničan ispis zvezdice i praznine. */
25              if (j % 2 == 0)
26                  printf("*");
27              else
28                  printf(" ");
29          else
30              /* Na ivicama kvadrata se iscrtavaju zvezdice, a na ostalim
   31               mestima beline. */
32              if (j % (2 * (m - 1)) == 0)
33                  printf("*");
34              else
35                  printf(" ");
36
37          printf("\n");
38      }
39
40      return 0;
41 }
```

Rešenje 1.5.56

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n;
6     int i, j;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Romb se crta crtanjem dva spojena trougla koji se nezavisno
13     iscrtavaju. */
14
15    /* Brojac i određuje koji red slike se trenutno ispisuje. */
16    for (i = 0; i < n; i++) {
17        /* Ispis zvezdica koje prethode karakterima -. */
18
19        /* Pisanje zvezdica u red. */
20        for (j = 0; j < i; j++)
21            printf("-");
22
23        /* Pisanje zvezdica u red. */
24        for (j = 0; j < n - i; j++)
25            printf("*");
26
27        /* Pisanje zvezdica u red. */
28        for (j = 0; j < i; j++)
29            printf("-");
30
31        /* Novi red. */
32        printf("\n");
33    }
34
35    return 0;
36 }
```

```

19     for (j = 0; j < n - i; j++)
20         printf("*");
21     /* Ispis karaktera -. */
22     for (j = 0; j < 2 * i; j++)
23         printf("-");
24     /* Ispis zvezdica koje su nakon karaktera -. */
25     for (j = 0; j < n - i; j++)
26         printf("*");
27     printf("\n");
28 }
29
30 /* II deo: crtanje donjeg trougla. Kako je prvi red donjeg
31    trougla vec ispisana (poslednji red gornjeg trougla), potrebno
32    je naciniti jednu iteraciju manje. */
33 for (i = n - 2; i >= 0; i--) {
34     /* Ispis zvezdica koje prethode karakterima -. */
35     for (j = 0; j < n - i; j++)
36         printf("*");
37     /* Ispis karaktera -. */
38     for (j = 0; j < 2 * i; j++)
39         printf("-");
40     /* Ispis zvezdica koje su nakon karaktera -. */
41     for (j = 0; j < n - i; j++)
42         printf("*");
43     printf("\n");
44 }
45 return 0;
}

```

Rešenje 1.5.57

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n;
6     int i, j;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Slika se sastoji iz dva dela, trougla i kvadrata i svaki deo
13       se nezavisno iscrtava. */
14
15    /* I deo: crtanje trougla (krova). */
16    for (i = 0; i < n - 1; i++) {
17        /* Ispis belina koje prethode zvezdicama. */
18        for (j = 0; j < n - i - 1; j++)
19            printf(" ");

```

```
21     /* Ispis trougla. */
22     for (j = 0; j < 2 * i + 1; j++)
23         if (j == 0 || j == 2 * i)
24             printf("*");
25         else
26             printf(" ");
27     printf("\n");
28 }
29
30 /* II deo: crtanje kvadrata. Da bi iscrtavanje bilo lakse
31 istovremeno se ispisuju dva karaktera. */
32 for (i = 0; i < n; i++) {
33     for (j = 0; j < n; j++)
34         /* Provera da li je ivica. */
35         if (j == 0 || j == n - 1 || i == 0 || i == n - 1)
36             printf("* ");
37         else
38             printf(" ");
39     printf("\n");
40 }
41
42 return 0;
43 }
```

Rešenje 1.5.58

```
1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n;
6     int i, j;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Prva petlja označava broj 'serija' koje će se ispisati. Na
13       primer, za n=5, prva serija je 1 2 3 4 5, druga serija je 2 3
14       4 i treća serija je 3. Kako se u svakoj sledećoj seriji broj
15       brojeva smanjuje za 2, do 0 karaktera u seriji se dolazi posle
16       n/2 koraka, ali zaokruzeno navise (5/2 = 2.5 --> 3), a to je
17       isto sto i celobrojno (n+1)/2. */
18    for (i = 1; i <= (n + 1) / 2; i++) {
19        /* U svakoj seriji se ispisuju brojevi izmedju i i n-i+1. */
20        for (j = i; j <= n + 1 - i; j++)
21            printf("%d ", j);
22    }
23
24    /* III nacin:
```

```

25 int levo = 1, desno = n-1;
26 while (levo <= desno) {
27     // Ispis jedne serije.
28     for (j = levo; j <= desno; j++)
29         printf(" %d", j);
30
31     // Pomeranje leve i desne granice.
32     levo++;
33     desno--;
34 } */
35
36 return 0;
37 }

```

Rešenje 1.5.59

```

1 #include <stdio.h>
2
3 int main() {
4     /* Deklaracije potrebnih promenljivih. */
5     unsigned int n;
6     int i, j;
7
8     /* Ucitavanje vrednosti broja n. */
9     printf("Unesite broj n: ");
10    scanf("%u", &n);
11
12    /* Brojac i je redni broj vrste koja se ispisuje. */
13    for (i = 1; i <= n; i++) {
14        /* Ispis brojeva izmedju 1 i n, sa korakom i. */
15        for (j = 1; j <= n; j += i)
16            printf("%d ", j);
17
18        printf("\n");
19    }
20
21    return 0;
22 }

```

1.7 Funkcije

Zadatak 1.7.1 Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji učitava tri cela broja i ispisuje njihov minimum.

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: 19 8 14
|| Minimum: 8

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite brojeve: -6 11 -12
|| Minimum: -12

[Rešenje 1.7.1]

Zadatak 1.7.2 Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja x . Napisati program koji učitava jedan realan broj i ispisuje njegov razlomljeni deo na šest decimala.

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 1.7.2]

Zadatak 1.7.3 Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja n . Napisati program koji učitava ceo pozitivan broj k i ispisuje zbir delilaca svakog broja od 1 do k . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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: neispravan unos.

[Rešenje 1.7.3]

Zadatak 1.7.4 Napisati funkciju `int je_stepen(unsigned x, unsigned n)` koja za dva broja x i n utvrđuje da li je x neki stepen broja n . Ukoliko jeste, funkcija vraća izložilac stepena, a u suprotnom vraća -1 . Napisati program

koji učitava dva neoznačena broja i ispisuje da li vrednost prvog broja odgovara vrednosti nekog stepena drugog broja (i kog).

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 81 3  
|| Jeste: 81 = 3^4
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 162 11  
|| Broj 162 nije stepen broja 11.
```

[Rešenje 1.7.4]

Zadatak 1.7.5 Napisati funkciju `int euklid(int x, int y)` koja za dva data cela broja određuje najveći zajednički delilac primenom Euklidovog algoritma. Napisati program koji učitava dva cela broja i ispisuje vrednost njihovog najvećeg zajedničkog delioca.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva cela broja: 1024 832  
|| Najveci zajednicki delilac: 64
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva cela broja: -900 112  
|| Najveci zajednicki delilac: 4
```

[Rešenje 1.7.5]

Zadatak 1.7.6 Napisati funkciju `float zbir_reciprocnih(int n)` koja za dato n vraća zbir recipročnih vrednosti brojeva od 1 do n . Napisati program koji učitava ceo pozitivan broj n i ispisuje odgovarajući zbir zaokružen na dve decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 10  
|| Zbir reciprocnih: 2.93
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 100  
|| Zbir reciprocnih: 5.19
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -100  
|| Greska: neispravan unos.
```

[Rešenje 1.7.6]

Zadatak 1.7.7 Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj x pojavljuje u nizu brojeva koji se unose sve do unosa broja nula. Napisati program koji učitava vrednost broja x i ispisuje koliko puta se njegova vrednost pojavila u unetom nizu.

1 Osnovni elementi imperativnog programiranja

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: 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: 0
```

[Rešenje 1.7.7]

Zadatak 1.7.8 Broj je prost ako je deljiv samo sa 1 i samim sobom.

- Napisati funkciju `int prost(int x)` koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati jedinicu ako je broj prost ili nulu u suprotnom.
- Napisati funkciju `void prvih_n_prostih(int n)` koja ispisuje prvih n prostih brojeva.
- Napisati funkciju `void prosti_brojevi_manji_od_n(int n)` koja ispisuje sve proste brojeve manje od broja n .

Napisati program koji učitava pozitivan ceo broj n i ispisuje prvih n prostih brojeva, kao i sve proste brojeve manje od n . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 5  
|| Prvih n prostih: 2 3 5 7 11  
|| Prosti manji od n: 2 3
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: 2  
|| Prvih n prostih: 2 3  
|| Prosti manji od n: ne postoje
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj n: -11  
|| Greska: neispravan unos.
```

[Rešenje 1.7.8]

Zadatak 1.7.9 Rešiti sledeće zadatke korišćenjem funkcija.

- Zadatak 1.1.2 rešiti korišćenjem funkcija `int kvadrat(int x)` koja računa kvadrat datog broja i `int kub(int x)` koja računa kub datog broja.
- Zadatak 1.3.2 rešiti korišćenjem funkcije `float apsolutna_vrednost(float x)` koja izračunava apsolutnu vrednost datog broja.
- Zadatak 1.5.7 rešiti korišćenjem funkcije `float stepen(float x, int n)` koja računa vrednost n -tog stepena realnog broja x .

- (d) Zadatak 1.5.28 rešiti korišćenjem funkcije `int fibonaci(int n)` koja računa n -ti element Fibonačijevog niza.

Zadatak 1.7.10 Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji učitava ceo broj i ispisuje aritmetičku sredinu njegovih cifara zaokruženu na tri decimale.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 461
Aritmeticka sredina: 3.667

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj: 1001
Aritmeticka sredina: 0.500

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj: -84723
Aritmeticka sredina: 4.800

[Rešenje 1.7.10]

Zadatak 1.7.11 Napisati funkciju `int sadrzi(int x, int c)` koja ispituje da li se cifra c nalazi u zapisu celog broja x . Funkcija treba da vrati jedinicu ako se cifra nalazi u broju, a nulu inače. Napisati program koji učitava jedan ceo broj i jednu cifru i u zavisnosti od toga da li se uneta cifra nalazi u zapisu unetog broja, ispisuje odgovarajuću poruku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite broj i cifru: 17890 7
Cifra 7 se nalazi u zapisu broja 17890.

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite broj i cifru: 19 6
Cifra 6 se ne nalazi u zapisu broja 19.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite broj i cifru: 17890 26
Greska: neispravan unos.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite broj i cifru: -1982 9
Cifra 9 se nalazi u zapisu broja -1982.

[Rešenje 1.7.11]

Zadatak 1.7.12 Napisati funkciju `int broj_neparnih_cifara(int x)` koja određuje broj neparnih cifara u zapisu datog celog broja. Napisati program koji učitava cele brojeve sve do unosa broja nula i ispisuje broj neparnih cifara svakog unetog broja.

1 Osnovni elementi imperativnog programiranja

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite cele brojeve:  
2341  
Broj neparnih cifara: 2  
78  
Broj neparnih cifara: 1  
800  
Broj neparnih cifara: 0  
-99761  
Broj neparnih cifara: 4  
0
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite cele brojeve:  
987611  
Broj neparnih cifara: 4  
135  
Broj neparnih cifara: 3  
-701  
Broj neparnih cifara: 2  
602  
Broj neparnih cifara: 0  
-884  
Broj neparnih cifara: 0  
79901  
Broj neparnih cifara: 4  
0
```

[Rešenje 1.7.12]

Zadatak 1.7.13 Napisati program za ispitivanje svojstava cifara datog celog broja.

- Napisati funkciju `int sve_parne_cifre(int x)` koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati jedinicu ako su sve cifre broja parne, a nulu inače.
- Napisati funkciju `int sve_cifre_jednake(int x)` koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati jedinicu ako su sve cifre broja jednake, a nulu inače.

Program učitava ceo broj i u zavisnosti od toga da li su navedena svojstva ispunjena ili ne, ispisuje odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj: 86422  
Sve cifre broja su parne.  
Cifre broja nisu jednake.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj: 55555  
Broj sadrzi bar jednu neparnu cifru.  
Cifre broja su jednake.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj: -88  
Sve cifre broja su parne.  
Cifre broja su jednake.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj i cifru: -342  
Broj sadrzi bar jednu neparnu cifru.  
Cifre broja nisu jednake.
```

[Rešenje 1.7.13]

Zadatak 1.7.14 Napisati funkciju `int ukloni(int n, int p)` koja menja broj n tako što iz njegovog zapisa uklanja cifru na poziciji p . Pozicije se broje sa desna na levo. Cifra jedinica ima poziciju 1. Napisati program koji učitava redni broj pozicije i zatim za cele brojeve koji se unose sve do unosa broja nula, ispisuje brojeve kojima je uklonjena cifra na poziciji p . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite poziciju: 3  
Unesite broj: 1210  
Novi broj: 110  
Unesite broj: 18  
Novi broj: 18  
Unesite broj: 3856  
Novi broj: 356  
Unesite broj: 0
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite poziciju: 1  
Unesite broj: -9632  
Novi broj: -963  
Unesite broj: -2  
Novi broj: 0  
Unesite broj: 246  
Novi broj: 24  
Unesite broj: -52  
Novi broj: -5  
Unesite broj: 0
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite poziciju: 0  
Greska: neispravan unos.
```

[Rešenje 1.7.14]

Zadatak 1.7.15 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 jedinicu ako je uslov ispunjen, a nulu inače. Napisati program koji učitava dva cela broja i ispisuje da li je za njih pomenuti uslov ispunjen ili ne.

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.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dva broja: -7777 77  
Uslov nije ispunjen.
```

[Rešenje 1.7.15]

Zadatak 1.7.16 Napisati funkciju `int neopadajuce(int n)` koja ispituje da li su cifre datog celog broja u neopadajućem poretku. Funkcija treba da vrati jedinicu ako cifre ispunjavaju uslov, a nulu inače. Napisati program koji učitava ceo broj i ispisuje poruku da li su cifre unetog broja u neopadajućem poretku.

1 Osnovni elementi imperativnog programiranja

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 2289
|| Cifre su u neopadajućem poretku.

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 5
|| Cifre su u neopadajućem poretku.

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 6628
|| Cifre nisu u neopadajućem poretku.

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -23
|| Cifre su u neopadajućem poretku.

[Rešenje 1.7.16]

Zadatak 1.7.17 Napisati funkciju `int par_nepar(int n)` koja ispituje da li su cifre datog celog broja naizmjenično parne i neparne. Funkcija treba da vrati jedinicu ako cifre ispunjavaju uslov, a nulu inače. Napisati program koji učitava ceo broj i ispisuje da li on ispunjava pomenuti uslov ili ne.

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 2749
|| Broj ispunjava uslov.

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: -963
|| Broj ispunjava uslov.

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 27449
|| Broj ne ispunjava uslov.

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 5
|| Broj ispunjava uslov.

[Rešenje 1.7.17]

Zadatak 1.7.18 Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog celog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sve do unosa broja nula ispisuje odgovarajuće rotirane brojeve.

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 146
|| Novi broj: 461
|| Unesite broj: 18
|| Novi broj: 81
|| Unesite broj: 3856
|| Novi broj: 8563
|| Unesite broj: 7
|| Novi broj: 7
|| Unesite broj: 0

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 89
|| Novi broj: 98
|| Unesite broj: -369
|| Novi broj: -693
|| Unesite broj: -55281
|| Novi broj: -52815
|| Unesite broj: 0

[Rešenje 1.7.18]

Zadatak 1.7.19 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 jedinicu ako je broj srećan, a nulu inače. Napisati program koji za uneti pozitivan ceo broj n ispisuje sve srećne brojeve od 1 do n . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 100
Srećni brojevi:
1 10 19 28 37 46 55 64 73 82 91 100
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: 0
Greska: neispravan unos.
```

[Rešenje 1.7.19]

Zadatak 1.7.20 Prirodan broj a je Armstrongov ako je jednak sumi n -tih stepena svojih cifara, pri čemu je n broj cifara broja a . Napisati funkciju `int armstrong(int x)` koja vraća jedinicu ako je broj Armstrongov, a nulu inače. Napisati program koji za učitani pozitivan ceo broj proverava da li je Armstrongov. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 1634
Broj je Armstrongov.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 118
Broj nije Armstrongov.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 0
Greska: neispravan unos.
```

[Rešenje 1.7.20]

Zadatak 1.7.21 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 sprovodi sve dok je sabirak po apsolutnoj vrednosti veći od date tačnosti eps . Napisati program koji učitava dva realna broja x i eps i ispisuje izračunatu vrednost e^x .

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj x: 5
Unesite eps: 0.001
Rezultat: 148.412951
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj x: -3
Unesite eps: 0.0001
Rezultat: 0.049796
```

[Rešenje 1.7.21]

1 Osnovni elementi imperativnog programiranja

Zadatak 1.7.22 Napisati funkciju `void ispis(float x, float y, int n)` koja za dva realna broja x i y i jedan pozitivan ceo broj n ispisuje vrednosti sinusne funkcije u n ravnomerno raspoređenih tačaka intervala $[x, y]$. Napisati program koji učitava granice intervala i broj tačaka i ispisuje odgovarajuće vrednosti sinusne funkcije zaokružene na četiri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dva realna broja: 7 31  
Unesite broj n: 6  
Rezultat:  
sin(7.0000) = 0.6570  
sin(11.8000) = -0.6935  
sin(16.6000) = -0.7784  
sin(21.4000) = 0.5573  
sin(26.2000) = 0.8759  
sin(31.0000) = -0.4040
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dva realna broja: -8.32 20.5  
Unesite broj n: 5  
Rezultat:  
sin(-8.3200) = -0.8934  
sin(-1.1150) = -0.8979  
sin(6.0900) = -0.1920  
sin(13.2950) = 0.6658  
sin(20.5000) = 0.9968
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dva realna broja: 8 8  
Greska: neispravan unos.
```

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dva realna broja: 7 32  
Unesite broj n: -10  
Greska: neispravan unos.
```

[Rešenje 1.7.22]

Zadatak 1.7.23 Napisati funkciju `char sifra(char c, int k)` koja za dati karakter c određuje šifru na sledeći način: ukoliko je c slovo, šifra je slovo koje se nalazi k pozicija pre njega u engleskoj 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 nenegativan ceo broj k , a zatim i karaktere sve do kraja ulaza i nakon svakog učitanog karaktera ispisuje njegovu šifru. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj k: 2  
Unesite tekst (CTRL+D za prekid):  
U svetu postoji jedno carstvo  
S qtcrs nmqrmbg hcblm aypqrtm  
U njemu caruje drugarstvo.  
S lhcks aypshc bpseypqrtm.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj k: -2  
Greska: neispravan unos.
```

[Rešenje 1.7.23]

Zadatak 1.7.24 Rešiti sledeće zadatke korišćenjem funkcija.

- (a) Zadatak 1.5.31 rešiti korišćenjem funkcije `char konverzija(char c)` koja malo slovo pretvara u odgovarajuće veliko i obrnuto.
- (b) Zadatak 1.5.32 rešiti korišćenjem funkcije `void prebrojavanje()` koja učitava karaktere sve do kraja ulaza i ispisuje broj malih slova, velikih slova, cifara, belina, kao i sumu svih unetih cifara.

Zadatak 1.7.25 Napisati program koji učitava tri cela broja koja predstavljaju dan, mesec i godinu i ispisuje datum sledećeg dana. Zadatak rešiti korišćenjem narednih funkcija.

- (a) `int prestupna(int godina)` koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati jedinicu ako je godina prestupna ili nulu ako nije.
- (b) `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu.
- (c) `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan.
- (d) `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum ispisuje datum sledećeg dana.

U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite datum: 24.8.1998.  
Datum sledeceg dana je: 25.8.1998.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite datum: 31.12.1789.  
Datum sledeceg dana je: 1.1.1790.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite datum: 28.2.2003.  
Datum sledeceg dana je: 1.3.2004.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite datum: 31.4.2004.  
Greska: neispravan unos.
```

[Rešenje 1.7.25]

Zadatak 1.7.26 Napisati funkciju `int od_nove_godine(int dan, int mesec, int godina)` koja određuje koliko je dana proteklo od Nove godine do datog datuma. Napisati program koji učitava tri cela broja koja predstavljaju

1 Osnovni elementi imperativnog programiranja

dan, mesec i godinu i ispisuje koliko dana je proteklo od Nove godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 24.8.1998.  
|| Broj dana od Nove godine je: 235
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.12.1680.  
|| Broj dana od Nove godine je: 366
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2003.  
|| Broj dana od Nove godine je: 58
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Greska: neispravan unos.
```

[Rešenje 1.7.26]

Zadatak 1.7.27 Napisati funkciju `int do_kraja_godine(int dan, int mesec, int godina)` koja određuje broj dana od datog datuma do kraja godine. Napisati program koji učitava tri cela broja koja predstavljaju dan, mesec i godinu i ispisuje broj dana do kraja godine. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 24.8.1998.  
|| Broj dana do Nove godine je: 129
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.12.1680.  
|| Broj dana do Nove godine je: 0
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2004.  
|| Broj dana do Nove godine je: 307
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Greska: neispravan unos.
```

[Rešenje 1.7.27]

Zadatak 1.7.28 Napisati funkciju `int broj_dana_izmedju(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati program koji učitava dva datuma u formatu `dd.mm.gggg` i ispisuje broj dana između ta dva datuma. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite prvi datum: 12.3.2008.  
Unesite drugi datum: 5.12.2008.  
Broj dana izmedju dva datuma je: 268
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite prvi datum: 26.9.1986.  
Unesite drugi datum: 2.2.1701.  
Broj dana izmedju dva datuma je: 104301
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite prvi datum: 24.8.1998.  
Unesite drugi datum: 12.10.2010.  
Broj dana izmedju dva datuma je: 4440
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite prvi datum: 24.8.1998.  
Unesite drugi datum: 31.4.2004.  
Greska: neispravan unos.
```

[Rešenje 1.7.28]

Zadatak 1.7.29 Napisati funkciju void romb(int n) koja iscrtava romb čija je stranica dužine n . Napisati program koji učitava ceo pozitivan broj i iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću 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: neispravan unos.
```

[Rešenje 1.7.29]

Zadatak 1.7.30 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 iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite brojeve: 8 -2 5 4  
Greska: neispravan unos.
```

[Rešenje 1.7.30]

Zadatak 1.7.31 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 iscrtava odgovarajuću sliku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite brojeve: 5 2 2 4  
*  
* *  
* *  
****
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite brojeve: 8 -2 5 4  
Greska: neispravan unos.
```

[Rešenje 1.7.31]

1.8 Rešenja

Rešenje 1.7.1

```

1 #include <stdio.h>

3 /* Funkcija racuna minimum tri cela broja. Promenljive u listi
   argumenata funkcije (x, y i z), kao i one deklarisane u samoj
   funkciji (minimum), lokalne su za tu funkciju, sto znaci da im
   se ne moze pristupiti nigde izvan funkcije min. */
7 int min(int x, int y, int z) {
   /* Inicijalizacija minimuma na vrednost broja x. */
9   int minimum = x;

11  /* Poredjenje sa druga dva broja i po potrebi azuriranje
    vrednosti minimuma. */
13  if (minimum > y)
14    minimum = y;
15  if (minimum > z)
16    minimum = z;

17  /* Vrednost minimuma se vraca kao povratna vrednost funkcije. */
19  return minimum;
21}

21 int main() {
23   /* Deklaracija potrebnih promenljivih. */
24   int x, y, z;

25   /* Ucitavanje vrednosti tri broja. */
26   printf("Unesite brojeve: ");
27   scanf("%d%d%d", &x, &y, &z);

28   /* Poziv funkcije i ispis rezultata. */
29   printf("Minimum: %d\n", min(x, y, z));

33   return 0;
}

```

Rešenje 1.7.2

```

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

3
4 /* Funkcija vraca razlomljeni deo prosledjenog broja. */
5 float razlomljeni_deo(float x) {
6   /* Napomena: Funkcija fabs racuna absolutnu vrednost realnog
      broja i njena deklaracija se nalazi u zaglavlu math.h.
      Funkcija abs racuna absolutnu vrednost celog broja i njena
      deklaracija se nalazi u zaglavlu stdlib.h. */
7
8
9

```

```
1    x = fabs(x);

11   /* Razlomljeni deo broja se dobija tako sto se od samog broja
13      oduzme njegov ceo deo. */
14      return x - (int) x;
15 }

17 int main() {
18   /* Deklaracija potrebne promenljive. */
19   float n;

21   /* Ucitavanje ulazne vrednosti. */
22   printf("Unesite broj:");
23   scanf("%f", &n);

25   /* Ispis rezultata. */
26   printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));
27
28   return 0;
29 }
```

Rešenje 1.7.3

```
1 #include <stdio.h>

3 /* Funkcija racuna zbir delilaca broja x. */
4 int zbir_delilaca(int x) {
5   int i;

7   /* Inicijalizacija zbira na 0. */
8   int zbir = 0;

9   /* Svaki broj i izmedju 1 i sqrt(x) koji deli broj x se dodaje
10      zbiru. Ako je u pitanju broj za koji vazi da je i*i
11      jednako x, onda se dodaje samo vrednost i. U suprotnom se
12      pored vrednosti i dodaje i/x/i.
13      Na primer, za x=6, kada je i=2, dodaju se i 2 i 6/2 = 3,
14      a za x = 4, kada je i=2, dodaje se samo 2. */
15      for (i = 1; i * i <= x; i++) {
16        if (x % i == 0) {
17          zbir += i;
18          if (i != x / i)
19            zbir += x / i;
20        }
21      }

23   /* Povratna vrednost funkcije je dobijeni zbir. */
24   return zbir;
25 }

27 int main() {
```

```

29  /* Deklaracija potrebnih promenljivih. */
30  int k, i;
31
32  /* Ucitavanje broja k. */
33  printf("Unesite broj k:");
34  scanf("%d", &k);
35
36  /* Provera ispravnosti ulaznih podataka. */
37  if (k <= 0) {
38      printf("Greska: neispravan unos.\n");
39      return 1;
40  }
41
42  /* Ispis zbir delilaca za svaki broj od 1 do k. */
43  for (i = 1; i <= k; i++)
44      printf("%d ", zbir_delilaca(i));
45  printf("\n");
46
47  return 0;
}

```

Rešenje 1.7.4

```

1 #include <stdio.h>
2
3 /* Funkcija za dva neoznacena broja x i n utvrdjuje da li je x neki
4  stepen broja n. Ukoliko jeste, funkcija vraca izlozilac stepena,
5  a u suprotnom vraca -1. */
6  int je_stepen(unsigned int x, unsigned int n) {
7      /* Na pocetku, s = n^i = n^1 = n. */
8      int i = 1;
9      unsigned int s = n;
10
11     /* U svakoj iteraciji petlje s se azurira tako da ima vrednost
12      n^i. Postupak se ponavlja dok je s manji od x. */
13     while (s < x) {
14         s = s * n;
15         i++;
16     }
17
18     /* Kako s ima vrednost n^i, ako vazi da je s jednako x, onda je
19      bas brojac i trazeni izlozilac. */
20     if (s == x)
21         return i;
22
23     /* Ako nije, onda se vraca -1. */
24     return -1;
25 }
26
27 int main() {
28     /* Deklaracije potrebnih promenljivih. */
29 }

```

```
29     unsigned int x, n;
30     int stepen;
31
32     /* Ucitavanje vrednosti x i n. */
33     printf("Unesite dva broja: ");
34     scanf("%u%u", &x, &n);
35
36     /* Poziv funkcije. */
37     stepen = je_stepen(x, n);
38
39     /* U zavisnosti od povratne vrednosti funkcije, vrsti se ispis
40      rezultata. */
41     if (stepen != -1)
42         printf("Jeste: %u=%u^%d\n", x, n, stepen);
43     else
44         printf("Broj %u nije stepen broja %u.\n", x, n);
45
46     return 0;
47 }
```

Rešenje 1.7.5

```
1 #include <stdio.h>
2
3 /* Funkcija racuna nzd(x,y) primenom Euklidovog algoritma. */
4 int euklid(int x, int y) {
5     int ostatak;
6     /* Euklidov algoritam: trazi se nzd(x,y), npr. nzd(12,18).
7        Postupak koji se primenjuje je sledeći:
8        1. ostatak = x % y = 12 % 18 = 12.
9        2. x postaje y => x = 18
10       3. y postaje ostatak => y = 12 =>
11
12       1. ostatak = x % y = 18 % 12 = 6
13       2. x postaje y => x = 6
14       3. y postaje ostatak => y = 6 =>
15
16       1. ostatak = x % y = 12 % 6 = 0
17       2. x postaje y => x = 6
18       3. y postaje ostatak => y = 0
19
20       Postupak se zavrsava kada y postane 0, a rezultat je
21       poslednji ne-nula ostatak, tj. x. */
22     while (y) {
23         ostatak = x % y;
24         x = y;
25         y = ostatak;
26     }
27
28     /* Kao povratna vrednost funkcije se vraca x. */
29     return x;
30 }
```

```

31   }
32
33 int main() {
34   /* Deklaracija potrebnih promenljivih. */
35   int a, b;
36
37   /* Ucitavanje vrednosti a i b. */
38   printf("Unesite dva cela broja:");
39   scanf("%d%d", &a, &b);
40
41   /* Ispis rezultata. */
42   printf("Najveci zajednicki delilac: %d\n", euklid(a, b));
43
44   return 0;
45 }
```

Rešenje 1.7.6 Pogledajte zadatak 1.7.3.

Rešenje 1.7.7

```

1 #include <stdio.h>
2
3 /* Funkcija broji koliko puta se realan broj x javlja u nizu unetih
4    brojeva. */
5 int prebrojavanje(float x) {
6   float y;
7   int broj_pojavljanja = 0;
8
9   /* Brojevi se ucitavaju sve do unosa broja nula. Svaki put kada
10    se unese broj koji je jednak broju x, brojac pojavljanja se
11    uveca za 1. */
12   printf("Unesite brojeve:\n");
13   while (1) {
14     scanf("%f", &y);
15
16     if (y == 0)
17       break;
18
19     if (x == y)
20       broj_pojavljanja++;
21   }
22
23   return broj_pojavljanja;
24 }
25
26 int main() {
27   /* Deklaracije potrebnih promenljivih. */
28   float x;
29   int rezultat;
30
31   /* Ucitavanje vrednosti broja x. */
```

```
33 printf("Unesite broj x: ");
34 scanf("%f", &x);

35 /* Ucitavanje brojeva i racunanje rezultata. */
36 rezultat = prebrojavanje(x);

37 /* Ispis rezultata. */
38 printf("Broj pojavljivanja broja %.2f: %d\n", x, rezultat);

39 return 0;
}
```

Rešenje 1.7.8

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

3 /* Funkcija vraca 1 ako je broj prost, a 0 u suprotnom. */
4 int prost(int x) {
5     int i;

7     /* Brojevi 2 i 3 su prosti. */
8     if (x == 2 || x == 3)
9         return 1;

11    /* Parni brojevi nisu prosti. */
12    if (x % 2 == 0)
13        return 0;

15    /* Ako se naidje na broj koji deli broj x, onda broj x nije
16       prost. Provera se vrsti za sve neparne brojeve izmedju 3 i
17       sqrt(x), jer kada bi x imao parnog delioca, onda bi i broj 2
18       delio x, a taj uslov je vec proveren. */
19    for (i = 3; i <= sqrt(x); i += 2)
20        if (x % i == 0)
21            return 0;

23    /* Ako nijedan od prethodnih uslova nije bio ispunjen, to znači
24       da nijedan broj ne deli x, pa je on prost. */
25    return 1;
26}

29 /* Funkcija ispisuje prvih n prostih brojeva. Ključna rec void
30    označava da funkcija nema povratnu vrednost. */
31 void prvih_n_prostih(int n) {
32     int broj_prostih = 0;
33     int k = 2;

35     /* Petlja se izvršava sve dok se ne ispise n prostih brojeva. */
36     while (broj_prostih < n) {
37         /* Ako se naidje na broj koji je prost, ispisuje se njegova
```

```

    vrednost i uvecava se brojac. */
39 if (prost(k)) {
40     printf("%d ", k);
41     broj_prostih++;
42 }
43
/* Prelazi se na sledeci broj. */
44 k++;
45
/* Napomena: Zbog prirode prostih brojeva, moze se krenuti i od
46 broja tri i vrsiti uvecavanje za dva, kako bi se preskocila
47 provera parnih brojeva. */
48 }
49 printf("\n");
50 }

/* Funkcija ispisuje sve proste brojeve cija je vrednost manja od
51 n. */
52 void prosti_brojevi_mani_iz_n(int n) {
53     /* Ukoliko je n manje ili jednako 2, onda nema prostih brojeva
54     koji su manji od njega. U tom slucaju se ispisuje odgovarajuca
55     poruka i naredbom return; se izlazi iz funkcije. */
56     if (n <= 2) {
57         printf("ne postoje\n");
58         return;
59     }
60
/* Za svaki broj k izmedju 2 i n-1 se vrsti provera da li je prost
61 i ako jeste, ispisuje se njegova vrednost. */
62     int k = 2;
63     while (k < n) {
64         if (prost(k))
65             printf("%d ", k);
66         k++;
67     }
68     printf("\n");
69 }
70

int main() {
/* Deklaracija potrebnih promenljivih. */
71     int n;
72
/* Ucitavanje broja n. */
73     printf("Unesite broj n:");
74     scanf("%d", &n);
75
/* Provera ispravnosti ulaza. */
76     if (n <= 0) {
77         printf("Greska: neispravan unos.\n");
78         return 1;
79     }
80
/* Ispis rezultata. */
81 }
```

```
91     printf("Prvih n prostih: ");
92     prvih_n_prostih(n);
93     printf("Prosti manji od n: ");
94     prosti_brojevi_manji_od_n(n);
95
96     return 0;
97 }
```

Rešenje 1.7.10

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija racuna aritmeticku sredinu cifara datog celog broja. */
5 float aritmeticka_sredina(int x) {
6     /* Aritmeticka sredina broja 0 je 0. */
7     if (x == 0)
8         return 0;
9
10    /* Deklaracija i inicializacija brojaca. */
11    int zbir_cifara = 0;
12    int broj_cifara = 0;
13
14    /* Izracunava se apsolutna vrednost broja x kako bi program
15       ispravno radio i za negativne brojeve. */
16    x = abs(x);
17
18    /* Sve dok ima neobradjenih cifara, na zbir se dodaje poslednja
19       cifra, brojac cifara se uvecava za 1 i iz broja x se uklanja
20       poslednja cifra. */
21    while (x) {
22        zbir_cifara += x % 10;
23        broj_cifara++;
24        x /= 10;
25    }
26
27    /* Kao povratna vrednost funkcije se vraca odgovarajuci
28       kolicnik. */
29    return (float) zbir_cifara / broj_cifara;
30}
31
32 int main() {
33     /* Deklaracija potrebne promenljive. */
34     int x;
35
36     /* Ucitavanje vrednosti broja x. */
37     printf("Unesite broj: ");
38     scanf("%d", &x);
39
40     /* Ispis rezultata. */
41     printf("Aritmeticka sredina: %.3f\n", aritmeticka_sredina(x));
```

```
43     return 0;
}
```

Rešenje 1.7.11

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija ispituje da li se cifra c nalazi u zapisu celog broja
5   x. Vraca 1 ako je uslov ispunjen i 0 u suprotnom. */
6 int sadrzi(int x, int c) {
7   /* Izracunava se apsolutna vrednost broja x. */
8   x = abs(x);
9
10  /* Izdvaja se cifra po cifra broja x. Ako se naidje na cifru cija
11    je vrednost c, onda se kao rezultat funkcije vraca 1 (jer x
12    sadrzi c). */
13  while (x) {
14    if (x % 10 == c)
15      return 1;
16    x /= 10;
17  }
18
19  /* Ako se petlja zavrsila, znaci da se nijednom nije naislo na
20    cifru c, sto znaci da broj x ne sadrzi cifru c i kao povratna
21    vrednost funkcije se vraca 0. */
22  return 0;
23}
24
25 int main() {
26   /* Deklaracija potrebnih promenljivih. */
27   int x, c;
28
29   /* Ucitavanje vrednosti x i c. */
30   printf("Unesite broj i cifru:");
31   scanf("%d%d", &x, &c);
32
33   /* Provera ispravnosti ulaza. */
34   if (c < 0 || c > 9) {
35     printf("Greska: neispravan unos.\n");
36     return 1;
37   }
38
39   /* Racunanje i ispis rezultata. */
40   if (sadrzi(x, c))
41     printf("Cifra %d se nalazi u zapisu broja %d\n", c, x);
42   else
43     printf("Cifra %d se ne nalazi u zapisu broja %d\n", c, x);
44
45   return 0;
}
```

}

Rešenje 1.7.12

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija određuje broj neparnih cifara u zapisu datog celog
5   broja. */
6 int broj_neparnih_cifara(int x) {
7     int brojac_neparnih = 0;
8     char cifra;
9     x = abs(x);
10
11    while (x) {
12        /* Izdvaja se poslednja cifra broja. */
13        cifra = x % 10;
14
15        /* Može se izbaci koriscenje naredbe if pomocu narednog izraza.
16           Naime, vrednost izraza cifra%2 je 1 kada je cifra neparna,
17           odnosno 0 kada je parna. Tako će na broj neparnih cifara
18           biti dodata jednica ako je cifra neparna, a ako je parna
19           bice dodata 0, što jeste zeljeno ponasanje. */
20        brojac_neparnih += (cifra % 2);
21        x /= 10;
22    }
23
24    return brojac_neparnih;
25}
26
27 int main() {
28     /* Deklaracija potrebne promenljive. */
29     int x;
30
31     /* Ucitavanje brojeva sve do unosa broja nula i ispis
32       broja neparnih cifara za svaki ucitani broj. */
33     printf("Unesite cele brojeve:\n");
34     while (1) {
35         scanf("%d", &x);
36         if (x == 0)
37             break;
38
39         printf("Broj neparnih cifara: %d\n", broj_neparnih_cifara(x));
40     }
41
42     return 0;
43 }
```

Rešenje 1.7.13

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija proverava da li su sve cifre broja x parne i vraca 1
   ako je uslov ispunjen i 0 ako nije. */
5 int sve_parni_cifre(int x) {
6     char cifra;
7     x = abs(x);
8
9     /* Ako se naidje na cifru koja nije parna, onda se kao povratna
10    vrednost funkcije vraca 0. */
11    while (x > 0) {
12        cifra = x % 10;
13        if (cifra % 2 == 1)
14            return 0;
15        x /= 10;
16    }
17
18    /* Ako se doslo do kraja petlje, znaci da se nije naislo ni na
19    jednu neparnu cifru, sto znaci da su sve cifre parne i da
20    treba da se vrati 1. */
21    return 1;
22}
23
24 /* Funkcija proverava da li su sve cifre broja x jednake i vraca 1
25   ako jesu, a 0 u suprotnom. */
26 int sve_cifre_jednake(int x) {
27     char poslednja_cifra;
28     x = abs(x);
29
30     /* Izdvajanje poslednje cifre broja x. */
31     poslednja_cifra = x % 10;
32     x /= 10;
33
34     /* Za sve ostale cifre se proverava da li su jednake poslednjoj.
35      Ako se naidje na neku koja nije, onda nisu sve cifre broja x
36      jednake i kao povratna vrednost se vraca 0. */
37     while (x) {
38         if (x % 10 != poslednja_cifra)
39             return 0;
40
41         x /= 10;
42     }
43
44     /* Ako se stiglo do kraja petlje, znaci da su sve cifre broja
45      bile jednake poslednjoj cifri, pa se kao povratna vrednost
46      vraca 1. */
47     return 1;
48}
49
50

```

```
1 int main() {
52  /* Deklaracija potrebne promenljive. */
53  int x;
54
55  /* Ucitavanje broja x. */
56  printf("Unesite broj:");
57  scanf("%d", &x);
58
59  /* U zavisnosti od povratne vrednosti napisanih funkcija vrsti se
60  ispis odgovarajucih poruka. */
61  if (sve_parne_cifre(x))
62    printf("Sve cifre broja su parne.\n");
63  else
64    printf("Broj sadrzi bar jednu neparnu cifru.\n");
65
66  if (sve_cifre_jednake(x))
67    printf("Cifre broja su jednake.\n");
68  else
69    printf("Cifre broja nisu jednake.\n");
70
71  return 0;
72 }
```

Rešenje 1.7.14

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 /* Funkcija uklanja cifru sa pozicije p iz broja n. Cifra jedinica
6  ima poziciju 1, desetica 2, itd. */
7 int ukloni(int n, int p) {
8    int znak, tezina_pozicije, levi_deo, desni_deo;
9
10   /* Racunanje znaka broja n. */
11   znak = n < 0 ? 1 : -1;
12
13   /* Racunanje absolutne vrednosti broja n. */
14   n = abs(n);
15
16   /* Racunanje tezina prosledjene pozicije. */
17   tezina_pozicije = pow(10, p - 1);
18
19   /* Broj se deli na dva dela - deo levo od cifre koja se izbacuje
20   i deo desno od cifre koja se izbacuje. */
21   levi_deo = n / (10 * tezina_pozicije);
22   desni_deo = n % tezina_pozicije;
23
24   /* Povratna vrednost funkcije se dobija spajanjem levog i desnog
25   dela i mnozenjem znakom pocetnog broja. */
26   return znak * (levi_deo * tezina_pozicije + desni_deo);
27 }
```

```

27 }
28
29 int main() {
30     /* Deklaracija potrebnih promenljivih. */
31     int broj, p;
32
33     /* Ucitavanje vrednosti pozicije. */
34     printf("Unesite poziciju: ");
35     scanf("%d", &p);
36
37     /* Provera ispravnosti ulaza. */
38     if (p <= 0) {
39         printf("Greska: neispravan unos.\n");
40         return 1;
41     }
42
43     /* Ucitavanje brojeva dok se ne unese nula i ispis brojeva
44      dobijenih izbacivanjem cifre na poziciji p. */
45     while (1) {
46         printf("Unesite broj: ");
47         scanf("%d", &broj);
48
49         if (broj == 0)
50             break;
51
52         printf("Novi broj: %d\n", ukloni(broj, p));
53     }
54
55     return 0;
56 }
```

Rešenje 1.7.15

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija proverava da li se neka cifra nalazi u zapisu celog
5    broja i ako se nalazi vraca odgovarajucu poziciju (tj. njenu
6    tezinu koja je neki stepen broja 10), a u suprotnom vraca -1. Na
7    primer, za broj = 1234 i cifra = 2, funkcija vraca 100. */
8 int pozicija_cifre(int broj, int cifra) {
9     int tezina_pozicije = 1;
10
11    while (broj) {
12        if (broj % 10 == cifra)
13            return tezina_pozicije;
14
15        tezina_pozicije *= 10;
16        broj /= 10;
17    }
18 }
```

```
19     return -1;
}
21
/* Funkcija iz zapisa broja izbacuje cifru koja se nalazi na
23 prosledjenoj poziciji. Pozicija je stepen broja 10. Na primer,
za x=1234 i pozicija = 10, treba da se izbaci 3.
25 levi_deo = 1234/(10*10) = 12
desni_deo = 1234%10 = 4
27 Povratna vrednost je 12*10 + 4 = 124. */
int izbaci_cifru(int broj, int pozicija) {
29     int levi_deo = broj / (pozicija * 10);
30     int desni_deo = broj % pozicija;
31     return levi_deo * pozicija + desni_deo;
}
33
/* Funkcija proverava da li su dva cela broja napisana pomocu istih
35 cifara. Vraca 1 ako je uslov ispunjen, a 0 u suprotnom. */
int zapis(int x, int y) {
37     int pozicija;
x = abs(x);
y = abs(y);

41     while (x) {
/* Provera da li y sadrzi poslednju cifru broja x. */
        pozicija = pozicija_cifre(y, x % 10);

45     /* Ako ne sadrzi, x i y se ne zapisuju pomocu istih cifara. */
        if (pozicija == -1)
            return 0;

49     /* Ako sadrzi, iz x se izbacuje poslednja cifra, a iz y se
51         izbacuje ista ta cifra (koja se nalazi na pronadjenoj
52         poziciji. */
        x /= 10;
53     y = izbaci_cifru(y, pozicija);
    }

55
/* Na kraju petlje iz x su izbacene sve cifre, a vazi da su
57     broevi zapisani pomocu istih cifara samo ukoliko ni u y nema
58     preostalih cifara. */
59     return y == 0;
}
61
int main() {
63     /* Deklaracija potrebnih promenljivih. */
64     int x, y;
65
66     /* Ucitavanje vrednosti x i y. */
67     printf("Unesite dva cela broja: ");
68     scanf("%d%d", &x, &y);
69
/* Ispis odgovarajuce poruke. */
```

```

71     if (zapis(x, y))
72         printf("Uslov je ispunjen.\n");
73     else
74         printf("Uslov nije ispunjen.\n");
75
76     return 0;
77 }
```

Rešenje 1.7.16

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija proverava da li se cifre u zapisu broja nalaze u
5    neopadajućem poretku. */
6 int neopadajuće(int n) {
7     int tekuća_cifra, prethodna_cifra;
8     n = abs(n);
9
10    /* Izvan petlje se izdvaja poslednja cifra u zapisu broja da bi u
11       petlji mogla da se poredi sa sledecom. */
12    prethodna_cifra = n % 10;
13    n /= 10;
14
15    /* U petlji se proverava poredak svake dve susedne cifre. Ukoliko
16       se detektuje da je poredak narusen, izlazi se iz funkcije i
17       vraca se vrednost 0. */
18    while (n) {
19        tekuća_cifra = n % 10;
20
21        if (tekuća_cifra > prethodna_cifra)
22            return 0;
23
24        /* Tekuća cifra postaje prethodna za narednu iteraciju. */
25        prethodna_cifra = tekuća_cifra;
26        n /= 10;
27    }
28
29    /* Nakon izlaska iz petlje povratna vrednost funkcije je 1 jer u
30       slučaju da je poredak u nekom trenutku narusen iz funkcije bi
31       se izaslo još u petlji. */
32    return 1;
33 }
34
35 int main() {
36     /* Deklaracija potrebne promenljive. */
37     int n;
38
39     /* Ucitavanje vrednosti broja n. */
40     printf("Unesite broj: ");
41     scanf("%d", &n);
```

```
43  /* Ispis odgovarajuce poruke. */
44  if (neopadajuce(n))
45      printf("Cifre su u neopadajucem poretku.\n");
46  else
47      printf("Cifre nisu u neopadajucem poretku.\n");
48
49  return 0;
}
```

Rešenje 1.7.17

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija proverava da li su cifre broja naizmenicno parne i
5    neparne. Ako je uslov ispunjen vraca 1, u suprotnom vraca 0. */
6 int par_nepar(int x) {
7     int prethodna_cifra, tekuca_cifra;
8     x = abs(x);
9
10    /* Poslednja cifra broja se izdvaja van petlje da bi u petlji
11       moglo da se vrsti poredjenje. */
12    prethodna_cifra = x % 10;
13    x /= 10;
14
15    while (x) {
16        tekuca_cifra = x % 10;
17
18        /* Ukoliko su uzastopne cifre iste parnosti, uslov nije
19           ispunjen, rad petlje i funkcije se prekida i vraca se 0. */
20        if (tekuca_cifra % 2 == prethodna_cifra % 2)
21            return 0;
22
23        /* Tekuca cifra postaje prethodna cifra za narednu iteraciju. */
24        prethodna_cifra = tekuca_cifra;
25        x /= 10;
26    }
27
28    /* Sve uzastopne cifre su razlicite parnosti jer ni jednom u
29       petlji uslov da su cifre iste parnosti nije bio ispunjen. */
30    return 1;
31}
32
33 int main() {
34     /* Deklaracija potrebne promenljive. */
35     int n;
36
37     /* Ucitavanje vrednosti broja n. */
38     printf("Unesite broj n: ");
39     scanf("%d", &n);
```

```

41  /* Ispis odgovarajuce poruke. */
42  if (par_nepar(n))
43      printf("Broj ispunjava uslov.\n");
44  else
45      printf("Broj ne ispunjava uslov.\n");
46
47  return 0;
}

```

Rešenje 1.7.18

```

1 #include <stdio.h>
2 #include <math.h>
3 #include <stdlib.h>
4
5 /* Funkcija racuna broj cifara celog broja n. */
6 int broj_cifara(int n) {
7     int brojac = 0;
8     n = abs(n);
9
10    if (n < 10)
11        return 1;
12
13    while (n) {
14        brojac++;
15        n /= 10;
16    }
17
18    return brojac;
19}
20
21 /* Funkcija racuna broj koji se dobija rotacijom broja n za jedno
22 mesto uлево. */
23 int rotacija(int n) {
24     int znak, prva_cifra, n_bez_prve_cifre, br_cifara;
25
26     znak = (n < 0) ? -1 : 1;
27     n = abs(n);
28     br_cifara = broj_cifara(n);
29
30     /* Izdvajaju se prva cifra i deo broja bez prve cifre.
31      Na primer: ako je n = 1234 onda je br_cifara = 4
32      prva_cifra se dobija sa:
33      n / (10 ^ (br_cifara - 1)) = 1234 / 1000 = 1.
34      n_bez_prve_cifre se dobija sa: n % 1000 = 234. */
35     int tezina_pozicije = pow(10, br_cifara - 1);
36     prva_cifra = n / tezina_pozicije;
37     n_bez_prve_cifre = n % tezina_pozicije;
38
39     /* Rezultat se dobija nadovezivanjem prve cifre na kraj i
40      ostatak. */
41
42     return (prva_cifra * 10 ^ (br_cifara - 1)) + n_bez_prve_cifre;
43}

```

```
40     mnozenjem znakom pocetnog broja. */
41     return znak * (n_bez_prve_cifre * 10 + prva_cifra);
42 }

43 int main() {
44     /* Deklaracija potrebne promenljive. */
45     int n;

46     /* Ucitavanje brojeva sve do unosa broja nula i ispis brojeva
47      dobijenih kao rezultat izvrsavanja funkcije rotacija nad
48      unetim brojevima. */
49     while (1) {
50         printf("Unesite broj: ");
51         scanf("%d", &n);

52         if (n == 0)
53             break;

54         printf("Novi broj: %d\n", rotacija(n));
55     }
56
57     return 0;
58 }
```

Rešenje 1.7.19

```
1 #include <stdio.h>

3 /* Funkcija vraca zbir cifara datog broja x. */
4 int zbir_cifara(int x) {
5     int zbir = 0;
6     while (x) {
7         zbir += x % 10;
8         x /= 10;
9     }
10    return zbir;
11}

13 /* Funkcija vraca 1 ako je broj srecan, a 0 u suprotnom. */
14 int srecan(int x) {
15     /* Sve dok broj x ima vise od jedne cifre, vrednost broja x se
16      zamenjuje zbirom njegovih cifara.
17      Na primer, pocetno x = 7698 nakon prve iteracije postaje
18      x = 7+6+9+8 = 30, nakon druge iteracije postaje x = 3 + 0 = 3, a
19      zatim se izlazi iz petlje. */
20     while (x >= 10)
21         x = zbir_cifara(x);

23     /* Broj je srecan ako na kraju x ima vrednost 1. */
24     return (x == 1);
25 }
```

```

27 int main() {
28     /* Deklaracija potrebnih promenljivih. */
29     int n, i;
30
31     /* Ucitavanje vrednosti broja n. */
32     printf("Unesite broj n: ");
33     scanf("%d", &n);
34
35     /* Provera ispravnosti ulaza. */
36     if (n <= 0) {
37         printf("Greska: neispravan unos.\n");
38         return 1;
39     }
40
41     /* Ispis svih srecnih brojeva koji su manji ili jednaki n. */
42     printf("Srecni brojevi: ");
43     for (i = 1; i <= n; i++)
44         if (srecan(i))
45             printf("%d ", i);
46
47     printf("\n");
48     return 0;
49 }
```

Rešenje 1.7.20

```

1 #include <stdio.h>
2 #include <math.h>
3 #include <stdlib.h>
4
5 /* Funkcija racuna broj cifara celog broja n. */
6 int broj_cifara(int n) {
7     int brojac = 0;
8     n = abs(n);
9
10    if (n < 10)
11        return 1;
12
13    while (n) {
14        brojac++;
15        n /= 10;
16    }
17
18    return brojac;
19}
20
21 /* Funkcija proverava da li je broj Armstrongov. */
22 int armstrong(int x) {
23     int suma = 0;
24     int n = broj_cifara(x);
25
26     while (n) {
27         int cifra = x % 10;
28         suma += cifra * cifra * cifra;
29         x /= 10;
30         n--;
31     }
32
33     if (suma == x)
34         return 1;
35     else
36         return 0;
37}
```

```
25 int x_pocetno = x;
26
27 /* Racunanje suma n-tih stepena cifara broja x. */
28 while (x) {
29     suma += pow(x % 10, n);
30     x /= 10;
31 }
32
33 /* Ako je suma jednaka pocetnoj vrednosti broja x, broj je
34    Armstrongov, u suprotnom nije. */
35 return x_pocetno == suma;
36
37 int main() {
38     /* Deklaracija potrebne promenljive. */
39     int x;
40
41     /* Ucitavanje vrednosti broja x. */
42     printf("Unesite broj: ");
43     scanf("%d", &x);
44
45     /* Ispis odgovarajuce poruke. */
46     if (armstrong(x))
47         printf("Broj je Armstrongov.\n");
48     else
49         printf("Broj nije Armstrongov.\n");
50
51     return 0;
52 }
```

Rešenje 1.7.21

```
#include <stdio.h>
2 #include <math.h>
3
4 /* Funkcija racuna vrednost e^x kao parcijalnu sumu reda
5    suma(x^n/n!), gde indeks n ide od od 0 do beskonacno, pri cemu
6    se sumiranje sprovodi sve dok je sabirak po apsolutnoj vrednosti
7    veci od date tacnosti eps. */
8 double e_na_x(double x, double eps) {
9     double s = 1, clan = 1;
10    int n = 1;
11
12    /* Parcijalnu sumu se formira tako sto se u svakoj iteraciji
13       petlje promenljivoj s doda jedan sabirak sume oblika (x^n)/n!
14       koji se cuva u promenljivoj clan.
15
16       Svaki sabirak se dobija na osnovu prethodnog tako sto se
17       prethodni pomnozi sa x i podeli sa n (n predstavlja redni broj
18       sabirka u sumi).
```

```

20     Prvi sabirak (kome odgovara n=0) iznosi 1; zbog toga
21     promenljive s i clan se inicijalizuju na vrednost 1.
22
23     Sumiranje se sprovodi sve dok je sabirak po absolutnoj
24     vrednosti veci od date tacnosti eps. */
25 do {
26     clan = (clan * x) / n;
27     s += clan;
28     n++;
29 } while (fabs(clan) > eps);
30
31     return s;
32 }
33
34 int main() {
35     /* Deklaracija potrebnih promenljivih. */
36     double x, eps;
37
38     /* Ucitavanje vrednosti x i eps. */
39     printf("Unesite broj x: ");
40     scanf("%lf", &x);
41     printf("Unesite eps: ");
42     scanf("%lf", &eps);
43
44     /* Ispis rezultata. */
45     printf("Rezultat: %f\n", e_na_x(x, eps));
46     return 0;
47 }
```

Rešenje 1.7.22

```

1 #include <stdio.h>
2 #include <math.h>
3
4 /* Funkcija ispisuje vrednosti funkcije sin(x) u n ravnomerno
5    rasporedjenih tacaka na intervalu [a,b]. */
6 void ispis(float a, float b, int n) {
7     double i;
8     double korak = (b - a) / (n - 1);
9
10    for (i = a; i <= b; i += korak, n--)
11        printf("sin(%4lf) = %4lf \n", i, sin(i));
12
13    /* Zapis realnih brojeva u racunaru ne mora da bude precisan
14       i sabiranje realnih brojeva moze, zbog akumuliranja greske,
15       da dovede do toga da iako je
16       korak = (b - a) / (n - 1)
17       ne vazi da je
18       b = a + korak + korak + ... + korak
19       (gde se korak sabira n-1 puta). Vrednost ovog zbira, u
20       zavisnosti od konkretnih brojeva, moze da bude i malo veca
```

```
    i malo manja od b. Zbog toga, dodatno proveravamo da li
22   treba da stampamo vrednost funkcije i u tacki b */
23   if(n != 0)
24     printf("sin(%lf) = %lf \n", b, sin(b));
25 }

26 int main() {
27   /* Deklaracije potrebnih promenljivih. */
28   float a, b;
29   int n;

30   /* Ucitavanje granica intervala i provera ispravnosti ulaza. */
31   printf("Unesite dva realna broja: ");
32   scanf("%f%f", &a, &b);
33   if (b <= a) {
34     printf("Greska: neispravan unos.\n");
35     return 1;
36   }

37   /* Ucitavanje broja n i provera ispravnosti ulaza. */
38   printf("Unesite broj n: ");
39   scanf("%d", &n);
40   if (n <= 1) {
41     printf("Greska: neispravan unos.\n");
42     return 1;
43   }

44   /* Ispis rezultata. */
45   printf("Rezultat:\n");
46   ispis(a, b, n);

47   return 0;
48 }
```

Rešenje 1.7.23

```
1 #include <stdio.h>

3 /* Funkcija vraca karakter koji se u engelskoj abecedi nalazi k mesta
4   pre
5   datog karaktera c. */
6 char sifra(char c, int k) {
7   /* Provera da li je karakter malo slovo. */
8   if (c >= 'a' && c <= 'z') {
9     /* Ako karakter koji je k pozicija pre datog karaktera isпада
10      iz opsega malih slova. */
11     if (c - k < 'a')
12       /* Od k se oduzima rastojanje izmedju c i 'a' (jer je za
13          toliko karaktera vec vraceno u nazad), kako bi se odredilo
           koliko preostali broj karaktera koji treba preskociti od
           karaktera 'z'. */
```

```

15     return 'z' - (k - (c - 'a') - 1);
16 else
17     /* U suprotnom, karakter c-k ne ispada iz opsega malih slova,
18      te je dovoljno njega vratiti. */
19     return c - k;
20 } else if (c >= 'A' && c <= 'Z') {
21     /* Postupak se ponavlja i za velika slova. */
22     if (c - k < 'A')
23         return 'Z' - (k - (c - 'A') - 1);
24     else
25         return c - k;
26 }
27
28 /* Ako nije ni malo ni veliko slovo, karakter se ne menja. */
29 return c;
30 }
31
32 int main() {
33     /* Deklaracije potrebnih promenljivih. */
34     int k;
35     char c;
36
37     /* Ucitavanje vrednosti k. */
38     printf("Unesite broj k: ");
39     scanf("%d", &k);
40
41     /* Ucitavanje karaktera sve do kraja ulaza i ispis njihove
42      sifre. */
43     printf("Unesite tekst (CTRL + D za prekid): ");
44     while ((c = getchar()) != EOF)
45         putchar(sifra(c, k));
46
47     return 0;
48 }
```

Rešenje 1.7.25

```

1 #include <stdio.h>
2
3 /* Funkcija 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 određuje broj dana u datom mesecu. */
12 int broj_dana(int mesec, int godina) {
13     switch (mesec) {
14         case 1:
```

```
15     case 3:  
16     case 5:  
17     case 7:  
18     case 8:  
19     case 10:  
20     case 12:  
21         return 31;  
22     case 4:  
23     case 6:  
24     case 9:  
25     case 11:  
26         return 30;  
27     case 2:  
28         if (prestupna(godina))  
29             return 29;  
30         else  
31             return 28;  
32     }  
33     return -1;  
34 }  
35 /* Funkcija proverava da li je datum ispravan. Ako je datum  
36    ispravan funkcija vraca 1, inace vraca 0. */  
37 int ispravan(int dan, int mesec, int godina) {  
38     /* Ako je godina negativna, datum nije ispravan. */  
39     if (godina < 0)  
40         return 0;  
41  
42     /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */  
43     if (mesec < 1 || mesec > 12)  
44         return 0;  
45  
46     /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,  
47        datum nije ispravan. */  
48     if (dan < 1 || dan > broj_dana(mesec, godina))  
49         return 0;  
50  
51     return 1;  
52 }  
53  
54 /* Funkcija racuna sledeci dan. */  
55 void sledeci_dan(int dan, int mesec, int godina) {  
56     /* Za kraj godine, odnosno za datum 31.12. sledeci datum je 1.1.  
57        i godina se uvecava za jedan. */  
58     if (mesec == 12 && dan == 31)  
59         printf("1.1.%d.\n", godina + 1);  
60     /* Ukoliko je dan jednak poslednjem danu u tom mesecu, odnosno  
61        ako je jednak broju dana u tom mesecu, onda je sledeci datum  
62        kada se mesec uveca za 1, a dan postane 1. Bitan je redosled  
63        ovih naredbi. Ako bi ovo ispitivanje bilo prvo, onda bi se  
64        mesec mogao uvecati na 13. sto ne bi bio ispravan datum. Zato  
65        se prvo proverava da li je kraj godine, pa tek onda da li je
```

```

67     kraj meseca. */
68 else if (dan == broj_dana(mesec, godina))
69     printf("1.%d.%d.\n", mesec + 1, godina);
70 /* Ako nije ni jedan od prethodna dva slučaja, onda se dan može
71    uvećati na 1, bez bojazni da će se prekoraciti broj dana u
72    datom mesecu. */
73 else
74     printf("%d.%d.%d.\n", dan + 1, mesec, godina);
75 }

77 int main() {
78 /* Deklaracija potrebnih promenljivih. */
79     int dan, mesec, godina;

81     /* Ucitavanje vrednosti dana, meseca i godine. */
82     printf("Unesite datum:");
83     scanf("%d.%d.%d.", &dan, &mesec, &godina);

85     /* Provera ispravnosti datuma. */
86     if (!ispravan(dan, mesec, godina)) {
87         printf("Greska: neispravan unos.\n");
88         return 1;
89     }

91     /* Poziv funkcije za ispis sledeceg dana. */
92     printf("Datum sledeceg dana je:");
93     sledeci_dan(dan, mesec, godina);

95     return 0;
}

```

Rešenje 1.7.26

Za rešavanje ovog zadatka koristi se funkcija `od_nove_godine` koja je definisana u rešenju zadatka [1.7.28](#).

Rešenje 1.7.27

Za rešavanje ovog zadatka koristi se funkcija `do_kraja_godine` koja je definisana u rešenju zadatka [1.7.28](#).

Rešenje 1.7.28

```

#include <stdio.h>

2 /* Funkcija proverava da li je godina prestupna. */
3 int prestupna(int godina) {
4     if ((godina % 100 != 0 && godina % 4 == 0) || godina % 400 == 0)
5         return 1;
6     else
7         return 0;
8 }

```

```
10  /* Funkcija određuje broj dana u datom mesecu. */
11 int broj_dana(int mesec, int godina) {
12     switch (mesec) {
13         case 1:
14         case 3:
15         case 5:
16         case 7:
17         case 8:
18         case 10:
19         case 12:
20             return 31;
21         case 4:
22         case 6:
23         case 9:
24         case 11:
25             return 30;
26         case 2:
27             if (prestupna(godina))
28                 return 29;
29             else
30                 return 28;
31     }
32     return -1;
33 }

36 /* Funkcija proverava da li je datum ispravan. Ako je datum
37    ispravan funkcija vraca 1, inace vraca 0. */
38 int ispravan(int dan, int mesec, int godina) {
39     /* Ako je godina negativna, datum nije ispravan. */
40     if (godina < 0)
41         return 0;
42
43     /* Ako mesec nije u opsegu od 1 do 12, datum nije ispravan. */
44     if (mesec < 1 || mesec > 12)
45         return 0;
46
47     /* Ako je dan manji od 1 ili veci od broja dana u datom mesecu,
48        datum nije ispravan. */
49     if (dan < 1 || dan > broj_dana(mesec, godina))
50         return 0;
51
52     return 1;
53 }

54 /* Funkcija određuje koliko dana je proteklo od pocetka godine. */
55 int od_nove_godine(int dan, int mesec, int godina) {
56     int suma_dana = 0, i;
57
58     /* Za sve mesece pre datog datuma dodaje se broj dana za dati
59        mesec. */
60     for (i = 1; i < mesec; i++)
```

```

62     suma_dana += broj_dana(i, godina);

64     /* Na kraju se dodaje koliko je dana proteklo u datom mesecu, a
       to je zadato promenljivom dan. */
66     return suma_dana + dan;
}
68
/* Funkcija određuje koliko dana ima do kraja godine. */
70 int do_kraja_godine(int dan, int mesec, int godina) {
    int suma_dana = 0, i;
72
    /* Za sve mesece posle datog datuma dodaje se broj dana za dati
       mesec. */
74    for (i = mesec + 1; i <= 12; i++)
        suma_dana += broj_dana(i, godina);
76
    /* Na kraju se dodaje koliko je dana je ostalo u datom mesecu. */
78    return suma_dana + broj_dana(mesec, godina) - dan;
}

80
/* Funkcija vraca 1 ako je prvi datum pre drugog datuma. U
   suprotnom vraca 0. */
82 int prethodi(int dan1, int mesec1, int godina1, int dan2,
              int mesec2, int godina2) {
84    if (godina1 < godina2)
        return 1;
86    else if (godina1 > godina2)
        return 0;
88    else if (mesec1 < mesec2)
        return 1;
90    else if (mesec1 > mesec2)
        return 0;
92    else if (dan1 < dan2)
        return 1;
94    else
        return 0;
}
96
98
100 /* Funkcija vraca broj dana u datoј godini. */
101 int broj_dana_u_godini(int godina) {
102    if (prestupna(godina))
        return 366;
104    else
        return 365;
}
106

108 /* Funkcija racuna broj dana izmedju dva datuma. */
109 int broj_dana_izmedju(int dan1, int mesec1, int godina1, int dan2,
                       int mesec2, int godina2) {
110    int pom, i;
111    int suma_dana = 0;
112

```

```
114  /* Provera koji od datuma je ranije. Ukoliko je potrebno,  
115   razmenjuju se vrednosti promenljivih tako da broj 1 ide uz raniji  
116   datum. */  
117  if (!prethodi(dani, mesec1, godina1, dan2, mesec2, godina2)) {  
118      pom = dani;  
119      dani = dan2;  
120      dan2 = pom;  
121  
122      pom = mesec1;  
123      mesec1 = mesec2;  
124      mesec2 = pom;  
125  
126      pom = godina1;  
127      godina1 = godina2;  
128      godina2 = pom;  
129  }  
130  
131  /* Ako su godine razlicite. */  
132  if (godina1 != godina2) {  
133      /* Za manji datum dodaje se broj dana do kraja godine. */  
134      suma_dana = do_kraja_godine(dani, mesec1, godina1);  
135  
136      /* Za sve godine koje su izmedju dve date godine dodaje se broj  
137       dana u tim godinama. */  
138      for (i = godina1 + 1; i < godina2; i++)  
139          suma_dana += broj_dana_u_godini(i);  
140  
141      /* Za veci datum dodaje se broj dana od pocetka godine. */  
142      suma_dana += od_nove_godine(dan2, mesec2, godina2);  
143  }  
144  /* Ako su godine iste, ali meseci razliciti. */  
145  else if (mesec1 != mesec2) {  
146      /* Dodaje se broj dana do kraja prvog meseca. */  
147      suma_dana = broj_dana(mesec1, godina1) - dani;  
148  
149      /* Dodaje se broj dana za svaki mesec koji je izmedju dva data  
150       meseca. Kako su godina1 i godina2 jednake svejedno je koja  
151       od ove dve promenljive se koristi u pozivu funkcije. */  
152      for (i = mesec1 + 1; i < mesec2; i++)  
153          suma_dana += broj_dana(i, godina1);  
154  
155      /* Dodaje se broj dana od pocetka meseca. */  
156      suma_dana += dan2;  
157  }  
158  /* Ako su i godine i meseci jednaki. */  
159  else  
160      suma_dana = dan2 - dani;  
161  
162  return suma_dana;  
163}  
164 int main() {
```

```

166  /* Deklaracija potrebnih promenljivih. */
167  int dani, mesec1, godina1, dan2, mesec2, godina2;
168
169  /* Ucitavanje datuma. */
170  printf("Unesite prvi datum:");
171  scanf("%d.%d.%d.", &dan1, &mesec1, &godina1);
172
173  printf("Unesite drugi datum:");
174  scanf("%d.%d.%d.", &dan2, &mesec2, &godina2);
175
176  /* Provera ispravnosti unetih datuma. */
177  if (!ispravan(dani, mesec1, godina1)
178      || !ispravan(dan2, mesec2, godina2)) {
179      printf("Greska: neispravan unos.\n");
180      return 1;
181  }
182
183  /* Ispis rezultata. */
184  printf("Broj dana izmedju dva datuma je: %d\n",
185         broj_dana_izmedju(dan1, mesec1, godina1, dan2, mesec2,
186                             godina2));
187
188  return 0;
}

```

Rešenje 1.7.29

```

1 #include <stdio.h>
2
3 /* Funkcija iscrtava romb. */
4 void romb(int n) {
5     int i, j;
6
7     /* Petlja iscrtava liniju po liniju romba. */
8     for (i = 0; i < n; i++) {
9         /* Ispis n-i-1 praznina. */
10        for (j = 0; j < n - i - 1; j++)
11            printf(" ");
12
13        /* Ispis n zvezdica. */
14        for (j = 0; j < n; j++)
15            printf("*");
16
17        /* Prelazak u sledeci red. */
18        printf("\n");
19    }
20
21    int main() {
22        /* Deklaracija potrebne promenljive. */
23        int n;

```

```
25     /* Ucitavanje vrednosti broja n. */
26     printf("Unesite broj n: ");
27     scanf("%d", &n);
28
29     /* Provera ispravnosti ulaza. */
30     if (n <= 0) {
31         printf("Greska: neispravan unos.\n");
32         return 1;
33     }
34
35     /* IsCRTavanje romba. */
36     romb(n);
37
38     return 0;
39 }
```

Rešenje 1.7.30

```
1 #include <stdio.h>
2
3 /* Funkcija stampa n zvezdica za kojima sledi znak za novi red. */
4 void stampaj_zvezdice(int n) {
5     int i;
6     for (i = 0; i < n; i++)
7         printf("*");
8
9     printf("\n");
10}
11
12/* Funkcija crta grafikon. */
13void grafikon_h(int a, int b, int c, int d) {
14    /* Prvo se ispisuje a zvezdica. */
15    stampaj_zvezdice(a);
16
17    /* Postupak se ponavlja za vrednosti b, c i d. */
18    stampaj_zvezdice(b);
19    stampaj_zvezdice(c);
20    stampaj_zvezdice(d);
21}
22
23int main() {
24    /* Deklaracija potrebnih promenljivih. */
25    int a, b, c, d;
26
27    /* Ucitavanje vrednosti a,b,c,d. */
28    printf("Unesite brojeve: ");
29    scanf("%d%d%d%d", &a, &b, &c, &d);
30
31    /* Provera ispravnosti ulaza i ispis rezultata. */
32    if (a < 0 || b < 0 || c < 0 || d < 0)
```

```

33     printf("Greska: neispravan unos.\n");
34     else
35         grafikon_h(a, b, c, d);
36
37     return 0;
}

```

Rešenje 1.7.31

```

1 #include <stdio.h>

3 /* Funkcija racuna najveci od 4 prosledjena broja. */
int maksimum(int a, int b, int c, int d) {
    int maks;

7     maks = a;
11    if (b > maks)
12        maks = b;
13    if (c > maks)
14        maks = c;
15    if (d > maks)
16        maks = d;

17    return maks;
}

19 /* Pomocna funkcija za ispis beline ili zvezdice. */
void ispisi_znak(int polje, int granica) {
    if (polje < granica)
        printf(" ");
    else
        printf("*");
}

25 /* Funkcija iscrtava vertikalni grafikon. */
void grafikon_v(int a, int b, int c, int d) {
    int i, maks;

29    /* Pronalazak najvece od zadate cetiri vrednosti. */
30    maks = maksimum(a, b, c, d);

33    /* Grafikon ukupno ima maks horizontalnih linija. */
34    for (i = 0; i < maks; i++) {
35        /* U svakoj od horizontalnih linija se nalazi po 4 polja: polje
36         za a, b, c i d uspravnu liniju. U svako od polja treba da se
37         upise ili zvezdica ili praznina, u zavisnosti od vrednosti i
38         toga koja linija se trenutno ispisuje. */

39        /* Ispis znaka za polje a. */
40        ispisi_znak(i, maks - a);
}

```

```
43     /* Ispis znaka za polje b. */
44     ispisi_znak(i, maks - b);

45     /* Ispis znaka za polje c. */
46     ispisi_znak(i, maks - c);

47     /* Ispis znaka za polje d. */
48     ispisi_znak(i, maks - d);

49     /* Na kraju svake horizontalne linije stampa se novi red. */
50     printf("\n");
51 }
52 }

53 int main() {
54     /* Deklaracija potrebnih promenljivih. */
55     int a, b, c, d;

56     /* Ucitavanje vrednosti cetiri broja. */
57     printf("Unesite brojeve: ");
58     scanf("%d%d%d%d", &a, &b, &c, &d);

59     /* Provera ispravnosti ulaza i poziv funkcije za ispis
60      grafikona. */
61     if (a < 0 || b < 0 || c < 0 || d < 0) {
62         printf("Greska: neispravan unos.\n");
63         return 1;
64     } else
65         grafikon_v(a, b, c, d);

66     return 0;
67 }
```

2

Napredni tipovi podataka

2.1 Nizovi

Zadatak 2.1.1 Napisati program koji učitava dimenziju niza, elemente niza i zatim ispisuje:

- (a) elemente niza koji se nalaze na parnim pozicijama.
- (b) parne elemente niza.

Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza:  
6  
Unesite elemente niza:  
1 8 2 -5 -13 75  
Elementi niza na parnim pozicijama:  
1 2 -13  
Parni elementi niza:  
8 2
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza:  
3  
Unesite elemente niza:  
11 81 -63  
Elementi niza na parnim pozicijama:  
11 -63  
Parni elementi niza:
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza:  
-4  
Greska: neispravan unos.
```

[Rešenje 2.1.1]

2 Napredni tipovi podataka

Zadatak 2.1.2 Napisati program koji učitava dimenziju niza, elemente niza i zatim menja uneti niz tako što kvadrira sve negativne elemente niza. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 6  
Unesite elemente niza:  
12.34 -6 1 8 32.4 -16  
Rezultujuci niz:  
12.34 36 1 8 32.4 256
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 9  
Unesite elemente niza:  
-8.25 6 17 2 -1.5 1 -7 2.65 -125.2  
Rezultujuci niz:  
68.0625 6 17 2 2.25 1 49 2.65 15675.04
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 4  
Unesite elemente niza:  
9.53 5 1 4.89  
Rezultujuci niz:  
9.53 5 1 4.89
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 104  
Greska: neispravan unos.
```

[Rešenje 2.1.2]

Zadatak 2.1.3 Ako su $a = (a_1, \dots, a_n)$ i $b = (b_1, \dots, b_n)$ vektori dimenzije n , njihov skalarni proizvod se definiše kao $a \cdot b = a_1 \cdot b_1 + \dots + a_n \cdot b_n$. Napisati program koji računa skalarni proizvod dva vektora. Vektori se zadaju kao celobrojni nizovi sa najviše 100 elemenata. Program učitava dimenziju i elemente nizova, a na izlaz ispisuje vrednost skalarnog proizvoda. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju vektora: 5  
Unesite koordinate vektora a:  
8 -2 0 2 4  
Unesite koordinate vektora b:  
35 12 5 -6 -1  
Skalarni proizvod: 240
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju vektora: 3  
Unesite koordinate vektora a:  
-1 0 1  
Unesite koordinate vektora b:  
5 5 5  
Skalarni proizvod: 0
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju vektora: 0  
Greska: neispravan unos.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju vektora: 1  
Unesite koordinate vektora a:  
-1  
Unesite koordinate vektora b:  
1  
Skalarni proizvod: -1
```

[Rešenje 2.1.3]

Zadatak 2.1.4 Napisati program koji učitava dimenziju niza, elemente niza, a potom i ceo broj k i ispisuje indekse elemenata koji su deljivi sa k . Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 4
Unesite elemente niza: 10 14 86 20
Unesite broj k: 5
Rezultat: 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
Rezultat: 0 3 4
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza: 1 2 3 4 5 6
Unesite broj k: 0
Greska: neispravan unos.
```

[Rešenje 2.1.4]

Zadatak 2.1.5 Autobusi su označeni rednim brojevima (počevši od 1) i u nizu se čuva vreme putovanja svakog autobusa u minutima. Međutim, zbog radova na putu između Požege i Užica, svi autobusi koji saobraćaju na tom potezu (autobusi označeni rednim brojevima od k do t) saobraćaju m minuta duže. Napisati program koji učitava broj autobusa n , n celih brojeva koji označavaju vreme putovanja tih autobusa i vrednosti k , t i m i ispisuje vreme putovanja svih autobusa nakon unetih izmena. Maksimalni broj autobusa je 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj autobusa: 8
Unesite vreme putovanja:
24 78 13 124 56 90 205 45
Unesite vrednosti k, t i m:
3 6 23
Vreme putovanja nakon izmena:
24 78 36 147 79 113 205 45
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj autobusa: 8
Unesite vreme putovanja:
24 78 13 124 56 90 205 45
Unesite vrednosti k, t i m:
3 15 3
Greska: neispravan unos.
```

[Rešenje 2.1.5]

2 Napredni tipovi podataka

Zadatak 2.1.6 Napisati program koji za učitani ceo broj ispisuje broj pojavljivanja svake od cifara u zapisu tog broja. UPUTSTVO: Za evidenciju broja pojavljivanja svake cifre pojedinačno, koristiti niz.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite ceo broj: 2355623  
|| U zapisu broja 2355623, cifra 2 se pojaviljuje 2 puta  
|| U zapisu broja 2355623, cifra 3 se pojaviljuje 2 puta  
|| U zapisu broja 2355623, cifra 5 se pojaviljuje 2 puta  
|| U zapisu broja 2355623, cifra 6 se pojaviljuje 1 puta
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite ceo broj: -39902  
|| U zapisu broja -39902, cifra 0 se pojaviljuje 1 puta  
|| U zapisu broja -39902, cifra 2 se pojaviljuje 1 puta  
|| U zapisu broja -39902, cifra 3 se pojaviljuje 1 puta  
|| U zapisu broja -39902, cifra 9 se pojaviljuje 2 puta
```

[Rešenje 2.1.6]

Zadatak 2.1.7 Napisati program koji učitava karaktere sve do unosa karaktera *, a zatim ih ispisuje u redosledu suprotnom od redosleda čitanja. Maksimalni broj karaktera je 500.

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
```

[Rešenje 2.1.7]

Zadatak 2.1.8 Napisati program koji učitava karaktere sve do kraja ulaza, a potom i izračunava koliko se puta u unetom tekstu pojavila svaka od cifara, svako malo slovo i svako veliko slovo. Ispisati broj pojavljivanja samo za karaktere koji su se u unetom tekstu pojavili barem jednom. UPUTSTVO: Za evidenciju broja pojavljivanja cifara, malih i velih slova korisiti pojedinačne nizove.

Primer 1

INTERAKCIJA SA PROGRAMOM:

Unesite tekst:

Mis je dobio grip.
Karakter b se pojavljuje 1 puta
Karakter d se pojavljuje 1 puta
Karakter e se pojavljuje 1 puta
Karakter g se pojavljuje 1 puta
Karakter i se pojavljuje 3 puta
Karakter j se pojavljuje 1 puta
Karakter o se pojavljuje 2 puta
Karakter p se pojavljuje 1 puta
Karakter r se pojavljuje 1 puta
Karakter s se pojavljuje 1 puta
Karakter M se pojavljuje 1 puta

Primer 2

INTERAKCIJA SA PROGRAMOM:

Unesite tekst:

Programiranje 1 je zanimljivo!!
Karakter 1 se pojavljuje 1 puta
Karakter a se pojavljuje 3 puta
Karakter e se pojavljuje 2 puta
Karakter g se pojavljuje 1 puta
Karakter i se pojavljuje 3 puta
Karakter j se pojavljuje 3 puta
Karakter l se pojavljuje 1 puta
Karakter m se pojavljuje 2 puta
Karakter n se pojavljuje 2 puta
Karakter o se pojavljuje 2 puta
Karakter r se pojavljuje 3 puta
Karakter v se pojavljuje 1 puta
Karakter z se pojavljuje 1 puta
Karakter P se pojavljuje 1 puta

[Rešenje 2.1.8]

Zadatak 2.1.9 Napisati program koji učitava jednu liniju teksta i ispisuje koliko puta se pojavilo svako od slova engleske abecede u unetom tekstu. Ne praviti razliku između malih i velikih slova.

Primer 1

INTERAKCIJA SA PROGRAMOM:

Tasi, tasi, TaNaNa i SVILENA marama.....

a:9 b:0 c:0 d:0 e:1 f:0 g:0 h:0 i:4 j:0 k:0 l:1 m:2
n:3 o:0 p:0 q:0 r:1 s:3 t:3 u:0 v:1 w:0 x:0 y:0 z:0

Primer 2

INTERAKCIJA SA PROGRAMOM:

Mihailo Petrović Alas (6 maj 1868 – 8 jun 1943)

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

Primer 3

INTERAKCIJA SA PROGRAMOM:

Alan Mathison Turing (London, 23. jun 1912 – Cesir, 7. jun 1954)

a:3 b:0 c:1 d:1 e:1 f:0 g:1 h:0 i:3 j:3 k:0 l:2 m:1
n:7 o:3 p:0 q:0 r:2 s:2 t:2 u:3 v:0 w:0 x:0 y:0 z:0

[Rešenje 2.1.9]

Zadatak 2.1.10 Takmičari na Beogradskom maratonu su označeni rednim brojevima počevši od 0. Vremena za koja su takmičari istrčali maraton izražena

2 Napredni tipovi podataka

u minutima se zadaju nizom celih brojeva u kojem indeks elementa niza označava redni broj takmičara. Napisati sledeće funkcije za obradu navedenih podataka:

- (a) `void ucitaj(int a[], int n)` koja učitava elemente niza a dimenzije n .
- (b) `void ispisi(int a[], int n)` koja ispisuje elemente niza a dimenzije n .
- (c) `int suma(int a[], int n)` koja računa i vraća ukupno vreme trčanja svih takmičara.
- (d) `float prosek(int a[], int n)` koja računa i vraća prosečno vreme (aritmetičku sredinu) trčanja takmičara.
- (e) `int maksimum(int a[], int n)` koja izračunava i vraća najduže vreme trčanja takmičara.
- (f) `int pozicija_minimum(int a[], int n)` koja vraća redni broj pobednika Beogradskog maratona, tj. onog takmičara koji je najkraće trčao. U slučaju da ima više takvih takmičara, vratiti onog sa najmanjim rednim brojem.

Napisati program koji učitava podatke o rezultatima takmičara na maratonu i ispisuje učitane podatke, ukupno, prosečno i maksimalno vreme trčanja, kao i redni broj pobednika maratona. Maksimalni broj takmičara je 1000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza:  
5  
Unesite elemente niza: 140 126 170 220 130  
Vreme trčanja takmicara: 140 126 170 220 130  
Ukupno vreme: 786  
Prosecko vreme trcanja: 157.20  
Maksimalno vreme trcanja: 220  
Indeks pobednika: 1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza:  
-5  
Greska: neispravan unos.
```

[Rešenje 2.1.10]

Zadatak 2.1.11 Napisati funkciju koja izračunava broj elemenata celobrojnog niza koji su manji od poslednjeg elementa niza. Napisati program koji učitava dimenziju niza i elemente niza, a zatim ispisuje broj elemenata koji zadovoljavaju pomenuti uslov. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 4  
Unesite elemente niza: 11 2 4 9  
Rezultat: 2
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 7  
Unesite elemente niza: 7 2 1 14 65 2 8  
Rezultat: 4
```

Primer 3

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite dimenziju niza: 5  
||| Unesite elemente niza: 25 18 29 30 14  
||| Rezultat: 0
```

Primer 4

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite dimenziju niza: -45  
||| Greska: neispravan unos.
```

[Rešenje 2.1.11]

Zadatak 2.1.12 Napisati funkciju koja izračunava broj parnih elemenata celobrojnog niza koji prethode maksimalnom elementu niza. Napisati program koji učitava dimenziju niza i elemente niza, a zatim ispisuje broj elemenata koji prethode maksimalnom elementu. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite dimenziju niza: 4  
||| Unesite elemente niza: 11 2 4 9  
||| Rezultat: 0
```

Primer 2

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite dimenziju niza: 7  
||| Unesite elemente niza: 7 2 1 14 65 2 8  
||| Rezultat: 2
```

Primer 3

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite dimenziju niza: 5  
||| Unesite elemente niza: 25 18 29 30 14  
||| Rezultat: 1
```

Primer 4

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite dimenziju niza: 105  
||| Greska: neispravan unos.
```

[Rešenje 2.1.12]

Zadatak 2.1.13 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 program koji učitava dimenziju niza, elemente niza i vrednosti i i j i zatim ispisuje zbir u datom opsegu. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite dimenziju 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 dimenziju niza: 3  
||| Unesite elemente niza: -2 8 1  
||| Unesite vrednosti za i i j: 1 12  
||| Greska: neispravan unos.
```

2 Napredni tipovi podataka

Primer 3

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

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 4  
Unesite elemente niza: 9 5 7 6  
Unesite vrednosti za i i j: 2 2  
Zbir je: 7
```

[Rešenje 2.1.13]

Zadatak 2.1.14 Napisati funkciju float zbir_positivnih(float a[], int n, int k) koja izračunava zbir prvih k pozitivnih elemenata realnog niza a dužine n . Napisati program koji učitava dimenziju niza, elemente niza i broj k , a zatim ispisuje zbir prvih k pozitivnih elemenata niza. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 3  
Unesite elemente niza:  
-6.598 -8.14 -15  
Unesite vrednost k: 4  
Zbir je: 0.00
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 7  
Unesite elemente niza:  
-35.11 5.29 -1.98 12.1 12.2 -3.33 -4.17  
Unesite vrednost k: 15  
Zbir je: 29.59
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 3  
Unesite elemente niza:  
-0.11 5.29 -4.17  
Unesite vrednost k: -15  
Greska: neispravan unos.
```

[Rešenje 2.1.14]

Zadatak 2.1.15 Napisati funkciju koja menja niz tako što razmenjuje mesta najmanjem i najvećem elementu niza. Ukoliko se neki od ovih elemenata javlja više puta, uzeti u obzir prvo pojavljivanje. Napisati program koji učitava dimenziju niza, elemente niza, a zatim ispisuje izmenjeni niz. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 5  
Unesite elemente niza: 8 -2 11 19 4  
Rezultujući niz:  
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  
Rezultujući niz:  
46 -2 51 8 66 -5 2 8 3 14
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 145  
Greska: neispravan unos.
```

[Rešenje 2.1.15]

Zadatak 2.1.16 Napisati program koji vrši pretragu niza nadmorskih visina.

- Napisati funkciju koja proverava da li niz sadrži zadati broj m . Povratna vrednost funkcije je 1 ako je vrednost sadržana u nizu ili 0 ako nije.
- 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.

Program učitava podatke o nadmorskim visinama i ceo broj m , a zatim ispisuje da li u nizu postoji podatak o unetoj nadmorskoj visini. Ukoliko postoji, ispisuje i poziciju prvog i poslednjeg pojavljivanja vrednosti m u nizu. Pozicije se broje od 0. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza:  
7  
Unesite podatke:  
800 1100 -200 1400 -200 1100 800  
Unesite vrednost m:  
1100  
Nadmorska visina 1100 se nalazi medju podacima.  
Pozicija prvog pojavljivanja: 1  
Pozicija poslednjeg pojavljivanja: 5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza:  
-5  
Greska: neispravan unos.
```

[Rešenje 2.1.16]

2 Napredni tipovi podataka

Zadatak 2.1.17 Marko skuplja sličice za Svetsko prvenstvo u fudbalu. Marko je primetio da mu se neke sličice ponavljaju i rešio je da ih razmeni sa drugarima. Napisati funkciju `int duplikati(int a[], int n, int b[])` koja od niza *a* dimenzije *n* formira niz *b* koji sadrži sve različite elemente niza *a* koji se pojavljuju bar dva puta u nizu. Funkcija kao povratnu vrednost vraća dimenziju niza *b*. Napisati program koji učitava brojeve Markovih sličica i ispisuje sve duplike. Maksimalni broj elemenata niza je 600. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju 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 dimenziju niza: 13  
|| Unesite elemente niza a:  
|| 8 26 7 2 1 1 7 2 2 2 7 5 1  
|| Elementi niza b: 7 2 1
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 2  
|| Unesite elemente niza a:  
|| 9 5  
|| Elementi niza b:
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 0  
|| Greska: neispravan unos.
```

[Rešenje 2.1.17]

Zadatak 2.1.18 Palindrom je tekst koji se isto čita i sa leve i sa desne strane. Napisati funkciju koja proverava da li je tekst zadat nizom karaktera palindrom (zanemariti razliku između malih i velikih slova). Napisati program koji učitava dužinu niza i niz karaktera, a zatim ispisuje da li je uneti tekst palindrom. Maksimalni broj elemenata niza je 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 15  
|| Unesite elemente niza:  
|| AnaVoljMilovana  
|| Niz jeste palindrom.
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 26  
|| Unesite elemente niza:  
|| Zanimljivo je programirati!  
|| Niz nije palindrom.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 1  
|| Unesite elemente niza:  
|| a  
|| Niz jeste palindrom.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dimenziju niza: 226  
|| Greska: neispravan unos.
```

[Rešenje 2.1.18]

Zadatak 2.1.19 Napisati funkciju koja proverava da li su elementi celobrojnog niza uređeni neopadajuće. Napisati program koji učitava dimenziju niza, elemente niza, a zatim ispisuje da li je pomenuti uslov ispunjen. Maksimalni broj elemenata niza je 300. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 7  
Unesite elemente niza: -40 -8 -8 2 30 30 46  
Niz jeste uredjen neopadajuce.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 4  
Unesite elemente niza: 4 23 15 30  
Niz nije uredjen neopadajuce.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 1  
Unesite elemente niza: 5  
Niz jeste uredjen neopadajuce.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 304  
Greska: neispravan unos.
```

[Rešenje 2.1.19]

Zadatak 2.1.20 U celobrojnem nizu se čuvaju informacije o prodaji artikala jedne prodavnice. Svaki indeks niza označava jedan dan u mesecu, a elementi niza predstavljaju broj artikala koji se prodao tog dana. Napisati funkciju koja računa najdužu uzastopnu seriju dana za koju važi da broj prodatih artikala nije opao. Napisati program koji učitava broj dana u mesecu, broj prodatih artikala za svaki dan u mesecu i zatim ispisuje dužinu izračunate serije. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 30  
Unesite broj prodatih artikala:  
89 171 112 67 119 36 181 157  
49 96 73 116 21 172  
140 0 23 71 157 135 11 166 21  
56 56 87 103 183 148 174  
Duzina najduzeg neopadajuceg  
prodavanja je 6.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 31  
Unesite broj prodatih artikala:  
215 223 262 95 18 116 334 97  
146 146 19 314 270 115 21 40  
253 27 210 68 96 175 41 242  
98 163 8 218 107 102  
Duzina najduzeg neopadajuceg  
prodavanja je 3.
```

2 Napredni tipovi podataka

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: -5
Greska: neispravan unos.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 31
Unesite broj prodatih artikala:
-215 223 262 95 18 116 334 97
146 146 19 314 -270 115 21 40
253 27 210 68 96 175 41 242
98 163 -8 218 107 102
Greska: neispravan unos.

[Rešenje 2.1.20]

Zadatak 2.1.21 Napisati funkciju koja određuje dužinu najduže serije jednakih uzastopnih elemenata u datom nizu brojeva. Napisati program koji učitava dimenziju niza i elemente niza, a zatim ispisuje dužinu najduže serije jednakih elemenata niza. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
9 -1 2 2 2 2 80 -200
Duzina najduze serije je 4.

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
9 9 0 -3 -3 -3 -3 72
Duzina najduze serije je 4.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza: 1 2 3 4 5 6 7 8
Duzina najduze serije je 1.

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 108
Greska: neispravan unos.

[Rešenje 2.1.21]

Zadatak 2.1.22 Napisati funkciju koja određuje da li se jedan niz javlja kao (uzastopni) podniz drugog niza.

- Niz b je uzastopni podniz niza a ako su elementi niza b uzastopni elementi niza a .
- Niz b je podniz niza a ako je redosled pojavljivanja elemenata niza b u nizu a isti i ne nužno uzastopan.

Napisati program koji učitava dimenzije i elemente dvaju nizova, a zatim ispisuje da li je drugi niz podniz prvog niza. Maksimalni broj elemenata nizova a i b je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 4
Unesite elemente niza: 90 -22 15 14
Elementi drugog niza cine
uzastopni podniz prvog niza.
Elementi drugog niza cine
podniz prvog niza.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 4
Unesite elemente niza: 2 7 15 7
Elementi drugog niza ne cine
uzastopni podniz prvog niza.
Elementi drugog niza cine
podniz prvog niza.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 4
Unesite elemente niza: 90 -22 200 1
Elementi drugog niza ne cine
uzastopni podniz prvog niza.
Elementi drugog niza ne
cine podniz prvog niza.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 8
Unesite elemente niza:
-4 2 7 90 -22 15 14 7
Unesite dimenziju niza: 1
Unesite elemente niza: 90
Elementi drugog niza cine
uzastopni podniz prvog niza.
Elementi drugog niza cine
podniz prvog niza.
```

[Rešenje 2.1.22]

Zadatak 2.1.23 Za celobrojni niz a dimenzije n kažemo da je *permutacija* ako sadrži sve brojeve od 1 do n .

- Napisati funkciju `void brojanje(int a[], int b[], int n)` koja na osnovu celobrojnog niza a dimenzije n formira niz b dimenzije n tako što i -ti element niza b odgovara broju pojavljivanja vrednosti i u nizu a .
- Napisati funkciju `int permutacija(int a[], int n)` koja proverava da li je zadati niz permutacija. Funkcija vraća vrednost 1 ako je svojstvo ispunjeno, odnosno 0 ako nije. UPUTSTVO: *Koristiti funkciju brojanje*.

Napisati program koji učitava dimenziju niza i elemente niza i ispisuje da li je uneti niz permutacija. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 5
Unesite elemente niza: 1 5 4 3 2
Uneti niz je permutacija.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza: 2 3 3 1 1 5
Uneti niz nije permutacija.
```

2 Napredni tipovi podataka

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 1
|| Unesite elemente niza: 1
|| Uneti niz je permutacija.

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 101
|| Greska: neispravan unos.

[Rešenje 2.1.23]

Zadatak 2.1.24 Napisati program koji učitava dva cela broja i proverava da li se uneti brojevi zapisuju pomoću istih cifara.

Primer 1

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

Primer 2

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

Primer 3

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

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dva broja: -1 1
|| Brojevi se zapisuju istim ciframa.

[Rešenje 2.1.24]

Zadatak 2.1.25 Napisati program koji vrši transformacije niza.

- Napisati funkciju koja obrće elemente niza.
- Napisati funkciju koja rotira niz ciklično za jedno mesto ulevo.
- Napisati funkciju koja rotira niz ciklično za k mesta ulevo.

Program učitava dimenziju niza, elemente niza i pozitivan ceo broj k , a zatim ispisuje niz koji se dobija nakon obrtanja početnog niza, niz koji se dobija rotiranjem tako dobijenog niza za jedno mesto ulevo i niz koji se dobija rotiranjem novodobijenog niza za k mesta ulevo. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 6
Unesite elemente niza: 7 -3 11 783 26 -19
Elementi niza nakon obrtanja:
-17 28 785 13 -1 9
Elementi niza nakon rotiranja za 1 mesto uлево:
28 785 13 -1 9 -17
Unesite jedan pozitivan ceo broj: 3
Elementi niza nakon rotiranja za 3 mesto uлево:
-1 9 -17 28 785 13
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju niza: 252
Greska: neispravan unos.
```

[Rešenje 2.1.25]

Zadatak 2.1.26 Napisati funkciju void ukrsti(int a[], int b[], int n, int c[]) koja formira niz c koji se dobija naizmeničnim raspoređivanjem elemenata nizova a i b, tj. $c = [a_0, b_0, a_1, b_1, \dots, a_{n-1}, b_{n-1}]$. Napisati program koji učitava dimenziju i elemente dvaju nizova i ispisuje niz koji se dobija ukrštanjem unetih nizova. Maksimalni broj elemenata niza a i b je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 5
Unesite elemente niza a: 2 -5 11 4 8
Unesite elemente niza b: 3 3 9 -1 17
Rezultujući niz:
2 3 -5 3 11 9 4 -1 8 17
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite dimenziju nizova: 105
Greska: neispravan unos.
```

[Rešenje 2.1.26]

Zadatak 2.1.27 Napisati funkciju void spoji(int a[], int b[], int n, int c[]) koja od nizova a i b dimenzije n formira niz c čija prva polovina odgovara elementima niza b, a druga polovina elementima niza a, tj. $c = [b_0, b_1, \dots, b_{n-1}, a_0, a_1, \dots, a_{n-1}]$. Napisati program koji učitava dimenziju i elemente dvaju nizova i ispisuje niz koji se dobija spajanjem unetih nizova na pomenuti način. Maksimalni broj elemenata niza a i b je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

2 Napredni tipovi podataka

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju nizova: 3  
Unesite elemente niza a: 4 -8 32  
Unesite elemente niza b: 5 2 11  
Rezultujući niz:  
5 2 11 4 -8 32
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju nizova: 145  
Greska: neispravan unos.
```

[Rešenje 2.1.27]

* **Zadatak 2.1.28** Napisati funkciju void spoji_sortirano(int a[], int b[], int n, int c[]) koja od nizova a i b dimenzije n koji su uređeni neopadajuće formira niz c koji je uređen na isti način. Napisati program koji učitava dimenziju i elemente uređenih nizova a i b i ispisuje niz koji se dobija spajanjem ovih nizova na pomenuti način. Maksimalni broj elemenata niza a i b je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju nizova: 5  
Unesite elemente sortiranog niza:  
2 11 28 40 63  
Unesite elemente sortiranog niza:  
-19 -5 5 11 52  
Rezultujući niz:  
-19 -5 2 5 11 11 28 40 52 63
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju nizova: 3  
Unesite elemente sortiranog niza:  
-2 4 8  
Unesite elemente sortiranog niza:  
6 15 19  
Rezultujući niz:  
-2 4 6 8 15 19
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju nizova: 145  
Greska: neispravan unos.
```

[Rešenje 2.1.28]

Zadatak 2.1.29 Napisati funkciju void promeni_redosled(int a[], int n) koja menja redosled elementima niza a dimenzije n tako da se parni elementi niza nalaze na početku niza, a neparni na kraju. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji je izmenjen na pomenuti način. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: Ne koristiti pomoćne nizove.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 10  
Unesite elemente niza:  
-2 8 11 53 59 20 17 -8 3 14  
Rezultujući niz:  
14 142 -6 -278 28 34 33 -69 -9 9
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 10  
Unesite elemente niza:  
9 142 -9 -278 -69 33 34 28 -6 14  
Rezultujući niz:  
-2 8 14 -8 20 59 17 53 3 11
```

[Rešenje 2.1.29]

Zadatak 2.1.30 Napisati funkciju koja iz datog niza briše sve elemente koji su prosti brojevi. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobija brisanjem pomenutih elemenata. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Zadatak rešiti uz korišćenje pomoćnog niza.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 5  
Unesite elemente niza: 11 5 6 48 8  
Rezultujući niz: 6 48 8
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 4  
Unesite elemente niza: 11 5 19 21  
Rezultujući niz: 21
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 5  
Unesite elemente niza: 12 18 9 31 7  
Rezultujući niz: 12 18 9
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 5  
Unesite elemente niza: -2 15 -11 8 7  
Rezultujući niz: 15 8
```

[Rešenje 2.1.30]

Zadatak 2.1.31 Napisati funkciju koja iz datog niza briše sve neparne elemente. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobija brisanjem neparnih elemenata. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Zadatak rešiti bez korišćenja pomoćnog niza.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 4  
Unesite elemente niza:  
8 9 15 12  
Rezultujući niz: 8 12
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite dimenziju niza: 6  
Unesite elemente niza:  
21 5 3 22 19 188  
Rezultujući niz: 22 188
```

2 Napredni tipovi podataka

Primer 3

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 4
|| Unesite elemente niza: 133 129 121 101
|| Rezultujuci niz:

Primer 4

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 8
|| Unesite elemente niza:
|| 15 -22 -23 13 18 46 14 -31
|| Rezultujuci niz: -22 18 46 14

[Rešenje 2.1.31]

Zadatak 2.1.32 Napisati funkciju koja iz datog niza briše sve elemente koji nisu deljivi svojom poslednjom cifrom. Izuzetak su elementi čija je poslednja cifra nula. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobija brisanjem pomenutih elemenata. Maksimalni broj elemenata niza je 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.
NAPOMENA: *Zadatak rešiti bez korišćenja pomoćnog niza.*

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 9
|| Unesite elemente niza a:
|| 173 -25 23 7 17 25 34 61 -4612
|| Rezultujuci niz: -25 7 25 61 -4612

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 0
|| Greska: neispravan unos.

[Rešenje 2.1.32]

Zadatak 2.1.33 Napisati funkciju koja iz datog niza briše sve brojeve koji nisu deljivi svojim indeksom. Ne razmatrati da li je u novom nizu, nakon brisanja i pomeranja, element deljiv svojim indeksom. Funkcija kao povratnu vrednost treba da vrati broj elemenata niza nakon brisanja. Napisati program koji učitava dimenziju niza i elemente niza i ispisuje niz koji se dobija brisanjem pomenutih elemenata. Maksimalni broj elemenata niza je 700. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Nulti element niza treba zadržati jer nije dozvoljeno deljenje nulom. Zadatak rešiti bez korišćenja pomoćnog niza.*

Primer 1

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 10
|| Unesite elemente niza:
|| 4 2 1 6 7 8 10 2 16 3
|| Rezultujuci niz: 4 2 6 16

Primer 2

|| INTERAKCIJA SA PROGRAMOM:
|| Unesite dimenziju niza: 10
|| Unesite elemente niza:
|| -8 5 10 6 7 10 8 2 16 27
|| Rezultujuci niz: -8 5 10 6 10 16 27

[Rešenje 2.1.33]

Zadatak 2.1.34 Korišćenjem nizova moguće je predstaviti skupove podataka. Napisati program koji demonstrira osnovne operacije nad skupovima (uniju, presek i razliku). Pomoću dva niza predstaviti dva skupa celih brojeva, a zatim ispisati njihovu uniju, presek i razliku. Maksimalni broj elemenata dva uneta niza je 500. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 5
Unesite elemente niza a: 1 2 3 4 5
Unesite broj elemenata niza b: 3
Unesite elemente niza b: 5 4 9
Unija: 1 2 3 4 5 9
Presek: 4 5
Razlika: 1 2 3
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 3
Unesite elemente niza a: 11 4 -5
Unesite broj elemenata niza b: 2
Unesite elemente niza b: 18 9
Unija: 11 4 -5 18 9
Presek:
Razlika: 11 4 -5
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: 6
Unesite elemente niza a: 12 7 9 12 5 1
Greska: skup ne moze imati duplike.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj elemenata niza a: -2
Greska: neispravan unos.
```

[Rešenje 2.1.34]

Zadatak 2.1.35 Da bi opsluživanje klijenata bilo efikasno i udobno, prilikom ulaska u banku svaki klijent dobija redni broj opsluživanja. Redni brojevi se čuvaju u nizu, počinju od vrednosti 1 i znova se generišu svakog radnog dana. Postoje i specijalni klijenti (npr. oni koji podižu stambeni kredit) koji mogu dobiti i negativni redni broj da bi se razlikovali od uobičajenih klijenata. Pomozite radniku obezbeđenja da lakše prati redosled opsluživanja klijenata.

- Napisati funkciju koja ubacuje redni broj klijenta x na kraj niza (klijenta koji je poslednji došao).
- Napisati funkciju koja ubacuje redni broj klijenta x na početak niza (klijenta koji će biti prvi uslužen, na primer, lica sa posebnim potrebama, trudnice ili stara lica).
- Napisati funkciju koja ubacuje redni broj klijenta x na poziciju k koju bira radnik obezbeđenja (manje prioritetna lica, recimo službena lica ili roditelji sa decom).
- Napisati funkciju koja izbacuje prvi redni broj iz niza (redni broj usluženog klijenta).

2 Napredni tipovi podataka

- (e) Napisati funkciju koja izbacuje poslednji redni broj iz niza (redni broj klijenta koji je odustao jer je shvatio da ima mnogo klijenata ispred njega).
- (f) Napisati funkciju koja izbacuje redni broj iz niza sa pozicije k (redni broj klijenta koji je odustao jer je dugo čekao).

Napisati program koji testira rad navedenih funkcija. Maksimalni broj klijenata u jednom danu je 2000. U slučaju neispravnog umosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite trenutni broj klijenata: 8  
Unesite niz sa rednim brojevima klijenata: 2 5 -2 16 33 19 8 11  
Unesite klijenta kojeg treba ubaciti u niz: 35  
Niz nakon ubacivanja klijenta: 2 5 -2 16 33 19 8 11 35  
Unesite prioritetnog klijenta kojeg treba ubaciti u niz: 36  
Niz nakon ubacivanja klijenta: 36 2 5 -2 16 33 19 8 11 35  
Unesite prioritetnog klijenta kojeg treba ubaciti u niz i njegovu poziciju: -6 2  
Niz nakon ubacivanja klijenta: 36 2 -6 5 -2 16 33 19 8 11 35  
Niz nakon odlaska klijenta: 2 -6 5 -2 16 33 19 8 11 35  
Niz nakon odlaska poslednjeg klijenta: 2 -6 5 -2 16 33 19 8 11  
Unesite redni broj klijenta koji je napustio red: -2  
Niz nakon odlaska klijenta: 2 -6 5 16 33 19 8 11
```

[Rešenje 2.1.35]

2.2 Rešenja

Rešenje 2.1.1

```

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

4 /* Predprocesorska direktiva kojom se definise maksimalan broj
   elemenata niza. */
6 #define MAKS 100

8 int main() {
10    /* Deklaracije potrebnih promenljivih. */
11    int a[MAKS];
12    int n, i;
13
14    /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
15    printf("Unesite dimenziju niza:\n");
16    scanf("%d", &n);
17    if (n <= 0 || n > MAKS) {
18        printf("Greska: neispravan unos.\n");
19        /* Za izlazak iz programa moze da se koristi i funkcija exit.
          Argument EXIT_FAILURE označava da je doslo do neke greske
          pri izvršavanju programa. Deklaracija ove funkcije se nalazi
          u zaglavlju stdlib.h. */
20        exit(EXIT_FAILURE);
21    }
22
23    /* Ucitavanje elemenata niza. */
24    printf("Unesite elemente niza:\n");
25    for (i = 0; i < n; i++)
26        scanf("%d", &a[i]);
27
28    /* Ispis elemenata niza na parnim pozicijama. */
29    printf("Elementi niza na parnim pozicijama:\n");
30    for (i = 0; i < n; i += 2)
31        printf("%d ", a[i]);
32    printf("\n");
33
34    /* Ispis parnih elemenata niza. */
35    printf("Parni elementi niza:\n");
36    for (i = 0; i < n; i++)
37        if (a[i] % 2 == 0)
38            printf("%d ", a[i]);
39    printf("\n");
40
41    /* Kada se funkciji exit prosledi EXIT_SUCCESS to znaci da se
       program uspesno zavrsio. Efekat je isti navodjenju return 0;
       naredbe na ovom mestu. */
42    exit(EXIT_SUCCESS);
43}
44
45
46

```

Rešenje 2.1.2

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 int main() {
7     /* Deklaracija potrebnih promenljivih. */
8     float brojevi[MAKS];
9     int n, i;
10
11    /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
12    printf("Unesite dimenziju niza: ");
13    scanf("%d", &n);
14    if (n <= 0 || n > MAKS) {
15        printf("Greska: neispravan unos.\n");
16        exit(EXIT_FAILURE);
17    }
18
19    /* Ucitavanje elemenata niza. */
20    printf("Unesite elemente niza:\n");
21    for (i = 0; i < n; i++)
22        scanf("%f", &brojevi[i]);
23
24    /* Ukoliko je i-ti element niza brojevi[i] negativan broj,
25       kvadrira se tako sto se pomnozi samim sobom. */
26    for (i = 0; i < n; i++)
27        if (brojevi[i] < 0)
28            brojevi[i] *= brojevi[i];
29
30    /* Ispis novodobijenog niza. */
31    printf("Rezultujuci niz: ");
32    for (i = 0; i < n; i++)
33        printf("%g ", brojevi[i]);
34    printf("\n");
35
36    exit(EXIT_SUCCESS);
37 }
```

Rešenje 2.1.3

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 int main() {
7     /* Deklaracija potrebnih promenljivih. */
8     int a[MAKS], b[MAKS];
9     int n, i, skalarni_proizvod;
```

```

11  /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
12  printf("Unesite dimenziju vektora: ");
13  scanf("%d", &n);
14  if (n <= 0 || n > MAKS) {
15      printf("Greska: neispravan unos.\n");
16      exit(EXIT_FAILURE);
17  }

18  /* Ucitavanje koordinata vektora. */
19  printf("Unesite koordinate vektora a: ");
20  for (i = 0; i < n; i++)
21      scanf("%d", &a[i]);
22  printf("Unesite koordinate vektora b: ");
23  for (i = 0; i < n; i++)
24      scanf("%d", &b[i]);

25  /* Racunanje skalarnog proizvoda po zadatoj formuli. */
26  skalarni_proizvod = 0;
27  for (i = 0; i < n; i++)
28      skalarni_proizvod += a[i] * b[i];

29  /* Ispis rezultata. */
30  printf("Skalarni proizvod: %d\n", skalarni_proizvod);

31  exit(EXIT_SUCCESS);
}

```

Rešenje 2.1.4

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 int main() {
7     /* Deklaracija potrebnih promenljivih. */
8     int brojevi[MAKS];
9     int n, i, k, indikator;
10
11    /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
12    printf("Unesite dimenziju niza: ");
13    scanf("%d", &n);
14    if (n <= 0 || n > MAKS) {
15        printf("Greska: neispravan unos.\n");
16        exit(EXIT_FAILURE);
17    }

18    /* Ucitavanje elemenata niza. */
19    printf("Unesite elemente niza: ");
20    for (i = 0; i < n; i++)
21

```

2 Napredni tipovi podataka

```
    scanf("%d", &brojevi[i]);

23   /* Ucitavanje broja k i provera ispravnosti ulaza. */
24   printf("Unesite broj k: ");
25   scanf("%d", &k);
26   if (k == 0) {
27     printf("Greska: neispravan unos.\n");
28     exit(EXIT_FAILURE);
29   }

30   /* Promenljiva koja cuva informaciju o tome da li u nizu
31      postoji element koji je deljiv brojem k. */
32   indikator = 0;

33   /* Ukoliko je element niza deljiv brojem k, indikator se
34      postavlja na 1 i ispisuje se indeks tog elementa. */
35   for (i = 0; i < n; i++) {
36     if (brojevi[i] % k == 0) {
37       if(!indikator) {
38         printf("Rezultat: ");
39         indikator = 1;
40       }
41       printf("%d ", i);
42     }
43   }

44   /* Ukoliko je indikator jednak nuli to znaci da ne postoji
45      element u nizu koji je deljiv brojem k. */
46   if (indikator == 0)
47     printf("U nizu nema elemenata koji su deljivi brojem %d.\n", k);
48   else
49     printf("\n");

50   exit(EXIT_SUCCESS);
51 }
```

Rešenje 2.1.5

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Indeksiranje autobusa pocinje od 1, pa zato maksimalna
5    dimenzija niza mora biti 201, a ne 200. */
6 #define MAKS 201
7
8 int main() {
9   /* Deklaracija potrebnih promenljivih. */
10  int n, niz[MAKS], i;
11  int k, t, m;
12
13  /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
```

```

15     printf("Unesite broj autobusa: ");
16     scanf("%d", &n);
17     if (n <= 0 || n > MAKS) {
18         printf("Greska: neispravan unos.\n");
19         exit(EXIT_FAILURE);
20     }
21
22     /* Ucitavanje vremena putovanja. */
23     printf("Unesite vreme putovanja:\n");
24     for (i = 1; i <= n; i++)
25         scanf("%d", &niz[i]);
26
27     /* Ucitavanje rednih brojeva autobusa cije se vreme putovanja
28      menja i vrednosti kasnjenja. */
29     printf("Unesite vrednosti k, t i m:\n");
30     scanf("%d%d%d", &k, &t, &m);
31
32     /* Provera ispravnosti ulaza. */
33     if (k <= 0 || k > n || t <= 0 || t > n || m < 0) {
34         printf("Greska: neispravan unos.\n");
35         exit(EXIT_FAILURE);
36     }
37
38     /* Azuriranje vremena putovanja. */
39     for (i = k; i <= t; i++)
40         niz[i] += m;
41
42     /* Ispis rezultata. */
43     printf("Vreme putovanja nakon izmena: ");
44     for (i = 1; i <= n; i++)
45         printf("%d ", niz[i]);
46     printf("\n");
47
48     exit(EXIT_SUCCESS);
49 }
```

Rešenje 2.1.6

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define BROJ_CIFARA 10
5
6 int main() {
7     /* Deklaracije potrebnih promenljivih. */
8     int x, x_original, cifra, i;
9
10    /* Svaki element niza brojac predstavlja brojac za jednu od
11       cifara: brojac[0] predstavlja broj nula u zapisu broja x
12       brojac[1] predstavlja broj jedinica u zapisu broja x ...
13       brojac[9] predstavlja broj devetki u zapisu broja x.
14 }
```

2 Napredni tipovi podataka

```
14     Brojace je potrebnoinicijalizovatipaocetku. */
15     /* I nacin: */
16     int brojaci[BROJ_CIFARA];
17     for(i=0; i<BROJ_CIFARA; i++)
18         brojaci[i] = 0;
19
20     /* II nacin: Inicijalizacija pri samoj deklaraciji.
21        Na ovaj nacin su svi elementi niza brojaci inicijalizovani na
22        nule.
23        int brojaci[BROJ_CIFARA] = {0}; */
24
25     /* Ucitavanje celog broja. */
26     printf("Unesite ceo broj:\n");
27     scanf("%d", &x);
28
29     /* Cuvanje pocetne vrednosti zbog finalnogispisa. */
30     x_original = x;
31     x = abs(x);
32
33     /* Obrada cifara. */
34     do {
35         /* Izdvajanje krajnjedesne cifre. */
36         cifra = x % 10;
37
38         /* Uvecavanje broja pojavljivanja izdvojene cifre. */
39         brojaci[cifra]++;
40
41         /* Prelazak na analizu sledece cifre. */
42         x /= 10;
43     } while (x);
44
45     /* Ispis informacija o ciframa koje se nalaze u zapisu broja x. */
46     for (i = 0; i < BROJ_CIFARA; i++)
47         if (brojaci[i]) {
48             printf("U zapisu broja %d, cifra %d se pojaviljuje %d putan\n",
49                   x_original, i, brojaci[i]);
50         }
51
52     exit(EXIT_SUCCESS);
53 }
```

Rešenje 2.1.7

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 int main() {
7     /* Deklaracije potrebnih promenljivih. */
8     char karakteri[MAKS];
```

```

9   char c;
10  int i, n;
11
12  /* Ucitavanje karaktera sve do unosa zvezdice ili do prekoracenja
13   maksimalnog broja karaktera. */
14  for (i = 0; i < MAKS; i++) {
15      printf("Unesite karakter: ");
16      scanf("%c", &c);
17      /* Citanje znaka za novi red nakon unetog karaktera. */
18      getchar();
19
20      /* Ukoliko je unet karakter '*' izlazi se iz petlje. */
21      if (c == '*')
22          break;
23
24      /* Smestanje procitanog karaktera u niz. */
25      karakteri[i] = c;
26  }
27
28  /* Broj unetih karaktera nakon izlaska iz petlje je i. */
29  n = i;
30
31  /* Ispis karaktera u obrnutom redosledu. */
32  for (i = n - 1; i >= 0; i--)
33      printf("%c ", karakteri[i]);
34  printf("\n");
35
36  exit(EXIT_SUCCESS);
37 }
```

Rešenje 2.1.8

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define BROJ_CIFARA 10
5 #define DUZINA_ABECEDA 26
6
7 /* Pomocna funkcija za ispis elemenata niza. Vrednost n označava
8  broj elemenata niza (može imati vrednost 10 ili 26), a karakter
9  c označava prvi karakter za datu kategoriju ('a' za mala slova,
10 'A' za velika slova i '0' za cifre). */
11 void ispisi(int niz[], int n, char c) {
12     int i;
13     for (i = 0; i < n; i++)
14         if (niz[i] != 0)
15             printf("Karakter %c se pojavljuje %d puta\n", c + i, niz[i]);
16
17 /* Funkcija inicijalizuje niz postavljajući vrednosti svih
18  elemenata na nulu. */
```

2 Napredni tipovi podataka

```
void inicijalizuj(int niz[], int n) {
21    int i;
22    for (i = 0; i < n; i++)
23        niz[i] = 0;
24    }

25
int main() {
26    /* Deklaracije nizova brojaca za cifre, mala i velika slova. */
27    int cifre[BROJ_CIFARA];
28    int mala_slova[DUZINA_ABECEDE];
29    int velika_slova[DUZINA_ABECEDE];

30
31    /* Deklaracije pomocnih promenljivih. */
32    int c;

33
34    /* Inicijalizacije brojaca nulama. */
35    inicijalizuj(cifre, BROJ_CIFARA);
36    inicijalizuj(mala_slova, DUZINA_ABECEDE);
37    inicijalizuj(velika_slova, DUZINA_ABECEDE);

38
39    /* Ucitavanje karaktera sve do kraja ulaza. */
40    printf("Unesite tekst:\n");
41    while ((c = getchar()) != EOF) {
42        if (c >= 'A' && c <= 'Z') {
43            /* Ako je procitani karakter veliko slovo uvecava se broj
44             pojavljivanja odgovarajuceg velikog slova. Indeks velikog
45             slova u nizu se odredjuje oduzimanjem slova 'A'.
46             Na taj nacin slovo 'A' ce imati indeks 0, slovo 'B' indeks
47             1, itd.*/
48            velika_slova[c - 'A']++;
49        } else if (c >= 'a' && c <= 'z') {
50            /* Ako je procitani karakter mali slovo uvecava se broj
51             pojavljivanja odgovarajuceg malog slova. */
52            mala_slova[c - 'a']++;
53        } else if (c >= '0' && c <= '9') {
54            /* Ako je procitani karakter cifra uvecava se broj
55             pojavljivanja odgovarajuce cifre. */
56            cifre[c - '0']++;
57        }
58    }

59
60    /* Ispis trazenih informacija. */
61    ispisi(cifre, BROJ_CIFARA, '0');
62    ispisi(mala_slova, DUZINA_ABECEDE, 'a');
63    ispisi(velika_slova, DUZINA_ABECEDE, 'A');

64
65    exit(EXIT_SUCCESS);
66}
```

Rešenje 2.1.9

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define DUZINA_ALFABETA 26
6
7 /* Pomocna funkcija za ispis elemenata niza. */
8 void ispisi(int niz[], int n) {
9     int i;
10    for (i = 0; i < n; i++) {
11        printf("%c:%d ", 'a' + i, niz[i]);
12    }
13}
14
15 int main() {
16    /* Deklaracije potrebnih promenljivih. */
17    int c;
18    int mala_slova[DUZINA_ALFABETA] = {0};
19
20    /* Ucitavanje karaktera sve do kraja ulaza. */
21    while ((c = getchar()) != EOF) {
22        /* Ako je procitani karakter slovo, broj pojavljivanja slova se
23         * uvecava. Kako se zanemaruje velicina slova, svako slovo se
24         * pretvori u malo i potom se element na odgovarajucoj poziciji
25         * u nizu uveca. */
26        if (isalpha(c))
27            mala_slova[tolower(c) - 'a']++;
28    }
29
30    /* Ispis rezultata. */
31    ispisi(mala_slova, DUZINA_ALFABETA);
32
33    exit(EXIT_SUCCESS);
34 }
```

Rešenje 2.1.10

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 1000
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12 }
```

2 Napredni tipovi podataka

```
14 /* Funkcija ispisuje elemente niza. */
15 void ispis(int a[], int n) {
16     int i;
17     for (i = 0; i < n; i++)
18         printf("%d ", a[i]);
19     printf("\n");
20 }
21
22 /* Funkcija racuna sumu elemenata niza. */
23 int suma(int a[], int n) {
24     int i, suma_elimenata = 0;
25
26     for (i = 0; i < n; i++)
27         suma_elimenata += a[i];
28
29     return suma_elimenata;
30 }
31
32 /* Funkcija racuna prosečnu vrednost elemenata niza. */
33 float prosek(int a[], int n) {
34     int suma_elimenata = suma(a, n);
35     return (float) suma_elimenata / n;
36 }
37
38 /* Funkcija izracunava maksimum elemenata niza. */
39 int maksimum(int a[], int n) {
40     int i, najveci = a[0];
41
42     for (i = 1; i < n; i++)
43         if (a[i] > najveci)
44             najveci = a[i];
45
46     return najveci;
47 }
48
49 /* Funkcija izracunava poziciju maksimalnog elementa u nizu. */
50 int pozicija_maksimuma(int a[], int n) {
51     int i, pozicija_najveceg = 0;
52
53     for (i = 1; i < n; i++)
54         if (a[i] > a[pozicija_najveceg])
55             pozicija_najveceg = i;
56
57     return pozicija_najveceg;
58 }
59
60 int main() {
61     /* Deklaracija potrebnih promenljivih. */
62     int a[MAKS];
63     int n;
64 }
```

```

66  /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
67  printf("Unesite dimenziju niza:");
68  scanf("%d", &n);
69  if (n <= 0 || n > MAKS) {
70      printf("Greska: neispravan unos.\n");
71      exit(EXIT_FAILURE);
72  }
73
74  /* Ucitavanje elemenata niza. */
75  ucitaj(a, n);
76
77  /* Ispis elemenata niza. */
78  printf("Vreme tricanja takmicara: ");
79  ispisi(a, n);
80
81  /* Ispis ukupnog, prosecnog i maksimalnog vremena. */
82  printf("Ukupno vreme: %d\n", suma(a, n));
83  printf("Prosecno vreme tricanja: %.2f\n", prosek(a, n));
84  printf("Maksimalno vreme tricanja: %d\n", maksimum(a, n));
85
86  /* Ispis indeksa pobednika. */
87  printf("Indeks pobednika: %d\n", pozicija_maksimuma(a, n));
88
89  exit(EXIT_SUCCESS);
}

```

Rešenje 2.1.11 Pogledajte zadatak 2.1.10.

Rešenje 2.1.12

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12    }
13
14 /* I nacin: Funkcija vraca poziciju najveceg elementa niza.
15    Prolazi se kroz niz i ako se naidje na element cija je
16    vrednost veca od trenutno najveceg elementa (a[pozicija]),
17    vrsi se azuriranje pozicije trenutno najveceg.
18    int pozicija_najveceg(int a[], int n) {
19        int i, pozicija = 0;
20        for(i=1; i<n; i++)
21            if(a[i] > a[pozicija])

```

2 Napredni tipovi podataka

```
        pozicija = i;
23     return pozicija;
}
25
Funkcija vraca broj parnih elemenata niza koji prethode
27 maksimalnom elementu niza.
int prebrojavanje(int a[], int n) {
29     int i;
30     int pozicija_maksimuma = pozicija_najveceg(a,n);
31     int broj_parnih = 0;
32     for (i = 0; i < pozicija_maksimuma; i++) {
33         if (a[i] % 2 == 0) {
34             broj_parnih++;
35         }
36     }
37     return broj_parnih;
}
39 */
40
41 /* II nacin:
42 Zadatak se moze resiti i jednim prolazom kroz niz. Ideja je da
43 se paralelno radi pretraga maksimalnog elementa i prebrojavanje
44 parnih elemenata koji mu prethode.
45
46 Ovo moze da se uradi sa dva brojacem parnih elemenata:
47 1. broj_parnih - brojac koji cuva broj parnih elemenata
48 koji prethode trenutnom maksimumu
49 2. broj_parnih_izmedju - brojac koji cuva broj parnih elemenata
50 koji se nalaze iza trenutnog maksimuma
51
52 Svaki put kada se maksimum azurira, na broj parnih se doda broj
53 parnih koji se prebrojao izmedju dva azuriranja, a
54 broj_parnih_izmedju se vraca na nulu. */
55 int prebrojavanje_jednim_prolazom(int a[], int n) {
56     int i;
57     int pozicija_maksimuma = 0;
58     int broj_parnih = 0;
59     int broj_parnih_izmedju = 0;
60
61     for (i = 0; i < n; i++) {
62         if (a[i] > a[pozicija_maksimuma]) {
63             pozicija_maksimuma = i;
64             broj_parnih += broj_parnih_izmedju;
65             broj_parnih_izmedju = 0;
66         }
67
68         if (a[i] % 2 == 0)
69             broj_parnih_izmedju++;
70     }
71
72     return broj_parnih;
73 }
```

```

75 int main() {
76     /* Deklaracija potrebnih promenljivih. */
77     int a[MAKS];
78     int n;
79
80     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
81     printf("Unesite dimenziju niza: ");
82     scanf("%d", &n);
83     if (n <= 0 || n > MAKST) {
84         printf("Greska: neispravan unos.\n");
85         exit(EXIT_FAILURE);
86     }
87
88     /* Ucitavanje elemenata niza. */
89     ucitaj(a, n);
90
91     /* Ispis rezultata. */
92     /* I nacin: printf("%d\n", prebrojavanje(a, n)); */
93
94     /* II nacin: Jednim prolazom kroz niz. */
95     printf("Rezultat: %d\n", prebrojavanje_jednim_prolazom(a, n));
96
97     exit(EXIT_SUCCESS);
98 }
```

Rešenje 2.1.13

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKST 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12    }
13
14 /* Funkcija sabira elemenate niza od pozicije i do pozicije j. */
15 int zbir(int a[], int i, int j) {
16     int k, rezultat = 0;
17
18     /* Obilazak elemenata niza koji pripadaju zadatom opsegu. */
19     for (k = i; k <= j; k++)
20         rezultat += a[k];
21
22     return rezultat;
23 }
```

```
25 int main() {
26     /* Deklaracije potrebnih promenljivih. */
27     int n, i, j;
28     int a[MAKS];
29
30     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
31     printf("Unesite dimenziju niza: ");
32     scanf("%d", &n);
33     if (n <= 0 || n > MAKS) {
34         printf("Greska: neispravan unos.\n");
35         exit(EXIT_FAILURE);
36     }
37
38     /* Ucitavanje elemenata niza. */
39     ucitaj(a, n);
40
41     /* Ucitavanje vrednosti granica i provera ispravnosti ulaza. */
42     printf("Unesite vrednosti za i i j: ");
43     scanf("%d%d", &i, &j);
44     if (i < 0 || j < 0 || i > n - 1 || j > n - 1 || i > j) {
45         printf("Greska: neispravan unos.\n");
46         exit(EXIT_FAILURE);
47     }
48
49     /* Ispis rezultata. */
50     printf("Zbir je: %d", zbir(a, i, j));
51
52     exit(EXIT_SUCCESS);
53 }
```

Rešenje 2.1.14

```
#include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(float a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%f", &a[i]);
12}
13
14 /* Funkcija racuna zbir prvih k pozitivnih elemenata niza. */
15 float zbir_pozitivnih(float a[], int n, int k) {
16     int i;
17     float zbir = 0;
18 }
```

```

1  /* Obilazi se element po element niza. Postupak se zavrsava
2   ukoliko se dodje do kraja niza ili ukoliko se sabere k
3   pozitivnih elemenata. */
4   for (i = 0; i < n && k > 0; i++)
5     if (a[i] >= 0) {
6       zbir += a[i];
7       /* Umanjuje se brojac pozitivnih elemenata. */
8       k--;
9     }
10
11   return zbir;
12 }

13 int main() {
14   /* Deklaracija potrebnih promenljivih. */
15   int n, k;
16   float a[MAKS];
17
18   /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
19   printf("Unesite dimenziju niza: ");
20   scanf("%d", &n);
21   if (n <= 0 || n > MAKS) {
22     printf("Greska: neispravan unos.\n");
23     exit(EXIT_FAILURE);
24   }
25
26   /* Ucitavanje elemenata niza. */
27   ucitaj(a, n);
28
29   /* Ucitavanje broja k i provera ispravnosti ulaza. */
30   printf("Unesite vrednost k: ");
31   scanf("%d", &k);
32   if (k < 0 || k > n) {
33     printf("Greska: neispravan unos.\n");
34     exit(EXIT_FAILURE);
35   }
36
37   /* Ispis rezultata. */
38   printf("Zbir je: %.2f\n", zbir_pozitivnih(a, n, k));
39
40   exit(EXIT_SUCCESS);
41 }
```

Rešenje 2.1.15

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
```

2 Napredni tipovi podataka

```
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12    }
13
14 /* Funkcija ispisuje elemente niza dimenzije n. */
15 void ispisi(int a[], int n) {
16     int i;
17     for (i = 0; i < n; i++)
18         printf("%d ", a[i]);
19     printf("\n");
20 }
21
22 /* Funkcija razmenjuje najmanji i najveci element niza. */
23 void razmeni_min_max(int brojevi[], int n) {
24     int i;
25     /* Najvecim, kao i najmanjim elementom niza, proglašava se multi
26      element niza. Pozicije najveceg i najmanjeg elementa se
27      postavljaju na 0. */
28     int najveci = brojevi[0], najmanji = brojevi[0];
29     int pozicija_najveceg = 0, pozicija_najmanjeg = 0;
30
31     /* U prolazu kroz niz trazi se najveci i najmanji element i pamte
32      se njihove pozicije. */
33     for (i = 1; i < n; i++) {
34         if (brojevi[i] > najveci) {
35             najveci = brojevi[i];
36             pozicija_najveceg = i;
37         }
38
39         if (brojevi[i] < najmanji) {
40             najmanji = brojevi[i];
41             pozicija_najmanjeg = i;
42         }
43     }
44
45     /* Zamenjuju se elementi na pozicijama pozicija_najmanjeg i
46      pozicija_najveceg. */
47     brojevi[posicija_najveceg] = najmanji;
48     brojevi[posicija_najmanjeg] = najveci;
49 }
50
51 int main() {
52     /* Deklaracija potrebnih promenljivih. */
53     int brojevi[MAKS];
54     int n;
55
56     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
57     printf("Unesite dimenziju niza: ");
58     scanf("%d", &n);
```

```

59   if (n <= 0 || n > MAKS) {
60     printf("Greska: neispravan unos.\n");
61     exit(EXIT_FAILURE);
62   }
63
64   /* Ucitavanje elemenata niza. */
65   ucitaj(brojevi, n);
66
67   /* Razmena najmanjeg i najveceg elementa. */
68   razmeni_min_max(brojevi, n);
69
70   /* Ispis rezultata. */
71   printf("Rezultujuci niz:\n");
72   ispisi(brojevi, n);
73
74   exit(EXIT_SUCCESS);
75 }
```

Rešenje 2.1.16

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8   int i;
9   printf("Unesite podatke: ");
10  for (i = 0; i < n; i++)
11    scanf("%d", &a[i]);
12 }
13
14 /* Funkcija proverava da li niz sadrzi zadatu vrednost m. */
15 int sadrzi(int a[], int n, int m) {
16   int i;
17   /* Prolazi se kroz sve elemente niza i ukoliko se naidje na
18   element cija je vrednost jednaka m, kao povratna vrednost
19   funkcije se vraca 1. */
20   for (i = 0; i < n; i++)
21     if (a[i] == m)
22       return 1;
23
24   /* Ako se stigne do kraja niza, znaci da se broj m ne nalazi
25   u nizu. */
26   return 0;
27 }
28
29 /* Funkcija vraca indeks prvog pojavljivanja elementa m u nizu a
30 ili -1 ukoliko se m ne nalazi u nizu a. */
31 int prvo_pojavljivanje(int a[], int n, int m) {
```

2 Napredni tipovi podataka

```
32 int i;
33 for (i = 0; i < n; i++)
34     if (a[i] == m)
35         return i;
36
37 /* Ako se stigne do kraja niza, znaci da se broj m ne nalazi
38    u nizu. */
39 return -1;
40 }

41 /* Funkcija vraca indeks poslednjeg pojavljivanja elementa m u nizu
42    a ili -1 ukoliko se m ne nalazi u nizu a. */
43 int poslednje_pojavljivanje(int a[], int n, int m) {
44     int i;
45
46     /* Polazi se od kraja niza i poredi se element po element sa
47        zadatim brojem m. */
48     for (i = n - 1; i >= 0; i--)
49         if (a[i] == m)
50             return i;
51
52     /* Ako se stigne do pocetka niza, znaci da se broj m ne nalazi
53        u nizu. */
54     return -1;
55 }

56 int main() {
57     /* Deklaracije potrebnih promenljivih. */
58     int a[MAKS];
59     int n, m, i;
60
61     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
62     printf("Unesite dimenziju niza: ");
63     scanf("%d", &n);
64     if (n <= 0 || n > MAKSS) {
65         printf("Greska: neispravan unos.\n");
66         exit(EXIT_FAILURE);
67     }
68
69     /* Ucitavanje elemenata niza. */
70     ucitaj(a, n);
71
72     /* Ucitavanje vrednosti za pretragu. */
73     printf("Unesite vrednost m:");
74     scanf("%d", &m);
75
76     /* Ispis rezultata pretrage. */
77     if (sadrzi(a, n, m)) {
78         printf("Nadmorska visina %d se nalazi medju podacima.\n", m);
79
80         i = prvo_pojavljivanje(a, n, m);
81         printf("Pozicija prvog pojavljivanja: %d\n", i);
82     }
83 }
```

```

84     i = poslednje_pojavljivanje(a, n, m);
85     printf("Pozicija poslednjeg pojavljivanja: %d\n", i);
86 } else
87     printf("Nadmorska visina %d se ne nalazi medju podacima.\n", m);
88
89 exit(EXIT_SUCCESS);
}

```

Rešenje 2.1.17

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 600
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza a: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12 }
13
14 /* Funkcija ispisuje elemente niza dimenzije n. */
15 void ispisi(int niz[], int n) {
16     int i;
17     for (i = 0; i < n; i++)
18         printf("%d ", niz[i]);
19     printf("\n");
20 }
21
22 /* Funkcija proverava da li niz a dimenzije n sadrzi zadatu
23    vrednost x. Pretraga se vrsti od prosledjene pozicije. */
24 int sadrzi(int niz[], int n, int od_pozicije, int x) {
25     int i;
26     for (i = od_pozicije; i < n; i++)
27         if (niz[i] == x)
28             return 1;
29
30     return 0;
31 }
32
33 /* Funkcija formira niz b tako sto u njega ubacuje sve elemente
34    niza a koji se u tom nizu pojavljuju bar dva puta. */
35 int duplikati(int a[], int n, int b[]) {
36     /* Promenljiva j je brojac elemenata rezultujuceg niza. */
37     int i, j = 0;
38
39     /* Obilazi se element po element niza a. Trenutni element je
40        duplikat ukoliko se javlja jos neki put u nizu a. Dovoljno je

```

2 Napredni tipovi podataka

```
41     gledati da li se nalazi iza tekuceg elementa jer ako se
42     nalazi ispred, onda je on vec obradjen (i duplikat je
43     detektovan). Element a[i] se dodaje u niz duplikata ako vazi:
44     1. a[i] je duplikat
45     2. a[i] se ne nalazi u nizu duplikata
46     Provera sadrzi(a, n, i+1, a[i]) proverava prvi uslov.
47     Provera !sadrzi(b, j, 0, a[i]) proverava drugi uslov. */
48     for (i = 0; i < n; i++)
49         if (sadrzi(a, n, i + 1, a[i]) && !sadrzi(b, j, 0, a[i])) {
50             b[j] = a[i];
51             j++;
52         }
53
54     /* Povratna vrednost funkcije je duzina niza b. */
55     return j;
56 }
57
58 int main() {
59     /* Deklaracija potrebnih promenljivih. */
60     int a[MAKS], b[MAKS];
61     int n_a, n_b;
62
63     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
64     printf("Unesite broj n: ");
65     scanf("%d", &n_a);
66     if (n_a <= 0 || n_a > MAKS) {
67         printf("Greska: neispravan unos.\n");
68         exit(EXIT_FAILURE);
69     }
70
71     /* Ucitavanje podataka o slicicama. */
72     ucitaj(a, n_a);
73
74     /* Popunjavanje niza b duplikatima niza a. */
75     n_b = duplikati(a, n_a, b);
76
77     /* Ispis rezultata. */
78     printf("Elementi niza b: ");
79     ispisi(b, n_b);
80
81     exit(EXIT_SUCCESS);
82 }
```

Rešenje 2.1.18

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS 200
```

```

7  /* Funkcija ucitava elemente niza dimenzije n. */
8  void ucitaj(char niz[], int n) {
9      int i;
10     printf("Unesite elemete niza: ");
11     for (i = 0; i < n; i++)
12         scanf("%c", &niz[i]);
13 }

15 /* Funkcija proverava da li je niz karaktera palindrom. */
16 int je_palindrom(char niz[], int n) {
17     int i, j;
18     /* U petlji se porede elementi niz[0] i niz[n-1], zatim niz[1] i
19        niz[n-2]
20        itd. Ako se nađe na par elemenata koji se razlikuju, može se
21        zaključiti da
22        niz nije palindrom. */
23     for (i = 0, j = n-1; i < j; i++, j--)
24         if (tolower(niz[i]) != tolower(niz[j]))
25             return 0;
26
27     /* Izvrsila se cela petlja pa se može zaključiti da je niz
28        palindrom. */
29     return 1;
30 }
31
32 int main() {
33     /* Deklaracije potrebnih promenljivih. */
34     char niz[MAKS];
35     int n;
36
37     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
38     printf("Unesite dimenziju niza: ");
39     scanf("%d", &n);
40     if (n <= 0 || n > MAKS) {
41         printf("Greska: neispravan unos.\n");
42         exit(EXIT_FAILURE);
43     }
44
45     /* Preskace se novi red nakon unosa dimenzije. Ovo se radi jer
46        sledi učitavanje karaktera i bez ove linije, prvi karakter
47        koji bi se upisao u niz bi bio novi red. */
48     getchar();
49
50     /* Ucitavanje elemenata niza. */
51     ucitaj(niz, n);
52
53     /* Ispis rezultata. */
54     if (je_palindrom(niz, n))
55         printf("Niz jeste palindrom.\n");
56     else
57         printf("Niz nije palindrom.\n");

```

```
57     exit(EXIT_SUCCESS);
}
```

Rešenje 2.1.19

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 300
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12    }
13
14 /* Funkcija proverava da li je niz uredjen neopadajuce. */
15 int uredjen_neopadajuce(int niz[], int n) {
16     int i;
17     for (i = 0; i < n - 1; i++)
18         if (niz[i] > niz[i + 1])
19             return 0;
20
21     return 1;
22 }
23
24 int main() {
25     /* Deklaracija potrebnih promenljivih. */
26     int n, niz[MAKS];
27
28     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
29     printf("Unesite dimenziju niza: ");
30     scanf("%d", &n);
31     if (n <= 0 || n > MAKS) {
32         printf("Greska: neispravan unos.\n");
33         exit(EXIT_FAILURE);
34     }
35
36     /* Ucitavanje elemenata niza. */
37     ucitaj(niz, n);
38
39     /* Ispis rezultata. */
40     if (uredjen_neopadajuce(niz, n))
41         printf("Niz jeste uredjen neopadajuce.\n");
42     else
43         printf("Niz nije uredjen neopadajuce.\n");
44
45     exit(EXIT_SUCCESS);
46 }
```

Rešenje 2.1.20

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Maksimalan broj dana u mesecu je 31, ali dani pocinju od 1, pa
   je potrebno odvojiti 32 mesta u nizu jer se nulti ne koristi. */
5 #define MAKS_DANA 32
6
7 /* Funkcija ucitava elemente niza dimenzije n. */
8 void ucitaj(int a[], int n) {
9     int i;
10    printf("Unesite broj prodatih artikala: ");
11    for (i = 0; i < n; i++) {
12        scanf("%d", &a[i]);
13        if (a[i] < 0) {
14            printf("Greska: neispravan unos.\n");
15            exit(EXIT_FAILURE);
16        }
17    }
18 }
19
20 /* Funkcija racuna duzinu najduzeg neopadajuceg podniza niza a. */
21 int najduzi_neopadajuci(int a[], int n) {
22     int i;
23     /* Na pocetku duzina trenutne serije i duzina maksimalne serije
       se inicijalizuju na 1. */
24     int duzina_trenutne_serije = 1;
25     int duzina_najduze_serije = 1;
26
27     for (i = 1; i < n; i++) {
28         /* Proverava se da li su uzastopni elementi u neopadajucem
            poretku. Ako je to slučaj uvecava se duzina serije, a
            ako nije, duzina trenutne serije se vraca na 1,
            kako bi se ispravno racunala duzina sledeće serije. */
29         if (a[i] >= a[i - 1])
30             duzina_trenutne_serije++;
31         else
32             duzina_trenutne_serije = 1;
33
34         /* Ukoliko je trenutna duzina serije veca od duzine do sada
            najduze serije, azurira se vrednost duzine najduze serije. */
35         if (duzina_trenutne_serije > duzina_najduze_serije)
36             duzina_najduze_serije = duzina_trenutne_serije;
37     }
38
39     return duzina_najduze_serije;
40 }
41
42 int main() {
43     /* Deklaracija potrebnih promenljivih. */
44     int a[MAKS_DANA], n;
45 }
```

```
52     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
53     printf("Unesite dimenziju niza: ");
54     scanf("%d", &n);
55     if (n <= 0 || n > MAKS_DANA) {
56         printf("Greska: neispravan unos.\n");
57         exit(EXIT_FAILURE);
58     }
59
60     /* Ucitavanje elemenata niza. */
61     ucitaj(a, n);
62
63     /* Ispis rezultata. */
64     printf("Duzina najduzeg neopadajućeg prodavanja je %d.\n",
65            najduzi_neopadajuci(a, n));
66
67     exit(EXIT_SUCCESS);
68 }
```

Rešenje 2.1.21

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12    }
13
14 /* Funkcija vraca duzinu najduze serije jednakih elemenata niza. */
15 int najduza_serija(int a[], int n) {
16     int i;
17     /* Na pocetku i duzina trenutne serije i duzina maksimalne serije
18      se inicializuju na 1. */
19     int trenutna_serija = 1;
20     int najduza_serija = 1;
21
22     for (i = 1; i < n; i++) {
23         /* Proverava se da li su uzastopni elementi jednaki. Ako je to
24          slucaj Uvecavanje duzina serije. Ako uzastopni elementi nisu
25          jednaki serija je prekinuta i vrednost duzine trenutne serije
26          se postavlja ponovo na 1 da bi mogla da se racuna duzina
27          sledeće serije. */
28         if (a[i] == a[i - 1])
29             trenutna_serija++;
30         else
```

```

31     trenutna_serija = 1;
33
34     /* Ukoliko je trenutna duzina serije veca od duzine do sada
35      najduze serije, parametar za duzinu najduze serije se
36      postavlja na novu, vecu vrednost. */
37     if (trenutna_serija > najduza_serija)
38         najduza_serija = trenutna_serija;
39
40     return najduza_serija;
41 }
42
43 int main() {
44     /* Deklaracija potrebnih promenljivih. */
45     int n, a[MAKS];
46
47     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
48     printf("Unesite dimenziju niza: ");
49     scanf("%d", &n);
50     if (n <= 0 || n > MAKS) {
51         printf("Greska: neispravan unos.\n");
52         exit(EXIT_FAILURE);
53     }
54
55     /* Ucitavanje elemenata niza. */
56     ucitaj(a, n);
57
58     /* Ispis rezultata. */
59     printf("Duzina najduze serije je %d.\n", najduza_serija(a, n));
60
61     exit(EXIT_SUCCESS);
62 }
```

Rešenje 2.1.22

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int niz[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &niz[i]);
12 }
13
14 /* a) */
15 int podniz_uzastopnih(int a[], int n, int b[], int m) {
16     int i, j;
```

```

18  /* Obilaze se elementi prvog niza. Svaki element prvog niza moze
20   * biti pocetak podniza, odnosno pocetak drugog niza. */
21   for (i = 0; i + m - 1 < n; i++) {
22     /* Prolaze se elementi drugog niza. Za svaki element niza b
23      proverava se da li je jednak odgovarajucem elementu niza a.
24      Za niz a razmatra se da li podniz pocinje od pozicije i.
25      Tako 0-ti element niza b je na poziciji i, 1. element je na
26      poziciji i+1, 2. na poziciji i+2, ..., j-ti na poziciji i+j.
27      Ako uslov nije ispunjen, petlja se prekida i proverava se da
28      li na sledecoj poziciji u nizu a pocinje podniz. */
29      for (j = 0; j < m; j++)
30        if (a[i + j] != b[j])
31          break;
32      /* Ako petlja nije prekinuta nakon ispitivanja, brojac za niz b
33      je jedanak dimenziji niza b, odnosno svi elementi niza b se
34      uzastopno nalaze u nizu a. */
35      if (j == m)
36        return 1;
37    }
38
39    /* Ukoliko niz b jeste uzastopni podniz uslov u petlji ce u nekom
40    trenutku biti ispunjen i iz petlje i funkcije ce se izaci sa
41    return naredbom. Ipak, ako se to nije desilo i dalje se
42    izvrsava funkcija, onda niz b nije uzastopni podniz. */
43    return 0;
44  }
45
46  /* b) */
47  int podniz(int a[], int n, int b[], int m) {
48    int i, j;
49
50    /* Obilaze se elementi niza a. */
51    for (i = 0, j = 0; i < n && j < m; i++) {
52      /* Svaki put kada se naidje na element niza b, brojac za niz b
53      se uvecava i proverava se da li se sledeci element niza b
54      nalazi u nizu a. */
55      if (a[i] == b[j])
56        j++;
57    }
58
59    /* Ukoliko se pronadju svi elementi niza b u nizu a, onda je
60    brojac za niz b jednak dimenziji niza b. U tom slucaju se
61    vraca vrednost 1, odnosno da niz jeste podniz. */
62    return j == m;
63  }
64
65  int main() {
66    /* Deklaracija potrebnih promenljivih. */
67    int n, a[MAKS];
68    int m, b[MAKS];

```

```

70  /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
71  printf("Unesite dimenziju niza: ");
72  scanf("%d", &n);
73  if (n <= 0 || n > MAKS) {
74      printf("Greska: neispravan unos.\n");
75      exit(EXIT_FAILURE);
76  }
77
78  /* Ucitavanje elemenata prvog niza. */
79  ucitaj(a, n);
80
81  /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
82  printf("Unesite dimenziju niza: ");
83  scanf("%d", &m);
84  if (m <= 0 || m > MAKS) {
85      printf("Greska: neispravan unos.\n");
86      exit(EXIT_FAILURE);
87  }
88
89  /* Ucitavanje elemenata drugog niza. */
90  ucitaj(b, m);
91
92  /* a) */
93  if (podniz_uzastopnih(a, n, b, m))
94      printf("Elementi drugog niza cine uzastopni podniz "
95             "prvog niza.\n");
96  else
97      printf("Elementi drugog niza ne cine uzastopni podniz "
98             "prvog niza.\n");
99
100 /* b) */
101 if (podniz(a, n, b, m))
102     printf("Elementi drugog niza cine podniz prvog niza.\n");
103 else
104     printf("Elementi drugog niza ne cine podniz prvog niza.\n");
105
106 exit(EXIT_SUCCESS);
107 }
```

Rešenje 2.1.23

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int niz[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++) {
```

2 Napredni tipovi podataka

```
11     scanf("%d", &niz[i]);  
  
13     /* Niz moze sadrzati elemente koji nisu u opsegu od 1 do n. U  
14      tom slucaju taj niz nije permutacija. */  
15     if (niz[i] <= 0 || niz[i] > n) {  
16         printf("Uneti niz nije permutacija.\n");  
17         exit(EXIT_SUCCESS);  
18     }  
19 }  
20  
21 /* Funkcija prebrojava koliko puta se pojavljuje svaki element niza  
22   a. */  
23 void brojanje(int a[], int b[], int n) {  
24     int i;  
  
25     /* Niz b se inicijalizuje tako sto se za svaki element postavi  
26       da se poljavljuje 0 puta u nizu a. */  
27     for (i = 1; i <= n; i++)  
28         b[i] = 0;  
  
29     /* Petljom se prolazi kroz niz a i za svaki element a[i] uvecava  
30       se broj njegovog pojavljivanja u nizu b. Na primer, ako je  
31       a[3] = 7, onda treba uvecati broj pojavljivanja broja 7, a to  
32       je b[7]++;
33       sto se krace moze zapisati kao b[a[3]]++.  
34       Prepostavlja se da je niz a dobro zadat, odnosno da su sve  
35       njegove vrednosti u intervalu od 1 do n. */  
36     for (i = 0; i < n; i++)  
37         b[a[i]]++;  
38 }  
39  
40 /* Funkcija proverava da li je niz a permutacija. */  
41 int permutacija(int a[], int n) {  
42     /* Niz b moze imati index MAKS (jer niz b se posmatra od 1 do  
43       MAKS), pa zato njegova dimenzija mora biti za jedan veca. */  
44     int b[MAKS + 1];  
45     int i;  
  
46     /* Racunanje broja pojavljivanja svakog broja niza a. */  
47     brojanje(a, b, n);  
  
48     /* Ukoliko se svaki element niza a javlja tacno jednom u nizu a,  
49       onda niz a jeste permutacija. Ovo svojstvo se proverava  
50       koriscenjem dobijenog niza b. */  
51     for (i = 1; i <= n; i++)  
52         if (b[i] != 1)  
53             return 0;  
  
54     return 1;  
55 }  
56  
57 int main() {
```

```

63  /* Deklaracija potrebnih promenljivih. */
64  int a[MAKS], n;
65
66  /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
67  printf("Unesite dimenziju niza: ");
68  scanf("%d", &n);
69  if (n <= 0 || n > MAKST) {
70      printf("Greska: neispravan unos.\n");
71      exit(EXIT_FAILURE);
72 }
73
74  /* Ucitavanje elemenata niza a. */
75  ucitaj(a, n);
76
77  /* Ispis rezultata. */
78  if (permutacija(a, n))
79      printf("Uneti niz je permutacija.\n");
80  else
81      printf("Uneti niz nije permutacija.\n");
82
83  exit(EXIT_SUCCESS);
}

```

Rešenje 2.1.24

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define BROJ_CIFARA 10
5
6 /* Funkcija inicijalizuje niz postavljajući vrednosti svih
7   elemenata na nulu. */
8 void inicijalizuj(int niz[], int n) {
9     int i;
10    for (i = 0; i < n; i++)
11        niz[i] = 0;
12 }
13
14 /* Funkcija izdvaja cifru po cifru broja i uvecava odgovarajuci
15   element niza koji odgovara brojacu za tu cifru. Na primer, za
16   broj=1123, po zavrsetku ove funkcije niz[1] ce imati vrednost 2
17   jer se cifra 1 pojavljuje 2 puta, niz[2] i niz[3] ce imati
18   vrednost 1, a svi ostali elementi niza ce imati vrednost 0. */
19 void analiza_cifara(int broj, int niz[]) {
20     int c;
21
22     /* Inicijalizacija svih brojaca na nule. */
23     inicijalizuj(niz, BROJ_CIFARA);
24
25     /* Uvecavanje odgovarajucih brojaca. */
26     do {

```

2 Napredni tipovi podataka

```
27     c = broj % 10;
28     niz[c]++;
29     broj /= 10;
30 } while (broj);
31 }

33 int main() {
34     /* Niz cifre_broja_x predstavlja brojace za cifre broja x.
35      Niz cifre_broja_y predstavlja brojace za cifre broja y. */
36     int cifre_broja_x[BROJ_CIFARA], cifre_broja_y[BROJ_CIFARA];
37     int x, y, i, indikator;

38     /* Ucitavanje brojeva x i y. */
39     printf("Unesite dva broja: ");
40     scanf("%d%d", &x, &y);

41     /* Za slucaj da su unete vrednosti negativne, posmatra se njihova
42      apsolutna vrednost. Ovo je opravdano iz razloga sto se brojevi
43      x i -x zapisuju istim ciframa. */
44     x = abs(x);
45     y = abs(y);

46     /* Popunjavaju se nizovi brojacima cifara. */
47     analiza_cifara(x, cifre_broja_x);
48     analiza_cifara(y, cifre_broja_y);

49     /* Promenljiva indikator sluzi za pracenje da li su oba broja
50      sastavljena od istih cifara. */
51     indikator = 1;

52     for (i = 0; i < BROJ_CIFARA; i++) {
53         /* Ako se broj pojavljivanja cifre i u zapisu broja x razlikuje
54            od broja pojavljivanja cifre i u zapisu broja y, brojevi se
55            ne zapisuju istim ciframa. Zato se vrednost indikatora moze
56            postaviti na 0 i prekinuti dalje uporedjivanje broja
57            pojavljivanja. */
58         if (cifre_broja_y[i] != cifre_broja_x[i]) {
59             indikator = 0;
60             break;
61         }
62     }
63     /* Ako je vrednost promenljive indikator ostala 1, to znaci da u
64      petlji nije pronadjena cifra koja se ne pojavljuje isti broj
65      puta u zapisima brojeva x i y. Zato se moze zakljucliti da se
66      brojevi zapisuju istim ciframa. */
67     if (indikator)
68         printf("Brojevi se zapisuju istim ciframa.\n");
69     else
70         printf("Brojevi se ne zapisuju istim ciframa.\n");

71     exit(EXIT_SUCCESS);
72 }
```

Rešenje 2.1.25

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12}
13
14 /* Funkcija ispisuje elemente niza dimenzije n. */
15 void ispisi(int a[], int n) {
16     int i;
17     for (i = 0; i < n; i++)
18         printf("%d ", a[i]);
19     printf("\n");
20}
21
22 /* Funkcija obrce elemente niza. */
23 void obrni(int a[], int n) {
24     int t, i, j;
25
26     /* Za niz a[0], a[1], ..., a[n-2], a[n-1] obrnuti niz je a[n-1],
27      a[n-2], ..., a[1], a[0]. Zato je potrebno razmeniti vrednosti
28      elemenata a[0] i a[n-1], a[1] i a[n-2], itd. i zaustaviti se
29      kada je vrednost indeksa prvog elementa veca od vrednosti
30      drugog elementa. */
31     for (i = 0, j = n - 1; i < j; i++, j--) {
32         t = a[i];
33         a[i] = a[j];
34         a[j] = t;
35     }
36}
37
38 /* Funkcija rotira niz ciklicno za jedno mesto u levo. */
39 void rotiraj_za_1(int a[], int n) {
40     int i, prvi = a[0];
41
42     /* Pomeranje preostalih elemenata niza za jedno mesto u levo. */
43     for (i = 0; i < n - 1; i++)
44         a[i] = a[i + 1];
45
46     /* Poslednjem elementu se dodeljuje sacuvana vrednost prvog
47      elementa. */
48     a[n - 1] = prvi;
49}

```

2 Napredni tipovi podataka

```
51 /* Funkcija rotira niz ciklicno za k mesta u levo. */
52 void rotiraj_za_k(int a[], int n, int k) {
53     int i;
54
55     /* Odredjuje se vrednost broja k koja je u opsegu od 0 do n-1
56      kako bi se izbegla suvisna pomeranja. */
57     k = k % n;
58
59     /* Niz se rotira za jednu poziciju uлево k puta. */
60     for (i = 0; i < k; i++)
61         rotiraj_za_1(a, n);
62 }
63
64 int main() {
65     /* Deklaracija potrebnih promenljivih. */
66     int a[MAKS];
67     int n, k;
68
69     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
70     printf("Unesite dimenziju niza: ");
71     scanf("%d", &n);
72     if (n <= 0 || n > MAKS) {
73         printf("Greska: neispravan unos.\n");
74         exit(EXIT_FAILURE);
75     }
76
77     /* Ucitavanje elemenata niza. */
78     ucitaj(a, n);
79
80     /* Obrtanje niza. */
81     printf("Elementi niza nakon obrtanja:\n");
82     obrni(a, n);
83     ispisi(a, n);
84
85     /* Rotiranje za jedno mesto u levo. */
86     printf("Elementi niza nakon rotiranja za 1 mesto uлево:\n");
87     rotiraj_za_1(a, n);
88     ispisi(a, n);
89
90     /* Rotiranje za k mesta u levo. */
91     printf("Unesite jedan pozitivan ceo broj:");
92     scanf("%d", &k);
93     if (k <= 0) {
94         printf("Greska: neispravan unos.\n");
95         exit(EXIT_FAILURE);
96     }
97     rotiraj_za_k(a, n, k);
98     printf("Elementi niza nakon rotiranja za %d mesto uлево:\n", k);
99     ispisi(a, n);
100
101    exit(EXIT_SUCCESS);
102 }
```

Rešenje 2.1.26

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int niz[], int n) {
8     int i;
9     for (i = 0; i < n; i++)
10        scanf("%d", &niz[i]);
11 }
12
13 /* Funkcija ispisuje elemente niza dimenzije n. */
14 void ispisi(int niz[], int n) {
15     int i;
16     for (i = 0; i < n; i++)
17         printf("%d ", niz[i]);
18     printf("\n");
19 }
20
21 /* Funkcija formira niz c ukrstanjem nizova a i b. */
22 void ukrsti(int a[], int b[], int n, int c[]) {
23     int i, j;
24     /* Formira se treći niz. Koriste se dva indeksa: indeks i
25        pomocu kojeg se pristupa elementima nizova a i b i koji treba
26        uvecati za 1 nakon svake iteracije i indeks j pomocu kojeg se
27        pristupa elementima rezultujuceg niza c; s obzirom da se u
28        svakoj iteraciji u niz c smestaju dva elementa, jedan iz niza
29        a i jedan iz niza b, indeks j se uvecava za 2 nakon svake
30        iteracije. */
31     for (i = 0, j = 0; i < n; i++, j += 2) {
32         c[j] = a[i];
33         c[j + 1] = b[i];
34     }
35 }
36
37 int main() {
38     /* Deklaracija potrebnih promenljivih. */
39     int a[MAKS], b[MAKS], c[2 * MAKS];
40     int n;
41
42     /* Ucitavanje dimenzije nizova i provera ispravnosti ulaza. */
43     printf("Unesite dimenziju nizova: ");
44     scanf("%d", &n);
45     if (n <= 0 || n > MAKS) {
46         printf("Greska: neispravan unos.\n");
47         exit(EXIT_FAILURE);
48     }
49
50     /* Ucitavanje elemenata nizova. */

```

2 Napredni tipovi podataka

```
51 printf("Unesite elemente niza a: ");
52 ucitaj(a, n);
53 printf("Unesite elemente niza b: ");
54 ucitaj(b, n);
55
56 /* Formiranje niza c. */
57 ukrsti(a, b, n, c);
58
59 /* Ispis elemenata rezultujuceg niza. */
60 printf("Rezultujuci niz:\n");
61 ispisi(c, 2 * n);
62
63 exit(EXIT_SUCCESS);
64 }
```

Rešenje 2.1.27

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int niz[], int n) {
8     int i;
9     for (i = 0; i < n; i++)
10         scanf("%d", &niz[i]);
11 }
12
13 /* Funkcija ispisi elemente niza dimenzije n. */
14 void ispisi(int niz[], int n) {
15     int i;
16     for (i = 0; i < n; i++)
17         printf("%d ", niz[i]);
18     printf("\n");
19 }
20
21 /* Funkcija formira niz c nadovezivanjem nizova a i b. */
22 void spoji(int a[], int b[], int n, int c[]) {
23     int i;
24
25     /* Niz c ima 2*n elemenata: prvih n elemenata su elementi niza b,
26     a narednih n elemenata elementi niza a. Elementi niza b se
27     nalaze na pozicijama 0,1,2,...n-1, a elementi niza a na
28     pozicijama n,n+1,...2*n-1. Jednim prolaskom kroz petlju na
29     poziciju i u nizu c se postavlja element b[i] niza b, a na
30     poziciju n+i element a[i] niza a. */
31     for (i = 0; i < n; i++) {
32         c[i] = b[i];
33         c[n + i] = a[i];
34     }
35 }
```

```

35 }
36
37 int main() {
38     /* Deklaracija potrebnih promenljivih. */
39     int a[MAKS], b[MAKS], c[2 * MAK];
40     int n;
41
42     /* Ucitavanje dimenzije nizova i provera ispravnosti ulaza. */
43     printf("Unesite dimenziju nizova: ");
44     scanf("%d", &n);
45     if (n <= 0 || n > MAK) {
46         printf("Greska: neispravan unos.\n");
47         exit(EXIT_FAILURE);
48     }
49
50     /* Ucitavanje elemenata nizova. */
51     printf("Unesite elemente niza a: ");
52     ucitaj(a, n);
53     printf("Unesite elemente niza b: ");
54     ucitaj(b, n);
55
56     /* Formiranje niza c. */
57     spoji(a, b, n, c);
58
59     /* Ispis elemenata rezultujuceg niza. */
60     printf("Rezultujuci niz:\n");
61     ispisi(c, 2 * n);
62
63     exit(EXIT_SUCCESS);
64 }
```

Rešenje 2.1.28

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int niz[], int n) {
8     int i;
9     printf("Unesite elemente sortiranog niza:\n");
10    for (i = 0; i < n; i++)
11        scanf("%d", &niz[i]);
12
13 /* Funkcija ispisuje elemente niza dimenzije n. */
14 void ispisi(int niz[], int n) {
15     int i;
16     for (i = 0; i < n; i++)
17         printf("%d ", niz[i]);
```

2 Napredni tipovi podataka

```
19     printf("\n");
}
21
int main() {
/* Deklaracija potrebnih promenljivih. */
23     int a[MAKS], b[MAKS], c[2 * MAKST];
25     int n;
/* Brojac u petlji za elemente niza a. */
27     int i = 0;
/* Brojac u petlji za elemente niza b. */
29     int j = 0;
/* Brojac u petlji za elemente niza c. */
31     int k = 0;

/* Ucitavanje dimenzije nizova i provera ispravnosti ulaza. */
33     printf("Unesite dimenziju nizova: ");
35     scanf("%d", &n);
36     if (n <= 0 || n > MAKST) {
37         printf("Greska: neispravan unos.\n");
38         exit(EXIT_FAILURE);
39     }

/* Ucitavanje elemenata nizova. */
41     ucitaj(a, n);
42     ucitaj(b, n);

/* Spajanje nizova. */
44     while (i < n && j < n) {
45         /* Porede se elementi nizova a i b i u niz c upisuje se samo
46             onaj koji je manji. Ako je upisan element iz niza a, onda se
47             vrsti i uvecavanje brojaca i (prelazak na sledeci element niza
48             a), a ako je upisan element iz niza b, onda se vrsti
49             uvecavanje brojaca j (prelazak na sledeci element niza b). */
50         if (a[i] < b[j]) {
51             c[k] = a[i];
52             i++;
53         } else {
54             c[k] = b[j];
55             j++;
56         }
57     }

/* U nizu c na poziciju k je upisan ili a[i] ili b[j]. Brojac k
   se uvecava. */
59     k++;
60 }
61
/* Ukoliko je ostalo elemenata u nizu a, upisuju se u niz c. */
62     while (i < n) {
63         c[k] = a[i];
64         k++;
65         i++;
66     }
67 }
```

```

71  /* Ukoliko je ostalo elemenata u nizu b, upisuju se u niz c. */
73  while (j < n) {
74      c[k] = b[j];
75      k++;
76      j++;
77  }

79  /* Ispis elemenata niza c cija dimenzija je zbir dimenzija nizova
80   * a i b. */
81  printf("Rezultujući niz:\n");
82  ispisi(c, 2 * n);
83
84  exit(EXIT_SUCCESS);
85 }

```

Rešenje 2.1.29

```

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

4 #define MAKS 100

6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12    }

14 /* Funkcija ispisuje elemente niza dimenzije n. */
15 void ispisi(int niz[], int n) {
16     int i;
17     for (i = 0; i < n; i++)
18         printf("%d ", niz[i]);
19     printf("\n");
20    }

22 /* Funkcija razmenjuje elemente niza tako da se na pocetku niza
23  nalaze svi parni elementi niza, a zatim svi neparni elementi
24  niza. */
25 void promeni_redosled(int niz[], int n) {
26     int i = 0, j = n - 1, pom;

28     /* Kreće se od pocetka niza (po brojacu i) i od kraja niza (po
29      brojacu j) i svaki put kada se nađe na elemente koji po
30      parnosti ne odgovaraju delu niza u kome treba da budu,
31      zamene se njihove vrednosti. */
32     while (i < j && i < n && j >= 0) {
33         if (niz[i] % 2 != 0 && niz[j] % 2 == 0) {

```

2 Napredni tipovi podataka

```
34     pom = niz[i];
35     niz[i] = niz[j];
36     niz[j] = pom;
37 }
38
39 /* Ukoliko je element na poziciji i paran, Prelazak na
40    sledeci element niza, brojac i se uvecava. */
41 if (niz[i] % 2 == 0)
42     i++;
43
44 /* Ukoliko je element na poziciji j neparan, Prelazak na
45    sledeci element niza, brojac j se smanjuje. */
46 if (niz[j] % 2 != 0)
47     j--;
48 }
49
50 int main() {
51     /* Deklaracija potrebnih promenljivih. */
52     int niz[MAKS];
53     int n;
54
55     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
56     printf("Unesite dimenziju niza: ");
57     scanf("%d", &n);
58     if (n <= 0 || n > MAKST) {
59         printf("Greska: neispravan unos.\n");
60         exit(EXIT_FAILURE);
61     }
62
63     /* Ucitavanje elemenata niza. */
64     ucitaj(niz, n);
65
66     /* Izmena niza na trazeni nacin. */
67     promeni_redosled(niz, n);
68
69     /* Ispis rezultata. */
70     printf("Rezultujuci niz:\n");
71     ispisi(niz, n);
72
73     exit(EXIT_SUCCESS);
74 }
```

Rešenje 2.1.30

```
#include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 #define MAKST 100
6
```

```

 8  /* Funkcija ucitava elemente niza dimenzije n. */
 9  void ucitaj(int a[], int n) {
10    int i;
11    printf("Unesite elemente niza: ");
12    for (i = 0; i < n; i++)
13      scanf("%d", &a[i]);
14  }
15
16  /* Funkcija ispisuje elemente niza dimenzije n. */
17  void ispisi(int a[], int n) {
18    int i;
19    for (i = 0; i < n; i++)
20      printf("%d ", a[i]);
21    printf("\n");
22  }
23
24  /* Funkcija vraca 1 ako je broj prost, a 0 u suprotnom. */
25  int prost(int x) {
26    int i;
27
28    /* Brojevi 2 i 3 su prosti. */
29    if (x == 2 || x == 3)
30      return 1;
31
32    /* Parni brojevi nisu prosti. */
33    if (x % 2 == 0)
34      return 0;
35
36    /* Ako se naidje na broj koji deli x, onda broj x nije
37     prost. Provera se vrsi za sve neparne brojeve izmedju 3 i
38     korena broja x, jer kada bi x imao parnog delioca, onda bi
39     i broj 2 delio x, a taj uslov je vec proveren. */
40    int koren_x = sqrt(x);
41    for (i = 3; i <= koren_x; i += 2)
42      if (x % i == 0)
43        return 0;
44
45    /* Ako nijedan od prethodnih uslova nije bio ispunjen, to znaci
46     da nijedan broj ne deli x, pa je on prost. */
47    return 1;
48  }
49
50  /* Funkcija od niza a formira niz b koji sadrzi sve elemente niza a
51     koji nisu prosti brojevi. Povratna vrednost funkcije je broj
52     elemenata niza b. */
53  int obrisi_proste(int a[], int n, int b[]) {
54    int i, j;
55
56    /* Kada se u nizu a naidje na prost element, on se upisuje u niz
57     b i Uvecavanje brojac za niz b. */
58    for (i = 0, j = 0; i < n; i++)
59      if (prost(a[i]) == 0) {

```

2 Napredni tipovi podataka

```
1         b[j] = a[i];
2         j++;
3     }
4
5     return j;
6 }
7
8 int main() {
9     /* Deklaracije potrebnih promenljivih. */
10    int a[MAKS], b[MAKS];
11    int n_a, n_b;
12
13    /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
14    printf("Unesite dimenziju niza: ");
15    scanf("%d", &n_a);
16    if (n_a <= 0 || n_a > MAKS) {
17        printf("Greska: neispravan unos.\n");
18        exit(EXIT_FAILURE);
19    }
20
21    /* Ucitavanje elemenata niza. */
22    ucitaj(a, n_a);
23
24    /* Formira se niz b brisanjem prostih brojeva iz niza a. */
25    n_b = obrisi_proste(a, n_a, b);
26
27    /* Ispis elemenata niza b. */
28    printf("Rezultujuci niz:\n");
29    ispisi(b, n_b);
30
31    exit(EXIT_SUCCESS);
32 }
```

Rešenje 2.1.31

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 100
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12
13    /* Funkcija ispisi elemente niza dimenzije n. */
14 void ispisi(int niz[], int n) {
15     int i;
```

```

17   for (i = 0; i < n; i++)
18     printf("%d ", niz[i]);
19   printf("\n");
20 }
21
/* Funkcija brise sve neparne elemente niza. */
22 int obrisi_neparne(int a[], int n) {
23   int i, j;
24   /* Promenljiva j predstavlja brojac prve slobodne pozicije na
25      koju se moze upisati element niza koji treba da ostane u nizu.
26      Kada se naidje na element koji je paran, on se kopira na
27      mesto a[j] i poveca se vrednost brojaca j. Ukoliko se naidje
28      na element koji je neparan, njega treba preskociti. */
29   for (i = 0, j = 0; i < n; i++) {
30     /* Ako je tekuci element niza a paran. */
31     if (a[i] % 2 == 0) {
32       /* Premesta se na poziciju j. */
33       a[j] = a[i];
34
35       /* Vrednost brojaca j se priprema za narednu iteraciju. */
36       j++;
37     }
38     /* Ako je tekuci element niza a neparan, sa njim nista ne treba
39       raditi. */
40   }
41
42   /* Rezultujuci niz ima j elemenata. */
43   return j;
44 }
45
46 int main() {
47   /* Deklaracija potrebnih promenljivih. */
48   int a[MAKS];
49   int n;
50
51   /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
52   printf("Unesite dimenziju niza: ");
53   scanf("%d", &n);
54   if (n <= 0 || n > MAKS) {
55     printf("Greska: neispravan unos.\n");
56     exit(EXIT_FAILURE);
57   }
58
59   /* Ucitavanje elemenata niza. */
60   ucitaj(a, n);
61
62   /* Brisanje neparnih elemenata niza. */
63   n = obrisi_neparne(a, n);
64
65   /* Ispis elemenata izmenjenog niza a. */
66   printf("Rezultujuci niz:\n");
67   ispisi(a, n);

```

2 Napredni tipovi podataka

```
69     exit(EXIT_SUCCESS);
71 }
```

Rešenje 2.1.32 Pogledajte zadatke 2.1.30 i 2.1.31.

Rešenje 2.1.33

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 700
5
6 /* Funkcija ucitava elemente niza dimenzije n. */
7 void ucitaj(int a[], int n) {
8     int i;
9     printf("Unesite elemente niza: ");
10    for (i = 0; i < n; i++)
11        scanf("%d", &a[i]);
12 }
13
14 /* Funkcija ispisuje elemente niza dimenzije n. */
15 void ispisi(int a[], int n) {
16     int i;
17     for (i = 0; i < n; i++)
18         printf("%d ", a[i]);
19     printf("\n");
20 }
21
22 /* Funkcija pomera za jedno mesto u levo elemente niza a pocevsi od
23 pozicije j. Element na poziciji j se brise i na njegovo mesto se
24 upisuje element na poziciji j+1, a u skladu sa tim svi ostali
25 elementi posle njega u nizu se pomeraju. */
26 void pomeri_za_jedno_mesto(int a[], int n, int j) {
27     int i;
28     for (i = j; i < n - 1; i++)
29         a[i] = a[i + 1];
30 }
31
32 /* Funkcija brise sve elemente niza koji nisu deljivi svojim
33 indeksom. Povratna vrednost funkcije je broj elemenata
34 rezultujuceg niza. */
35 int brisanje(int niz[], int n) {
36     int i;
37
38     /* Potrebno je krenuti od poslednjeg elementa niza i petljom ici
39     ka pocetku niza (element na poziciji 0 se ne razmatra).
40     Proverava se da li je element potrebno obrisati i ako jeste
41     vrsti se pomeranje elemenata niza za jedno mesto u levo.
42     Prednost ovog resenja u odnosu na resenje kada se krene od
43     pocetka niza je u tome sto element koji se ispituje sigurno
```

```

    nije promenio svoju poziciju usled pomeranja zbog brisanja.
    Problem se može resiti i koriscenjem pomocnog niza (uraditi za
    vezbu). To rešenje je efikasnije, ali troši vise resursa. */
45   for (i = n - 1; i > 0; i--)
46     if (niz[i] % i != 0) {
47       pomeri_za_jedno_mesto(niz, n, i);
48       /* Nakon brisanja elementa, smanjuje se i dimenzija niza. */
49       n--;
50     }
51
52   return n;
53 }

54 int main() {
55   /* Deklaracija potrebnih promenljivih. */
56   int n, niz[MAKS];
57
58   /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
59   printf("Unesite dimenziju niza: ");
60   scanf("%d", &n);
61   if (n <= 0 || n > MAKS) {
62     printf("Greska: neispravan unos.\n");
63     exit(EXIT_FAILURE);
64   }
65
66   /* Ucitavanje elemenata niza. */
67   ucitaj(niz, n);
68
69   /* Brisanje traženih elemenata. */
70   n = brisanje(niz, n);
71
72   /* Ispis rezultujuceg niza. */
73   printf("Rezultujući niz:\n");
74   ispisi(niz, n);
75
76   exit(EXIT_SUCCESS);
77 }

```

Rešenje 2.1.34

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 500
5
6 /* Funkcija vraca 1 ukoliko broj x postoji u nizu, 0 inace. */
7 int postoji(int niz[], int n, int x) {
8   int i;
9   for (i = 0; i < n; i++)
10     if (niz[i] == x)
11       return 1;

```

2 Napredni tipovi podataka

```
13     return 0;
14 }
15 /* Funkcija ucitava elemente niza dimenzije n. */
16 void ucitaj(int niz[], int n) {
17     int i, element;
18     printf("Unesite elemente niza: ");
19     for (i = 0; i < n; i++) {
20         scanf("%d", &element);
21         if (postoji(niz, i, element)) {
22             printf("Greska: skup ne moze imati duplike.\n");
23             exit(EXIT_FAILURE);
24         }
25         niz[i] = element;
26     }
27 }
28
29 /* Funkcija ispisuje elemente niza dimenzije n. */
30 void ispisi(int niz[], int n) {
31     int i;
32     for (i = 0; i < n; i++)
33         printf("%d ", niz[i]);
34     printf("\n");
35 }
36
36 int main() {
37     /* Deklaracija potrebnih promenljivih. */
38     int a[MAKS], b[MAKS], unija[2 * MAKRS], presek[MAKS],
39         razlika[MAKS];
40     int i, n_a, n_b, n_unija, n_presek, n_razlika;
41
42     /* Ucitavanje dimenzije prvog niza i provera ispravnosti ulaza. */
43     printf("Unesite dimenziju niza: ");
44     scanf("%d", &n_a);
45     if (n_a <= 0 || n_a > MAKRS) {
46         printf("Greska: neispravan unos.\n");
47         exit(EXIT_FAILURE);
48     }
49
50     /* Ucitavanje elemenata niza. */
51     ucitaj(a, n_a);
52
53     /* Ucitavanje dimenzije drugog niza i provera ispravnosti
54      ulaza. */
55     printf("Unesite dimenziju niza: ");
56     scanf("%d", &n_b);
57     if (n_b <= 0 || n_b > MAKRS) {
58         printf("Greska: neispravan unos.\n");
59         exit(EXIT_FAILURE);
60     }
61 }
62
63
```

```

  /* Ucitavanje elemenata niza. */
65  ucitaj(b, n_b);

  /* Brojac elemenata u nizovima unija, presek i razlika. */
67  n_unija = n_presek = n_razlika = 0;

69  for (i = 0; i < n_a; i++) {
71    /* Svi elementi niza a se dodaju u uniju. */
72    unija[n_unija] = a[i];
73    n_unija++;

75    /* Ukoliko se element a[i] nalazi u nizu b dodaje se nizu presek
76    i
77    povecava se brojac elemenata u nizu presek. */
78    if (postoji(b, n_b, a[i]) == 1) {
79      presek[n_presek] = a[i];
80      n_presek++;
81    }

82    /* Ukoliko element a[i] ne postoji u nizu b dodaje se nizu
83    razlika i
84    povecava se brojac elemenata u nizu razlika. */
85    if (postoji(b, n_b, a[i]) == 0) {
86      razlika[n_razlika] = a[i];
87      n_razlika++;
88    }
89  }

90  /* Elemente niza b koji nisu uneti u uniju dodaju se u uniju. */
91  for (i = 0; i < n_b; i++)
92    if (postoji(unija, n_unija, b[i]) == 0) {
93      unija[n_unija] = b[i];
94      n_unija++;
95    }

96  /* Ispis rezultata. */
97  printf("Unija: ");
98  ispisi(unija, n_unija);

100  printf("Presek: ");
101  ispisi(presek, n_presek);

103  printf("Razlika: ");
104  ispisi(razlika, n_razlika);

106  exit(EXIT_SUCCESS);
}

```

Rešenje 2.1.35

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 2000
5
6 /* Funkcija ispisuje elemente niza dimenzije n. */
7 void ispis(int niz[], int n) {
8     int i;
9     for (i = 0; i < n; i++)
10        printf("%d ", niz[i]);
11    printf("\n");
12}
13
14 /* Funkcija ubacuje element x na kraj niza i vraca novu dimenziju
15   niza. */
16 int ubaci_na_kraj(int niz[], int n, int x) {
17     if (n == MAKS) {
18         printf("Greska: prekoracen je maksimalan broj elemenata niza.");
19         exit(EXIT_FAILURE);
20     }
21
22     niz[n] = x;
23     return n + 1;
24}
25
26 /* Funkcija ubacuje element x na pocetak niza i vraca novu dimenziju
27   niza. */
28 int ubaci_na_pocetak(int niz[], int n, int x) {
29     if (n == MAKS) {
30         printf("Greska: prekoracen je maksimalan broj elemenata niza.");
31         exit(EXIT_FAILURE);
32     }
33
34     int i;
35     /* Prvo se svi elementi niza pomere za jednu poziciju u desno da
36       bi se oslobođio prostor za prvi element niza. Poslednji
37       element niza se pomera sa pozicije (n-1) na poziciju (n).
38       Slicno se pomeraju i ostali elementi. */
39     for (i = n; i > 0; i--)
40         niz[i] = niz[i - 1];
41
42     /* Na prvu poziciju se upisuje novi element. Bitan je redosled
43       naredbi: ako bi prvo bio upisan novi element, a tek onda
44       izvršeno pomeranje, element na poziciji niz[0] bi bio obrisan
45       i ne bi mogao biti upisan na poziciju niz[1]. */
46     niz[0] = x;
47
48     return n + 1;
49}
```

```

51 /* Funkcija ubacuje element x na neku poziciju u nizu i vraca novu
   dimenziju niza. */
53 int ubaci_na_poziciju(int niz[], int n, int x, int pozicija) {
54     if (n == MAKS) {
55         printf("Greska: prekoracen je maksimalan broj elemenata niza.");
56         exit(EXIT_FAILURE);
57     }
58
59     int i;
60     /* Prvo se svi elementi niza od pozicije do kraja pomere za jedno
61      mesto u desno da bi se osloboudio prostor za novi element niza.
62      */
63     for (i = n; i > pozicija; i--)
64         niz[i] = niz[i - 1];
65
66     /* Na poziciju se upisuje novi element. */
67     niz[pozicija] = x;
68
69     return n + 1;
70 }
71
71 /* Funkcija brise prvi element niza i vraca novu dimenziju niza. */
72 int brisi_prvog(int niz[], int n) {
73     if (n == 0) {
74         printf("Greska: nije moguce brisanje iz praznog niza.\n");
75         exit(EXIT_FAILURE);
76     }
77
78     int i;
79     /* Svi elementi niza pomjeraju se za jedno mesto u levo. */
80     for (i = 0; i < n - 1; i++)
81         niz[i] = niz[i + 1];
82
83     return n - 1;
84 }
85
85 /* Funkcija brise poslednji element niza i vraca novu dimenziju
   niza. */
86 int brisi_poslednjeg(int niz[], int n) {
87     if (n == 0) {
88         printf("Greska: nije moguce brisanje iz praznog niza.\n");
89         exit(EXIT_FAILURE);
90     }
91
92     /* Dovoljno je smanjiti dimenziju niza, elemente niza nije
       potrebno brisati. */
93     return n - 1;
94 }
95
96 /* Funkcija brise element x i vraca novu dimenziju niza.
   Prepostavlja se da element ima samo jedno pojavljivanje. */
97 int brisi_element(int niz[], int n, int x) {
98
99
100}

```

2 Napredni tipovi podataka

```
int i, j;

103 /* Prvo treba pronaci poziciju elementa u nizu. */
105 for (i = 0; i < n; i++)
106     if (niz[i] == x)
107         break;

109 /* Provera da li element postoji u nizu. Ako je brojac stigao do
110    kraja niza, onda element ne postoji u nizu. */
111 if (i == n) {
112     printf("Klijent sa rednim brojem %d ne postoji u nizu.\n", x);
113     return n;
114 }

115 /* Ukoliko element postoji u nizu, svi elementi niza nakon njega
116    se pomjeraju za jedno mesto u levo. */
117 for (j = i; j < n - 1; j++)
118     niz[j] = niz[j + 1];

119 return n - 1;
120 }

123 int main() {
124     int n, niz[MAKS], i, klijent, pozicija;

127     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
128     printf("Unesite trenutni broj klijenata: ");
129     scanf("%d", &n);
130     if (n <= 0 || n > MAKS) {
131         printf("Greska: neispravan unos.\n");
132         exit(EXIT_FAILURE);
133     }

135     /* Ucitavanje elemenata niza. */
136     printf("Unesite niz sa rednim brojevima klijenata: ");
137     for (i = 0; i < n; i++)
138         scanf("%d", &niz[i]);

139     /* Ubacivanje klijenta na kraj. */
140     printf("Unesite broj klijenta kojeg treba ubaciti u niz: ");
141     scanf("%d", &klijent);
142     n = ubaci_na_kraj(niz, n, klijent);
143     printf("Niz nakon ubacivanja klijenta:\n");
144     ispis(niz, n);

147     /* Ubacivanje klijenta na pocetak. */
148     printf("Unesite prioritetnog klijenta kojeg treba"
149           "ubaciti u niz: ");
150     scanf("%d", &klijent);
151     n = ubaci_na_pocetak(niz, n, klijent);
152     printf("Niz nakon ubacivanja klijenta:\n");
153     ispis(niz, n);
```

```
155  /* Ubacivanje klijenta na zadatu poziciju. */
156  printf("Unesite prioritetnog klijenta kojeg treba ubaciti "
157      "u niz i njegovu poziciju:");
158  scanf("%d%d", &klijent, &pozicija);
159  if (pozicija < 0 || pozicija > n) {
160      printf("Greska: neispravan unos.\n");
161      exit(EXIT_FAILURE);
162  } else {
163      n = ubaci_na_poziciju(niz, n, klijent, pozicija);
164      printf("Niz nakon ubacivanja klijenta:\n");
165      ispis(niz, n);
166  }
167
168  /* Brisanje prvog klijenta. */
169  n = brisi_prvog(niz, n);
170  printf("Niz nakon odlaska klijenta:\n");
171  ispis(niz, n);
172
173  /* Brisanje poslednjeg klijenta. */
174  n = brisi_poslednjeg(niz, n);
175  printf("Niz nakon odlaska klijenta:\n");
176  ispis(niz, n);
177
178  /* Brisanje klijenta sa datim rednim brojem. */
179  printf("Unesite redni broj klijenta koji je napustio red: ");
180  scanf("%d", &klijent);
181  n = brisi_element(niz, n, klijent);
182  printf("Niz nakon odlaska klijenta:\n");
183  ispis(niz, n);
184
185  exit(EXIT_SUCCESS);
}
```

2.3 Pokazivači

Zadatak 2.3.1 Napisati funkciju void uredi(int *pa, int *pb) koja uređuje svoja dva celobrojna argumenta tako da se u prvom nalazi manja vrednost, a u drugom veća. Napisati program koji učitava dva cela broja i ispisuje uređene brojeve.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 2 5
Uredjene promenljive: 2, 5

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite dva broja: 11 -4
Uredjene promenljive: -4, 11

[Rešenje 2.3.1]

Zadatak 2.3.2 Napisati funkciju void rgb_u_cmy(int r, int g, int b, float *c, float *m, float *y) koja datu boju u *rgb* formatu konvertuje u boju u *cmy* formatu po sledećim formulama:

$$c = 1 - r/255, \quad m = 1 - g/255, \quad y = 1 - b/255$$

Napisati program koji učitava boju u *rgb* formatu i ispisuje vrednosti unete boje u *cmy* formatu. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *Vrednosti boja u *rgb* formatu su u opsegu [0, 255].*

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite boju u *rgb* formatu: 56 111 24
cmy: (0.78, 0.56, 0.91)

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite boju u *rgb* formatu: 156 -90 5
Greska: neispravan unos.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite boju u *rgb* formatu: 9 0 237
cmy: (0.96, 1.00, 0.07)

Primer 4

INTERAKCIJA SA PROGRAMOM:
Unesite boju u *rgb* formatu: 300 11 27
Greska: neispravan unos.

[Rešenje 2.3.2]

Zadatak 2.3.3 Napisati funkciju int presek(float k1, float n1, float k2, float n2, float *px, float *py) koja za dve razne prave date svojim koeficijentima pravca i slobodnim članovima određuje njihovu tačku preseka. Funkcija treba da vrati jedinicu ako se prave seku, a nulu ako nemaju tačku

2.3 Pokazivači

preseka (ako su paralelne). Napisati program koji učitava podatke o pravama i ukoliko prave imaju presek, ispisuje koordinate tačke preseka, a ako nemaju, ispisuje odgovarajuću poruku.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite k i n za prvu pravu: 4 5
Unesite k i n za drugu pravu: 11 -4
Prave se sekut u tacki (1.29, 10.14).

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite k i n za prvu pravu: 0.5 -4.7
Unesite k i n za drugu pravu: 0.5 9.1
Prave su paralelne.

[Rešenje 2.3.3]

Zadatak 2.3.4 Napisati funkciju koja za dva cela broja izračunava njihov količnik i ostatak pri deljenju. Funkcija treba da vrati jedinicu ukoliko je uspešno izračunala vrednosti, a nulu ukoliko deljenje nije moguće. Napisati program koji učitava dva cela broja i ispisuje njihov količnik i ostatak pri deljenju. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 4 5
Kolicnik: 0
Ostatak: 4

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: 4 0
Greska: neispravan unos.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite brojeve: -123 11
Kolicnik: -11
Ostatak: -2

[Rešenje 2.3.4]

Zadatak 2.3.5 Napisati funkciju koja za dužinu trajanja filma koja je data u sekundama, određuje ukupno trajanje filma u satima, minutima i sekundama. Napisati program koji učitava trajanje filma u sekundama i ispisuje odgovarajuće vreme trajanja u formatu *broj_sati:h:minutam:broj_sekundis*. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Trajanje fima u sekundama: 5000
1h:23m:20s

Primer 2

INTERAKCIJA SA PROGRAMOM:
Trajanje fima u sekundama: -300
Greska: neispravan unos.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Trajanje fima u sekundama: 2500
0h:41m:40s

Primer 4

INTERAKCIJA SA PROGRAMOM:
Trajanje fima u sekundama: 7824
2h:10m:24s

[Rešenje 2.3.5]

2 Napredni tipovi podataka

Zadatak 2.3.6 Napisati funkciju koja sa ulaza učitava karakter po karakter sve do kraja ulaza, a zatim prebrojava sva pojavljivanja karaktera tačka i sva pojavljivanja karaktera zarez. Napisati program koji za uneti tekst ispisuje koliko puta se pojavila tačka, a koliko puta se pojavio zarez.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
Bio jednom jedan lav...  
Kakav lav?  
Strasan lav,  
narogusen i ljut sav!  
Broj tacaka: 3  
Broj zareza: 1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
Bavite se sportom,  
ne moze da skodi,  
sportisti su bili  
i bice u modi.  
Kondicije puni,  
uvek vedri, zdravi.  
Svako dete treba  
sportom da se bavi.  
Broj tacaka: 3  
Broj zareza: 4
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
Na sirokom carskom drumu  
sto preseca prasumu  
sreli se beli slon  
i jedan crni telefon!  
Broj tacaka: 0  
Broj zareza: 0
```

[Rešenje 2.3.6]

Zadatak 2.3.7 Napisati funkciju void par_nepar(int a[], int n, int parni[], int *np, int neparni[], int *nn) koja razbija niz *a* na niz parnih i niz neparnih brojeva. Vrednost na koju pokazuje pokazivač *np* treba da bude jednak broju elemenata niza *parni*, a vrednost na koju pokazuje pokazivač *nn* treba da bude jednak broju elemenata niza *neparni*. Maksimalan broj elemenata niza je 50. Napisati program koji učitava dimenziju niza, a zatim i elemente niza i ispisuje odgovarajuće nizove parnih, odnosno neparnih elemenata unetog niza. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: 8  
Unesite elemente niza:  
1 8 9 -7 -16 24 77 4  
Niz parnih brojeva: 8 -16 24 4  
Niz neparnih brojeva: 1 9 -7 77
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: 5  
Unesite elemente niza:  
2 4 6 8 -11  
Niz parnih brojeva: 2 4 6 8  
Niz neparnih brojeva: -11
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: 2  
Unesite elemente niza: -15 15  
Niz parnih brojeva:  
Niz neparnih brojeva: -15 15
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: 0  
Greska: neispravan unos.
```

[Rešenje 2.3.7]

Zadatak 2.3.8 Napisati funkciju koja izračunava najmanji i najveći element niza realnih brojeva. Napisati program koji učitava niz realnih brojeva maksimalne dužine 50 i ispisuje vrednosti najmanjeg i najvećeg elementa niza, zaokružene na tri decimale. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: 5  
Unesite elemente niza:  
24.16 -32.11 999.25 14.25 11  
Najmanji: -32.110  
Najveci: 999.250
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: 4  
Unesite elemente niza:  
-5.126 -18.29 44 29.268  
Najmanji: -18.290  
Najveci: 44.000
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: 1  
Unesite elemente niza: 4.16  
Najmanji: 4.160  
Najveci: 4.160
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj elemenata niza: -3  
Greska: neispravan unos.
```

[Rešenje 2.3.8]

2.4 Rešenja

Rešenje 2.3.1

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Argumenti funkcije uredi_pogresno, promenljive a i b,
5    predstavljaju lokalne promenljive za ovu funkciju i prestaju da
6    postoje po zavrsetku funkcije. Zbog toga se efekti razmene
7    vrednosti promenljivih a i b u slucaju da je a>b ne vide u
8    glavnom programu.
9    void uredi_pogresno(int a, int b) {
10       int pom;
11       if (a > b) {
12           pom = a;
13           a = b;
14           b = pom;
15       }
16   } */
17
18 /* Argumenti funkcije uredi, promenljive pa i pb, takodje su
19    lokalne promenljive za ovu funkciju i prestaju da postoje kada
20    se funkcija zavrsi. Razlika je u tome sto su one adrese
21    promenljivih a i b koje zelimo da razmenimo u slucaju da je a>b.
22
23    Promenljivoj a se pristupa preko pokazivacke promenljive pa sa
24    *pa i slicno, promenljivoj b sa *pb.
25
26    Vrednosti promenljivih *pa i *pb se razmenjuju kao i vrednosti
27    bilo koje dve celobrojne promenljive. */
28 void uredi(int *pa, int *pb) {
29     int pom;
30     if (*pa > *pb) {
31         pom = *pa;
32         *pa = *pb;
33         *pb = pom;
34     }
35 }
36
37 int main() {
38     /* Deklaracija potrebnih promenljivih. */
39     int a, b;
40
41     /* Ucitavanje vrednosti dva cela broja. */
42     printf("Unesite dva broja:");
43     scanf("%d%d", &a, &b);
44
45     /* Neispravan nacin:
46        uredi_pogresno(a, b);
47        printf("Uredjene promenljive: %d, %d\n", a, b); */
```

```

49  /* Funkcija uredi kao argumente prima dve pokazivacke promenljive
50   (int*,int*). Zbog toga je u pozivu funkcije neophodno
51   proslediti adrese promenljivih koje zelimo da uredimo rastuce:
52   &a i &b. */
53   uredi(&a, &b);
54   printf("Uredjene promenljive: %d, %d\n", a, b);
55
56   exit(EXIT_SUCCESS);
57 }
```

Rešenje 2.3.2

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4
5 #define MIN_RGB 0
6 #define MAKS_RGB 255
7
8 /* Funkcija vrši konverziju boje iz rgb formata u cmy format.
9   Kako se pomocu naredbe return ne može vratiti više od jedne
10  vrednosti, neophodno je da se promenljive cije se vrednosti
11  racunaju prenesu preko pokazivaca. */
12 void rgb_u_cmy(int r, int g, int b, float *c, float *m, float *y) {
13     *c = 1 - r / 255.0;
14     *m = 1 - g / 255.0;
15     *y = 1 - b / 255.0;
16 }
17
18 /* Funkcija proverava da li je vrednost boje u ispravnom opsegu. */
19 int ispravna_rgb_vrednost(int boja) {
20     if (boja < MIN_RGB || boja > MAKS_RGB)
21         return 0;
22     return 1;
23 }
24
25 int main() {
26     /* Deklaracije potrebnih promenljivih. */
27     int r, g, b;
28     float c, m, y;
29
30     /* Ucitavanje vrednosti boje u rgb formatu. */
31     printf("Unesite boju u rgb formatu: ");
32     scanf("%d%d%d", &r, &g, &b);
33
34     /* Provera ispravnosti ulaza. */
35     if (!ispravna_rgb_vrednost(r) || !ispravna_rgb_vrednost(g) ||
36         !ispravna_rgb_vrednost(b)) {
37         printf("Greska: neispravan unos.\n");
38         exit(EXIT_FAILURE);
39     }
40 }
```

```
39     /* Konverzija boje i ispis rezultata. Funkciji se kao argumenti
40      prosledjuju vrednosti brojeva r, g, i b, kao i adrese na koje
41      treba da se upisu izracunate c, m, y vrednosti. */
42     rgb_u_cmy(r, g, b, &c, &m, &y);
43     printf("cmy: (%.2f, %.2f, %.2f)\n", c, m, y);
44
45     exit(EXIT_SUCCESS);
46 }
```

Rešenje 2.3.3

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija racuna presek pravih
5   y = k1 * x + n1 i y = k2 * x + n2.
6   Koordinate preseka (ako postoji) se upisuju na adrese px i
7   py. Kao povratna vrednost funkcije se vraca jedinica ukoliko
8   presek postoji, a nula inace. */
9 int presek(float k1, float n1, float k2, float n2, float *px,
10            float *py) {
11     /* Ako je koeficijent pravca jednak, prave su paralelne. */
12     if (k1 == k2)
13         return 0;
14
15     /* Koordinate preseka se upisuju na adrese (px, py). */
16     *px = -(n1 - n2) / (k1 - k2);
17     *py = k1 * (*px) + n1;
18
19     /* Funkcija vraca 1 kao indikator da presek postoji. */
20     return 1;
21 }
22
23 int main() {
24     /* Deklaracije potrebnih promenljivih. */
25     float k1, k2, n1, n2;
26     float x, y;
27
28     /* Ucitavanje parametara za dve prave. */
29     printf("Unesite k i n za prvu pravu: ");
30     scanf("%f%f", &k1, &n1);
31     printf("Unesite k i n za drugu pravu: ");
32     scanf("%f%f", &k2, &n2);
33
34     /* Ispis rezultata. */
35     if (presek(k1, n1, k2, n2, &x, &y))
36         printf("Prave se sekaju tacki (%.2f, %.2f).\n", x, y);
37     else
38         printf("Prave su paralelne.\n");
```

```
40     exit(EXIT_SUCCESS);
}
```

Rešenje 2.3.4 Pogledajte zadatke 2.3.2 i 2.3.3.

Rešenje 2.3.5

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija koja dato trajanje izrazeno u ukupnom broju sekundi
5    konvertuje u trajanje koje je izrazeno u broju sati, minuta i
6    sekundi.*/
7 void konverzija(int trajanje, int *psati, int *pminuti,
8                  int *psekunde) {
9     *psati = trajanje / 3600;
10    trajanje -= *psati * 3600;
11
12    *pminuti = trajanje / 60;
13    trajanje -= *pminuti * 60;
14
15    *psekunde = trajanje;
16}
17
18 int main() {
19     /* Deklaracija potrebnih promenljivih. */
20     int trajanje, sati, minuti, sekunde;
21
22     /* Ucitavanje trajanja u sekundama i provera ispravnosti ulaza. */
23     printf("Trajanje filma u sekundama: ");
24     scanf("%d", &trajanje);
25     if (trajanje < 0) {
26         printf("Greska: neispravan unos.\n");
27         exit(EXIT_FAILURE);
28     }
29
30     /* Racunanje i ispis rezultata. */
31     konverzija(trajanje, &sati, &minuti, &sekunde);
32     printf("%dh:%dm:%ds\n", sati, minuti, sekunde);
33
34     exit(EXIT_SUCCESS);
35 }
```

Rešenje 2.3.6

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija ucitava karakter po karakter sa ulaza i prebrojava
5   koliko puta se pojавio karakter '.' i koliko puta se pojavit
```

2 Napredni tipovi podataka

```
6     karakter ',' . */
7 void interpunkcija(int *br_tacaka, int *br_zareza) {
8     int tacke = 0, zarezi = 0;
9     char c;
10
11    /* Ucitavanje i prebrojavanje trazenih karaktera. */
12    while ((c = getchar()) != EOF) {
13        if (c == '.')
14            tacke++;
15
16        if (c == ',')
17            zarezi++;
18    }
19
20    /* Smestanje rezultata na prosledjene adrese. */
21    *br_tacaka = tacke;
22    *br_zareza = zarezi;
23}
24
25int main() {
26    /* Deklaracije potrebnih promenljivih. */
27    int br_tacaka, br_zareza;
28
29    /* Ucitavanje i obrada teksta. */
30    printf("Unesite tekst: \n");
31    interpunkcija(&br_tacaka, &br_zareza);
32
33    /* Ispis rezultata. */
34    printf("Broj tacaka: %d\n", br_tacaka);
35    printf("Broj zareza: %d\n", br_zareza);
36
37    exit(EXIT_SUCCESS);
38}
```

Rešenje 2.3.7

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija od niza a formira dva niza: niz parnih elemenata
7   niza a i niz neparnih elemenata niza a. Duzine rezultujucih
8   nizova se upisuju na adrese np i nn. */
9 void par_nepar(int a[], int n, int parni[], int *np,
10                 int neparni[], int *nn) {
11     int i, j, k;
12
13     /* Promenljiva i je brojac u originalnom nizu i on se uvecava u
14       svakoj iteraciji. Promenljiva j je projac za niz parnih
15       brojeva i on treba da se uveca svaki put kada se naidje na
```

```

17     novi element ovog niza. Promenljiva k je brojac za niz
18     neparnih brojeva i on treba da se uveca sveki put kada se
19     nađe na novi element ovog niza. */
20     for (i = 0, j = 0, k = 0; i < n; i++) {
21         if (a[i] % 2 == 0) {
22             parni[j] = a[i];
23             j++;
24         } else {
25             neparni[k] = a[i];
26             k++;
27         }
28     }

29     /* Na kraju petlje, u promenljivoj j se nalazi podatak o broju
30     elemenata niza parni[], a u promenljivoj k podatak o broju
31     elemenata niza neparni[]. Ove vrednosti se upisuju na adrese np
32     i nn. */
33     *np = j;
34     *nn = k;
35 }

36 /* Funkcija ispisuje elemente niza. */
37 void ispisi(int niz[], int n) {
38     int i;
39     for (i = 0; i < n; i++)
40         printf("%d ", niz[i]);
41     printf("\n");
42 }

43 int main() {
44     /* Deklaracije potrebnih promenljivih. */
45     int n, n1, n2, i;
46     int a[MAKS], parni[MAKS], neparni[MAKS];

47     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
48     printf("Unesite broj elemenata niza: ");
49     scanf("%d", &n);
50     if (n < 0 || n > MAKSS) {
51         printf("Greska: neispravan unos.\n");
52         exit(EXIT_FAILURE);
53     }

54     /* Ucitavanje elemenata niza. */
55     printf("Unesite elemente niza: ");
56     for (i = 0; i < n; i++)
57         scanf("%d", &a[i]);

58     /* Popunjavanje rezultujucih nizova odgovarajucim
59     vrednostima. */
60     par_nepar(a, n, parni, &n1, neparni, &n2);

61     /* Ispis niza parni[] koji ima n1 elemenata. */
62 }
```

```
69     printf("Niz parnih brojeva: ");
70     ispisi(parni, n1);

71     /* Ispis niza neparnih koji ima n2 elemenata. */
72     printf("Niz neparnih brojeva: ");
73     ispisi(neparni, n2);

74     exit(EXIT_SUCCESS);
75 }
```

Rešenje 2.3.8

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija racuna najmanji i najveci element niza a duzine n. */
7 void min_maks(float a[], int n, float *najmanji, float *najveci) {
8     int i;
9
10    /* Vrednosti minimuma i maksimuma se inicijalizuju na vrednost
11       prvog clana niza. */
12    *najmanji = a[0];
13    *najveci = a[0];
14
15    /* U petlji se prolazi kroz ostale clanove niza i po potrebi se
16       vrsti azuriranje najmanje i najvece vrednosti. */
17    for (i = 1; i < n; i++) {
18        if (a[i] > *najveci)
19            *najveci = a[i];
20
21        if (a[i] < *najmanji)
22            *najmanji = a[i];
23    }
24
25    /* Na kraju petlje, na adresama najmanji i najveci se nalaze
26       trazene vrednosti. */
27 }
28
29 int main() {
30     /* Deklaracija potrebnih promenljivih. */
31     int i, n;
32     float a[MAKS], min, maks;
33
34     /* Ucitavanje dimenzije niza i provera ispravnosti ulaza. */
35     printf("Unesite broj elemenata niza: ");
36     scanf("%d", &n);
37     if (n < 0 || n > MAKS) {
38         printf("Greska: neispravan unos.\n");
39         exit(EXIT_FAILURE);
40     }
41 }
```

```
41     }
42
43     /* Ucitavanje elemenata niza. */
44     printf("Unesite elemente niza:\n");
45     for (i = 0; i < n; i++)
46         scanf("%f", &a[i]);
47
48     /* Racunanje vrednosti najmanjeg i najveceg elementa. */
49     min_maks(a, n, &min, &maks);
50
51     /* Ispis rezultata. */
52     printf("Najmanji: %.3f\n", min);
53     printf("Najveci: %.3f\n", maks);
54
55     exit(EXIT_SUCCESS);
56 }
```

2.5 Niske

Zadatak 2.5.1 Napisati funkciju void konvertuj(char s[]) koja menja nisku s tako što mala slova zamenjuje odgovarajućim velikim slovima, a velika slova zamenjuje odgovarajućim malim slovima. Napisati program koji učitava nisku maksimalne dužine 10 karaktera i ispisuje konvertovanu nisku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: BeoGrad  
Konvertovana niska: bEOgRAD
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: A+B+C  
Konvertovana niska: a+b+c
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: 12345  
Konvertovana niska: 12345
```

[Rešenje 2.5.1]

Zadatak 2.5.2 Napisati funkciju void ubaci_zvezdice(char s[]) koja menja nisku s tako što u njoj svaki drugi karakter zamenjuje zvezdicom. Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje izmenjenu nisku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: *a*b*c*  
Izmenjena niska: *****
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: zimA  
Izmenjena niska: z*m*
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: 123ab*c789  
Izmenjena niska: 1*3*b*7*9
```

[Rešenje 2.5.2]

Zadatak 2.5.3 Napisati program koji vrši poređenje niski. Napisati funkcije:

- int jednake(char s1[], char s2[]) koja vraća jedinicu ako su s_1 i s_2 jednake niske, a nulu inače.
- void u_velika_slova(char s[]) koja pretvara sva slova niske s u velika slova, a ostale karaktere ne menja.

Program učitava dve reči maksimalne dužine 20 karaktera i ispituje da li su unete reči jednake. Pri poređenju treba zanemariti razliku između malih i velikih slova.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite niske:  
isPit2010  
IsPiT2010  
Niske su jednake.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite niske:  
Prog1  
prog2  
Niske nisu jednake.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite niske:  
jun  
JUNSKI  
Niske nisu jednake.
```

[Rešenje 2.5.3]

Zadatak 2.5.4 Napisati program koji proverava da li se uneta niska završava samoglasnikom. Napisati funkcije:

- (a) `int samoglasnik(char c)` koja ispituje da li je karakter c samoglasnik i vraća 1 ako jeste ili 0 ako nije.
- (b) `int samoglasnik_na_kraju(char s[])` koja ispituje da li se niska s završava samoglasnikom.

Program učitava reč maksimalne dužine 20 karaktera i ispisuje da li se reč završava samoglasnikom ili ne.

Primer 1

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku: kestenje  
||| Niska se zavrsava samoglasnikom.
```

Primer 2

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku: vетар  
||| Niska se ne zavrsava samoglasnikom.
```

Primer 3

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku: OLUJA  
||| Niska se zavrsava samoglasnikom.
```

Primer 4

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku: Programiranje1  
||| Niska se ne zavrsava samoglasnikom.
```

[Rešenje 2.5.4]

Zadatak 2.5.5 Napisati funkciju `int sadrzi_veliko(char s[])` koja proverava da li niska s sadrži veliko slovo. Napisati program koji za učitanu nisku maksimalne dužine 20 karaktera proverava da li sadrži veliko slovo i ispisuje odgovarajuću poruku.

Primer 1

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku:  
||| naocare  
||| Ne sadrzi veliko slovo.
```

Primer 2

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku:  
||| DiopTrija0.75  
||| Sadrzi veliko slovo.
```

Primer 3

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku:  
||| 21.06.2017.  
||| Ne sadrzi veliko slovo.
```

[Rešenje 2.5.5]

Zadatak 2.5.6 Napisati program koji za učitanu nisku s i karakter c ispituje da li se karakter c pojavljuje u niski s . Ako je to slučaj, program treba da ispiše indeks prvog pojavljivanja karaktera c u niski s , a u suprotnom -1. Pretpostaviti da niska može da ima najviše 20 karaktera.

Primer 1

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku: bazen  
||| Unesite karakter: z  
||| Pozicija: 2
```

Primer 2

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku: lezaljka  
||| Unesite karakter: a  
||| Pozicija: 3
```

Primer 3

```
||| INTERAKCIJA SA PROGRAMOM:  
||| Unesite nisku: limunada  
||| Unesite karakter: b  
||| Pozicija: -1
```

[Rešenje 2.5.6]

2 Napredni tipovi podataka

Zadatak 2.5.7 Napisati funkciju `int podniska(char s[], char t[])` koja proverava da li je niska `t` uzastopna podniska niske `s`. Napisati program koji učitava dve niske maksimalne dužine 10 karaktera i ispisuje da li je druga niska podniska prve.

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 2.5.7]

Zadatak 2.5.8 Napisati funkciju `void skrati(char s[])` koja uklanja beline sa kraja date niske. Napisati program koji učitava liniju maksimalne dužine 100 karaktera i ispisuje učitanu i izmenjenu nisku između zvezdica.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku:  
rep belina  
Ucitana niska:  
*rep belina *  
Izmenjena niska:  
*rep belina*
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku:  
tri tabulatora na kraju  
Ucitana niska:  
*tri tabulatora na kraju *  
Izmenjena niska:  
*tri tabulatora na kraju*
```

[Rešenje 2.5.8]

Zadatak 2.5.9 Napisati funkciju `void ukloni_slova(char s[])` koja iz niske `s` uklanja sva mala i sva velika slova. Napisati program koji za učitanu nisku maksimalne dužine 20 karaktera ispisuje odgovarajuću izmenjenu nisku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: a1b2c3def  
Rezultat: 123
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: 1+2=3  
Rezultat: 1+2=3
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: malavelika  
Rezultat:
```

[Rešenje 2.5.9]

Zadatak 2.5.10 Napisati funkciju `void ukloni(char *s)` koja iz niske uklanja sva slova iza kojih sledi slovo koje je u engleskoj abecedi nakon njih, pri čemu se veličina slova zanemaruje. Pravilo se ne primenjuje na nisku dobijenu uklanjanjem. Napisati program koji učitava liniju teksta koja ima najviše 100

karaktera i ispisuje liniju koja se dobije nakon uklanjanja pomenutih karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
Zdravo svima!
Izmenjena niska:
Zrvo vma!
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
Danas je 10 stepeni.
Izmenjena niska:
Dns j 10 tpni.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku:
Ima vetra, kise i hladnoce.
Izmenjena niska:
ma vtra, kse i loe.
```

[Rešenje 2.5.10]

Zadatak 2.5.11 Napisati program koji učitava nisku s maksimalne dužine 30 karaktera i formira nisku t trostrukim nadovezivanjem niske s .

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: dan
Rezultujuca niska:
dandandan
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 3sesira
Rezultujuca niska:
3sesira3sesira3sesira
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: a-b=5
Rezultujuca niska:
a-b=5a-b=5a-b=5
```

[Rešenje 2.5.11]

Zadatak 2.5.12 Napisati program koji za unetu reč maksimalne dužine 20 karaktera i pozitivan broj n manji od 10, formira rezultujuću reč tako što unetu reč kopira n puta pri čemu se između svaka dva kopiranja umeće crtica. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: ana
Unesite broj n: 4
Rezultujuca niska:
ana-ana-ana-ana
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 123
Unesite broj n: 1
Rezultujuca niska:
123
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: x*y
Unesite broj n: 3
Rezultujuca niska:
x*y-x*y-x*y
```

[Rešenje 2.5.12]

Zadatak 2.5.13 Napisati funkciju `void kopiraj_n(char t[], char s[], int n)` koja kopira najviše n karaktera niske s u nisku t . Napisati program koji testira rad napisane funkcije. Prepostaviti da je maksimalna dužina niske s 20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: petar
Unesite broj n: 3
Rezultujuca niska: pet
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite nisku: gromobran
Unesite broj n: 4
Rezultujuca niska: grom
```

Primer 3

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

[Rešenje 2.5.13]

2 Napredni tipovi podataka

Zadatak 2.5.14 Napisati funkciju void dupliranje(char t[], char s[]) koja na osnovu niske *s* formira nisku *t* tako što duplira svaki karakter niske *s*. Napisati program koji učitava nisku maksimalne dužine 20 karaktera i ispisuje nisku koja se dobije nakon dupliranja karaktera.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: zima
Rezultujuća niska: zziimmaa

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: C++
Rezultujuća niska: CC++++

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: C
Rezultujuća niska: CC

[Rešenje 2.5.14]

Zadatak 2.5.15 Napisati program koji učitava nisku cifara sa eventualnim vodećim znakom i pretvara je u ceo broj. NAPOMENA: Prepostaviti da je unos ispravan.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: -1238
Rezultat: -1238

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 73
Rezultat: 73

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: +1
Rezultat: 1

[Rešenje 2.5.15]

Zadatak 2.5.16 Napisati program koji učitava ceo broj, pretvara ga u nisku i ispisuje dobijenu nisku.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj: -6543
Rezultat: -6543

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj: 84
Rezultat: 84

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite ceo broj: 5
Rezultat: 5

[Rešenje 2.5.16]

Zadatak 2.5.17 Napisati funkciju int heksadekadni_broj(char s[]) koja proverava da li je niskom *s* zadat korektan heksadekadni broj. Funkcija treba da vrati vrednost 1 ukoliko je uslov ispunjen, odnosno 0 ako nije. Napisati program koji za učitanu nisku maksimalne dužine 7 karaktera ispisuje da li je korektan heksadekadni broj. UPUTSTVO: Heksadekadni broj je korektno zadat ako počinje prefiksom 0x ili 0X i ako sadrži samo cifre i mala ili velika slova A, B, C, D, E i F.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 0x12EF
Korekstan heksadekadni broj.

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 0X22af
Korekstan heksadekadni broj.

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: 0xEra9
Nekorekstan heksadekadni broj.

[Rešenje 2.5.17]

Zadatak 2.5.18 Napisati funkciju `int dekadna_vrednost(char s[])` koja izračunava dekadnu vrednost heksadekadnog broja zadatog niskom *s*. Napisati program koji za učitanu nisku maksimalne dužine 7 karaktera ispisuje odgovarajuću dekadnu vrednost. Pretpostaviti da je uneta niska korektan heksadekadni broj.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: 0Xff2  
Rezultat: 4082
```

Primer 3

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

[Rešenje 2.5.18]

Zadatak 2.5.19 Napisati funkciju `int ucitaj_liniju(char s[], int n)` koja učitava liniju maksimalne dužine *n* u nisku *s* i vraća dužinu učitane linije. Napisati program koji učitava linije do kraja ulaza i ispisuje najdužu liniju i njenu dužinu. Ukoliko ima više linija maksimalne dužine, ispisati prvu. Pretpostaviti da svaka linija sadrži najviše 80 karaktera. NAPOMENA: *Linija može da sadrži blanko znakove, ali ne može sadržati znak za novi red ili EOF.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
Dobar dan!  
Kako ste, sta ima novo?  
Ja sam dobro.  
Najduza linija:  
Kako ste, sta ima novo?  
Duzina: 23
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
Pruva linija  
Druga linija  
Treca linija  
Najduza linija:  
Druga linija  
Duzina: 12
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite tekst:  
Danas je lep dan.  
Najduza linija:  
Danas je lep dan.  
Duzina: 17
```

[Rešenje 2.5.19]

* **Zadatak 2.5.20** Napisati funkcije za rad sa rečenicama:

- int `procitaj_recenicu(char s[], int n)` koja učitava rečenicu sa ulaza i smešta je u nisku *s*. Funkcija vraća dužinu učitane rečenice. Učitavanje se završava nakon učitanog karaktera `.`, nakon *n* učitanih karaktera ili ako se dođe do kraja ulaza.
- void `prebroj(char s[], int *broj_malih, int *broj_velikih)` koja prebrojava mala i velika slova u niski *s*.

2 Napredni tipovi podataka

Napisati program koji učitava rečenice do kraja ulaza i ispisuje onu rečenicu kod koje je apsolutna razlika broja malih i velikih slova najveća. Pri učitavanju rečenica zanemariti sve beline koje se nalaze između dve rečenice. Prepostaviti da jedna rečenica sadrži najviše 80 karaktera.

Primer 1

INTERAKCIJA SA PROGRAMOM:

Unesite tekst:

U ovom poglavlju se govori o niskama. Niske su nizovi karaktera ciji je poslednji element terminalna nula.

U ovom zadatku je potrebno učitati rečenice. Svaka rečenica pocinje sa bilo kojim karakterom koji nije belina. Na kraju rečenice se nalazi tacka.

Rezultujuća rečenica:

Niske su nizovi karaktera ciji je poslednji element terminalna nula.

[Rešenje 2.5.20]

Zadatak 2.5.21 Napisati funkciju `char* strchr_klon(char s[], char c)` koja vraća pokazivač na prvo pojavljivanje karaktera `c` u niski `s` ili `NULL` ukoliko se karakter `c` ne pojavljuje u niski `s`.¹ Napisati program koji za učitanu nisku maksimalne dužine 20 karaktera i karakter `c` ispisuje indeks prvog pojavljivanja karaktera `c` u okviru učitane niske ili `-1` ukoliko učitana niska ne sadrži uneti karakter.

Primer 1

INTERAKCIJA SA PROGRAMOM:

Unesite nisku s: programiranje
Unesite karakter c: a
Pozicija: 5

Primer 2

INTERAKCIJA SA PROGRAMOM:

Unesite nisku s: 123456789
Unesite karakter c: y
Pozicija: -1

Primer 3

INTERAKCIJA SA PROGRAMOM:

Unesite nisku s: letov2017
Unesite karakter c: 0
Pozicija: 5

Primer 4

INTERAKCIJA SA PROGRAMOM:

Unesite nisku s: jedrilica
Unesite karakter c: I
Pozicija: -1

[Rešenje 2.5.21]

Zadatak 2.5.22 Napisati funkciju `int strspn_klon(char t[], char s[])` koja izračunava dužinu prefiksa niske `t` sastavljenog od karaktera niske `s`. Napisati program koji za učitane dve niske maksimalne dužine 20 karaktera ispisuje rezultat poziva napisane funkcije.

¹Funkcija `strchr_klon` odgovara funkciji `strchr` čija se deklaracija nalazi u zaglavlju `string.h`. Slično važi i za ostale `klon` funkcije iz narednih zadataka.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku t: program  
Unesite nisku s: pero  
Rezultat: 3
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku t: Barselona  
Unesite nisku s: Brazil  
Rezultat: 3
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku t: 24.10.2017.  
Unesite nisku s: 0123456789  
Rezultat: 2
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku t: 12345  
Unesite nisku s: 9876543210  
Rezultat: 5
```

[Rešenje 2.5.22]

Zadatak 2.5.23 Napisati funkciju `int strcspn_klon(char t[], char s[])` koja izračunava dužinu prefiksa niske *t* sastavljenog isključivo od karaktera koji se ne nalaze u niski *s*. Napisati program koji testira ovu funkciju za dve unete niske maksimalne dužine 100 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku t:  
programiranje  
Unesite nisku s:  
pero  
Rezultat: 0
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku t:  
programiranje  
Unesite nisku s:  
analiza  
Rezultat: 5
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku t:  
programiranje  
Unesite nisku s:  
1.10.  
Rezultat: 13
```

[Rešenje 2.5.23]

Zadatak 2.5.24 Napisati funkciju `char* strstr_klon(char s[], char t[])` koja vraća pokazivač na prvo pojavljivanje niske *t* u niski *s* ili *NULL* ukoliko se niska *t* ne pojavljuje u niski *s*. Napisati program koji testira napisanu funkciju tako što učitava pet linija i ispisuje redne brojeve svih linija koje sadrže nisku *program*. Ukoliko ne postoji linija sa niskom *program*, ispisati odgovarajuću poruku. Prepostaviti da je svaka linija maksimalne dužine 100 karaktera kao i da se linije numerišu od broja 1.

2 Napredni tipovi podataka

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
tv program
c prog. jezik
c++ programskih jezik
Programski odbor
program
Rezultat: 1 3 5

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
Programske paradigme
su predmet na
trecoj godini
programerskih
smerova.
Rezultat: 4

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite pet linija:
U narednim
linijama
necemo navoditi
nisku koja se
trazi.
Nijedna linija ne sadrzi
nisku program.

[Rešenje 2.5.24]

Zadatak 2.5.25 Napisati funkciju `int strcmp_klon(char s[], char t[])` koja vraća 0 ako su niske *s* i *t* jednake, neku pozitivnu vrednost ako je *s* leksikografski iza *t*, a neku negativnu vrednost inače. Napisati program koji učitava dve niske maksimalne dužine 20 karaktera i ako su različite, ispisuje učitane niske u rastućem leksikografskom poretku, a ako su jednake, ispisuje samo jednu nisku.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: Beograd
Unesite nisku t: Amsterdam
Rezultat:
Amsterdam
Beograd

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: Beograd
Unesite nisku t: Beograd
Rezultat:
Beograd

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite nisku s: radnik
Unesite nisku t: radnica
Rezultat:
radnica
radnik

[Rešenje 2.5.25]

Zadatak 2.5.26 Napisati funkciju `void obrni(char s[])` koja obrće nisku *s*. Napisati program koji obrće učitanu nisku maksimalne dužine 20 karaktera i ispisuje obrnutu nisku.

Primer 1

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: kisobran
Rezultat: narbosik

Primer 2

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: Aleksandar
Rezultat: radnaskela

Primer 3

INTERAKCIJA SA PROGRAMOM:
Unesite nisku: kajak
Rezultat: kajak

[Rešenje 2.5.26]

Zadatak 2.5.27 Napisati funkciju void rotiraj(char s[], int k) koja rotira nisku s za k mesta uлево. Napisati program koji учиства nisku maksimalne dužine 20 karaktera i nenegativan ceo broj k i ispisuje rotiranu nisku. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku i broj k:  
sveska 2  
Rezultat: eskasv
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku i broj k:  
olovka 6  
Rezultat: olovka
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku i broj k:  
rezac 8  
Rezultat: acrez
```

[Rešenje 2.5.27]

Zadatak 2.5.28 Napisati program koji šifruje unetu nisku tako što svako slovo zamenjuje sledećim slovom engleske abecede (slova 'z' i 'Z' zamenjuje, redom, sa 'a' i 'A'), a ostale karaktere ostavlja nepromjenjene. Ispisati nisku dobijenu na ovaj način. Pretpostaviti da uneta niska nije duža od 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: bundeva  
Rezultat: cvoefwb
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: zimzelen  
Rezultat: ajnafmfo
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: Oktobar17  
Rezultat: Plupcbs17
```

[Rešenje 2.5.28]

Zadatak 2.5.29 Napisati funkciju void sifruj(char rec[], char sifra[]) koja na osnovu date reči formira šifru tako što se svako slovo u reči zameni sa naredna tri slova engleske abecede (nakon slova 'z' tj. 'Z' sledi slovo 'a' tj. 'A'). Napisati program koji testira napisanu funkciju za reč maksimalne dužine 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: tamo  
Rezultat: uvwbcnoppqr
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: Zec  
Rezultat: ABCfghdef
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: a+b=c  
Rezultat: bcd+cde=def
```

[Rešenje 2.5.29]

2 Napredni tipovi podataka

Zadatak 2.5.30 Napisati funkciju void formiraj(char s1[], char s2[], char c1, char c2) koja na osnovu niske s_1 formira nisku s_2 udvajanjem svih karaktera c_1 u niski s_1 i izbacivanjem svih karaktera c_2 iz niske s_1 , dok ostali karakteri ostaju nepromenjeni. Napisati program koji testira ovu funkciju za unetu nisku i dva uneta karaktera. Prepostaviti da uneta niska nije duža od 20 karaktera.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: flomaster  
Unesite prvi karakter: s  
Unesite drugi karakter: m  
Rezultat: floasster
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: bojica  
Unesite prvi karakter: b  
Unesite drugi karakter: a  
Rezultat: bbojic
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku: patentara  
Unesite prvi karakter: t  
Unesite drugi karakter: a  
Rezultat: pitenttr
```

[Rešenje 2.5.30]

* **Zadatak 2.5.31** Napisati program za rad sa brojevima zapisanim u različitim brojevnim sistemima.

- Napisati funkciju unsigned int u_dekadni_sistem(char broj[], unsigned int osnova) koja određuje dekadnu vrednost zapisa datog neoznačenog broja $broj$ u datoju osnovi $osnova$.
- Napisati funkciju void iz_dekadnog_sistema(unsigned int broj, unsigned int osnova, char rezultat[]) koja datu dekadnu vrednost $broj$ zapisuje u datoju osnovi $osnova$ i smešta rezultat u nisku $rezultat$. Prepostaviti da je $0 < osnova \leq 16$.

Napisati program koji učitava broj n koji se zadaje kao niska cifara i osnove o_1 i o_2 i ispisuje dekadnu vrednost broja n u osnovi o_1 , kao i zapis tako dobijene dekadne vrednosti u osnovi o_2 . Prepostaviti da je maksimalna dužina zapisa broj 20 karaktera i da će svi brojevi biti ispravno zadati tj. u opsegu tipa unsigned.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite n, o1 i o2: 10101011 2 16  
Dekadna vrednost broja 10101011: 171  
Vrednost broja 171 u osnovi 16: AB
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite n, o1 i o2: 1067 8 3  
Dekadna vrednost broja 1067: 567  
Zapis broja 567 u osnovi 3: 210000
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite n, o1 i o2: 1010111001010 2 3  
Dekadna vrednost broja 1010111001010: 5578  
Zapis broja 5578 u osnovi 3: 21122121
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite n, o1 i o2: 111 3 5  
Dekadna vrednost broja 111: 13  
Zapis broja 13 u osnovi 5: 23
```

[Rešenje 2.5.31]

2.6 Rešenja

Rešenje 2.5.1

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>

5 /* Poslednji karakter svake niske je terminirajuća nula '\0',
6    specijalni karakter čiji je ASCII kod 0.
7
8    Ukoliko je pretpostavka da niska sadrži najviše 10 karaktera,
9    neophodno je deklarisati niz od 11 karaktera, pri čemu se
10   dodatni karakter izdvaja za terminirajuću nulu. */
11 #define MAKS_NISKA 11

13 /* Funkcija vrši konverziju svakog malog slova niske u odgovarajuće
14   veliko slovo i obrnuto. Ostali karakteri ostaju nepromjenjeni. */
15 void konvertuj(char s[]) {
16     int i;
17
18     /* Prolazi se kroz nisku, karakter po karakter, sve dok se ne
19       dodje do terminirajuće nule koja sluzi kao oznaka kraja niske.
20       */
21     for (i = 0; s[i] != '\0'; i++) {
22         /* Svako malo slovo se pretvara u veliko i obrnuto. */
23         if (islower(s[i]))
24             s[i] = toupper(s[i]);
25         else if (isupper(s[i]))
26             s[i] = tolower(s[i]);
27     }
28
29     /* II nacin: Uslov u petlji može krace da se zapise sa s[i] jer
30       ASCII kod terminirajuće nule ima vrednost 0.
31     for (i = 0; s[i]; i++) {
32         if (islower(s[i]))
33             s[i] = toupper(s[i]);
34         else if (isupper(s[i]))
35             s[i] = tolower(s[i]);
36     } */
37 }
38
39 int main() {
40     /* Deklaracija potrebne promenljive. */
41     char s[MAKS_NISKA];
42
43     /* Za razliku od nizova koji se ucitavaju i stampaju element po
44       element, niske se mogu ucitati i odstampati pomocu jedne
45       scanf/printf naredbe koriscenjem specifikatora %s. */
46
47     printf("Unesite nisku: ");
48     scanf("%s", s);

```

2 Napredni tipovi podataka

```
47     /* Izmena niske. */
48     konvertuj(s);
49
50     /* Ispis rezultata. */
51     printf("Konvertovana niska: %s\n", s);
52
53     exit(EXIT_SUCCESS);
54 }
```

Rešenje 2.5.2

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS_NISKA 21
5
6 /* Funkcija ubacuje zvezdice na svako drugo mesto niske s. */
7 void ubaci_zvezdice(char s[]) {
8     int i;
9
10    for (i = 0; s[i] != '\0' && s[i + 1] != '\0'; i += 2)
11        s[i + 1] = '*';
12 }
13
14 int main() {
15     /* Deklaracija potrebne promenljive. */
16     char s[MAKS_NISKA];
17
18     /* Ucitavanje niske. */
19     printf("Unesite nisku: ");
20     scanf("%s", s);
21
22     /* Izmena niske. */
23     ubaci_zvezdice(s);
24
25     /* Ispis rezultata. */
26     printf("Izmenjena niska: %s\n", s);
27
28     exit(EXIT_SUCCESS);
29 }
```

Rešenje 2.5.3

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_NISKA 21
```

```

7  /* Funkcija pretvara sva slova niske s u velika slova. */
8  void u_velika_slova(char s[]) {
9      int i;
10     for (i = 0; s[i]; i++)
11         s[i] = toupper(s[i]);
12 }
13
14 /* Funkcija vraca 1 ako su niske s1 i s2 jednake, a nulu inace. */
15 int jednake(char s1[], char s2[]) {
16     int i;
17
18     /* Prolazi se kroz obe niske dok god ima neobradjenih karaktera u
19      bilo kojoj od njih. Ukoliko se naidje na karaktere koji su
20      razliciti, kao povratna vrednost se vraca 0 jer u tom slucaju
21      niske nisu jednake. */
22     for (i = 0; s1[i] || s2[i]; i++)
23         if (s1[i] != s2[i])
24             return 0;
25
26     /* Ako se doslo do kraja petlje znaci da su se svi karakteri
27      poklopili, a da se pri tom doslo do kraja obe niske, tako da
28      se kao povratna vrednost funkcije vraca 1 jer su niske s1 i s2
29      jednake. */
30     return 1;
31 }
32
33 int main() {
34     /* Deklaracija potrebnih promenljivih. */
35     char s1[MAKS_NISKA], s2[MAKS_NISKA];
36
37     /* Ucitavanje niski s1 i s2. */
38     printf("Unesite niske:\n");
39     scanf("%s%s", s1, s2);
40
41     /* Kako bi se pri poredjenju zanemarila razlika izmedju malih i
42      velikih slova, sva slova obe niske se pretvaraju u velika. */
43     u_velika_slova(s1);
44     u_velika_slova(s2);
45
46     /* Ispis rezultata. */
47     if (jednake(s1, s2))
48         printf("Niske su jednake.\n");
49     else
50         printf("Niske nisu jednake.\n");
51
52     exit(EXIT_FAILURE);
53 }

```

Rešenje 2.5.4

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4 #include <string.h>
5
6 #define MAKS_NISKA 21
7
8 /* Funkcija proverava da li je karakter c samoglasnik. */
9 int samoglasnik(char c) {
10     /* Karakter se pretvara u veliko slovo kako bi se izbegle posebne
11        provere za mala i velika slova. */
12     c = toupper(c);
13
14     /* Samoglasnici su slova a, e, i, o i u */
15     if (c == 'A' || c == 'E' || c == 'I' || c == 'O' || c == 'U')
16         return 1;
17
18     return 0;
19 }
20
21 /* Funkcija proverava da li se niska s zavrsava samoglasnikom. */
22 int samoglasnik_na_kraju(char s[]) {
23     /* Funkcija strlen racuna duzinu date niske. Njena deklaracija se
24        nalazi u zaglavlju string.h. */
25     int duzina = strlen(s);
26
27     /* Ako je niska prazna, ne zavrsava se samoglasnikom. */
28     if (duzina == 0)
29         return 0;
30
31     /* Provera da li je poslednji karakter niske samoglasnik. */
32     return samoglasnik(s[duzina - 1]);
33 }
34
35 int main() {
36     /* Deklaracija potrebne promenljive. */
37     char s[MAKS_NISKA];
38
39     /* Ucitavanje niske. */
40     printf("Unesite nisku: ");
41     scanf("%s", s);
42
43     /* Ispis rezultata. */
44     if (samoglasnik_na_kraju(s))
45         printf("Niska se zavrsava samoglasnikom.\n");
46     else
47         printf("Niska se ne zavrsava samoglasnikom.\n");
48
49     exit(EXIT_SUCCESS);
50 }
```

Rešenje 2.5.5 Pogledajte zadatak 2.5.1.

Rešenje 2.5.6

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS_NISKA 21
5
6 /* Funkcija vraca indeks prvog pojavljivanja karaktera c u okviru
7    niske s. Ukoliko se ne pojavljuje, funkcija vraca -1. */
8 int pozicija(char s[], char c) {
9     int i;
10
11    for (i = 0; s[i]; i++)
12        if (s[i] == c)
13            return i;
14
15    return -1;
16}
17
18 int main() {
19     /* Deklaracije potrebnih promenljivih. */
20     char s[MAKS_NISKA];
21     char c;
22
23     /* Ucitavanje niske i karaktera. */
24     printf("Unesite nisku: ");
25     scanf("%s", s);
26     getchar();
27     printf("Unesite karakter: ");
28     c = getchar();
29
30     /* I nacin: */
31     printf("Pozicija: %d\n", pozicija(s, c));
32
33     /* II nacin: Funkcija strchr(s,c) je funkcija koja vraca adresu
34        prvog pojavljivanja karaktera c u niski s, ako se c pojavljuje
35        u s, a NULL inace.
36
37        Vrednost promenljive s je zapravo vrednost adrese prvog
38        karaktera niske s.
39
40        Ako treba da se ispise indeks prvog pojavljivanja, to moze da
41        se uradi tako sto se od adrese koji je vratila funkcija
42        strchr oduzme adresu prvog karaktera.
43
44        Na primer:
45        s = "koliba" ==> s je adresa karaktera 'k'
46        p = strchr(s, 'l') ==> p je adresa karaktera 'l'
47        |k|o|l|i|b|a|
48        ^      ^

```

```
    |    |
    s    p
50   Izraz p-s ima vrednost 2 (jer je rastojanje izmedju ove dve
52   adrese 2).
53   Tip promenljive p je char* jer predstavlja adresu jednog
54   karaktera.

55   char *p = strchr(s, c);
56   if (p != NULL)
57       printf("Pozicija: %d\n", p - s);
58   else
59       printf("-1\n"); /*

60   exit(EXIT_SUCCESS);
61 }
```

Rešenje 2.5.7

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

4 #define MAKS_NISKA 11

6 /* Funkcija proverava da li je niska t podniska niske s. */
int podniska(char s[], char t[]) {
8     int i, j;

10    /* Spoljasnja petlja ide redom po niski s. */
11    for (i = 0; s[i] != '\0'; i++) {
12        /* Unutrasnja petlja ide redom po niski t pomocu brojaca j i
13           proverava da li se cela niska t poklapa sa delom niske s
14           koji pocinje na poziciji i.

16           Cim se naidje na situaciju da se karakteri ne poklapaju,
17           izlazi se iz unutrasnje petlje. */
18        for (j = 0; t[j] != '\0'; j++)
19            if (s[i + j] != t[j])
20                break;

22        /* Ako je unutrasnja petlja dosla do kraja niske t, to znaci
23           da su se svi karakteri iz t poklopili sa karakterima iz s i
24           t je podniska od s. */
25        if (t[j] == '\0')
26            return 1;
27    }
28    return 0;
29}

32 int main() {
33     /* Deklaracija potrebnih promenljivih. */
```

```

34     char s[MAKS_NISKA], t[MAKS_NISKA];
36
37     /* Ucitavanje niski s i t. */
38     printf("Unesite nisku s: ");
39     scanf("%s", s);
40     printf("Unesite nisku t: ");
41     scanf("%s", t);
42
43     /* Ispis rezultata. */
44     if (podniska(s, t))
45         printf("t je podniska niske s.\n");
46     else
47         printf("t nije podniska niske s.\n");
48
49     /* II nacin: Funkcija strstr(t, s) proverava da li je t podniska
50      od s i kao povratnu vrednost vraca adresu prvog pojavljenja
51      t u s ili NULL ukoliko se t ne pojavljuje u s. Deklaracija
52      ove funkcije se nalazi u zaglavlju string.h
53
54     char* p = strstr(t, s);
55     if(p == NULL)
56         printf("t nije podniska od s.\n");
57     else
58         printf("t je podniska od s.\n"); */
59
60     exit(EXIT_SUCCESS);
61 }
```

Rešenje 2.5.8

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>
5
6 #define MAKS_LINIJA 101
7
8 /* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
9   Funkcija ne smesta znak za novi red na kraj linije. */
10 void ucitaj_liniyu(char s[], int n) {
11     int i = 0, c;
12
13     /* Ucitavanje karakter po karakter dok se ne unese novi red ili
14       oznaka za kraj ulaza ili dok se ne dostigne maksimalan broj
15       karaktera. */
16     while ((c = getchar()) != '\n' && i < n - 1 && c != EOF) {
17         s[i] = c;
18         i++;
19     }
20
21     /* Maksimalan broj karaktera za liniju je n-1 jer na kraju treba
```

2 Napredni tipovi podataka

```
    ostaviti i jedno mesto za terminirajucu nulu. */
23   s[i] = '\0';
}
25
/* Funkcija uklanja beline sa kraja niske s. */
27 void skrati(char s[]) {
    int i;
    /* Vrsi se prolazak kroz nisku sa desna na levo i trazi se
       pozicija prvog karaktera koji nije belina.
31
       Funkcija isspace proverava da li je dati karakter neka od
33   belina (blanko, tabulator ili novi red) i njena deklaracija se
       nalazi u zaglavlju ctype.h. */
35   for (i = strlen(s) - 1; i >= 0; i--)
        if (!isspace(s[i]))
            break;
37
/* Nakon izlaska iz petlje, brojac i se nalazi na poziciji prvog
   karaktera sa desne strane koji nije belina. Iz tog razloga se
   na poziciju i+1 upisuje terminirajuca nula kao oznaka da se sada
   tu nalazi kraj niske. */
43   s[i + 1] = '\0';
}
45
int main() {
    /* Deklaracija potrebne promenljive. */
47   char s[MAKS_LINIJA];
49
    /* Ucitavanje cele linije sa ulaza. */
51   printf("Unesite nisku:\n");
52   ucitaj_liniju(s, MAKS_LINIJA);
53
    /* Napomena: Postoji vise nacina za ucitavanje linije sa
       standardnog ulaza koriscenjem funkcija iz standardne C
       biblioteke. Jedan od njih je koriscenjem funkcije gets:
       gets(s); Postoje razlozi zasto ova funkcija nije bezbedna za
       koriscenje i oni ce biti objasnjeni u kasnjim poglavljima. U
       poglavljju "Datoteke" ce biti predstavljeni i bezbedni nacini
       da se to uradi koriscenjem nekih drugih funkcija. */
57
    /* Ispis rezultata. */
61   printf("Ucitana niska:\n%s\n", s);
62   skrati(s);
63   printf("Izmenjena niska:\n%s\n", s);
64
65   exit(EXIT_SUCCESS);
}
```

Rešenje 2.5.9

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>
5
6 #define MAKS_LINIJA 21
7
8 /* Funkcija uklanja sva slova iz niske s. */
9 void ukloni_slova(char s[]) {
10     int i, j;
11
12     /* Prolazi se kroz nisku s karakter po karakter i vrsti se provera
13      da li trenutni karakter treba da se zadrzi. Karakter treba da
14      se zadrzi ukoliko nije ni malo ni veliko slovo.
15
16      Brojac j sluzi da pamti gde se upisuje sledeći karakter koji
17      treba da se zadrzi i svaki put kada se naidje na takav karakter,
18      on se upisuje na poziciju j, a brojac j se uvecava. */
19     for (i = 0, j = 0; s[i]; i++) {
20         if (!islower(s[i]) && !isupper(s[i])) {
21             s[j] = s[i];
22             j++;
23         }
24
25         /* Na kraju se na poziciji j upisuje i terminirajuca nula, kako bi
26          se naznacilo da se kraj niske nalazi nakon poslednjeg
27          zadrzanog karaktera. */
28         s[j] = '\0';
29     }
30
31     int main() {
32         /* Deklaracija potrebne promenljive. */
33         char s[MAKS_LINIJA];
34
35         /* Ucitavanje niske s. */
36         printf("Unesite nisku:\n");
37         scanf("%s", s);
38
39         /* Ispis rezultata. */
40         ukloni_slova(s);
41         printf("Rezultat: %s\n", s);
42
43         exit(EXIT_SUCCESS);
44     }

```

Rešenje 2.5.10

```

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

```

2 Napredni tipovi podataka

```
3 #include <ctype.h>
5
5 #define MAKS_LINIJA 101
7
7 /* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
   Funkcija ne smesta znak za novi red na kraj linije. */
9 void ucitaj_liniiju(char s[], int n) {
10    int i = 0, c;
11
12    while ((c = getchar()) != '\n' && i < n - 1 && c != EOF) {
13        s[i] = c;
14        i++;
15    }
16    s[i] = '\0';
17}
18
19 /* Pomocna funkcija koja proverava da li karakter c1 treba zadrzati
20   ako vazi da se iza njega nalazi karakter c2. */
21 int treba_zadrzati(char c1, char c2) {
22    /* Ako neki od karaktera nije slovo, c1 se ne izbacuje. */
23    if (!isalpha(c1) || !isalpha(c2))
24        return 1;
25
26    /* Oba karaktera se pretvaraju u veliko slovo kako bi se smanjio
27       broj poredjenja. */
28    c1 = toupper(c1);
29    c2 = toupper(c2);
30
31    /* c1 se zadrzava ako se c2 ne nalazi iza njega u engleskoj abecedi
32       . */
33    return c2 <= c1;
34}
35
35 /* Funkcija uklanja sva slova za koja vazi da se neposredno nakon
36   njih nalazi slovo koje je u engleskoj abecedi iza njih. */
37 void ukloni(char s[]) {
38    int i, j;
39    for (i = 0, j = 0; s[i]; i++) {
40        if (treba_zadrzati(s[i], s[i + 1])) {
41            s[j] = s[i];
42            j++;
43        }
44    }
45    s[j] = '\0';
46}
47
47 int main() {
48    /* Deklaracija potrebne promenljive. */
49    char s[MAKS_LINIJA];
50
51    /* Ucitavanje linije sa ulaza. */
52    printf("Unesite nisku:\n");
53}
```

```

55     ucitaj_linijsu(s, MAKS_LINIJA);
56
57     /* Ispis rezultata. */
58     ukloni(s);
59     printf("Izmenjena niska:\n%s\n", s);
60
61     exit(EXIT_SUCCESS);
62 }
```

Rešenje 2.5.11

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_NISKA 31
6 #define MAKS_REZULTAT 91
7
8 /* Niske se ne kopiraju naredbom dodele. Ukoliko je potrebno da
9    neka niska ima isti sadrzaj kao i neka druga niska, moze se
10   koristiti funkcija strcpy(t, s) koja kopira karaktere niske s u
11   nisku t zajedno za terminirajucom nulom. Deklaracija ove
12   funkcije se nalazi u zaglavlju string.h.
13
14 Funkcija strcpy_klon predstavlja jednu implementaciju funkcije
15 strcpy. */
16 void strcpy_klon(char kopija[], char original[]) {
17     int i;
18     for (i = 0; original[i]; i++)
19         kopija[i] = original[i];
20
21     kopija[i] = '\0';
22 }
23
24 int main() {
25     /* Deklaracija potrebnih promenljivih. */
26     char s[MAKS_NISKA], t[MAKS_REZULTAT];
27
28     /* Ucitavanje niske. */
29     printf("Unesite nisku: ");
30     scanf("%s", s);
31
32     /* Niska s se kopira u nisku t. */
33     strcpy_klon(t, s);
34
35     /* Funkcija strcat(s,t) nadovezuje karaktere niske s na kraj
36     niske t i novu nisku terminira karakterom '\0'. Deklaracija
37     ove funkcije se nalazi u zaglavlju string.h. */
38
39     /* Niska s se jos dva puta nadovezuje na nisku t. */
40     strcat(t, s);
```

2 Napredni tipovi podataka

```
    strcat(t, s);

42   /* Ispis rezultata. */
43   printf("Rezultujuca niska: %s\n", t);
44
45   exit(EXIT_SUCCESS);
46 }
```

Rešenje 2.5.12

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>

5 #define MAKS_NISKA 21
6 #define MAKS_N 10
7 /* Rezultat se dobija nadovezivanjem niske maksimalne duzine
MAKS_NISKA-1 i karaktera '-' najvise MAKS_N puta. Odavde je
maksimalna duzina rezultata: (MAKS_NISKA - 1 + 1) * MAKS_N =
MAKS_NISKA*MAKS_N. Na ovo treba dodati jos 1 karakter zbog
terminirajuce nule. */
8 #define MAKS_REZULTAT (MAKS_NISKA*MAKS_N + 1)

13 int main() {
14   /* Deklaracija potrebnih promenljivih. */
15   char s[MAKS_NISKA], t[MAKS_REZULTAT];
16   int i, n;

19   /* Ucitavanje niske. */
20   printf("Unesite nisku: ");
21   scanf("%s", s);

23   /* Ucitavanje broja ponavljanja i provera ispravnosti ulaza. */
24   printf("Unesite broj n: ");
25   scanf("%d", &n);
26   if (n <= 0 || n > MAKS_N) {
27     printf("Greska: neispravan unos.\n");
28     exit(EXIT_FAILURE);
29   }

31   /* Formiranje rezultata. Prvi karakter rezultujuce niske se
32   postavlja na terminirajucu nulu. Ovo se radi jer strcat
33   funkcione tako sto krene od pocetka niske, ide do
34   terminirajuce nule i zatim povepsi od tog mesta nadovezuje
35   nisku koja je prosledjena kao drugi argument. Na ovaj nacin
36   je obezbedjeno da ce prvi poziv funkcije strcat krenuti da
37   nadovezuje od pocetka niske t. U petlji se na t nadovezuje prvo
38   niska s, a zatim niska "-". Ovo se ponavlja n-1 puta jer nakon
39   poslednjeg nadovezivanja niske s ne treba da se nadje "-". Iz
40   tog razloga se po zavrsetku petlje vrsti jos jedno nadovezivanje
41   niske s, ali ne i niske "-". */
42 }
```

```

1 t[0] = '\0';
2 for (i = 0; i < n - 1; i++) {
3     strcat(t, s);
4     strcat(t, "-");
5 }
6 strcat(t, s);

7 /* Ispis rezultata. */
8 printf("Rezultujuca niska: %s\n", t);
9
10 exit(EXIT_SUCCESS);
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

```

Rešenje 2.5.13

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_NISKA 21
6
7 /* Funkcija kopira prvih n karaktera niske s u nisku t. */
8 void kopiraj_n(char t[], char s[], int n) {
9     int i;
10    /* Kopiranje se vrsti ili dok se ne dodje do terminirajuće nule u s
11       ili dok se ne prekopira n karaktera. */
12    for (i = 0; i < n && s[i] != '\0'; i++)
13        t[i] = s[i];
14
15    /* Na kraju rezultujuće niske se upisuje terminirajuća nula. */
16    t[i] = '\0';
17 }
18
19 int main() {
20    /* Deklaracije potrebnih promenljivih. */
21    int n;
22    char s[MAKS_NISKA], t[MAKS_NISKA];
23
24    /* Ucitavanje niske. */
25    printf("Unesite nisku: ");
26    scanf("%s", s);
27
28    /* Ucitavanje broja n i provera ispravnosti ulaza. */
29    printf("Unesite broj n: ");
30    scanf("%d", &n);
31
32    if (n < 0 || n > MAKS_NISKA - 1) {
33        printf("Greska: neispravan unos.\n");
34        exit(EXIT_FAILURE);
35    }
36
37    /* Formiranje rezultata. */
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53

```

```
1     kopiraj_n(t, s, n);
38
39    /* II nacin: Koriscenjem funkcije strncpy(t, s, n), cija se
40       deklaracija nalazi u zaglavlu string.h, kopira najvise n
41       karaktera niske s u nisku t.
42
43    strncpy(t,s,n); */
44
45    /* Ispis rezultata. */
46    printf("Rezultujuca niska: %s\n", t);
47
48    exit(EXIT_SUCCESS);
49}
```

Rešenje 2.5.14

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Duzina niske koja se ucitava, bez terminirajuće nule. */
5 #define MAKS_DUZINA 20
6
7 /* Duzine originalne i rezultujuće niske. */
8 #define MAKS_NISKA (MAKS_DUZINA + 1)
9 #define MAKS_REZULTAT (2 * MAKS_DUZINA + 1)
10
11 /* Funkcija formira nisku t od niske s dupliranjem svakog
12    karaktera. Npr. abc postaje aabbcc. */
13 void dupliranje(char t[], char s[]) {
14     int i, j;
15
16     /* Brojac i označava tekucu poziciju u niski s, a brojac j
17        označava tekucu poziciju u niski t. */
18     for (i = 0, j = 0; s[i] != '\0'; i++, j += 2) {
19         t[j] = s[i];
20         t[j + 1] = s[i];
21
22         /* Kraci nacin: t[j] = t[j + 1] = s[i]; */
23     }
24
25     /* Upisuje se terminirajuća nula na kraj rezultujuće niske. */
26     t[j] = '\0';
27 }
28
29 int main() {
30     /* Deklaracija potrebnih promenljivih. */
31     char s[MAKS_NISKA], t[MAKS_REZULTAT];
32
33     /* Ucitavanje niske. */
34     printf("Unesite nisku: ");
35     scanf("%s", s);
```

```

37  /* Formiranje niske t. */
38  dupliranje(t, s);
39
40  /* Ispis rezultata. */
41  printf("Rezultujuca niska: %s\n", t);
42
43  exit(EXIT_SUCCESS);
}

```

Rešenje 2.5.15

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_NISKA 10
6
7 /* Funkcija formiraj_broj na osnovu niske b formira ceo broj ciji
8 je to zapis.
9
10 Ako su cifre broja a, b, c i d, tada broj mozemo formirati kao:
11 a*10^3 + b*10^2 + c*10^1 + d*10^0. Medjutim, efikasnije je
12 koristiti Hornerovu sumu: 10*(10*(10*(10*0 + a)+b)+c)+d. */
13 int formiraj_broj(char b[]) {
14     int i;
15     int broj = 0, znak;
16
17     /* Odredjivanje znaka broja i pozicije prve cifre. */
18     if (b[0] == '-') {
19         znak = -1;
20         i = 1;
21     } else if (b[0] == '+') {
22         znak = 1;
23         i = 1;
24     } else {
25         i = 0;
26         znak = 1;
27     }
28
29     /* Prolazak kroz cifre broja i racunanje vrednosti broja
30     koriscenjem Hornerove sheme. Vrednost trenutne cifre se dobija
31     kada se od trenutnog karaktera (b[i]) oduzme karakter '0'.
32     Ako se naidje na karakter koji nije cifra, petlja se prekida.
33     Na primer, za b="123abc", rezultat treba da bude 123. */
34     for (; b[i] != '\0'; i++) {
35         if (isdigit(b[i]))
36             broj = broj * 10 + (b[i] - '0');
37         else
38             break;
39     }

```

2 Napredni tipovi podataka

```
41     return broj * znak;
42 }
43
44 int main() {
45     /* Deklaracija potrebne promenljive. */
46     char s[MAKS_NISKA];
47
48     /* Broj se ucitava kao niska. */
49     printf("Unesite nisku: ");
50     scanf("%s", s);
51
52     /* Ispis rezultata. */
53     printf("Rezultat: %d\n", formiraj_broj(s));
54
55     /* II nacin: Koriscenjem funkcije atoi. Deklaracija ove funkcije
56      se nalazi u zaglavlju stdlib.h.
57
58     printf("%d\n", atoi(s)); */
59
60     exit(EXIT_SUCCESS);
61 }
```

Rešenje 2.5.16

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS_NISKA 10
5
6 /* Funkcija racuna broj cifara broja n. */
7 int broj_cifara(int n) {
8     int brojac = 0;
9     do {
10         brojac++;
11         n /= 10;
12     } while (n);
13
14     return brojac;
15 }
16
17 /* Funkcija od prosledjenog broja formira nisku. */
18 void broj_u_nisku(int broj, char s[]) {
19     int n, cifra, i;
20
21     /* Promenljiva n cuva informaciju o duzini niske. Duzina niske
22      odgovara broju cifara prosledjenog broja. Ukoliko je broj
23      negativan, onda se duzina uvecava za 1 i na prvo mesto se
24      upisuje znak '-'. */
25     n = broj_cifara(broj);
26     if (broj < 0) {
```

```

27     s[0] = '-';
28     n++;
29 }
30
31 /* U nastavku se radi sa apsolutnom vrednoscu broja. */
32 broj = abs(broj);
33
34 /* Cifre broja se upisuju u nisku s sa desna na levo. */
35 s[n] = '\0';
36 i = n - 1;
37 do {
38     /* Karakter koji odgovara trenutnoj cifri se dobija izrazom '0'
39     + cifra. Na primer, '0' + 5 je '5' jer se karakter '5'
40     nalazi 5 mesta nakon karaktera '0' u ASCII tablici. */
41     cifra = broj % 10;
42     broj = broj / 10;
43     s[i] = '0' + cifra;
44     i--;
45 } while (broj);
46 }
47
48 int main() {
49     /* Deklaracije potrebnih promenljivih. */
50     int n;
51     char s[MAKS_NISKA];
52
53     /* Ucitavanje broja. */
54     printf("Unesite ceo broj: ");
55     scanf("%d", &n);
56
57     /* Formiranje niske. */
58     broj_u_nisku(n, s);
59
60     /* Ispis rezultata. */
61     printf("Rezultat: %s\n", s);
62
63     exit(EXIT_SUCCESS);
64 }
```

Rešenje 2.5.17

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_NISKA 8
6
7 /* Funkcija proverava da li je prosledjeni karakter ispravna
8  heksadekadna cifra. */
9 int heksa_cifra(char c) {
10    c = toupper(c);
```

2 Napredni tipovi podataka

```
11  /* Karakter je ispravan ako je cifra ili ako je neko od slova:
12   A, B, C, D, E ili F. */
13   return isdigit(c) || (c >= 'A' && c <= 'F');
14 }
15
16 /* Funkcija proverava da li prosledjena niska s predstavlja
17  ispravan heksadekadni broj. */
18 int heksadekadni_broj(char s[]) {
19     int i;
20
21     /* Svaki heksadekasni broj pocinje sa 0x ili 0X. */
22     if (s[0] != '0' || toupper(s[1]) != 'X')
23         return 0;
24
25     /* Za svaki karakter niske s se proverava da li predstavlja
26      ispravnu heksadekadnu cifru. Ako se naidje na karakter koji
27      ne zadovoljava taj uslov, onda se kao povratna vrednost vraca
28      nula. */
29     for (i = 2; s[i]; i++)
30         if (!heksa_cifra(s[i]))
31             return 0;
32
33     /* Ako su svi karakteri isravne heksadekadne cifre, onda je i s
34      ispravan heksadekadni broj i funkcija vraca jedinicu. */
35     return 1;
36 }
37
38 int main() {
39     /* Deklaracija potrebne promenljive. */
40     char s[MAKS_NISKA];
41
42     /* Ucitavanje niske. */
43     printf("Unesite nisku: ");
44     scanf("%s", s);
45
46     /* Ispis rezultata. */
47     if (heksadekadni_broj(s))
48         printf("Korektni heksadekadni broj.\n");
49     else
50         printf("Nekorektni heksadekadni broj.\n");
51
52     exit(EXIT_SUCCESS);
53 }
```

Rešenje 2.5.18

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>
```

```

5 #define MAKS_NISKA 8
7
9 /* Funkcija racuna dekadnu vrednost jedne heksadekadne cifre. Ako
10 je c dekadna cifra, vrednost se dobija oduzimanjem '0'. Ako je c
11 veliko slovo, vrednost se dobija oduzimanjem 'A' i dodavanjem 10
12 (npr. vrednost karaktera 'B' je  $10 + 'B' - 'A' = 11$ ). */
13 int vrednost_heksa_cifre(char c) {
14     if (isdigit(c))
15         return c - '0';
16     else
17         return 10 + toupper(c) - 'A';
18 }
19
20 /* Funkcija racuna dekadnu vrednost heksadekadnog broja. */
21 int dekadna_vrednost(char s[]) {
22     int i, tezina_pozicije = 1, rezultat = 0;
23     int n = strlen(s);
24
25     /* Vrsi se prolazak kroz nisku sa desna na levo. Heksadekadna
26     cifra najvece tezine se nalazi na poziciji n-1, a cifra najmanje
27     tezine se nalazi na poziciji 2 (jer su prva dva karaktera 0x).
28
29     U svakoj iteraciji, na rezultat se dodaje vrednost tekuce
30     cifre pomnozene vrednoscu tezine njene pozicije.
31     Na primer, za s = "0x1a8e", n=6
32     i = 5, rezultat += vrednost('e')*1 => rezultat += 11*1
33     i = 4, rezultat += vrednost('8')*16 => rezultat += 8*16
34     i = 3, rezultat += vrednost('a')*256 => rezultat += 10*256
35     i = 2, rezultat += vrednost('1')*4096 => rezultat += 1*4096 */
36     for (i = n - 1; i >= 2; i--) {
37         rezultat += tezina_pozicije * vrednost_heksa_cifre(s[i]);
38         tezina_pozicije *= 16;
39     }
40
41     /* II nacin: Koriscenjem Hornerove sheme.
42     for (i = 2; i < n; i++)
43         rezultat = rezultat * 16 + vrednost_heksa_cifre(s[i]); */
44
45     return rezultat;
46 }
47
48 int main() {
49     /* Deklaracija potrebne promenljive. */
50     char s[MAKS_NISKA];
51
52     /* Ucitavanje niske. */
53     printf("Unesite nisku: ");
54     scanf("%s", s);
55
56     /* Ispis rezultata. */
57     printf("Rezultat: %d\n", dekadna_vrednost(s));

```

2 Napredni tipovi podataka

```
57     exit(EXIT_SUCCESS);
59 }
```

Rešenje 2.5.19

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#define MAKS_LINIJA 81
/* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
   Funkcija ne smesta znak za novi red na kraj linije. */
int ucitaj_liniiju(char s[], int n) {
    int i = 0;
    int c;
    /* Ucitava se karakter po karakter dok se ne unese novi red ili
       oznaka za kraj ulaza ili dok se ne dostigne maksimalan broj
       karaktera. */
    while ((c = getchar()) != '\n' && i < n - 1 && c != EOF) {
        s[i] = c;
        i++;
    }
    /* Maksimalan broj karaktera za liniju je n-1 jer na kraju treba
       ostaviti i jedno mesto za terminirajucu nulu. */
    s[i] = '\0';
    return i;
}
int main() {
    /* Deklaracije potrebnih promenljivih. */
    char linija[MAKS_LINIJA], najduza_liniija[MAKS_LINIJA];
    int duzina_najduze = 0, duzina;
    /* U petlji se ucitavaju linije sve dok se ne unese prazna
       linija. Ukoliko se unese linija koja je duza od trenutno
       najduze, vrsti se azuriranje duzine najduze linije, kao i same
       linije. */
    printf("Unesite tekst:\n");
    while ((duzina = ucitaj_liniiju(linija, MAKS_LINIJA)) > 0)
        if (duzina_najduze < duzina) {
            duzina_najduze = duzina;
            strcpy(najduza_liniija, linija);
        }
    /* Ispis rezultata. */
    if (duzina_najduze == 0)
```

```

46     printf("Nije uneta nijedna linija.\n");
47 else
48     printf("Najduza linija:\n%s\nDuzina: %d\n", najduza_linija,
49           duzina_najduze);
50
51 exit(EXIT_SUCCESS);
52 }

```

Rešenje 2.5.20

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>
5
6 #define MAKS_RECENICA 81
7
8 /* Funkcija ucitava recenicu maksimalne duzine n. */
9 int ucitaj_recenicu(char s[], int n) {
10    int i = 0, c;
11
12    /* Ako postoje, preskaku se beline sa pocetka. Po zavrsetku ove
13       petlje u c se nalazi prvi sledeci karakter koji nije belina. */
14    do {
15        c = getchar();
16    } while (isspace(c));
17
18    /* Ako je taj karakter EOF, zavrsava se ucitavanje. */
19    if (c == EOF)
20        return 0;
21
22    /* U nisku se smesta karakter, prelazi se na sledeci karakter i
23       postupak se ponavlja sve dok se ne unese tacka, EOF ili dok se
24       ne smesti maksimalan broj karaktera koje recenica moze da
25       sadrzi. */
26    do {
27        s[i] = c;
28        i++;
29        c = getchar();
30    } while (c != '.' && i < n - 2 && c != EOF);
31
32    /* Ako je poslednji uneti karakter EOF, zavrsava se ucitavanje. */
33    if (c == EOF)
34        return 0;
35
36    /* Na kraju svake recenice stoji tacka za kojom sledi '\0'. */
37    s[i] = '.';
38    s[i + 1] = '\0';
39
40    return i + 1;
41 }

```

2 Napredni tipovi podataka

```
42  /* Funkcija prebrojava mala i velika slova. */
44 void prebroj(char s[], int *broj_malih, int *broj_velikih) {
45     int i, mala = 0, velika = 0;
46
47     for (i = 0; s[i]; i++) {
48         if (islower(s[i]))
49             mala++;
50         else if (isupper(s[i]))
51             velika++;
52     }
53
54     *broj_malih = mala;
55     *broj_velikih = velika;
56 }
57
58 int main() {
59     /* Deklaracija potrebnih promenljivih. */
60     char recenica[MAKS_RECENICA];
61     char rezultujuca_recenica[MAKS_RECENICA];
62     int najveca_razlika = -1, trenutna_razlika;
63     int mala, velika;
64     int ucitana_bar_jedna = 0;
65
66     /* U petlji se ucitavaju recenice sve dok se ne unese EOF. */
67     while (ucitaj_recenicu(recenica, MAKS_RECENICA) > 0) {
68         /* Prebrojavanje malih i velikih slova. */
69         prebroj(recenica, &mala, &velika);
70
71         /* Racunanje njihove absolutne razlike. */
72         trenutna_razlika = abs(mala - velika);
73
74         /* Ako je razlika veca od trenutno najvece, azurira se vrednost
75            najvece razlike i pamti se trenutna recenica. */
76         if (trenutna_razlika > najveca_razlika) {
77             najveca_razlika = trenutna_razlika;
78             strcpy(rezultujuca_recenica, recenica);
79         }
80
81         /* Indikator koji označava da se petlja bar jednom izvrsila,
82            tj. da korisnik nije odmah zadao EOF. */
83         ucitana_bar_jedna = 1;
84     }
85
86     /* Ispis rezultata. */
87     if (ucitana_bar_jedna)
88         printf("Rezultujuca recenica:\n%s\n", rezultujuca_recenica);
89     else
90         printf("Nije uneta nijedna recenica. ");
91
92     exit(EXIT_SUCCESS);
93 }
```

Rešenje 2.5.21

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS_NISKA 21
5
6 /* Funkcija vraca adresu prvog pojavljivanja karaktera c u niski s
7 ili NULL ukoliko se c ne pojavljuje u s.
8
9 Trazeni rezultat se moze dobiti koriscenjem funkcije strchr cija
10 se deklaracija nalazi u zaglavlju string.h. Funkcija
11 strchr_klon predstavlja jednu mogucu implementaciju ove
12 funkcije. */
13
14 char *strchr_klon(char s[], char c) {
15     int i;
16
17     /* Za svaki karakter se proverava da li je jednak karakteru c.
18     Ako se naidje na takav karakter, kao povratna vrednost
19     funkcije se vraca njegova adresa (&s[i]). */
20     for (i = 0; s[i]; i++)
21         if (s[i] == c)
22             return &s[i];
23
24     /* Ako je petlja zavrsena, znaci da nije pronadjen karakter koji
25     je jednak karakteru c pa se kao povratna vrednost funkcije
26     vraca NULL pokazivac. */
27     return NULL;
28 }
29
30 int main() {
31     /* Deklaracije potrebnih promenljivih. */
32     char s[MAKS_NISKA];
33     char c;
34
35     /* Ucitavanje niske s. */
36     printf("Unesite nisku s: ");
37     scanf("%s", s);
38
39     /* Preskace se novi red koji je unet nakon niske s i ucitava se
40     karakter c. */
41     getchar();
42     printf("Unesite karakter c: ");
43     scanf("%c", &c);
44
45     /* Racunanje i ispis rezultata. */
46     char *p = strchr_klon(s, c);
47     if (p == NULL)
48         printf("Pozicija: -1\n");
49     else
50         printf("Pozicija: %ld\n", p - s);

```

```
    exit(EXIT_SUCCESS);
52 }
```

Rešenje 2.5.22

```
#include <stdio.h>
2 #include <stdlib.h>
# include <string.h>
4
#define MAKS_NISKA 21
6
/* Funkcija racuna duzinu prefiksa niske t koji se moze zapisati
8 pomocu karaktera niske s. Na primer, t="programiranje",
s="grupacija", rezultat je 2 jer niska s sadrzi prva dva
10 karaktera niske t, ali ne i treći.
Trazeni rezultat moze se dobiti koriscenjem funkcije strspn cija
12 se deklaracija nalazi u zaglavlju string.h. Funkcija
strspn_klon predstavlja jednu mogucu implementaciju ove
14 funkcije.*/
int strspn_klon(char t[], char s[]) {
16     int i, brojac = 0;

18     /* Ide se redom po karakterima niske t i za svaki karakter se
vrsi provera da li se on nalazi u zapisu niske s. Za ovo se
20     koristi funkcija strchr. Ako se nalazi, uvecava se brojac, a
ako se ne nalazi, prekida se petlja.*/
22     for (i = 0; t[i]; i++) {
24         if (strchr(s, t[i]) != NULL)
            brojac++;
        else
            break;
26     }

28     return brojac;
30 }

32 int main() {
34     /* Deklaracija potrebnih promenljivih.*/
36     char s[MAKS_NISKA], t[MAKS_NISKA];

38     /* Ucitavanje niski.*/
39     printf("Unesite nisku t: ");
40     scanf("%s", t);
41     printf("Unesite nisku s: ");
42     scanf("%s", s);

44     /* Racunanje i ispis rezultata.*/
45     printf("Rezultat: %d\n", strspn_klon(t, s));
46
        exit(EXIT_SUCCESS);
}
```

Rešenje 2.5.23 Rešenje ovog zadatka se svodi na rešenje zadatka 2.5.22, uz razliku da se ovde prebrojavaju karakteri koji se ne nalaze u zapisu niske s.

Rešenje 2.5.24

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>

5 #define MAKS_NISKA 101

7 /* Funkcija ucitava liniju maksimalne duzine n i upisuje je u s.
   Funkcija ne smesta znak za novi red na kraj linije. */
9 void ucitaj_liniju(char s[], int n) {
    int i = 0, c;

11 while ((c = getchar()) != '\n' && i < n - 1 && c != EOF) {
13     s[i] = c;
14     i++;
15 }

17     s[i] = '\0';
18 }
19
/* Funkcija vraca pokazivac na prvo pojavljivanje niske t u okviru
21 niske s ili NULL ukoliko se t ne nalazi u s.

23 Trazeni rezultat moze se dobiti koriscenjem funkcije strstr cija
se deklaracija nalazi u zaglavlju string.h. Funkcija
25 strstr_klon predstavlja jednu mogucu implementaciju ove
funkcije. */
27 char *strstr_klon(char s[], char t[]) {
    int i, j;

29
/* Spoljasnja petlja ide redom po niski s. */
31 for (i = 0; s[i] != '\0'; i++) {
    /* Unutrasnja petlja ide redom po niski t pomocu brojaca j i
       proverava da li se cela niska t poklapa sa delom niske s
       koji pocinje na poziciji i.
       Cim se naidje na situaciju da se karakteri ne poklapaju,
       izlazi se iz unutrasnje petlje. */
37     for (j = 0; t[j] != '\0'; j++)
        if (s[i + j] != t[j])
            break;

41
/* Ako je unutrasnja petlja dosla do kraja niske t, to znaci
   da su se svi karakteri iz t poklopili sa karakterima iz s i
   t je podniska od s. Kao povratna vrednost se vraca adresa
   gde t pocinje u s. */
45     if (t[j] == '\0')
        return &s[i];
}

```

2 Napredni tipovi podataka

```
49     return NULL;
50 }
51
52 int main() {
53     /* Deklaracije potrebnih promenljivih. */
54     char linija[MAKS_NISKA];
55     int i, bar_jedna = 0;
56
57     /* Ucitavanje linija i ispis rednih brojeva linija koje sadrze
58      rec "program". */
59     printf("Unesite pet linija:\n");
60     for (i = 1; i <= 5; i++) {
61         ucitaj_liniju(linija, MAKS_NISKA);
62         if (strstr_klon(linija, "program") != NULL) {
63             if (!bar_jedna)
64                 printf("Rezultat: ");
65             printf("%d ", i);
66             bar_jedna = 1;
67         }
68         /* II nacin: Koriscenjem funkcije strstr cija se deklaracija
69          nalazi u zaglavlju string.h.
70          if(strstr(linija, "program") != NULL){
71              printf("%d ", i);
72              bar_jedna = 1;
73          }*/
74     }
75     printf("\n");
76
77     /* Ako indikator bar_jedna i dalje ima vrednost 0, znaci da nije
78      uneta nijedna linija koja sadrzi rec "program". */
79     if (!bar_jedna)
80         printf("Nijedna linija ne sadrzi nisku program.\n");
81
82     exit(EXIT_SUCCESS);
83 }
```

Rešenje 2.5.25

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

4 #define MAKS_NISKA 21

6 /* Funkcija poredi dve niske i vraca nulu ukoliko su jednake, neku
8  pozitivnu vrednost ukoliko je niska s1 leksikografski iza s2,
a neku negativnu vrednost inace.

10 Trazeni rezultat moze se dobiti koriscenjem funkcije strcmp cija
12  se deklaracija nalazi u zaglavlju string.h. Funkcija
strcmp_klon predstavlja jednu mogucu implementaciju ove
```

```

    funkcije. */
14 int strcmp_klon(char s1[], char s2[]) {
15     int i;
16
17     /* Prolazi se kroz obe niske dok god se odgovarajuci karakteri
18      poklapaju. Ako se u ovom prolasku desi da je petlja dosla do
19      kraja obe niske, onda su one jednake i kao povratna vrednost
20      funkcije se vraca 0. */
21     for (i = 0; s1[i] == s2[i]; i++)
22         if (s1[i] == '\0')
23             return 0;
24
25     /* Ako niske nisu jednake, znaci da je brojac i stao na prvom
26      mestu gde se niske s1 i s2 razlikuju. Posto funkcija treba da
27      vrati pozitivnu vrednost ako je niska s1 laksikografski iza
28      s2, a negativnu u suprotnom, ovo moze biti realizovano
29      vracanjem razlike ASCII kodova. Na primer: s1 = "pero", s2 =
30      "program" Nakon petlje, brojac i ima vrednost 1 (jer je tu
31      prva razlika). Kao povratna vrednost se vraca s1[1] - s2[1] =
32      'e' - 'r' = -13 sto kao negativna vrednost govori da se s1
33      nalazi leksikografski ispred s2. */
34     return s1[i] - s2[i];
35 }
36
37 int main() {
38     /* Deklaracije potrebnih promenljivih. */
39     char s[MAKS_NISKA], t[MAKS_NISKA];
40     int rezultat;
41
42     /* Ucitavanje niski s i t. */
43     printf("Unesite nisku s: ");
44     scanf("%s", s);
45     printf("Unesite nisku t: ");
46     scanf("%s", t);
47
48     /* Poredjenje niski i ispis rezultata. */
49     rezultat = strcmp_klon(s, t);
50
51     /* II nacin: Koriscenjem funkcije strcmp cija se deklaracija
52      nalazi u zaglavlju string.h: rezultat = strcmp(s, t); */
53
54     /* Ispis rezultata. */
55     printf("Rezultat:\n");
56     if (rezultat == 0)
57         printf("%s\n", s);
58     else if (rezultat < 0)
59         printf("%s\n%s\n", s, t);
60     else
61         printf("%s\n%s\n", t, s);
62
63     exit(EXIT_SUCCESS);
64 }

```

Rešenje 2.5.26

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_NISKA 21
6
7 /* Funkcija obrće nisku s. */
8 void obrni(char s[]) {
9     int i, j;
10    int n = strlen(s);
11    char c;
12
13    /* Brojac i ide od prvog karaktera niske s, a brojac j od
14       poslednjeg i dok god se ne sretnu, vrsti se zamena karaktera
15       koji se nalaze na njihovim pozicijama. */
16    for (i = 0, j = n - 1; i < j; i++, j--) {
17        c = s[i];
18        s[i] = s[j];
19        s[j] = c;
20    }
21}
22
23 int main() {
24     /* Deklaracija potrebne promenljive. */
25     char s[MAKS_NISKA];
26
27     /* Ucitavanje niske. */
28     printf("Unesite nisku: ");
29     scanf("%s", s);
30
31     /* Racunanje i ispis rezultata. */
32     obrni(s);
33     printf("Rezultat: %s\n", s);
34
35     exit(EXIT_SUCCESS);
36 }
```

Rešenje 2.5.27

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_NISKA 21
6
7 /* Funkcija rotira nisku za jedno mesto ulevo. */
8 void rotiraj1(char s[], int n) {
9     int i;
10    /* Pamti se prvi karakter. */
```

```

12   char prvi = s[0];
13
14   /* Svaki sledeći karakter se pomera za jedno mesto ulevo. */
15   for (i = 0; i < n - 1; i++)
16     s[i] = s[i + 1];
17
18   /* Prvi karakter se upisuje na kraj niske. */
19   s[n - 1] = prvi;
20 }
21
22 /* Funkcija rotira nisku s za k mesta ulevo. */
23 void rotiraj(char s[], int k) {
24   int i;
25   int n = strlen(s);
26
27   for (i = 0; i < k; i++)
28     rotiraj1(s, n);
29 }
30
31 int main() {
32   /* Deklaracija potrebnih promenljivih. */
33   char s[MAKS_NISKA];
34   int k;
35
36   /* Ucitavanje niske i vrednosti broja k. */
37   printf("Unesite nisku i broj k: ");
38   scanf("%s%d", s, &k);
39
40   /* Provera ispravnosti ulaza. */
41   if (k < 0) {
42     printf("Greska: neispravan unos.\n");
43     exit(EXIT_FAILURE);
44   }
45
46   /* Racunanje i ispis rezultata. */
47   rotiraj(s, k);
48   printf("Rezultat: %s\n", s);
49
50   exit(EXIT_SUCCESS);
51 }
```

Rešenje 2.5.28

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>
5
6 #define MAKS_NISKA 21
7
8 /* Funkcija svako slovo niske s menja slovom koje se u ASCII
```

2 Napredni tipovi podataka

```
9     tablici nalazi neposredno iza njega. Specijalan slučaj je slovo
10    z koje treba da se zameni slovom a. Ostali karakteri ostaju
11    nepromjenjeni. */
12    void sifruj(char s[]) {
13        int i;
14
15        for (i = 0; s[i]; i++)
16            if (isalpha(s[i])) {
17                if (s[i] == 'z')
18                    s[i] = 'a';
19                else if (s[i] == 'Z')
20                    s[i] = 'A';
21                else
22                    s[i] = s[i] + 1;
23            }
24    }
25
26    int main() {
27        /* Deklaracija potrebne promenljive. */
28        char s[MAKS_NISKA];
29
30        /* Ucitavanje niske. */
31        printf("Unesite nisku: ");
32        scanf("%s", s);
33
34        /* Racunanje i ispis rezultata. */
35        sifruj(s);
36        printf("Rezultat: %s\n", s);
37
38        exit(EXIT_SUCCESS);
39    }
```

Rešenje 2.5.29

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_DUZINA 20
6 #define MAKS_NISKA (MAKS_DUZINA + 1)
7 #define MAKS_REZULTAT (3*MAKS_DUZINA + 1)
8
9 /* Pomocna funkcija koja za prosledjeno slovo vraca slovo koje ide
10   posle njega. */
11 char sledeci(char c) {
12     if (c == 'z')
13         return 'a';
14
15     if (c == 'Z')
16         return 'A';
17 }
```

```

19     return c + 1;
}

21 /* Funkcija od niske s formira rezultujuću nisku koja se dobija na
22    sledeći način:
23    1. ako je s[i] slovo, onda se u rezultujuću nisku upisuju naredna
24       tri slova engelske abecede (kada se stigne do kraja engleske
25          abecede, ide se u
26          krug, tj. nakon slova z sledi slovo a)
27    2. ako s[i] nije slovo, s[i] se samo prepisuje u rezultat. */
28 void sifruj(char s[], char rezultat[]) {
29     int i, j;
30
31     /* Brojac i se koristi za nisku s, a brojac j za rezultujuću
32        nisku. */
33     for (i = 0, j = 0; s[i]; i++) {
34         if (isalpha(s[i])) {
35             /* Ako je s[i] slovo, onda se u rezultat upisuju 3 slova koja
36                sledi nakon njega. */
37             rezultat[j] = sledeci(s[i]);
38             rezultat[j + 1] = sledeci(rezultat[j]);
39             rezultat[j + 2] = sledeci(rezultat[j + 1]);
40             j += 3;
41         } else {
42             /* Ako s[i] nije slovo, onda se samo prepisuje u rezultat. */
43             rezultat[j] = s[i];
44             j++;
45         }
46
47         /* Na kraj rezultata se dopisuje terminirajuća nula. */
48         rezultat[j] = '\0';
49     }
50
51     int main() {
52         /* Deklaracija potrebnih promenljivih. */
53         char s[MAKS_NISKA], rezultat[MAKS_REZULTAT];
54
55         /* Ucitavanje niske. */
56         printf("Unesite nisku: ");
57         scanf("%s", s);
58
59         /* Racunanje i ispis rezultata. */
60         sifruj(s, rezultat);
61         printf("Rezultat: %s\n", rezultat);
62
63         exit(EXIT_SUCCESS);
64     }

```

Rešenje 2.5.30

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>

5 #define MAKS_DUZINA 20
6 #define MAKS_NISKA (MAKS_DUZINA + 1)
7 #define MAKS_REZULTAT (2*MAKS_DUZINA + 1)

9 /* Funkcija od niske s formira rezultujuću nisku na sledeći nacin:
10   1. Svi karakteri niske s koji su jednaki c1 se dupliraju. 2. Svi
11   karakteri niske s koji su jednaki c2 se brišu. 3. Ostali
12   karakteri se samo prepisuju. */
13 void formiraj(char s[], char rezultat[], char c1, char c2) {
14     int i, j;
15     /* Brojac i se koristi za nisku s, a brojac j za rezultujuću
16     nisku. */
17     for (i = 0, j = 0; s[i]; i++) {
18         if (s[i] == c1) {
19             /* Ako je s[i] jednako c1, duplira se u rezultatu. */
20             rezultat[j] = s[i];
21             rezultat[j + 1] = s[i];
22             j += 2;
23         } else if (s[i] != c2) {
24             /* Ako s[i] razlicito od c2, upisuje se u rezultat. */
25             rezultat[j] = s[i];
26             j++;
27         }
28     }
29     /* Na kraj rezultata se dopisuje terminirajuća nula. */
30     rezultat[j] = '\0';
31 }
32
33 int main() {
34     /* Deklaracija potrebnih promenljivih. */
35     char s[MAKS_NISKA], rezultat[MAKS_REZULTAT];
36     char c1, c2;
37
38     /* Ucitavanje niske i karaktera. */
39     printf("Unesite nisku: ");
40     scanf("%s", s);
41     getchar();
42     printf("Unesite prvi karakter: ");
43     scanf("%c", &c1);
44     getchar();
45     printf("Unesite drugi karakter: ");
46     scanf("%c", &c2);
47
48     /* Provera ispravnosti ulaza. */
49     if (c1 == c2) {
```

```

51     printf("Greska: neispravan unos.\n");
52     exit(EXIT_FAILURE);
53 }

55 /* Racunanje i ispis rezultata. */
56 formiraj(s, rezultat, c1, c2);
57 printf("Rezultat: %s\n", rezultat);

59 exit(EXIT_SUCCESS);
}

```

Rešenje 2.5.31

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>

5 #define MAKS_NISKA 20

6 /* Pomocna funkcija koja racuna dekadnu vrednost prosledjenog
7  karaktera (npr. '1' ima vrednost 1, 'C' ima vrednost 12). */
8 unsigned vrednost_cifre(char c) {
9     c = toupper(c);
10    if (isdigit(c))
11        return c - '0';
12    else
13        return c - 'A' + 10;
14}

15 /* Funkcija racuna dekadnu vrednost neoznacenog broja zapisanog u
16  datoju osnovi. */
17 unsigned int u_dekadni_sistem(char broj[], unsigned int osnova) {
18     int i, n = strlen(broj);
19     int rezultat = 0, tezina_pozicije = 1;

20     for (i = n - 1; i >= 0; i--) {
21         rezultat += vrednost_cifre(broj[i]) * tezina_pozicije;
22         tezina_pozicije *= osnova;
23     }

24     return rezultat;
25 }

26 /* Funkcija obrće nisku s. */
27 void obrni(char s[]) {
28     int i, j;
29     int n = strlen(s);
30     char c;

31     for (i = 0, j = n - 1; i < j; i++, j--) {
32         c = s[i];
33         s[i] = s[j];
34         s[j] = c;
35     }
36 }

```

2 Napredni tipovi podataka

```
        c = s[i];
40      s[i] = s[j];
41      s[j] = c;
42    }
43 }
44 /* Pomocna funkcija koja dekadnu vrednost cifre pretvara u
45  odgovarajuci karakter (npr. 12 u 'C', 5 u '5'). */
46 char ostatak_u_char(int ostatak) {
47   if (ostatak < 10)
48     return '0' + ostatak;
49   else
50     return 'A' + ostatak - 10;
51 }

52 */

53 /* Funkcija datu dekadnu vrednost broja prebacuje u broj u dator
54  osnovi. */
55 void iz_dekadnog_sistema(unsigned int broj, unsigned int osnova,
56                           char rezultat[]) {
57   int i = 0;
58   int ostatak;
59
60   do {
61     ostatak = broj % osnova;
62     broj = broj / osnova;
63     rezultat[i] = ostatak_u_char(ostatak);
64     i++;
65   } while (broj);

66   rezultat[i] = '\0';
67   obrni(rezultat);
68 }

69 int main() {
70   /* Deklaracije potrebnih promenljivih. */
71   char broj[MAKS_NISKA], broj2[MAKS_NISKA];
72   unsigned int osnova1, osnova2;
73
74   /* Ucitavanje ulaznih podataka. */
75   printf("Unesite n, o1 i o2: ");
76   scanf("%s%u%u", &broj, &osnova1, &osnova2);

77   /* Ispis rezultata. */
78   unsigned dekadna_vrednost = u_dekadni_sistem(broj, osnova1);
79   printf("Dekadna vrednost broja %s: %u\n", broj, dekadna_vrednost);

80   iz_dekadnog_sistema(dekadna_vrednost, osnova2, broj2);
81   printf("Zapis broja %u u osnovi %u: %s\n", dekadna_vrednost,
82         osnova2, broj2);

83   exit(EXIT_SUCCESS);
84 }
```

2.7 Višedimenzioni nizovi

Zadatak 2.7.1 Napisati program koji učitava i zatim ispisuje elemente učitane matrice. Sa ulaza se najpre učitavaju dva cela broja m i n , a potom i elementi matrice celih brojeva dimenzije $m \times n$. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Matrica je:
1 2 3 4
5 6 7 8
9 10 11 12
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Matrica je:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
500 3
Greska: neispravan unos.
```

[Rešenje 2.7.1]

Zadatak 2.7.2 Napisati program koji za učitanu celobrojnu matricu² dimenzije $m \times n$ izračunava i štampa na tri decimale njenu euklidsku normu. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. UPUTSTVO: *Euklidска норма матрице је квадратни корен суме квадрата свих елемената матрице.*

²Pod pojmom *učitati matricu* ili *za datu matricu* uvek se podrazumeva da se prvo unose dimenzije matrice, a potom i sama matrica.

2 Napredni tipovi podataka

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
3 4  
Unesite elemente matrice:  
1 2 3 4  
5 6 7 8  
9 10 11 12  
Euklidska norma: 25.495
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
5 3  
Unesite elemente matrice:  
1 1 2  
5 0 2  
7 8 9  
1 2 4  
0 1 1  
Euklidska norma: 15.875
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
500 3  
Greska: neispravan unos.
```

[Rešenje 2.7.2]

Zadatak 2.7.3 Napisati funkcije za rad sa celobrojnim matricama:

- `void ucitaj(int a[][] [MAKS], int n, int m)` kojom se učitavaju elementi matrice celih brojeva a dimenzije $m \times n$,
- `void ispisi(int a[][] [MAKS], int n, int m)` kojom se ispisuju elementi matrice a dimenzije $m \times n$.

Napisati program koji najpre učitava, a zatim i ispisuje elemente učitane matrice. Prepostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: *U ovom i u narednim zadacima, konstanta MAKS u prototipu funkcije označava maksimalni broj kolona date matrice i potrebno ju je definisati u rešenju direktivom #define.*

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
3 4  
Unesite elemente matrice:  
1 2 3 4  
5 6 7 8  
9 10 11 12  
Matrica je:  
1 2 3 4  
5 6 7 8  
9 10 11 12
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
2 5  
Unesite elemente matrice:  
1 1 2 3 4  
5 0 2 5 7  
Matrica je:  
1 1 2 3 4  
5 0 2 5 7
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
500 3  
Greska: neispravan unos.
```

[Rešenje 2.7.3]

Zadatak 2.7.4 Napisati funkciju void transponovana(int a[] [MAKS], int m, int n, int b[] [MAKS]) koja određuje matricu b koja je dobijena transponovanjem matrice a . Napisati program koji za učitanu matricu celih brojeva ispisuje odgovarajuću transponovanu matricu. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
3 4  
Unesite elemente matrice:  
1 2 3 4  
5 6 7 8  
9 10 11 12  
Transponovana matrica je:  
1 5 9  
2 6 10  
3 7 11  
4 8 12
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
5 3  
Unesite elemente matrice:  
1 1 2  
5 0 2  
7 8 9  
1 2 4  
0 1 1  
Transponovana matrica je:  
1 5 7 1 0  
1 0 8 2 1  
2 2 9 4 1
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
500 3  
Greska: neispravan unos.
```

[Rešenje 2.7.4]

Zadatak 2.7.5 Napisati funkciju void razmeni(int a[] [MAKS], int m, int n, int k, int t) u kojoj se razmenjuju elementi k -te i t -te vrste matrice a dimezije $m \times n$. Napisati program koji za učitanu matricu celih brojeva i dva cela broja k i t ispisuje matricu dobijenu razmenjivanjem k -te i t -te vrste ulazne matrice. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
3 4  
Unesite elemente matrice:  
1 2 3 4  
5 6 7 8  
9 10 11 12  
Unesite indekse vrsta:  
0 2  
Rezultujuca matrica:  
9 10 11 12  
5 6 7 8  
1 2 3 4
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
5 3  
Unesite elemente matrice:  
1 1 2  
5 0 2  
7 8 9  
1 2 4  
0 1 1  
Unesite indekse vrsta:  
1 3  
Rezultujuca matrica:  
1 1 2  
1 2 4  
7 8 9  
5 0 2  
0 1 1
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
5 3  
Unesite elemente matrice:  
1 1 2  
5 0 2  
7 8 9  
1 2 4  
0 1 1  
Unesite indekse vrsta:  
-1 50  
Greska: neispravan unos.
```

[Rešenje 2.7.5]

2 Napredni tipovi podataka

Zadatak 2.7.6 Napisati program koji za učitanu matricu celih brojeva ispisuje indekse onih elemenata matrice koji su jednaki zbiru svih svojih susednih elemenata. Prepostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

```
- - - - - s b s -  
- s s s - - s s s -  
- s a s - - - - -  
- s s s - - - - -  
- - - - - - s s  
- - - - - - s c
```

Slika 2.1: Susedni elementi u matrici.

UPUTSTVO: Elementi matrice m susedni elementu $m[i][j]$ su svi elementi matrice čiji se indeksi, po apsolutnoj vrednosti, razlikuju najviše za jedan. Element matrice može imati najviše osam suseda: $m[i-1][j-1]$, $m[i-1][j]$, $m[i-1][j+1]$, $m[i][j-1]$, $m[i][j+1]$, $m[i+1][j-1]$, $m[i+1][j]$ i $m[i+1][j+1]$. U zavisnosti od položaja u matrici, element matrice može imati i tri ili pet suseda. Na slici 2.1 su slovom *s* obeleženi susedni elementi matrice za elemente $m[2][2]$ (element je na slici obeležen sa *a*), $m[0][7]$ (element je na slici obeležen sa *b*) i $m[5][9]$ (element je na slici obeležen sa *c*).

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
4 5  
Unesite elemente matrice:  
1 1 2 1 3  
0 8 1 9 0  
1 1 1 0 0  
0 3 0 2 2  
Indeksi elemenata koji su  
jednaki zbiru suseda su:  
1 1  
3 1  
3 4
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
3 4  
Unesite elemente matrice:  
7 10 12 20  
-1 -3 1 7  
0 -47 2 0  
Indeksi elemenata koji su  
jednaki zbiru suseda su:  
0 3  
1 2
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
5 -3  
Greska: neispravan unos.
```

[Rešenje 2.7.6]

Zadatak 2.7.7 Napisati funkciju koja formira niz b_0, b_1, \dots, b_{n-1} od matrice $n \times m$ tako što element niza b_i izračunava kao srednju vrednost elemenata i -te vrste matrice. Napisati program koji za učitanu matricu celih brojeva ispisuje

dobijeni niz. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
4 5
Unesite elemente matrice:
1 1 2 1 3
0 8 1 9 0
1 1 1 0 0
0 3 0 2 2
Dobijeni niz je:
1.6 3.6 0.6 1.4
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
3 4
Unesite elemente matrice:
7 10 12 20
-1 -3 1 7
0 -47 2 0
Dobijeni niz je:
12.25 1 -11.25
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
51 13
Greska: neispravan unos.
```

[Rešenje 2.7.7]

Zadatak 2.7.8 Relacija se može predstaviti kvadratnom matricom nula i jedinica na sledeći način: element i je u relaciji sa elementom j ukoliko se u preseku i -te vrste i j -te kolone nalazi jedinica, a nije u relaciji ukoliko se tu nalazi nula. Napisati funkcije:

- `int refleksivna(int a[][] [MAKS], int n)` kojom se za relaciju zadatu matricom a dimenzije $n \times n$ ispituje da li je refleksivna;
- `int simetricna(int a[][] [MAKS], int n)` kojom se za relaciju zadatu matricom a dimenzije $n \times n$ ispituje da li je simetrična;
- `int tranzitivna(int a[][] [MAKS], int n)` kojom se za relaciju zadatu matricom a ispituje dimenzije $n \times n$ da li je tranzitivna;
- `int ekvivalencija(int a[][] [MAKS], int n)` kojom se za relaciju zadatu matricom a dimenzije $n \times n$ ispituje da li je relacija ekvivalencije.

Napisati program koji za učitanu dimenziju n i kvadratnu matricu dimenzije $n \times n$ ispisuje osobine odgovarajuće relacije. Pretpostaviti da je maksimalna dimenzija matrice 50×50 i da matrica za vrednosti elemenata može imati samo nule i jedinice. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 4
Unesite elemente matrice:
1 0 0 0
0 1 1 0
0 0 1 0
0 0 0 0
Relacija nije refleksivna.
Relacija nije simatricna.
Relacija jeste tranzitivna.
Relacija nije ekvivalencija.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 4
Unesite elemente matrice:
1 1 0 0
1 1 1 0
0 1 1 0
0 0 0 1
Relacija jeste refleksivna.
Relacija jeste simatricna.
Relacija nije tranzitivna.
Relacija nije ekvivalencija.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 54
Greska: neispravan unos.
```

[Rešenje 2.7.8]

Zadatak 2.7.9 Data je kvadratna matrica dimenzije $n \times n$.

- Napisati funkciju `float trag(float a[] [MAKS], int n)` koja računa trag matrice, odnosno zbir elemenata na glavnoj dijagonali matrice.
- Napisati funkciju `float suma_sporedna(float a[] [MAKS], int n)` koja računa zbir elemenata na sporednoj dijagonali matrice.
- Napisati funkciju `float suma_iznad(float a[] [MAKS], int n)` koja određuje sumu elemenata iznad glavne dijagonale.
- Napisati funkciju `float suma_ispod(float a[] [MAKS], int n)` koja određuje sumu elemenata ispod sporedne dijagonale matrice.

Napisati program koji za učitanu matricu realnih brojeva ispisuje na tri decimale trag matrice, sumu na sporednoj dijagonali, sumu iznad glavne dijagonale i sumu elemenata ispod sporedne dijagonale. Prepostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 4
Unesite elemente matrice:
6 12.08 -1 20.5
8 90 -33.4 19.02
7.02 5 -20 14.5
8.8 -1 3 -22.8
```

Primer 1 (nastavak)

```
Trag: 53.20
Suma na sporednoj dijagonali: 0.90
Suma iznad glavne dijagonale: 31.70
Suma ispod sporedne dijagonale: -7.28
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 5
Unesite elemente matrice:
1 2 3 5 5
7 8 9 0 1
6 4 3 2 2
8 9 1 3 4
0 3 1 8 6
```

Primer 2 (nastavak)

```
Trag: 21.00
Suma na sporednoj dijagonali: 17.00
Suma iznad glavne dijagonale: 33.00
Suma ispod sporedne dijagonale: 31.00
```

[Rešenje 2.7.9]

Zadatak 2.7.10 Kvadratna matrica je donje trougaona ako se u gornjem trouglu (iznad glavne dijagonale, ne uključujući dijagonalu) nalaze sve nule. Napisati program koji za učitanu kvadratnu matricu proverava da li je ona donje trougaona i ispisuje odgovarajuću poruku. Pretpostaviti da je maksimalna dimenzija matrice 100×100 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice:
5
Unesite elemente matrice:
-1 0 0 0 0
2 10 0 0 0
0 1 5 0 0
7 8 20 14 0
-23 8 5 1 11
Matrica jeste donje trougaona.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice:
3
Unesite elemente matrice:
2 -2 1
1 2 2
2 1 -2
Matrica nije donje trougaona.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice:
200
Greska: neispravan unos.
```

[Rešenje 2.7.10]

Zadatak 2.7.11 Napisati program koji za učitanu celobrojnu kvadratnu matricu ispisuje redni broj kolone koja ima najveći zbir elemenata. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

2 Napredni tipovi podataka

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
3  
Unesite elemente matrice:  
1 2 3  
7 3 4  
5 3 1  
Indeks kolone je: 0
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
4  
Unesite elemente matrice:  
7 8 9 10  
7 6 11 4  
3 1 2 -2  
8 3 9 9  
Indeks kolone je: 2
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
104  
Greska: neispravan unos.
```

[Rešenje 2.7.11]

Zadatak 2.7.12 Napisati program koji za učitanu kvadratnu matricu realnih brojeva izračunava i ispisuje na dve decimale razliku između zbira elemenata gornjeg trougla i zbira elemenata donjeg trougla matrice. Gornji trougao čine svi elementi matrice koji su iznad glavne i sporedne dijagonale (ne računajući dijagonale), a donji trougao čine svi elementi ispod glavne i sporedne dijagonale (ne računajući dijagonale). Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
3  
Unesite elemente matrice:  
2 3.2 4  
7 8.8 1  
2.3 1 1  
Razlika je: 2.20
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
5  
Unesite elemente matrice:  
2.3 1 12 8 -20  
4 -8.2 7 14.5 19  
1 -2.5 9 11 33  
3 4.3 -5.7 2 8  
9 56 1.08 7 5.5  
Razlika je: -30.38
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
52  
Greska: neispravan unos.
```

[Rešenje 2.7.12]

Zadatak 2.7.13 Napisati program koji za učitanu celobrojnu matricu dimenzije $m \times n$ i uneta dva broja p i k ($p \leq m$, $k \leq n$) ispisuje sume svih podmatrica dimenzije $p \times k$ unete matrice. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice: 3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Unesite dva cela broja: 3 3
Sume podmatrica su: 54 63
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice: 3 4
Unesite elemente matrice:
1 2 3 4
5 6 7 8
9 10 11 12
Unesite dva cela broja: 2 3
Sume podmatrica su: 24 30 48 54
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice: 5 3
Unesite elemente matrice:
1 1 2
5 0 2
7 8 9
1 2 4
0 1 1
Unesite dva cela broja: 2 2
Sume podmatrica su: 7 5 20 19 18 23 4 8
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice: -3 200
Greska: neispravan unos.
```

[Rešenje 2.7.13]

Zadatak 2.7.14 Napisati program koji za učitanu celobrojnu kvadratnu matricu ispituje da li su njeni elementi po kolonama, vrstama i dijagonalama (glavnoj i sporednoj) sortirani strogo rastuće. Pretpostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 2
Unesite elemente matrice:
6 9
4 10
Elementi nisu sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi nisu sortirani po dijagonalama.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 3
Unesite elemente matrice:
1 2 3
4 5 6
7 8 9
Elementi su sortirani po kolonama.
Elementi su sortirani po vrstama.
Elementi su sortirani po dijagonalama.
```

2 Napredni tipovi podataka

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice: 4  
Unesite elemente matrice:  
5 5 7 9  
6 10 11 13  
8 12 14 15  
13 15 16 20  
Elementi su sortirani po kolonama.  
Elementi nisu sortirani po vrstama.  
Elementi su sortirani po dijagonalama.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice: 1  
Unesite elemente matrice:  
5  
Elementi su sortirani po kolonama.  
Elementi su sortirani po vrstama.  
Elementi su sortirani po dijagonalama.
```

[Rešenje 2.7.14]

Zadatak 2.7.15 Napisati program koji za učitanu celobrojnu kvadratnu matricu ispituje da li su zbroji elemenata njenih kolona uređeni u strogo raštućem poretku. Pretpostaviti da je maksimalna dimenzija matrice 10×10 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice: 4  
Unesite elemente matrice:  
1 0 0 0  
0 0 1 0  
0 0 0 1  
0 1 0 0  
Sume nisu uredjene strogo rastuce.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice: 3  
Unesite elemente matrice:  
1 2 3  
4 5 6  
7 8 9  
Sume jesu uredjene strogo rastuce.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice: 3  
Unesite elemente matrice:  
2 -2 1  
1 2 2  
2 1 -2  
Sume nisu uredjene strogo rastuce.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice: 5  
Unesite elemente matrice:  
-1 0 3 0 20  
0 0 0 10 0  
0 0 -1 0 0  
0 1 0 0 0  
0 0 0 0 -1  
Sume jesu uredjene strogo rastuce.
```

[Rešenje 2.7.15]

Zadatak 2.7.16 Matrica je *ortonormirana* ako je vrednost skalarnog proizvoda svakog para različitih vrsta jednak nuli, a vrednost skalarnog proizvoda vrste sa samom sobom jednak jedinici. Napisati program koji za unetu celobrojnu

kvadratnu matricu proverava da li je ortonormirana. Prepostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: Skalarni proizvod vektora $a = (a_1, a_2, \dots, a_n)$ i $b = (b_1, b_2, \dots, b_n)$ je $a_1 \cdot b_1 + a_2 \cdot b_2 + \dots + a_n \cdot b_n$.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 4
Unesite elemente matrice:
1 0 0 0
0 0 1 0
0 0 0 1
0 1 0 0
Matrica jeste ortonormirana.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 3
Unesite elemente matrice:
1 2 3
4 5 6
7 8 9
Matrica nije ortonormirana.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 3
Unesite elemente matrice:
2 -2 1
1 2 2
2 1 -2
Matrica nije ortonormirana.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 5
Unesite elemente matrice:
-1 0 0 0 0
0 0 0 1 0
0 0 -1 0 0
0 1 0 0 0
0 0 0 0 -1
Matrica jeste ortonormirana.
```

[Rešenje 2.7.16]

Zadatak 2.7.17 Kvadratna matrica je *magični kvadrat* ako su sume elemenata u svim vrstama i kolonama jednake. Napisati program koji proverava da li je data celobrojna kvadratna matrica magični kvadrat i ispisuje odgovarajuću poruku na standardni izlaz. Prepostaviti da je maksimalna dimenzija matrice 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 4
Unesite elemente matrice:
1 5 3 1
2 1 2 5
3 2 2 3
4 2 3 1
Matrica jeste magični kvadrat.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta matrice: 3
Unesite elemente matrice:
1 2 3
4 5 6
-1 3 3
Matrica nije magični kvadrat.
```

[Rešenje 2.7.17]

2 Napredni tipovi podataka

* **Zadatak 2.7.18** Napisati program koji učitava celobrojnu kvadratnu matricu i ispisuje elemente matrice u grupama koje su paralelne sa njenom sporednom dijagonalom, počevši od gornjeg levog ugla. Prepostaviti da je maksimalna dimenzija matrice 100×100 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
3  
Unesite elemente matrice:  
1 2 3  
4 5 6  
7 8 9  
Ispis je:  
1  
2 4  
3 5 7  
6 8  
9
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
5  
Unesite elemente matrice:  
7 -8 1 2 3  
90 11 0 5 4  
12 -9 14 23 8  
80 6 88 17 62  
-22 10 44 57 -200  
Ispis je:  
7  
-8 90  
1 11 12  
2 0 -9 80  
3 5 14 6 -22  
4 23 88 10  
8 17 44  
62 57  
-200
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta matrice:  
-5  
Greska: neispravan unos.
```

[Rešenje 2.7.18]

* **Zadatak 2.7.19** Napisati funkciju `void mnozenje(int a[][] [MAKS], int m, int n, int b[][] [MAKS], int k, int t, int c[][] [MAKS])` koja računa matricu c kao proizvod matrica a i b . Dimenzija matrice a je $n \times m$, a dimenzija matrice b je $k \times t$. Napisati program koji ispisuje proizvod učitanih matrica. Prepostaviti da je maksimalna dimenzija matrica 50×50 . Ukoliko množenje matrica nije moguće ili je došlo do greške prilikom unosa podataka, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice a:
3 4
Unesite elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Unesite broj vrsta i broj kolona matrice b:
4 2
Unesite elemente matrice:
11 5
6 7
8 9
0 -3
Rezultat mnozenja je:
87 64
2 24
145 83
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice a:
5 2
Unesite elemente matrice:
1 7
9 0
-10 2
92 3
14 -8
Unesite broj vrsta i broj kolona matrice b:
2 4
Unesite elemente matrice:
7 8 9 10
-11 2 34 78
Rezultat mnozenja je:
-70 22 247 556
63 72 81 90
-92 -76 -22 56
611 742 930 1154
186 96 -146 -484
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice a:
3 4
Unesite elemente matrice:
1 2 8 9
-4 5 2 3
7 6 4 10
Unesite broj vrsta i broj kolona matrice b:
5 2
Mnozenje matrica nije moguce.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i broj kolona matrice a:
-3 4
Greska: neispravan unos.
```

[Rešenje 2.7.19]

* **Zadatak 2.7.20** Element matrice naziva se *sedlo* ako je istovremeno najmanji u svojoj vrsti, a najveći u svojoj koloni. Napisati program koji ispisuje indekse i vrednosti onih elemenata matrice realnih brojeva koji su sedlo. Maksimalna dimenzija matrice je 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
2 3
Unesite elemente matrice:
1 2 3
0 5 6
Sedlo: 0 0 1
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
3 3
Unesite elemente matrice:
10 3 20
15 5 100
30 -1 200
Sedlo: 1 1 5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj vrsta i
broj kolona matrice:
3 -3
Greska: neispravan unos.
```

[Rešenje 2.7.20]

2 Napredni tipovi podataka

* **Zadatak 2.7.21** Napisati program koji ispisuje elemente matrice celih brojeva u spiralnom redosledu počevši od gornjeg levog ugla krećući se u smeru kazaljke na satu. Maksimalna dimenzija matrice je 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
3 3  
Unesite elemente matrice:  
1 2 3  
4 5 6  
7 8 9  
Ispis je:  
1 2 3 6 9 8 7 4 5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i  
broj kolona matrice:  
5 7  
Unesite elemente matrice:  
7 -8 1 2 3 -54 87  
90 11 0 5 4 9 18  
12 -9 14 23 8 -22 74  
80 6 88 17 62 38 41  
-22 10 44 57 -200 39 55  
Ispis je:  
7 -8 1 2 3 -54 87 18 74 41 55  
39 -200 57 44 10 -22 80 12 90  
11 0 5 4 9 -22 38 62 17 88 6  
-9 14 23 8
```

[Rešenje 2.7.21]

* **Zadatak 2.7.22** Matrica a se sadrži u matrici b ukoliko postoji podmatrica matrice b identična matrici a . Napisati program koji za dve učitane matrice celih brojeva proverava da li se druga matrica sadrži u prvoj učitanoj matrici. Maksimalna dimenzija matrica je 50×50 . U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i broj kolona matrice A:  
3 4  
Unesite elemente matrice:  
1 2 8 9  
-4 5 2 3  
7 6 4 10  
Unesite broj vrsta i broj kolona matrice B:  
2 2  
Unesite elemente matrice:  
2 3  
4 10  
Druga matrica je sadrzana u prvoj matrici.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i broj kolona matrice A:  
3 4  
Unesite elemente matrice:  
1 2 8 9  
-4 5 2 3  
7 6 4 10  
Unesite broj vrsta i broj kolona matrice B:  
2 2  
Unesite elemente matrice:  
2 8  
6 4  
Druga matrica nije sadrzana  
u prvoj matrici.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i broj kolona matrice A:  
5 5  
Unesite elemente matrice:  
7 -8 1 2 3  
90 11 0 5 4  
12 -9 14 23 8  
80 6 88 17 62  
-22 10 44 57 -200  
Unesite broj vrsta i broj kolona matrice B:  
3 4  
Unesite elemente matrice:  
90 11 0 5  
12 -9 14 23  
80 6 88 17  
Druga matrica je sadrzana u prvoj matrici.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj vrsta i broj kolona matrice A:  
5 5  
Unesite elemente matrice:  
7 -8 1 2 3  
90 11 0 5 4  
12 -9 14 23 8  
80 6 88 17 62  
-22 10 44 57 -200  
Unesite broj vrsta i broj kolona matrice B:  
53 4  
Greska: neispravan unos.
```

[Rešenje 2.7.22]

2.8 Rešenja

Rešenje 2.7.1

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 int main() {
7     /* Deklaracije potrebnih promenljivih. */
8     int a[MAKS][MAKS];
9     int i, j, m, n;
10
11    /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
12    printf("Unesite broj vrsta i broj kolona matrice: ");
13    scanf("%d%d", &m, &n);
14    if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
15        printf("Greska: neispravan unos.\n");
16        exit(EXIT_FAILURE);
17    }
18
19    /* Ucitavanje elemenata matrice. */
20    printf("Unesite elemente matrice:\n");
21    for (i = 0; i < m; i++)
22        for (j = 0; j < n; j++)
23            scanf("%d", &a[i][j]);
24
25    /* Ispis elemenata matrice. */
26    printf("Matrica je:\n");
27    for (i = 0; i < m; i++) {
28        for (j = 0; j < n; j++)
29            printf("%d ", a[i][j]);
30        printf("\n");
31    }
32
33    exit(EXIT_SUCCESS);
34}
```

Rešenje 2.7.2

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 #define MAKS 50
6
7 int main() {
8     /* Deklaracije potrebnih promenljivih. */
9     int a[MAKS][MAKS];
```

```

11   int i, j, m, n, suma = 0;
12
13  /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
14  printf("Unesite broj vrsta i broj kolona matrice: ");
15  scanf("%d%d", &m, &n);
16  if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
17      printf("Greska: neispravan unos.\n");
18      exit(EXIT_FAILURE);
19  }
20
21  /* Ucitavanje elemenata matrice. */
22  printf("Unesite elemente matrice:\n");
23  for (i = 0; i < m; i++)
24      for (j = 0; j < n; j++)
25          scanf("%d", &a[i][j]);
26
27  /* Racunanje sume kvadrata svih elemenata. */
28  for (i = 0; i < m; i++)
29      for (j = 0; j < n; j++)
30          suma += a[i][j] * a[i][j];
31
32  /* Ispis rezultata. */
33  printf("Euklidska norma: %.3lf\n", sqrt(suma));
34
35  exit(EXIT_SUCCESS);
}

```

Rešenje 2.7.3

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int m, int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < m; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13}
14
15 /* Funkcija ispisuje elemente matrice dimenzije mxn. */
16 void ispisi(int a[][MAKS], int m, int n) {
17    int i, j;
18    printf("Matrica je:\n");
19    for (i = 0; i < m; i++) {
20        for (j = 0; j < n; j++)
21            printf("%d ", a[i][j]);
22        printf("\n");
23    }
24}

```

2 Napredni tipovi podataka

```
23     }
24 }
25
26 int main() {
27     /* Deklaracije potrebnih promenljivih. */
28     int a[MAKS][MAKS];
29     int m, n;
30
31     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
32     printf("Unesite broj vrsta i broj kolona matrice: ");
33     scanf("%d%d", &m, &n);
34     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
35         printf("Greska: neispravan unos.\n");
36         exit(EXIT_FAILURE);
37     }
38
39     /* Ucitavanje elemenata matrice. */
40     ucitaj(a, m, n);
41
42     /* Ispis ucitane matrice. */
43     ispisi(a, m, n);
44
45     exit(EXIT_SUCCESS);
46 }
```

Rešenje 2.7.4

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int m, int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < m; i++) {
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13    }
14
15 /* Funkcija ispisiuje elemente matrice dimenzije mxn. */
16 void ispisi(int a[][MAKS], int m, int n) {
17     int i, j;
18     for (i = 0; i < m; i++) {
19         for (j = 0; j < n; j++)
20             printf("%d ", a[i][j]);
21         printf("\n");
22     }
23 }
```

```

25 /* Funkcija formira maticu t transponovanjem matrice a. */
26 void transponovana(int a[][][MAKS], int m, int n, int t[][][MAKS]) {
27     int i, j;
28     for (i = 0; i < m; i++)
29         for (j = 0; j < n; j++)
30             t[j][i] = a[i][j];
31 }

32 int main() {
33     /* Deklaracije potrebnih promenljivih. */
34     int a[MAKS][MAKS], t[MAKS][MAKS];
35     int m, n;
36
37     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
38     printf("Unesite broj vrsta i broj kolona matrice: ");
39     scanf("%d%d", &m, &n);
40     if (n <= 0 || n > MAKST || m <= 0 || m > MAKST) {
41         printf("Greska: neispravan unos.\n");
42         exit(EXIT_FAILURE);
43     }
44
45     /* Ucitavanje elemenata matrice. */
46     ucitaj(a, m, n);
47
48     /* Formiranje transponovane matrice. */
49     transponovana(a, m, n, t);
50
51     /* Ispis rezultata. */
52     printf("Transponovana matrica je:\n");
53     ispisi(t, n, m);
54
55     exit(EXIT_SUCCESS);
56 }

```

Rešenje 2.7.5

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKST 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][][MAKS], int m, int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < m; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13
14 /* Funkcija ispisuje elemente matrice dimenzije mxn. */

```

2 Napredni tipovi podataka

```
void ispisi(int a[][MAKS], int m, int n) {
17    int i, j;
18    for (i = 0; i < m; i++) {
19        for (j = 0; j < n; j++)
20            printf("%d ", a[i][j]);
21        printf("\n");
22    }
23}

25 /* Funkcija razmenjuje elemente k-te i t-te vrste. */
void razmeni(int a[][MAKS], int m, int n, int k, int t) {
27    int j, pom;
28    for (j = 0; j < n; j++) {
29        pom = a[k][j];
30        a[k][j] = a[t][j];
31        a[t][j] = pom;
32    }
33}

35 int main() {
36    /* Deklaracije potrebnih promenljivih. */
37    int a[MAKS][MAKS];
38    int m, n, k, t;
39
40    /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
41    printf("Unesite broj vrsta i broj kolona matrice: ");
42    scanf("%d%d", &m, &n);
43    if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
44        printf("Greska: neispravan unos.\n");
45        exit(EXIT_FAILURE);
46    }
47
48    /* Ucitavanje elemenata matrice. */
49    ucitaj(a, m, n);
50
51    /* Ucitavanje indeksa vrsta i provera ispravnosti ulaza. */
52    printf("Unesite indekse vrsta: ");
53    scanf("%d%d", &k, &t);
54    if (k < 0 || k >= m || t < 0 || t >= m) {
55        printf("Greska: neispravan unos.\n");
56        exit(EXIT_FAILURE);
57    }
58
59    /* Razmena k-te i t-te vrste. */
60    razmeni(a, m, n, k, t);
61
62    /* Ispis rezultata. */
63    printf("Rezultujuca matrica:\n");
64    ispisi(a, m, n);
65
66    exit(EXIT_SUCCESS);
67}
```

Rešenje 2.7.6

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int m, int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < m; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13}
14
15 int main() {
16     /* Deklaracije potrebnih promenljivih. */
17     int a[MAKS][MAKS];
18     int m, n, i, j, suma_suseda;
19     int k, t;
20
21     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
22     printf("Unesite broj vrsta i broj kolona matrice: ");
23     scanf("%d%d", &m, &n);
24     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
25         printf("Greska: neispravan unos.\n");
26         exit(EXIT_FAILURE);
27     }
28
29     /* Ucitavanje elemenata matrice. */
30     ucitaj(a, m, n);
31
32     /* Izracunavanje i ispis rezultata. */
33     printf("Indeksi elemenata koji su jednaki zbiru suseda su:\n");
34     for (i = 0; i < m; i++) {
35         for (j = 0; j < n; j++) {
36             suma_suseda = 0;
37
38             /* Racunanje sume elemenata podmatrice velicine 3*3 ciji je
39                 centralni element a[i][j]. Pri racunanju ove sume vodi se
40                 racuna da se ne izadje iz okvira matice a. Preciznije,
41                 ukoliko su sacunate neodgovarajuce vrednosti za indekse
42                 k i t (npr. kada je i=0 ili kada je i=m-1), te vrednosti
43                 zahvaljuci uslovu k >= 0 && k < m && t >= 0 && t < n nece
44                 uci u sumu. */
45             for (k = i - 1; k <= i + 1; k++)
46                 for (t = j - 1; t <= j + 1; t++)
47                     if (k >= 0 && k < m && t >= 0 && t < n)
48                         suma_suseda += a[k][t];
49
50             /* Od ukupne sume se oduzima tekuci element kako bi se dobio
51             rezultat. */
52             printf("%d ", suma_suseda - a[i][j]);
53         }
54     }
55 }

```

2 Napredni tipovi podataka

```
51     zbir elemenata koji su njegovi susedi. */
52     suma_suseda -= a[i][j];
53
54     /* Ukoliko je suma suseda jednaka tekucem elementu, ispisuju
55      se indeksi tekuceg elementa matrice. */
56     if (suma_suseda == a[i][j])
57         printf("%d %d\n", i, j);
58     }
59 }
60 exit(EXIT_SUCCESS);
61 }
```

Rešenje 2.7.7

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int m, int n) {
8     int i, j;
9
10    printf("Unesite elemente matrice:\n");
11    for (i = 0; i < m; i++)
12        for (j = 0; j < n; j++)
13            scanf("%d", &a[i][j]);
14
15    /* Funkcija formira niz b tako sto element b[i] ima vrednost
16       prosecne vrednosti i-te vrste matrice. */
17    void kreiraj_niz(int a[][MAKS], int m, int n, double b[]) {
18        int i, j, suma;
19
20        for (i = 0; i < m; i++) {
21            suma = 0;
22            for (j = 0; j < n; j++)
23                suma += a[i][j];
24
25            b[i] = (double) suma / n;
26        }
27    }
28
29    int main() {
30        /* Deklaracije potrebnih promenljivih. */
31        int a[MAKS][MAKS];
32        double b[MAKS];
33        int m, n, i;
34
35        /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
36        printf("Unesite broj vrsta i broj kolona matrice: ");
37    }
```

```

39     scanf("%d%d", &m, &n);
40     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
41         printf("Greska: neispravan unos.\n");
42         exit(EXIT_FAILURE);
43     }
44
45     /* Ucitavanje elemenata matrice. */
46     ucitaj(a, m, n);
47
48     /* Formiranje niza b. */
49     kreiraj_niz(a, m, n, b);
50
51     /* Ispis rezultata. */
52     printf("Dobijeni niz je:\n");
53     for (i = 0; i < m; i++)
54         printf("%g ", b[i]);
55     printf("\n");
56
57     exit(EXIT_SUCCESS);
}

```

Rešenje 2.7.8

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13
14    /* Relacija je refleksivna ukoliko je za svako i a[i][i] = 1.
15       Funkcija proverava da li je relacija zadata matricom a
16       refleksivna i vraca 1 ukoliko jeste, a 0 inace. */
17    int refleksivna(int a[][MAKS], int n) {
18        int i;
19        for (i = 0; i < n; i++)
20            if (a[i][i] != 1)
21                return 0;
22
23        return 1;
24    }
25
26    /* Relacija je simetricna ukoliko za svaki par i, j vazi da je
27       a[i][j] = a[j][i]. Funkcija proverava da li je relacija zadata
28       matricom a i vraca 1 ukoliko jeste, a 0 inace. */

```

2 Napredni tipovi podataka

```
    matricom a simetricna i vraca 1 ukoliko jeste, a 0 inace. */
30 int simetricna(int a[][MAKS], int n) {
31     int i, j;
32     for (i = 0; i < n; i++)
33         for (j = i + 1; j < n; j++)
34             if (a[i][j] != a[j][i])
35                 return 0;
36
37     return 1;
38 }

40 /* Relacija je tranzitivna ukoliko za svaku trojku i, j, k vazi da
41    ako je a[i][j] = 1 i a[j][k] = 1, onda je i a[i][k] = 1.
42    Funkcija proverava da li je relacija zadata matricom a
43    tranzitivna i vraca 1 ukoliko jeste, a 0 inace. */
44 int tranzitivna(int a[][MAKS], int n) {
45     int i, j, k;
46     for (i = 0; i < n; i++)
47         for (j = 0; j < n; j++)
48             for (k = 0; k < n; k++)
49                 if (a[i][j] == 1 && a[j][k] == 1 && a[i][k] == 0)
50                     return 0;
51
52     return 1;
53 }

54 /* Relacija je relacija ekvivalencije ukoliko je refleksivna,
55    simetricna i tranzitivna. Funkcija proverava da li je relacija
56    zadata matricom a relacija ekvivalencije i vraca 1 ukoliko
57    jeste, a 0 inace. */
58 int ekvivalencija(int a[][MAKS], int n) {
59     if (refleksivna(a, n) && simetricna(a, n) && tranzitivna(a, n))
60         return 1;
61
62     return 0;
63 }

64 int main() {
65     /* Deklaracije potrebnih promenljivih. */
66     int a[MAKS][MAKS];
67     int n;
68
69     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
70     printf("Unesite broj vrsta matrice: ");
71     scanf("%d", &n);
72     if (n <= 0 || n > MAKST) {
73         printf("Greska: neispravan unos.\n");
74         exit(EXIT_FAILURE);
75     }
76
77     /* Ucitavanje elemenata matrice. */
78     ucitaj(a, n);
79 }
```

```

82  /* Racunanje i ispis rezultata. */
83  if (refleksivna(a, n))
84      printf("Relacija jeste refleksivna.\n");
85  else
86      printf("Relacija nije refleksivna.\n");
87
88  if (simetricna(a, n))
89      printf("Relacija jeste simetricna.\n");
90  else
91      printf("Relacija nije simetricna.\n");
92
93  if (tranzitivna(a, n))
94      printf("Relacija jeste tranzitivna.\n");
95  else
96      printf("Relacija nije tranzitivna.\n");
97
98  if (ekvivalencija(a, n))
99      printf("Relacija jeste ekvivalencija.\n");
100 else
101     printf("Relacija nije ekvivalencija.\n");
102
103 exit(EXIT_SUCCESS);
104 }
```

Rešenje 2.7.9

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(float a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%f", &a[i][j]);
13 }
14
15 /* Funkcija racuna trag matrice. */
16 float trag(float a[][MAKS], int n) {
17     float suma = 0;
18     int i;
19
20     for (i = 0; i < n; i++)
21         suma += a[i][i];
22
23     return suma;
24 }
```

```
26 /* Funkcija racuna sumu elemenata koji se nalaze na sporednoj
   dijagonali matrice. */
28 float suma_sporedna(float a[][][MAKS], int n) {
29     float suma = 0;
30     int i;
31
32     for (i = 0; i < n; i++)
33         suma += a[i][n - i - 1];
34
35     return suma;
36 }
37
38 /* Funkcija racuna sumu elemenata koji se nalaze iznad glavne
   dijagonale matrice. */
39 float suma_iznad(float a[][][MAKS], int n) {
40     float suma = 0;
41     int i, j;
42
43     for (i = 0; i < n; i++)
44         for (j = i + 1; j < n; j++)
45             suma += a[i][j];
46
47     return suma;
48 }
49
50 /* Funkcija racuna sumu elemenata koji se nalaze ispod sporedne
   dijagonale matrice. */
51 float suma_ispod(float a[][][MAKS], int n) {
52     float suma = 0;
53     int i, j;
54
55     for (i = 0; i < n; i++)
56         for (j = n - i; j < n; j++)
57             suma += a[i][j];
58
59     return suma;
60 }
61
62
63 int main() {
64     /* Deklaracije potrebnih promenljivih. */
65     float a[MAKS][MAKS];
66     int n;
67
68     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
69     printf("Unesite broj vrsta matrice: ");
70     scanf("%d", &n);
71     if (n <= 0 || n > MAKS) {
72         printf("Greska: neispravan unos.\n");
73         exit(EXIT_FAILURE);
74     }
75 }
76
```

```

78    /* Ucitavanje elemenata matrice. */
79    ucitaj(a, n);

80    /* Ispis rezultata. */
81    printf("Trag: %.2f\n", trag(a, n));
82    printf("Suma na sporednoj dijagonalni: %.2f\n",
83           suma_sporedna(a, n));
84    printf("Suma iznad glavne dijagonale: %.2f\n",
85           suma_iznad(a, n));
86    printf("Suma ispod sporedne dijagonale: %.2f\n",
87           suma_ispod(a, n));

88    exit(EXIT_SUCCESS);
89 }

```

Rešenje 2.7.10

```

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

4 #define MAKS 50

6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13 }

14 /* Funkcija proverava da li je matrica donje trougaona i vraca
15 jedinicu ukoliko jeste, a nulu inace. */
16 int donje_trougaona(int a[][MAKS], int n) {
17     int i, j;

18     /* Prolazi se kroz sve elemente iznad glavne dijagonale i ukoliko
19      se nađe na element koji je razlicit od nule, onda matrica
20      nije donje trougaona. */
21     for (i = 0; i < n; i++)
22         for (j = i + 1; j < n; j++)
23             if (a[i][j] != 0)
24                 return 0;

25     /* Ukoliko su svi elementi iznad glavne dijagonale nule, matrica
26      jeste donje trougaona. */
27     return 1;
28 }

29 int main() {
30     /* Deklaracije potrebnih promenljivih. */
31 }

```

2 Napredni tipovi podataka

```
36     int a[MAKS][MAKS];
37     int n;
38
39     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
40     printf("Unesite broj vrsta matrice: ");
41     scanf("%d", &n);
42     if (n <= 0 || n > MAKS) {
43         printf("Greska: neispravan unos.\n");
44         exit(EXIT_FAILURE);
45     }
46
47     /* Ucitavanje elemenata matrice. */
48     ucitaj(a, n);
49
50     /* Ispis rezultata. */
51     if (donje_trougaona(a, n))
52         printf("Matrica jeste donje trougaona.\n");
53     else
54         printf("Matrica nije donje trougaona.\n");
55
56     exit(EXIT_SUCCESS);
57 }
```

Rešenje 2.7.11

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13 }
14
15 int main() {
16     /* Deklaracije potrebnih promenljivih. */
17     int a[MAKS][MAKS];
18     int n, i, j;
19     int maksimalni_zbir, trenutni_zbir = 0, indeks_kolone;
20
21     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
22     printf("Unesite broj vrsta matrice: ");
23     scanf("%d", &n);
24     if (n <= 0 || n > MAKS) {
25         printf("Greska: neispravan unos.\n");
26         exit(EXIT_FAILURE);
27     }
28
29     /* ... ostatak funkcije */
30 }
```

```

    }

28 /* Ucitavanje elemenata matrice. */
29 ucitaj(a, n);

32 /* Maksimalni zbir se inicijalizuje na vrednost zbiru prve
33 kolone. U ovom slučaju bi bilo pogresno da se maksimalni zbir
34 inicijalizuje na nulu jer može da se desi da su svi elementi
35 matrice negativni. Drugi nacin da se ispravno inicijalizuje
36 maksimalni zbir jeste da mu se dodeli vrednost konstante
37 INT_MIN cija se definicija nalazi u zaglavljku limits.h. */
38 for (i = 0; i < n; i++)
39     trenutni_zbir += a[i][0];

40 maksimalni_zbir = trenutni_zbir;
41 indeks_kolone = 0;

44 /* Racunanje zbiru svake sledeće kolone i azuriranje vrednosti
45 maksimalnog zbiru. */
46 for (j = 1; j < n; j++) {
47     /* Racunanje zbiru kolone j. */
48     trenutni_zbir = 0;
49     for (i = 0; i < n; i++)
50         trenutni_zbir += a[i][j];

52     /* Ukoliko je taj zbir veci od trenutno maksimalnog zbiru,
53     azurira se vrednost maksimalnog zbiru i pamti se tekuća
54     kolona. */
55     if (trenutni_zbir > maksimalni_zbir) {
56         maksimalni_zbir = trenutni_zbir;
57         indeks_kolone = j;
58     }
59 }

60 /* Ispis rezultata. */
61 printf("Indeks kolone je: %d\n", indeks_kolone);

64 exit(EXIT_SUCCESS);
}

```

Rešenje 2.7.12

```

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

4 #define MAKS 50

6 /* Funkcija učitava elemente matrice dimenzije mxn. */
7 void ucitaj(float a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");

```

2 Napredni tipovi podataka

```
10   for (i = 0; i < n; i++)
11     for (j = 0; j < n; j++)
12       scanf("%f", &a[i][j]);
13   }
14
15 int main() {
16   /* Deklaracije potrebnih promenljivih. */
17   float a[MAKS][MAKS];
18   int n, i, j;
19   float gornji_trougao = 0, donji_trougao = 0;
20
21   /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
22   printf("Unesite broj vrsta matrice: ");
23   scanf("%d", &n);
24   if (n <= 0 || n > MAKST) {
25     printf("Greska: neispravan unos.\n");
26     exit(EXIT_FAILURE);
27   }
28
29   /* Ucitavanje elemenata matrice. */
30   ucitaj(a, n);
31
32   /* Racunanje sume gornjeg trougla. */
33   for (i = 0; i < n / 2; i++)
34     for (j = i + 1; j < n - i - 1; j++)
35       gornji_trougao += a[i][j];
36
37   /* Racunanje sume donjeg trougla. */
38   for (i = n / 2; i < n; i++)
39     for (j = n - i; j < i; j++)
40       donji_trougao += a[i][j];
41
42   /* Ispis rezultata. */
43   printf("Razlika je: %.2f\n", gornji_trougao - donji_trougao);
44
45   exit(EXIT_SUCCESS);
46 }
```

Rešenje 2.7.13

```
#include <stdio.h>
2 #include <stdlib.h>
4
5 #define MAKST 50
6
7 /* Funkcija ucitava elemente matrice dimenzije mxn. */
8 void ucitaj(int a[][MAKS], int m, int n) {
9   int i, j;
10  printf("Unesite elemente matrice:\n");
11  for (i = 0; i < m; i++)
12    for (j = 0; j < n; j++)
```

```

12     scanf("%d", &a[i][j]);
13 }
14
15 int main() {
16     /* Deklaracije potrebnih promenljivih. */
17     int a[MAKS][MAKS];
18     int n, i, j, m, x, y, p, k;
19     int suma;
20
21     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
22     printf("Unesite broj vrsta i broj kolona matrice: ");
23     scanf("%d%d", &m, &n);
24     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
25         printf("Greska: neispravan unos.\n");
26         exit(EXIT_FAILURE);
27     }
28
29     /* Ucitavanje elemenata matrice. */
30     ucitaj(a, m, n);
31
32     /* Ucitavanje brojeva p i k i provera ispravnosti ulaza. */
33     printf("Unesite dva cela broja: ");
34     scanf("%d%d", &p, &k);
35     if (p <= 0 || p > m || k <= 0 || k > n) {
36         printf("Greska: neispravan unos.\n");
37         exit(EXIT_FAILURE);
38     }
39
40     /* Racunanje i ispis rezultata. */
41     printf("Sume podmatrica su: ");
42     for (i = 0; i <= m - p; i++) {
43         for (j = 0; j <= n - k; j++) {
44             /* Za svaku poziciju (i,j), racunana se suma podmatrice
45              dimenzije p x k, ciji je gornji levi ugao a[i][j]. */
46             suma = 0;
47             for (x = 0; x < p; x++)
48                 for (y = 0; y < k; y++)
49                     suma += a[i + x][j + y];
50
51             printf("%d ", suma);
52         }
53     }
54     printf("\n");
55
56     exit(EXIT_SUCCESS);
57 }
```

Rešenje 2.7.14

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13 }
14
15 /* Funkcija proverava da li je kolona j sortirana rastuce i vraca
16 jedinicu ukoliko jeste, a nulu inace. */
17 int sortirana_kolona(int a[][][MAKS], int n, int j) {
18     int i;
19
20     for (i = 0; i < n - 1; i++)
21         if (a[i][j] >= a[i + 1][j])
22             return 0;
23
24     return 1;
25 }
26
27 /* Funkcija proverava da li je svaka kolona matrice sortirana
28 rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
29 int sortirani_po_kolonama(int a[][][MAKS], int n) {
30     int j;
31
32     for (j = 0; j < n; j++)
33         if (!sortirana_kolona(a, n, j))
34             return 0;
35
36     return 1;
37 }
38
39 /* Funkcija proverava da li je i-ta vrsta sortirana rastuce i vraca
40 jedinicu ukoliko jeste, a nulu inace. */
41 int sortirana_vrsta(int a[][][MAKS], int n, int i) {
42     int j;
43
44     for (j = 0; j < n - 1; j++)
45         if (a[i][j] >= a[i][j + 1])
46             return 0;
47
48     return 1;
49 }
```

```

51 /* Funkcija proverava da li je svaka vrsta matrice sortirana
   rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
53 int sortirani_po_vrstama(int a[][MAKS], int n) {
54     int i;
55
56     for (i = 0; i < n; i++)
57         if (!sortirana_vrsta(a, n, i))
58             return 0;
59
60     return 1;
61 }
62
63 /* Funkcija proverava da li je glavna dijagonalna matrice sortirana
   rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
64 int sortirana_glavna(int a[][MAKS], int n) {
65     int i;
66
67     for (i = 0; i < n - 1; i++)
68         if (a[i][i] >= a[i + 1][i + 1])
69             return 0;
70
71     return 1;
72 }
73
74 /* Funkcija proverava da li je sporedna dijagonalna matrice
   sortirana rastuce i vraca jedinicu ukoliko jeste, a nulu inace. */
75 int sortirana_sporedna(int a[][MAKS], int n) {
76     int i;
77
78     for (i = 0; i < n - 1; i++)
79         if (a[i][n - i - 1] >= a[i + 1][n - i - 2])
80             return 0;
81
82     return 1;
83 }
84
85 /* Funkcija proverava da li su obe dijagonale matrice sortirane
   rastuce i vraca jedinicu ukoliko jesu, a nulu inace. */
86 int sortirani_po_dijagonalama(int a[][MAKS], int n) {
87     return sortirana_glavna(a, n) && sortirana_sporedna(a, n);
88 }
89
90 int main() {
91     /* Deklaracije potrebnih promenljivih. */
92     int a[MAKS][MAKS];
93     int n;
94
95     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
96     printf("Unesite broj vrsta matrice: ");
97     scanf("%d", &n);
98
99     if (n <= 0 || n > MAKS) {
100         printf("Greska: neispravan unos.\n");

```

2 Napredni tipovi podataka

```
103     exit(EXIT_FAILURE);
104 }
105
106 /* Ucitavanje elemenata matrice. */
107 ucitaj(a, n);
108
109 /* Ispis rezultata. */
110 if (sortirani_po_kolonama(a, n))
111     printf("Elementi su sortirani po kolonama.\n");
112 else
113     printf("Elementi nisu sortirani po kolonama.\n");
114
115 if (sortirani_po_vrstama(a, n))
116     printf("Elementi su sortirani po vrstama.\n");
117 else
118     printf("Elementi nisu sortirani po vrstama.\n");
119
120 if (sortirani_po_dijagonalama(a, n))
121     printf("Elementi su sortirani po dijagonalama.\n");
122 else
123     printf("Elementi nisu sortirani po dijagonalama.\n");
124
125 exit(EXIT_SUCCESS);
126 }
```

Rešenje 2.7.15

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 10
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13}
14
15 /* Funkcija racuna sumu elemenata kolone j. */
16 int suma_kolone(int a[][MAKS], int n, int j) {
17     int suma = 0, i;
18
19     for (i = 0; i < n; i++)
20         suma += a[i][j];
21
22     return suma;
23}
```

```

25 /* Funkcija proverava da li su sume kolona uredjene rastuce i vraca
   jedinicu ako jesu, a nulu inace. */
26 int uredjene_sume(int a[][][MAKS], int n) {
27     int prethodna_suma, trenutna_suma, j;
28
29     /* Prva suma se inicializuje na sumu prve kolone. */
30     prethodna_suma = suma_kolone(a, n, 0);
31
32     for (j = 1; j < n; j++) {
33         /* Racunanje sume trenutne kolone. */
34         trenutna_suma = suma_kolone(a, n, j);
35
36         /* Ukoliko je ta suma manja ili jednaka prethodnoj, poredak
            suma nije rastuci. */
37         if (trenutna_suma <= prethodna_suma)
38             return 0;
39
40         /* Suma trenutne kolone postaje suma prethodne kolone za
            narednu iteraciju. */
41         prethodna_suma = trenutna_suma;
42     }
43
44     return 1;
45 }
46
47 int main() {
48     /* Deklaracije potrebnih promenljivih. */
49     int a[MAKS][MAKS];
50     int n;
51
52     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
53     printf("Unesite broj vrsta matrice: ");
54     scanf("%d", &n);
55     if (n <= 0 || n > MAKS) {
56         printf("Greska: neispravan unos.\n");
57         exit(EXIT_FAILURE);
58     }
59
60     /* Ucitavanje elemenata matrice. */
61     ucitaj(a, n);
62
63     /* Ispis rezultata. */
64     if (uredjene_sume(a, n))
65         printf("Sume jesu uredjene strogo rastuce.\n");
66     else
67         printf("Sume nisu uredjene strogo rastuce.\n");
68
69     exit(EXIT_SUCCESS);
70 }
71
72 }
```

Rešenje 2.7.16

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 200
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13 }
14
15 /* Funkcija racuna skalarni proizvod i-te i j-te vrste matrice. */
16 int skalarni_proizvod(int a[][MAKS], int n, int i, int j) {
17     int suma = 0, k;
18
19     for (k = 0; k < n; k++)
20         suma += a[i][k] * a[j][k];
21
22     return suma;
23 }
24
25 /* Matrica je ortonormirana ukoliko je skalarni proizvod svakog
26 para razlicitih vrsta jednak nuli, a skalarni proizvod svake
27 vrste same sa sobom jednak jedinici. Funkcija proverava da li je
28 matrica ortonormirana i vraca jedinicu ukoliko jeste, a nulu
29 inace. */
30 int ortonormirana(int a[][MAKS], int n) {
31     int i, j;
32
33     /* Za svaki par vrsta se racuna skalarni proizvod i proverava da
34     li je uslov ispunjen. Ukoliko nije, kao povratna vrednost
35     funkcije se vraca nula. */
36     for (i = 0; i < n; i++) {
37         for (j = i; j < n; j++) {
38             /* Provera za slucaj kada se racuna skalarni proizvod vrste
39             same sa sobom. */
40             if (i == j && skalarni_proizvod(a, n, i, i) != 1)
41                 return 0;
42
43             /* Provera za par razlicitih vrsta. */
44             if (i != j && skalarni_proizvod(a, n, i, j) != 0)
45                 return 0;
46         }
47
48     /* Ako je izvrsavanje stiglo do kraja petlje, znaci da je uslov
49     ispunjen za sve vrste, tj. da je matrica ortonormirana. */
50     return 1;
51 }
```

```

51 }
53 int main() {
54     /* Deklaracije potrebnih promenljivih. */
55     int a[MAKS][MAKS];
56     int n;
57
58     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
59     printf("Unesite broj vrsta matrice: ");
60     scanf("%d", &n);
61     if (n <= 0 || n > MAKS) {
62         printf("Greska: neispravan unos.\n");
63         exit(EXIT_FAILURE);
64     }
65
66     /* Ucitavanje elemenata matrice. */
67     ucitaj(a, n);
68
69     /* Ispis rezultata. */
70     if (ortonormirana(a, n))
71         printf("Matrica jeste ortonormirana.\n");
72     else
73         printf("Matrica nije ortonormirana.\n");
74
75     exit(EXIT_SUCCESS);
76 }
```

Rešenje 2.7.17

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13}
14
15 /* Funkcija racuna sumu kolone j. */
16 int suma_kolone(int a[][MAKS], int n, int j) {
17     int i, suma = 0;
18
19     for (i = 0; i < n; i++)
20         suma += a[i][j];
21
22     return suma;
23}
```

2 Napredni tipovi podataka

```
23 }
25 /* Funkcija racuna sumu i-te vrste. */
26 int suma_vrste(int a[][][MAKS], int n, int i) {
27     int j, suma = 0;
28
29     for (j = 0; j < n; j++)
30         suma += a[i][j];
31
32     return suma;
33 }

35 /* Funkcija proverava da li elementi matrice predstavljaju magicni
36    kvadrat. */
37 int magicni_kvadrat(int a[][][MAKS], int n) {
38     /* Da bi matrica bila magicni kvadrat, sume svih vrsta i kolona
39        treba da budu jednake. Suma se zato inicijalizuje na sumu prve
40        kolone. */
41     int suma = suma_kolone(a, n, 0);
42     int i, j;
43
44     /* Proverava se da li su sume ostalih kolona jednake izracunatoj
45        sumi. Ukoliko se naidje na kolonu koja ne zadovoljava ovaj
46        uslov, matrica nije magicni kvadrat. */
47     for (j = 1; j < n; j++)
48         if (suma_kolone(a, n, j) != suma)
49             return 0;
50
51     /* Proverava se i da li su sume svih vrsta jednake izracunatoj
52        sumi. Ukoliko se naidje na vrstu koja ne zadovoljava ovaj
53        uslov, matrica nije magicni kvadrat. */
54     for (i = 0; i < n; i++)
55         if (suma_vrste(a, n, i) != suma)
56             return 0;
57
58     /* Ako sve vrste i kolone imaju jednake sume, matrica je magicni
59        kvadrat. */
60     return 1;
61 }

63 int main() {
64     /* Deklaracije potrebnih promenljivih. */
65     int a[MAKS][MAKS];
66     int n;
67
68     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
69     printf("Unesite broj vrsta matrice: ");
70     scanf("%d", &n);
71     if (n <= 0 || n > MAKS) {
72         printf("Greska: neispravan unos.\n");
73         exit(EXIT_FAILURE);
74     }
75 }
```

```

75  /* Ucitavanje elemenata matrice. */
76  ucitaj(a, n);

79  /* Ispis rezultata. */
80  if (magicni_kvadrat(a, n))
81      printf("Matrica jeste magicni kvadrat.\n");
82  else
83      printf("Matrica nije magicni kvadrat.\n");

85  exit(EXIT_SUCCESS);
}

```

Rešenje 2.7.18

```

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

4 #define MAKS 100

6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < n; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13 }

14 int main() {
15     /* Deklaracije potrebnih promenljivih. */
16     int a[MAKS][MAKS];
17     int n, i, j, k;

18     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
19     printf("Unesite broj vrsta matrice: ");
20     scanf("%d", &n);
21     if (n <= 0 || n > MAKS) {
22         printf("Greska: neispravan unos.\n");
23         exit(EXIT_FAILURE);
24     }

25     /* Ucitavanje elemenata matrice. */
26     ucitaj(a, n);

27     /* Petlja kojom se ispisuju dijagonale iznad sporedne dijagonale,
28     uključujući i sporednu dijagonalu.
29     Npr. za n=4, indeksi elemenata u matrici su:
30     (0,0) (0,1) (0,2) (0,3)
31     (1,0) (1,1) (1,2) (1,3)
32     (2,0) (2,1) (2,2) (2,3)
33
34
35
36

```

2 Napredni tipovi podataka

```
38     (3,0) (3,1) (3,2) (3,3)
Dakle, ispis elemenata ide u sledecem redosledu:
(0,0)
(0,1) (1,0)
(0,2) (1,1) (2,0)
(0,3) (1,2) (2,1) (3,0)
Za k-ti ispis vazi da indeksi vrsta imaju vrednosti od nula do
k, a indeksi kolona od k do nula. */
printf("Ispis je:\n");
for (k = 0; k < n; k++) {
    /* Indeks kolone se inicijalizuje na k, a indeks vrste na 0. */
    j = k;
    i = 0;

    /* Ispisuju se odgovarajuci elementi, indeks vrste se povecava,
       a indeks kolone se smanjuje. */
    while (j >= 0) {
        printf("%d ", a[i][j]);
        i++;
        j--;
    }
    printf("\n");
}

/* Petlja kojom se ispisuju dijagonale ispod sporedne dijagonale.
Npr. za n=4, indeksi elemenata u matrici su:
(0,0) (0,1) (0,2) (0,3)
(1,0) (1,1) (1,2) (1,3)
(2,0) (2,1) (2,2) (2,3)
(3,0) (3,1) (3,2) (3,3)
Dakle, ispis elemenata ide u sledecem redosledu:
(1,3) (2,2) (3,1)
(2,3) (3,2)
(3,3)
Za k-ti ispis vazi da indeksi vrsta imaju vrednosti od k do
n-1, a indeksi kolona od n-1 do 1. */
for (k = 1; k < n; k++) {
    i = k;
    j = n - 1;

    while (i < n) {
        printf("%d ", a[i][j]);
        i++;
        j--;
    }
    printf("\n");
}

exit(EXIT_SUCCESS);
}
```

Rešenje 2.7.19

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5
6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int m, int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < m; i++) {
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13    }
14
15 /* Funkcija ispisuje elemente matrice dimenzije mxn. */
16 void ispisi(int a[][MAKS], int m, int n) {
17     int i, j;
18     for (i = 0; i < m; i++) {
19         for (j = 0; j < n; j++)
20             printf("%d ", a[i][j]);
21         printf("\n");
22     }
23
24 /* Funkcija vrsti mnozenje matrica a i b i rezultat smesta u matricu
25   c. */
26 void mnozenje(int a[][MAKS], int m, int n,
27                 int b[][MAKS], int k, int t, int c[][][MAKS]) {
28     int i, j, w;
29
30     for (i = 0; i < m; i++)
31         for (j = 0; j < t; j++) {
32             /* Element c[i][j] se dobija kao skalarni proizvod i-te vrste
33              matrice a i j-te kolone matrice b. */
34             c[i][j] = 0;
35             for (w = 0; w < n; w++)
36                 c[i][j] += a[i][w] * b[w][j];
37         }
38     }
39
40 int main() {
41     /* Deklaracije potrebnih promenljivih. */
42     int a[MAKS][MAKS], b[MAKS][MAKS], c[MAKS][MAKS];
43     int m, n, k, t;
44
45     /* Ucitavanje dimenzija matrice a i provera ispravnosti ulaza. */
46     printf("Unesite broj vrsta i broj kolona matrice a: ");
47     scanf("%d%d", &m, &n);
48     if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
49         printf("Greska: neispravan unos.\n");
50     }

```

2 Napredni tipovi podataka

```
    exit(EXIT_FAILURE);
}

/* Ucitavanje elemenata prve matrice. */
ucitaj(a, m, n);

/* Ucitavanje dimenzija matrice b i provera ispravnosti ulaza. */
printf("Unesite broj vrsta i broj kolona matrice b: ");
scanf("%d%d", &k, &t);
if (k <= 0 || k > MAKS || t <= 0 || t > MAKS) {
    printf("Greska: neispravan unos.\n");
    exit(EXIT_FAILURE);
}

/* Provera da li se odgovarajuće dimenzije matrica poklapaju. */
if (n != k) {
    printf("Mnozenje matrica nije moguce.\n");
    exit(EXIT_FAILURE);
}

/* Ucitavanje elemenata druge matrice. */
ucitaj(b, k, t);

/* Racunanje proizvoda. */
mnozenje(a, m, n, b, k, t, c);

/* Ispis rezultata. */
printf("Rezultat mnozenja je:\n");
ispisi(c, m, t);

exit(EXIT_SUCCESS);
}
```

Rešenje 2.7.20

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

#define MAKS 50

/* Funkcija ucitava elemente matrice dimenzije mxn. */
void ucitaj(double a[][MAKS], int m, int n) {
    int i, j;
    printf("Unesite elemente matrice:\n");
    for (i = 0; i < m; i++)
        for (j = 0; j < n; j++)
            scanf("%lf", &a[i][j]);
}

int main() {
    /* Deklaracije potrebnih promenljivih. */
```

```

17 double a[MAKS][MAKS];
18 int m, n, k, i, j, indeks_kolone;
19 double maks_kolone, min_vrste;
20
21 /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
22 printf("Unesite broj vrsta i broj kolona matrice: ");
23 scanf("%d%d", &m, &n);
24 if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
25     printf("Greska: neispravan unos.\n");
26     exit(EXIT_FAILURE);
27 }
28
29 /* Ucitavanje elemenata matrice. */
30 ucitaj(a, m, n);
31
32 /* Pronalazak elemenata koji su sedlo. */
33 for (i = 0; i < m; i++) {
34     /* Pronalazi se najmanji element u tekućoj vrsti. Pamti se
35      kolona kojoj taj element pripada. */
36     min_vrste = a[i][0];
37     indeks_kolone = 0;
38
39     for (j = 1; j < n; j++)
40         if (a[i][j] < min_vrste) {
41             min_vrste = a[i][j];
42             indeks_kolone = j;
43         }
44
45     /* Pronalazi se najveći element u zapamćenoj koloni. */
46     maks_kolone = a[0][indeks_kolone];
47
48     for (k = 1; k < m; k++)
49         if (a[k][indeks_kolone] > maks_kolone)
50             maks_kolone = a[k][indeks_kolone];
51
52     /* Element je sedlo ukoliko je on istovremeno najmanji u svojoj
53      vrsti i najveći u svojoj koloni. */
54     if (min_vrste == maks_kolone)
55         printf("Sedlo: %d %d %g\n", i, indeks_kolone, min_vrste);
56 }
57
58 exit(EXIT_SUCCESS);
59 }

```

Rešenje 2.7.21

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS 50
5

```

```

1  /* Funkcija ucitava elemente matrice dimenzije mxn. */
2  void ucitaj(int a[][][MAKS], int m, int n) {
3      int i, j;
4      printf("Unesite elemente matrice:\n");
5      for (i = 0; i < m; i++)
6          for (j = 0; j < n; j++)
7              scanf("%d", &a[i][j]);
8
9
10
11
12
13
14
15 int main() {
16     /* Deklaracije potrebnih promenljivih. */
17     int a[MAKS][MAKS];
18     int m, n, brojac, i, j, pravac;
19     int gornja_granica, donja_granica, leva_granica, desna_granica;
20
21     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
22     printf("Unesite broj vrsta i broj kolona matrice: ");
23     scanf("%d%d", &m, &n);
24     if (n <= 0 || n > MAKST || m <= 0 || m > MAKST) {
25         printf("Greska: neispravan unos.\n");
26         exit(EXIT_FAILURE);
27     }
28
29     /* Ucitavanje elemenata matrice. */
30     ucitaj(a, m, n);
31
32     /* Ciklicni ispis elemenata matrice:
33      Npr. za n=4, indeksi elemenata u matrici su:
34      (0,0) (0,1) (0,2) (0,3)
35      (1,0) (1,1) (1,2) (1,3)
36      (2,0) (2,1) (2,2) (2,3)
37      (3,0) (3,1) (3,2) (3,3)
38      Ispis treba da ide sledecim redosledom:
39      1. krece se sa leva na desno (0,0) (0,1) (0,2) (0,3)
40      2. zatim se ide na dole (1,3) (2,3) (3,3)
41      3. zatim na levo (3,2) (3,1) (3,0)
42      4. zatim na gore (2,0) (1,0) (ovde se staje jer je (0,0) vec
43      ispisano) i prelazi se opet na levo. Koraci 1-4 se ponavljaju
44      dok god se ne ispisu svi elementi. Ideja je da kada se ispisu
45      elementi prve vrste (kada se ide sa leva na desno), da se
46      pomeri "gornja granica ispisa" za 1, kako bi se naznacilo da
47      je taj red vec ispisano. Slicno, kada se vrsi ispis odozgo na
48      dole, uspesno je ispisana jedna kolona pa je potrebno pomeriti
49      "desnu granicu ispisa" za jedan u levo. Kada se ispise jedna
50      vrsta sa desna na levo, vrsi se pomeranje donje granice ispisa
51      za jedan na gore. Slicno, kada se ispise jedna kolona odozdo na
52      gore, pomera se leva granica ispisa za jedan u desno. */
53     gornja_granica = 0;
54     donja_granica = m - 1;
55     leva_granica = 0;
56     desna_granica = n - 1;
57

```

```

 59  /* Promenljiva pravac govori u kom smeru ispis ide. */
60  pravac = 1;

61  /* Promenljive i i j su indeksi elementa koji se ispisuje. */
62  i = 0;
63  j = 0;

64  printf("Ispis je:\n");
65  for (brojac = 0; brojac < m * n; brojac++) {
66      printf("%d ", a[i][j]);

67      switch (pravac) {
68          /* Ako je pravac = 1, trenutni smer ispisa je sa leva na
69          /* desno. */
70          case 1:
71              /* Ako je ispisan element na desnoj granici, onda se menja
72              /* pravac ispisa. */
73              if (j == desna_granica) {
74                  /* Prelazi se na pravac odozgo na dole. */
75                  pravac = 2;
76                  /* Pomera se gornja granica za jedan na dole. */
77                  gornja_granica++;
78                  /* Pomera se vrednost vrste za jedan na dole. */
79                  i++;
80              } else {
81                  /* Ako jos uvek nije ispisan element na desnoj granici,
82                  /* vrsti se pomeranje na sledeci element u trenutnoj vrsti. */
83                  j++;
84              }
85          break;

86          /* Ako je pravac = 2, trenutni smer ispisa je odozgo na dole.
87          /* Slicno kao i u prethodnom slučaju, ako se dodje do donje
88          /* granice, menja se pravac i pomera se desna granica za
89          /* jedno mesto u levo. U suprotnom se samo prelazi na narednu
90          /* vrstu. */
91          case 2:
92              if (i == donja_granica) {
93                  pravac = 3;
94                  desna_granica--;
95                  j--;
96              } else {
97                  i++;
98              }
99          break;

100         /* Ako je pravac = 3, trenutni smer ispisa je sa desna na
101         /* levo. Slicno kao i u prethodnom slučaju, ako se dodje do
102         /* leve granice, menja se pravac i pomera se donja granica za
103         /* jedno mesto na gore. U suprotnom se samo prelazi na
104         /* narednu kolonu. */
105         case 3:
106
107
108
109

```

2 Napredni tipovi podataka

```
111     if (j == leva_granica) {
112         pravac = 4;
113         donja_granica--;
114         i--;
115     } else {
116         j--;
117     }
118     break;

119     /* Ako je pravac = 4, trenutni smer ispisa je odozdo na gore.
120      Slicno kao i u prethodnim slucajevima, ako se dodje do
121      gornje granice, menja se pravac i pomera se leva granica
122      za jedno mesto u desno. U suprotnom se samo prelazi na
123      narednu vrstu. */
124
125     case 4:
126         if (i == gornja_granica) {
127             pravac = 1;
128             leva_granica++;
129             j++;
130         } else {
131             i--;
132         }
133     }
134     printf("\n");
135
136     exit(EXIT_SUCCESS);
137 }
```

Rešenje 2.7.22

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

4 #define MAKS 50

6 /* Funkcija ucitava elemente matrice dimenzije mxn. */
7 void ucitaj(int a[][MAKS], int m, int n) {
8     int i, j;
9     printf("Unesite elemente matrice:\n");
10    for (i = 0; i < m; i++)
11        for (j = 0; j < n; j++)
12            scanf("%d", &a[i][j]);
13
14    /* Funkcija proverava da li je matrica b podmatrica matrice a i
15       vraca jedinicu ukoliko jeste, a nulu inace. */
16    int podmatrica(int a[][MAKS], int m, int n,
17                   int b[][MAKS], int k, int t) {
18        int i, j, x, y;
19        int jeste_podmatrica;
```

```

22   for (i = 0; i <= m - k; i++) {
23     for (j = 0; j <= n - t; j++) {
24       /* Za svaku poziciju (i,j) se proverava da li je podmatrica
25          dimenzije k*t ciji je gornji levi ugao a[i][j] jednaka
26          matrici b. */
27       jeste_podmatrica = 1;
28       for (x = 0; x < k && jeste_podmatrica; x++)
29         for (y = 0; y < t && jeste_podmatrica; y++)
30           if (a[i + x][j + y] != b[x][y])
31             jeste_podmatrica = 0;
32
33       if (jeste_podmatrica)
34         return 1;
35     }
36   }
37
38   return 0;
39 }
40
int main() {
41   /* Deklaracije potrebnih promenljivih. */
42   int a[MAKS][MAKS], b[MAKS][MAKS];
43   int m, n, k, t;
44
45   /* Ucitavanje dimenzije matrice A i provera ispravnosti ulaza. */
46   printf("Unesite broj vrsta i broj kolona matrice A: ");
47   scanf("%d%d", &m, &n);
48   if (n <= 0 || n > MAKS || m <= 0 || m > MAKS) {
49     printf("Greska: neispravan unos.\n");
50     exit(EXIT_FAILURE);
51   }
52
53   /* Ucitavanje elemenata prve matrice. */
54   ucitaj(a, m, n);
55
56   /* Ucitavanje dimenzije matrice B i provera ispravnosti ulaza. */
57   printf("Unesite broj vrsta i broj kolona matrice B: ");
58   scanf("%d%d", &k, &t);
59   if (k <= 0 || k > MAKS || t <= 0 || t > MAKS) {
60     printf("Greska: neispravan unos.\n");
61     exit(EXIT_FAILURE);
62   }
63
64   /* Ucitavanje elemenata druge matrice. */
65   ucitaj(b, k, t);
66
67   /* Ispis rezultata. */
68   if (podmatrica(a, m, n, b, k, t))
69     printf("Druga matrica je sadrzana u prvoj matrici.\n");
70   else
71     printf("Druga matrica nije sadrzana u prvoj matrici.\n");
72 }
```

2 Napredni tipovi podataka

```
74     exit(EXIT_SUCCESS);  
}
```

2.9 Strukture

Zadatak 2.9.1 Definisati strukturu kojom se opisuje kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja. Napisati program koji za učitana dva kompleksna broja ispisuje vrednost zbira, razlike, proizvoda i količnika. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

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 2.9.1]

Zadatak 2.9.2 Definisati strukturu kojom se opisuje razlomak. Napisati funkcije koje izračunavaju zbir i proizvod dva razlomka. Napisati program koji za uneti ceo broj n i unetih n razlomaka ispisuje njihov ukupan zbir i proizvod. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj razlomaka: 5
Unesite razlomke:
1 2
7 8
3 4
5 6
2 9
Suma svih razlomaka: 229/72
Proizvod svih razlomaka: 35/576
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj razlomaka: 10
Unesite razlomke:
4 3
12 25
3 8
1 3
8 9
2 3
5 6
-24 50
7 18
-7 19
Suma svih razlomaka: 6089/1368
Proizvod svih razlomaka: 1568/577125
```

[Rešenje 2.9.2]

Zadatak 2.9.3 Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vocka* sadrži ime voćke i količinu vitamina C u miligramima (realan broj). Napisati funkcije:

2 Napredni tipovi podataka

- (a) `int ucitaj(Vocka niz[])` koja učitava voćke sa standardnog ulaza sve do kraja ulaza i kao povratnu vrednost vraća broj učitanih voćki;
- (b) `Vocka vocka_sa_najvise_vitamina(Vocka niz[], int n)` koja pronađe voćku koja ima najviše vitamina C.

Napisati program koji učitava podatke o voćkama i ispisuje ime voćke sa najviše vitamina C. Pretpostaviti da broj voćki neće biti veći od 50, kao i da je ime voćke niska od najviše 20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite ime voćke i njenu kolicinu vitamina C: jabuka 4.6  
Unesite ime voćke i njenu kolicinu vitamina C: limun 83.5  
Unesite ime voćke i njenu kolicinu vitamina C: kivi 71  
Unesite ime voćke i njenu kolicinu vitamina C: banana 8.7  
Unesite ime voćke i njenu kolicinu vitamina C: pomorandza 70.8  
Unesite ime voćke i njenu kolicinu vitamina C:  
Voće sa najviše vitamina C je: limun
```

[Rešenje 2.9.3]

Zadatak 2.9.4 Definisati strukturu `Grad` koja sadrži ime grada i njegovu prosečnu temperaturu u toku decembra. Napisati funkcije:

- (a) `void ucitaj(Grad gradovi[], int n)` koja učitava sa standardnog ulaza podatke o n gradova.
- (b) `void ispisi(Grad gradovi[], int n)` koja ispisuje podatke o gradovima koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni celzijusa.

Napisati program koji učitava imena n gradova i njihove prosečne temperature, a zatim ispisuje imena gradova sa idealnom temperaturom za klizanje. Pretpostaviti da je maksimalan broj gradova 50 i da je maksimalna dužina imena grada 20 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

[Rešenje 2.9.4]

Zadatak 2.9.5 Definisati strukturu `ParReci` koja sadrži reč na srpskom jeziku i odgovarajući prevod na engleski jezik. Napisati program koji do kraja ulaza učitava sve parove reči, a potom za rečenicu koja se zadaće u jednoj liniji ispisuje prevod. Ako je reč u rečenici nepoznata umesto nje ispisati nisku zvezdica čija dužina odgovara dužini nepoznate reči. Pretpostaviti da je maksimalna dužina reči 50 karaktera, maksimalan broj parova reči 100, a maksimalna dužina rečenice 100 karaktera.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite reci i njihove prevode:
je is
zima winter
pas dog
sreca happiness
prijatelj friend
solja cup
covek man
Unesite recenicu za prevod:
pas je covekov najbolji prijatelj
dog is ***** ***** friend
```

[Rešenje 2.9.5]

Zadatak 2.9.6 Statistički zavod Srbije istražuje kako rade obdaništa u Srbiji. Za svako obdanište poznat je spisak koji sadrži broj dece u grupi, a zatim i ocene koje je svako dete dalo o radu obdaništa. Definisati strukturu `Dete` koja sadrži polja pol (`m` ili `z`), broj godina (od 3 do 6) i ocenu koju je dete dalo radu

2 Napredni tipovi podataka

obdaništa (od 1 do 5). Napisati program koji učitava broj dece u grupi, a zatim i informacije o svakom detetu. Ispisati, na tri decimale, prosečnu ocenu koje je obdanište dobilo od dece sa unetim polom i brojem godina. Prepostaviti da je maksimalan broj dece u obdaništu 200. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj dece u grupi: 5  
Unesite podatke za svako dete (pol,  
broj godina i ocenu):  
m 3 5  
z 3 4  
m 4 2  
m 5 4  
m 3 4  
Unesite pol i broj godina za  
statistiku: m 3  
Prosečna ocena je: 4.500.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj dece u grupi: 10  
Unesite podatke za svako dete (pol,  
broj godina i ocenu):  
m 3 5  
z 4 4  
m 5 4  
z 4 3  
z 3 2  
z 4 5  
m 6 5  
z 4 4  
z 4 5  
m 6 3  
Unesite pol i broj godina za  
statistiku: z 4  
Prosečna ocena je: 4.200.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj dece u grupi: 15  
Unesite podatke za svako dete (pol,  
broj godina i ocenu):  
m 3 2  
z 7 5  
Greska: neispravan broj godina.
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj dece u grupi: 2  
Unesite podatke za svako dete (pol,  
broj godina i ocenu):  
m 3 2  
z 3 5  
Unesite pol i broj godina za  
statistiku: h 5  
Greska: neispravan pol.
```

[Rešenje 2.9.6]

Zadatak 2.9.7 Definisati strukturu kojom se opisuje student. Student se opisuje svojim imenom i prezimenom, smerom (R, I, V, N, T, M) i prosečnom ocenom. Napisati program koji učitava podatke o n studenata, a zatim i informaciju o smeru i ispisuje imena i prezimena onih studenata koji su sa datog smera, kao i podatke studenata koji ima najveći prosek. Ako ima više takvih studenata ispisati podatke o svima. Prepostaviti da je maksimalan broj studenata 2000, a maksimalna dužina imena i prezimena po 30 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj studenata: 5
Unesite podatke o studentima:
0. student: Kocic Marija R 9.14
1. student: Tanja Mratinkovic R 7.88
2. student: Mihailo Simic N 8.44
3. student: Milena Medar I 9.14
4. student: Ljubica Mihic N 9.00
Unesite smer: R
Studenti sa R smera:
Kocic Marija
Tanja Mratinkovic
-----
Svi studenti koji imaju maksimalni prosek:
Kocic Marija, R, 9.14
Milena Medar, I, 9.14

```

Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj studenata: 4
Unesite podatke o studentima:
0. student: Djordje Lazarevic N 9.05
1. student: Minja Peric W 7.70
Greska: neispravan unos smera.

```

[Rešenje 2.9.7]

Zadatak 2.9.8 Definisati strukturu Djak koja sadrži ime đaka i 9 ocena (ocene su celi brojevi od 1 do 5). Napisati program koji učitava podatke o đacima sve do kraja ulaza i na standardni izlaz ispisuje prvo imena nedovoljnih đaka, a zatim imena odličnih đaka. Đak je nedovoljan ako ima barem jednu jedinicu, a odličan ako ima prosek ocena veći ili jednak 4.5. Prepostaviti da je maksimalna dužina imena đaka 20 karaktera, kao i da je maksimalan broj đaka 30. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite podatke o djaku:
Maja 4 5 2 3 4 4 3 3 4
Unesite podatke o djaku:
Nikola 5 4 5 5 5 4 4 5 5
Unesite podatke o djaku:
Jasmina 2 2 1 1 2 3 3 1 3
Unesite podatke o djaku:
Pera 5 4 5 3 5 5 1 5 5
Unesite podatke o djaku:
Pavle 4 3 2 4 3 2 4 3 2
Unesite podatke o djaku:

NEDOVOLJNI: Jasmina Pera
ODLICNI: Nikola

```

Primer 2

```

INTERAKCIJA SA PROGRAMOM:
Unesite podatke o djaku:
Uros 3 4 2 3 4 2 3 4 4
Unesite podatke o djaku:
Nebojsa 4 5 5 5 4 5 5 5 5
Unesite podatke o djaku:
Sreten 2 3 2 4 5 4 4 4 2
Unesite podatke o djaku:

NEDOVOLJNI:
ODLICNI: Nebojsa

```

Primer 3

```

INTERAKCIJA SA PROGRAMOM:
Unesite podatke o djaku:
Mirko 2 3 4 4 4 3 3 3 4
Unesite podatke o djaku:
Mihailo 2 3 10 5 5 2 3 4 2
Greska: neispravna ocena.

```

[Rešenje 2.9.8]

2 Napredni tipovi podataka

Zadatak 2.9.9 Definisati strukturu **Osoba** kojom se opisuje jedan unos u imenik. Za svaku osobu su dati podaci: ime, prezime i imejl adresa. Napisati program koji učitava ceo broj n , a zatim podatke o n osoba. Ispisati imena i prezimena svih osoba koje imaju imejl adresu koja se završava sa @gmail.com. Prepostaviti da je maksimalan broj osoba 50, kao i da je maksimalna dužina imena osobe 20 karaktera, prezimena 30 karaktera, a imejl adrese 50 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci. NAPOMENA: Može se smatrati da je svaka imejl adresa dobro zadata i sadrži samo jedno pojavljivanje znaka @.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj osoba: 3  
Unesite podatke o osobama  
(ime, prezime i imejl adresu):  
Dusko Dugousko dusk0@yahoo.com  
Pink Panter panter@gmail.com  
Pera Detlic pd@gmail.com  
Vlasnici gmail naloga su:  
Pink Panter  
Pera Detlic
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj osoba: 3  
Unesite podatke o osobama  
(ime, prezime i imejl adresu):  
Homer Simpson homer@yahoo.com  
Mardz Simpson mardz@matf.bg.ac.rs  
Nema vlasnika gmail naloga.
```

[Rešenje 2.9.9]

* **Zadatak 2.9.10** Napisati program koji izračunava prosečnu cenu jedne potrošačke korpe. Potrošačka korpa se sastoji od broja kupljenih artikala i niza kupljenih artikala. Svaki artikal određen je svojim nazivom, količinom i cenom. Program treba da učita broj potrošača n , zatim podatke za n potrošačkih korpi i da na osnovu učitanih podataka izračuna prosečnu cenu potrošačke korpe. Program ispisuje na dve decimale račune svake potrošačke korpe i na kraju ispisuje prosečnu cenu potrošačke korpe. Prepostaviti da je maksimalan broj potrošačkih korpi 100, maksimalan broj artikala u korpi 20 i da naziv svakog artikla sadrži maksimalno 30 karaktera. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```

INTERAKCIJA SA PROGRAMOM:
Unesite broj potrosackih korpi: 3
Unesite podatke o korpi:
Broj artikala: 4
Unesite artikal (naziv, kolicinu i cenu): jabuke 10 22.4
Unesite artikal (naziv, kolicinu i cenu): dezodorans 1 120.99
Unesite artikal (naziv, kolicinu i cenu): C_supra 3 36.56
Unesite artikal (naziv, kolicinu i cenu): sunka 1 230.99
Unesite podatke o korpi:
Broj artikala: 2
Unesite artikal (naziv, kolicinu i cenu): Jafa_keks 1 55.78
Unesite artikal (naziv, kolicinu i cenu): Najlepse_zelje 1 62.99
Unesite podatke o korpi:
Broj artikala: 3
Unesite artikal (naziv, kolicinu i cenu): prasak_za_ves 1 1199.99
Unesite artikal (naziv, kolicinu i cenu): omeksivac 1 279.99
Unesite artikal (naziv, kolicinu i cenu): protiv_kamenca 1 699.99

Korpa 0:
    jabuke 10 22.40
    dezodorans 1 120.99
    C_supra 3 36.56
    sunka 1 230.99
-----
    ukupno: 685.66

Korpa 1:
    Jafa_keks 1 55.78
    Najlepse_zelje 1 62.99
-----
    ukupno: 118.77

Korpa 2:
    prasak_za_ves 1 1199.99
    omeksivac 1 279.99
    protiv_kamenca 1 699.99
-----
    ukupno: 2179.97

Prosecna cena potrosacke korpe: 994.80

```

[Rešenje 2.9.10]

Zadatak 2.9.11 Definisati strukturu Lopta sa poljima poluprecnik (ceo broj u centimetrima) i boja (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Napisati funkcije:

- (a) void ucitaj(Lopta niz[], int n) koja učitava podatke o n lopti u niz.
- (b) double ukupna_zapremina(Lopta niz[], int n) koja računa ukupnu zapreminu svih lopti.

2 Napredni tipovi podataka

- (c) `int broj_crvenih(Lopta niz[], int n)` koja prebrojava koliko ima crvenih lopti u nizu.

Napisati program koji učitava informacije o n lopti i ispisuje ukupnu zapreminu i broj crvenih lopti. Pretpostaviti da je maksimalan broj lopti 50. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj lopti: 8  
Unesite poluprecnike i boje lopti  
(1-plava, 2-zuta, 3-crvena, 4-zelena):  
1. lopta: 1 2  
2. lopta: 2 10  
Greska: neispravan unos.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj lopti: 8  
Unesite poluprecnike i boje lopti  
(1-plava, 2-zuta, 3-crvena, 4-zelena):  
1. lopta: 2 1  
2. lopta: 30 3  
3. lopta: 7 3  
4. lopta: 4 1  
5. lopta: 5 2  
6. lopta: 6 2  
7. lopta: 12 3  
8. lopta: 14 2  
Ukupna zapremina: 134996.34  
Ukupno crvenih lopti: 3
```

[Rešenje 2.9.11]

Zadatak 2.9.12 Napisati program za predstavljanje poligona i izračunavanje dužine njegovih stranica i obima.

- Definisati strukturu `Tacka` kojom se opisuje tačka dekartovske ravni čije su x i y koordinate podaci tipa `double`.
- Definisati funkciju `double rastojanje(const Tacka *A, const Tacka *B)` koja izračunava rastojanje između dve tačke.
- Definisati funkciju `int ucitaj_poligon(Tacka poligon[], int n)` koja učitava maksimalno n puta po dve vrednosti tipa `double` (koje predstavljaju koordinate temena poligona) i upisuje ih u zadati niz tačaka. Funkcija vraća broj uspešno učitanih tačaka.

- (d) Definisati funkciju double `obim_poligona(Tacka poligon[], int n)` koja izračunava obim poligona sa n temena u zadatom nizu. UPUTSTVO: *Prilikom računanja obima ne zaboraviti stranicu koja spaja poslednje i prvo teme.*
- (e) Definisati funkciju double `maksimalna_stranica(Tacka poligon[], int n)` koja izračunava dužinu najduže stranice poligona sa n temena u zadatom nizu.
- (f) Napisati funkciju double `povrsina_trougla(const Tacka *A, const Tacka *B, const Tacka *C)` koja izračunava površinu trougla čija su temena A, B i C.
- (g) Napisati funkciju double `povrsina_poligona(Tacka poligon[], int n)` koja izračunava površinu konveksnog poligona. UPUTSTVO: *Zadatak se može rešiti podelom poligona na trouglove i korišćenjem funkcije `povrsina_trougla`.*

Napisati program koji učitava poligon sa maksimalno n temena i za učitani poligon ispisuje na tri decimale obim, dužinu najduže stranice i površinu. Prepostaviti da je uneti poligon konveksan. Poligon mora imati barem tri temena. Prepostaviti da je maksimalan broj temena 1000. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite maksimalan broj temena poligona: 10
Unesite temena poligona:
0 0
0 6
3 3
Obim poligona je 14.485.
Duzina maksimalne stranice je 6.000.
Povrsina poligona je 9.000.
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite maksimalan broj temena poligona: 4
0 0
Greska: poligon mora imati bar tri tacke.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite maksimalan broj temena poligona: 10
Unesite temena poligona:
0 0
12 0
13 2
16 5
20 10
18 15
15 20
10 20
8 15
3 4
Obim poligona je 63.566.
Duzina maksimalne stranice je 12.083.
Povrsina poligona je 247.500.
```

[Rešenje 2.9.12]

*** Zadatak 2.9.13** Definisati strukturu `Izraz` kojom se opisuje numerički izraz nad celim brojevima koji se sastoji od dva celobrojna operanda i numeričke operacije (sabiranje, oduzimanje, množenje ili celobrojno deljenje).

2 Napredni tipovi podataka

- (a) Napisati funkciju `int korektan_izraz(const Izraz *izraz)` koja ispituje da li je dati izraz korektno zadat i vraća jedinicu ako jeste, a nulu inače. Podrazumeva se da je izraz korektno zadat ako je operacija `+`, `-`, `*` ili `/` i u slučaju deljenja drugi operand je različit od 0.
- (b) Napisati funkciju `int vrednost(const Izraz *izraz)` koja za dati izraz određuje vrednost izraza.
- (c) Napisati funkciju `void ucitaj(Izraz izrazi[], int n)` koja učitava izraze. Funkcija treba da učita sa standardnog ulaza n izraza koji su zadati prefiksno — prvo operacija, a potom dva operanda.

Napisati program koji učitava prirodan broj n , a zatim n izraza u prefiksnoj notaciji. Program treba da ispiše maksimalnu vrednost unetih izraza i sve izraze čija vrednost je manja od polovine maksimalne vrednosti. Prepostaviti da je maksimalan broj izraza 100. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj izraza: 4  
Unesite izraze u prefiksnoj notaciji:  
+ 10 4  
- 9 2  
* 11 2  
/ 7 3  
Maksimalna vrednost izraza: 22  
Izrazi cija je vrednost manja  
od polovine maksimalne vrednosti:  
9 - 2 = 7  
7 / 3 = 2
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj izraza: 3  
Unesite izraze u prefiksnoj notaciji:  
* 1 2  
/ 3 0  
Greska: deljenje nulom.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite broj izraza: 10  
Unesite izraze u prefiksnoj notaciji:  
+ 10 2  
- -678 34  
* 77 2  
+ 1000 -23  
+ 102 4  
- 200 23  
/ 67 12  
/ 1000 2  
* 44 6  
/ 13 1  
Maksimalna vrednost izraza: 977  
Izrazi cija je vrednost manja  
od polovine maksimalne vrednosti:  
10 + 2 = 12  
-678 - 34 = -712  
77 * 2 = 154  
102 + 4 = 106  
200 - 23 = 177  
67 / 12 = 5  
44 * 6 = 264  
13 / 1 = 13
```

[Rešenje 2.9.13]

* **Zadatak 2.9.14** Definisati strukturu kojom se opisuje polinom. Polinom je dat svojim stepenom i realnim koeficijentima.

- (a) Napisati funkciju `int ucitaj(Polinom niz[])` koja sa standardnog ulaza učitava polinome sve do kraja ulaza. Polinomi su zadati stepenom i koeficijentima počevši od slobodnog člana. Funkcija kao povratnu vrednost vraća broj učitanih polinoma.
- (b) Napisati funkciju `void ispis(const Polinom *p)` koja ispisuje polinom stepena n sa koeficijentima k_0, k_1, \dots, k_n u obliku $k_0 \pm k_1 * x \pm k_2 * x^2 \pm k_3 * x^3 \pm \dots \pm k_n * x^n$. Na mesto znaka \pm zapisati odgovarajući znak, $+$ ili $-$, u zavisnosti od znaka odgovarajućeg koeficijenta. Koeficijente ispisivati na dve decimale. Koeficijente koji su jednaki 0 ne ispisivati.
- (c) Napisati funkciju `void integral(const Polinom *p, Polinom *tekuci_integral)` koja za dati polinom p određuje njegov integral $tekuci_integral$. Za vrednost slobodnog člana integrala uzeti vrednost 0.

Napisati program koji učitava polinome do kraja ulaza i za svaki učitani polinom određuje i ispisuje njegov integral. Prepostaviti da je maksimalan broj polinoma 100, a maksimalan stepen polinoma 10. U slučaju neispravnog unosa, ispisati odgovarajuću poruku o grešci.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite stepen: 3
Unesite koeficijente polinoma:
1 0 3 1
Unesite stepen: 4
Unesite koeficijente polinoma:
7 9 4 0 4
Unesite stepen:
Integrali su:
1.00*x + 1.00*x^3 + 0.25*x^4
7.00*x + 4.50*x^2 + 1.33*x^3 + 0.80*x^5
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite stepen: 3
Unesite koeficijente polinoma:
1 0 -4 1
Unesite stepen: 2
Unesite koeficijente polinoma:
1 2 -3
Unesite stepen: 1
Unesite koeficijente polinoma:
0 -1
Unesite stepen:
Integrali su:
1.00*x - 1.33*x^3 + 0.25*x^4
1.00*x + 1.00*x^2 - 1.00*x^3
-0.50*x^2
```

[Rešenje 2.9.14]

2.10 Rešenja

Rešenje 2.9.1

```
1 #include <stdio.h>
2 #include <math.h>
3
4 /* Struktura koja opisuje kompleksni broj. */
5 typedef struct {
6     float re;
7     float im;
8 } KompleksanBroj;
9
10 /* Kada se neka promenljiva zadaje kao argument funkcije, obicno se
11    prenosi po vrednosti (bez pokazivaca), ako se ona nece menjati u
12    funkciji ili po adresi (preko pokazivaca), ako ce se njena
13    vrednost promeniti u funkciji.
14
15 Prilikom poziva funkcije, za svaki argument funkcije kreira se
16    promenljiva koja predstavlja lokalnu kopiju argumenta i koja
17    prestaje da postoji po zavrsetku funkcije. S obzirom da se
18    strukture sastoje od vise polja, zauzimaju vise memorije nego
19    nestrukturne promenljive. Zbog toga je za njihovo kopiranje
20    potrebno vise vremena i vise memorijskih resursa nego za
21    kopiranje nestrukturnih promenljivih.
22
23 Da bi program bio efikasniji, korisno je da se struktura uvek
24    prenosi po adresi (preko pokazivaca), bez obzira da li ce se
25    ona u toj funkciji menjati ili ne. Pokazivac na strukturu
26    zauzima manje memorije nego sama struktura pa je izrada njegove
27    kopije brza, a kopija pokazivaca uzima manji memorijski prostor
28    nego kopija strukture.
29
30 Kada se struktura promenljiva prenosi u funkciju po adresi
31    (preko pokazivaca), tada postoji mogucnost da se njena polja
32    menjaju u funkciji. Ukoliko to nije potrebno, uz argument se
33    dodaje kljucna rec const. Na taj nacin, u slucaju pokusaja
34    izmene strukturalne promenljive koja je prosledjena kao const,
35    kompjajler ce prijaviti gresku. Na ovaj nacin se obezbedjuje da
36    promenljiva koja je preneta po adresi ne bude cak ni slucajno
37    izmenjena u funkciji. */
38
39 /* Funkcija izracunava zbir kompleksnih brojeva. */
40 KompleksanBroj saberi(const KompleksanBroj *a,
41                      const KompleksanBroj *b) {
42
43     KompleksanBroj c;
44     c.re = a->re + b->re;
45     c.im = a->im + b->im;
46
47     return c;
48 }
```

```

/* Funkcija izracunava razliku kompleksnih brojeva. */
49 KompleksanBroj oduzmi(const KompleksanBroj *a,
                         const KompleksanBroj *b) {
51     KompleksanBroj c;
52     c.re = a->re - b->re;
53     c.im = a->im - b->im;
54     return c;
55 }

57 /* Funkcija izracunava proizvod kompleksnih brojeva. */
58 KompleksanBroj pomnozi(const KompleksanBroj *a,
                           const KompleksanBroj *b) {
59     KompleksanBroj c;
60     c.re = a->re * b->re - a->im * b->im;
61     c.im = b->re * a->im + a->re * b->im;
62     return c;
63 }

65 /* Funkcija izracunava kolicnik kompleksnih brojeva. */
66 KompleksanBroj podeli(const KompleksanBroj *a,
                        const KompleksanBroj *b,
68                     int *postoji_kolicnik) {
69     KompleksanBroj c;
70
71     if (b->re != 0 || b->im != 0) {
72         c.re = (a->re * b->re + a->im * b->im) /
73             (b->re * b->re + b->im * b->im);
74         c.im = (b->re * a->im - a->re * b->im) /
75             (b->re * b->re + b->im * b->im);
76     } else {
77         printf("Kolicnik ne postoji.\n");
78         *postoji_kolicnik = 0;
79     }
80
81     return c;
82 }

85 /* Funkcija ispisuje kompleksan broj. */
86 void ispisi(const KompleksanBroj *c){
87     /* Ukoliko je imaginarni deo negativan, njegov zapis vec
88      uključuje znak, pa se zato uzima njegova absolutna
89      vrednost. */
90     printf("%.2f%c%.2f*i\n", c->re, c->im > 0 ? '+' : '-',
91           fabs(c->im));
92 }

93 int main() {
94     /* Deklaracije potrebnih promenlivih. */
95     KompleksanBroj a, b, c;
96     int postoji_kolicnik = 1;
97
98     /* Ucitavanje kompleksnih brojeva. */
99

```

2 Napredni tipovi podataka

```
101 printf("Unesite realni i imaginarni deo prvog broja: ");
102 scanf("%f%f", &a.re, &a.im);
103 printf("Unesite realni i imaginarni deo drugog broja: ");
104 scanf("%f%f", &b.re, &b.im);

105 /* Ispis zbiru. */
106 c = saberi(&a, &b);
107 printf("Zbir: ");
108 ispisi(&c);

109 /* Ispis razlike. */
110 c = oduzmi(&a, &b);
111 printf("Razlika: ");
112 ispisi(&c);

113 /* Ispis proizvoda. */
114 c = pomnozi(&a, &b);
115 printf("Proizvod: ");
116 ispisi(&c);

117 /* Ispis kolicnika. */
118 c = podeli(&a, &b, &postoji_kolicnik);
119 if (postoji_kolicnik) {
120     printf("Kolicnik: ");
121     ispisi(&c);
122 }
123
124
125
126
127 return 0;
128 }
```

Rešenje 2.9.2

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Struktura koja opisuje razlomak. */
5 typedef struct {
6     int brojilac;
7     int imenilac;
8 } Razlomak;
9
10 /* Funkcija Euklidovim algoritmom racuna najveci zajednicki delilac
11    brojeva a i b. */
12 int nzd(int a, int b) {
13     int ostatak;
14
15     while (b != 0) {
16         ostatak = a % b;
17         a = b;
18         b = ostatak;
19     }
20 }
```

```

21     return a;
}
23
24 /* Funkcija vraca razlomak koji se dobija deljenjem imenioca i
25    brojioca njihovim najvecim zajednickim deliocem.*/
26 void skrati(Razlomak *r) {
27     int nzd_razlomka = nzd(r->brojilac, r->imenilac);
28     r->brojilac /= nzd_razlomka;
29     r->imenilac /= nzd_razlomka;
}
30
31 /* Funkcija racuna zbir razlomaka a i b. */
32 Razlomak saberi(const Razlomak *a, const Razlomak *b) {
33     Razlomak c;
34
35     c.brojilac = a->brojilac * b->imenilac +
36                 b->brojilac * a->imenilac;
37     c.imenilac = a->imenilac * b->imenilac;
38     skrati(&c);
39
40     return c;
}
41
42
43 /* Funkcija racuna proizvod razlomaka a i b. */
44 Razlomak pomnozi(const Razlomak *a, const Razlomak *b) {
45     Razlomak c;
46
47     c.brojilac = a->brojilac * b->brojilac;
48     c.imenilac = a->imenilac * b->imenilac;
49     skrati(&c);
50
51     return c;
}
52
53
54 int main() {
55     /* Deklaracije potrebnih promenljivih. */
56     int n, i;
57     Razlomak suma, proizvod, r;
58
59     /* Ucitavanje broja razlomaka i provera ispravnosti ulaza. */
60     printf("Unesite broj razlomaka: ");
61     scanf("%d", &n);
62     if (n <= 0) {
63         printf("Greska: neispravan unos.\n");
64         exit(EXIT_FAILURE);
65     }
66
67     /* Inicijalizacija sume i proizvoda. */
68     suma.brojilac = 0;
69     suma.imenilac = 1;
70     proizvod.brojilac = 1;
71 }
```

2 Napredni tipovi podataka

```
    proizvod.imenilac = 1;
73
74    /* Ucitavanje razlomaka i racunanje rezultata. */
75    printf("Unesite razlomke:\n");
76    for (i = 0; i < n; i++) {
77        scanf("%d%d", &r.brojilac, &r.imenilac);
78
79        if (r.imenilac == 0) {
80            printf("Greska: neispravan unos.\n");
81            exit(EXIT_FAILURE);
82        }
83
84        suma = saberi(&suma, &r);
85        proizvod = pomnozi(&proizvod, &r);
86    }
87
88    /* Ispis rezultata. */
89    printf("Suma svih razlomaka: %d/%d\n", suma.brojilac,
90           suma.imenilac);
91    printf("Proizvod svih razlomaka: %d/%d\n", proizvod.brojilac,
92           proizvod.imenilac);
93
94    exit(EXIT_SUCCESS);
95}
```

Rešenje 2.9.3

```
#include <stdio.h>
2 #include <string.h>
3
4 #define MAKS_IME 21
# define MAKS_VOCKI 50
5
6 /* Struktura koja opisuje vocku. */
7 typedef struct {
8     char ime[MAKS_IME];
9     float vitamin;
10 } Vocka;
11
12 /* Funkcija ucitava podatke o vockama u niz struktura. Kao
13    povratnu vrednost vraca broj ucitanih vocki. */
14 int ucitaj(Vocka niz[]) {
15     int i = 0;
16
17     /* Ucitavanje vocki do kraja ulaza ili do popunjavanja niza. */
18     do {
19         printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
20         if (scanf("%s%f", niz[i].ime, &niz[i].vitamin) == EOF)
21             break;
22
23         i++;
24     }
```

```

26     } while (i < MAKS_VOCKI);
27
28 }
29
30 /* Funkcija pronalazi vocku sa najvise vitamina C. */
31 Vocka vocka_sa_najvise_vitamina(Vocka niz[], int n) {
32     /* Pronalazak pozicije vocke sa najvise vitamina C. */
33     int maks_i = 0, i;
34     for (i = 1; i < n; i++)
35         if (niz[i].vitamin > niz[maks_i].vitamin)
36             maks_i = i;
37
38     /* Kao povratna vrednost se vraca vocka na poziciji maks_i. */
39     return niz[maks_i];
40 }
41
42 int main() {
43     /* Deklaracije potrebnih promenljivih. */
44     Vocka vocke[MAKS_VOCKI], najzdravija;
45     int n;
46
47     /* Ucitavanje ulaza. */
48     n = ucitaj(vocke);
49
50     /* Ispis rezultata. */
51     najzdravija = vocka_sa_najvise_vitamina(vocke, n);
52     printf("Voca sa najvise vitamina C je: %s\n", najzdravija.ime);
53
54     return 0;
55 }
```

Rešenje 2.9.4 Pogledajte zadatak 2.9.3.

Rešenje 2.9.5

```

1 #include <stdio.h>
2 #include <string.h>
3
4 #define MAKS_REC 21
5 #define MAKS_BROJ_REC 100
6
7 /* Struktura koja opisuje par reci. */
8 typedef struct {
9     char sr[MAKS_REC];
10    char en[MAKS_REC];
11} ParReci;
12
13 /* Funkcija ucitava parove reci u recnik. */
14 int ucitaj(ParReci recnik[]) {
15     int i = 0;
```

2 Napredni tipovi podataka

```
16     char sr[MAKS_REC], en[MAKS_REC];  
  
18     /* Ucitavanje parovi reci sa standardnog ulaza sve do kraja  
19      ulaza. */  
20     printf("Unesite reci i njihove prevode:\n");  
21     while (scanf("%s %s", sr, en) != EOF) {  
22         if (i == MAKS_BROJ_REC) {  
23             break;  
24         }  
25         strcpy(recnik[i].sr, sr);  
26         strcpy(recnik[i].en, en);  
27         i++;  
28     }  
29     return i;  
30 }  
  
34 /*  
35  Funkcija u recniku koji sadrzi n reci trazi prevod reci rec i  
36  upisuje ga u prevod. Ukoliko se rec ne nalazi u recniku, prevod  
37  se sastoji od zvezdica pri cemu broj zvezdica odgovara duzini  
38  nepoznate reci. */  
39 void pronadji_prevod(ParReci recnik[], int n, char rec[],  
40                      char prevod[]) {  
41     int i;  
42     /* Pretraga reci. */  
43     for (i = 0; i < n; i++) {  
44         if (strcmp(recnik[i].sr, rec) == 0) {  
45             strcpy(prevod, recnik[i].en);  
46             return;  
47         }  
48     }  
49     /* Ukoliko rec nije pronadjena, formira se prevod reci koji se  
50      sastoji od zvezdica. */  
51     for (i = 0; rec[i]; i++)  
52         prevod[i] = '*';  
53     prevod[i] = '\0';  
54 }  
  
58 int main() {  
59     /* Deklaracije potrebnih promenljivih. */  
60     ParReci recnik[MAKS_BROJ_REC];  
61     int n;  
62     char rec[MAKS_REC], prevod[MAKS_REC];  
63     char c;  
64     /* Ucitavanje parova reci u recnik. */  
65     n = ucitaj(recnik);  
66 }
```

```

68  /* Ucitavanje recenice i ispis njenog prevoda. */
69  printf("Unesite recenicu za prevod: \n");
70  do {
71      /* Ucitava se rec po rec date recenice i pronalazi se njen
72          prevod. */
73      scanf("%s", rec);
74      pronadji_prevod(recnik, n, rec, prevod);
75      printf("%s ", prevod);
76
77      /* Ukoliko je karakter iza reci znak za novi red, onda se
78          prekida sa unosom, a ako nije ucitava se sledeca rec. */
79      c = getchar();
80  } while (c != '\n');
81
82  putchar('\n');
83
84  return 0;
}

```

Rešenje 2.9.6

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS_DECE 200
5
6 /* Struktura koja opisuje dete. */
7 typedef struct {
8     char pol;
9     int broj_godina;
10    int ocena;
11} Dete;
12
13 /* Funkcija ucitava podatke o deci i proverava ispravnost unetih
14   podataka. */
15 void ucitaj(Dete niz[], int n) {
16     char blanko;
17     int i;
18     printf("Unesite podatke za svako dete (pol, broj godina i "
19           "ocenu):\n");
20     for (i = 0; i < n; i++) {
21         scanf("%c%c%d%d", &blanko, &niz[i].pol, &niz[i].broj_godina,
22               &niz[i].ocena);
23
24         /* Provera ispravnosti unosa. */
25         if (niz[i].pol != 'm' && niz[i].pol != 'z') {
26             printf("Greska: neispravan pol.\n");
27             exit(EXIT_FAILURE);
28         }
29         if (niz[i].broj_godina > 6 || niz[i].broj_godina < 3) {
30             printf("Greska: neispravan broj godina.\n");
31         }
32     }
33 }

```

2 Napredni tipovi podataka

```
31         exit(EXIT_FAILURE);
32     }
33     if (niz[i].ocena < 1 || niz[i].ocena > 5) {
34         printf("Greska: neispravna ocena.\n");
35         exit(EXIT_FAILURE);
36     }
37 }
38
39 int main() {
40     /* Deklaracija potrebnih promenljivih. */
41     int n, i, broj_godina;
42     Dete niz[MAKS_DECE];
43     char blanko, pol;
44     int suma, broj_dece;
45
46     /* Ucitavanje broja dece i provera ispravnosti ulaza. */
47     printf("Unesite broj dece u grupi: ");
48     scanf("%d", &n);
49     if (n <= 0 || n > MAKS_DECE) {
50         printf("Greska: neispravan unos.\n");
51         exit(EXIT_FAILURE);
52     }
53
54     /* Ucitavanje podataka o deci. */
55     ucitaj(niz, n);
56
57     /* Ucitavanje trazenih podataka. */
58     printf("Unesite pol i broj godina za statistiku: ");
59     scanf("%c%c%d", &blanko, &pol, &broj_godina);
60
61     /* Provera ispravnosti unetih podataka. */
62     if (pol != 'm' && pol != 'z') {
63         printf("Greska: neispravan pol.\n");
64         exit(EXIT_FAILURE);
65     }
66     if (broj_godina > 6 || broj_godina < 3) {
67         printf("Greska: neispravan broj godina.\n");
68         exit(EXIT_FAILURE);
69     }
70
71     /* Racunanje prosecne ocene dece ciji se pol i broj godina
72      poklapaju sa unetim. */
73     suma = 0;
74     broj_dece = 0;
75     for (i = 0; i < n; i++) {
76         if (niz[i].pol == pol && niz[i].broj_godina == broj_godina) {
77             suma += niz[i].ocena;
78             broj_dece++;
79         }
80
81     /* Ispis rezultata. */
```

```

83     if (broj_dece == 0)
84         printf("Ne postoji deca sa takvima karakteristikama.\n");
85     else
86         printf("Prosecna ocena je: %.3lf.\n",
87                (double) suma / broj_dece);
88
89     exit(EXIT_SUCCESS);
}

```

Rešenje 2.9.7

```

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

4 #define MAKS_STUDENATA 2000
#define MAKS_NISKA 31
6
/* Struktura koja opisuje studenta. */
8 typedef struct Student {
    char ime[MAKS_NISKA];
    char prezime[MAKS_NISKA];
    char smer;
    float prosek;
} Student;
14
/* Funkcija ucitava podatke o studentima u niz. */
16 void ucitaj(Student niz[], int n) {
    int i;
18
    printf("Unesite podatke o studentima:\n");
20    for (i = 0; i < n; i++) {
        printf("%d. student: ", i);
22        scanf("%s %s %c %f", niz[i].ime, niz[i].prezime,
               &niz[i].smer, &niz[i].prosek);
24
        if (niz[i].smer != 'R' && niz[i].smer != 'I' &&
            niz[i].smer != 'V' && niz[i].smer != 'N' &&
            niz[i].smer != 'T' && niz[i].smer != 'O') {
28            printf("Greska: neispravan unos smera.\n");
            exit(EXIT_FAILURE);
        }
    }
32}
34
/* Funkcija ispisuje podatke o studentu. */
void ispisi(const Student *s) {
36    printf("%s %s, %c, %.2f\n", s->ime, s->prezime, s->smer,
           s->prosek);
38}
40
/* Funkcija racuna najveci prosek. */

```

2 Napredni tipovi podataka

```
42     float najveci_prosek(Student studenti[], int n) {
43         float maks_prosek;
44         int i;
45
46         maks_prosek = studenti[0].prosek;
47         for (i = 1; i < n; i++)
48             if (maks_prosek < studenti[i].prosek)
49                 maks_prosek = studenti[i].prosek;
50
51         return maks_prosek;
52     }
53
54     int main() {
55         /* Deklaracija potrebnih promenljivih. */
56         Student studenti[MAKS_STUDENATA];
57         int n, i;
58         float maks_prosek;
59         char smer;
60
61         /* Ucitavanje broja studenata i provera ispravnosti ulaza. */
62         printf("Unesite broj studenata: ");
63         scanf("%d", &n);
64         if (n < 0 || n > MAKS_STUDENATA) {
65             printf("Greska: neispravan unos.\n");
66             exit(EXIT_FAILURE);
67         }
68
69         /* Ucitavanje podataka o studentima. */
70         ucitaj(studenti, n);
71
72         /* Ucitavanje smera. Pre smera se preskace novi red koji je unet
73          nakon podataka o poslednjem studentu. */
74         printf("Unesite smer: ");
75         getchar();
76         scanf("%c", &smer);
77         if (smer != 'R' && smer != 'I' && smer != 'V' && smer != 'N' &&
78             smer != 'T' && smer != 'O') {
79             printf("Greska: neispravan unos smera.\n");
80             exit(EXIT_FAILURE);
81         }
82
83         /* Ispis studenata sa unetog smera. */
84         printf("Studenti sa %c smera:\n", smer);
85         for (i = 0; i < n; i++)
86             if (studenti[i].smer == smer)
87                 printf("%s %s\n", studenti[i].ime, studenti[i].prezime);
88         printf("-----\n");
89
90         /* Racunanje najveceg proseka. */
91         maks_prosek = najveci_prosek(studenti, n);
92
93         /* Ispis svih studenata sa najvecim prosekom. */
```

```

94     printf("Svi studenti koji imaju maksimalni prosek:\n");
95     for (i = 0; i < n; i++)
96         if (studenti[i].prosek == maks_prosek)
97             ispisi(&studenti[i]);
98
99     exit(EXIT_SUCCESS);
}

```

Rešenje 2.9.8

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS_IME 21
5 #define BROJ_OCENA 9
6 #define MAKS_DJAKA 30
7
8 /* Struktura koja opisuje djaka. */
9 typedef struct {
10     char ime[MAKS_IME];
11     int ocena[BROJ_OCENA];
12 } Djak;
13
14 /* Funkcija proverava ispravnost date ocene. */
15 void provera_ocene(int ocena) {
16     if (ocena < 1 || ocena > 5) {
17         printf("Greska: neispravna ocena.\n");
18         exit(EXIT_FAILURE);
19     }
20 }
21
22 /* Funkcija ucitava podatke o djacima u niz. */
23 int ucitaj(Djak niz[]) {
24     int i = 0, j;
25
26     while (i < MAKS_DJAKA) {
27         printf("Unesite podatke o djaku: ");
28         /* Ucitavanje imena. */
29         if (scanf("%s", niz[i].ime) == EOF)
30             break;
31
32         /* Ucitavanje ocena. */
33         for (j = 0; j < BROJ_OCENA; j++) {
34             scanf("%d", &niz[i].ocena[j]);
35             provera_ocene(niz[i].ocena[j]);
36         }
37         i++;
38     }
39
40     return i;
}

```

```
42  /* Funkcija racuna prosecnu ocenu datog djaka. */
43  float prosecna_ocena(const Djak *djak) {
44      int j;
45      float suma = 0;
46      for (j = 0; j < BROJ_OCENA; j++)
47          suma += djak->ocena[j];
48
49      return suma / BROJ_OCENA;
50  }
51
52  int main() {
53      /* Deklaracija potrebnih promenljivih. */
54      Djak niz[MAKS_DJAKA];
55      int i, j, n;
56      float prosek;
57
58      /* Ucitavanje podataka o djacima. */
59      n = ucitaj(niz);
60
61      /* Ispis imena nedovoljnih djaka. */
62      printf("\n\nNEDOVOLJNI: ");
63      for (i = 0; i < n; i++)
64          for (j = 0; j < BROJ_OCENA; j++)
65              if (niz[i].ocena[j] == 1) {
66                  printf("%s ", niz[i].ime);
67                  break;
68              }
69      printf("\n");
70
71      /* Ispis imena odlicnih djaka. */
72      printf("ODLICNI: ");
73      for (i = 0; i < n; i++) {
74          prosek = prosecna_ocena(&niz[i]);
75          if (prosek >= 4.5)
76              printf("%s ", niz[i].ime);
77      }
78      printf("\n");
79
80      exit(EXIT_SUCCESS);
81  }
```

Rešenje 2.9.9

```
1 #include <stdio.h>
2 #include <string.h>
3 #include <stdlib.h>
4
5 #define MAKS_IME 21
6 #define MAKS_PREZIME 31
7 #define MAKS_EMAIL 51
```

```

#define MAKS_OSOBA 50
9
/* Struktura koja opisuje osobu. */
11 typedef struct {
12     char ime[MAKS_IME];
13     char prezime[MAKS_PREZIME];
14     char email[MAKS_EMAIL];
15 } Osoba;
16
17 /* I nacin: Funkcija proverava da li se prosledjeni email zavrsava
18    sa "gmail.com" koriscenjem funkcije strtok. */
19 int gmail(char email[])
20 {
21     /* Funkcija strtok "deli" nisku u podniske tako sto ih razdvaja
22        na mestu na kom se nalazi prosledjeni delimiter (u ovom
23        slucaju je to "@"). Na primer, ukoliko je
24        email="pera.peric@gmail.com", funkcija deli ovu nisku na
25        "pera.peric" i "gmail.com". */
26     char *deo = strtok(email, "@");
27
28     /* Kada se funkcija sledeci put pozove i pri tom pozivu se kao
29        prvi argument navede NULL, tada funkcija vraca sledeci token u
30        nizu, a to je u ovom slucaju "gmail.com". */
31     deo = strtok(NULL, "");
32
33     /* Ako se email zavrsava na "gmail.com", funkcija vraca 1, a u
34        suprotnom 0. */
35     return strcmp(deo, "gmail.com") == 0;
36 }
37
38 /* II nacin:
39 int gmail2(char email[])
40 {
41     //Pronalazi se pokazivac na znak @.
42     char* desni_deo = strchr(email, '@');
43
44     //Poredi se niska koja pocinje jedan karakter posle @ sa
45     //niskom "gmail.com".
46     return strcmp(desni_deo+1, "gmail.com") == 0;
47 }
48
49 int main()
50 {
51     /* Deklaracije potrebnih promenljivih. */
52     int n, i, postoji_gmail_adresa = 0;
53     Osoba osobe[MAKS_OSOBA];
54
55     /* Ucitavanje broja osoba i provera ispravnosti ulaza. */
56     printf("Unesite broj osoba: ");
57     scanf("%d", &n);
58     if (n < 0 || n > MAKS_OSOBA) {
59         printf("Greska: neispravan unos.\n");
60         exit(EXIT_FAILURE);
61     }
62 }
```

2 Napredni tipovi podataka

```
    }

61  /* Ucitavanje podataka o osobama. */
63  printf("Unesite podatke o osobama (ime, prezime i imejl adresu):\n"
       );
65  for (i = 0; i < n; i++)
    scanf("%s%s%s", osobe[i].ime, osobe[i].prezime, osobe[i].email);

67  /* Ispis rezultata. */
68  for (i = 0; i < n; i++) {
69      if (gmail(osobe[i].email)) {
70          if (!postoji_gmail_adresa) {
71              /* U ovu granu ce se uci samo kada se nadjde na prvog
72               vlasnika gmail naloga. */
73              printf("Vlasnici gmail naloga su:\n");
74              postoji_gmail_adresa = 1;
75          }
76          printf("%s %s\n", osobe[i].ime, osobe[i].prezime);
77      }
78  }

79  /* Ukoliko se nije naislo ni na jednog vlasnika gmail naloga,
80   promenljiva postoji_gmail_adresa ce ostati 0 i u tom slucaju
81   se ispisuje odgovarajuca poruka. */
82  if (!postoji_gmail_adresa)
83      printf("Nema vlasnika gmail naloga.\n");
84
85  exit(EXIT_SUCCESS);
86 }
```

Rešenje 2.9.10

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

4 #define MAKS_ARTIKALA 20
5 #define MAKS_KORPI 100
6 #define MAKS_NAZIV 31

8 /* Struktura koja opisuje artikal. */
9 typedef struct {
10     char naziv[MAKS_NAZIV];
11     int kolicina;
12     float cena;
13 } Artikal;

14 /* Struktura koja opisuje korpu. */
15 typedef struct {
16     int broj_artikala;
17     Artikal artikli[MAKS_ARTIKALA];
18 } Korpa;
```

```

20  /* Funkcija ucitava jedan artikal i proverava ispravnost ucitanih
21   * podataka. */
22  void ucitaj_artikal(Artikal *a) {
23      printf("Unesite artikal (naziv, kolicinu i cenu): ");
24      scanf("%s%d%f", a->naziv, &a->kolicina, &a->cena);
25
26      if (a->kolicina <= 0) {
27          printf("Greska: neispravan unos kolicine (%d).\n", a->kolicina);
28          exit(EXIT_FAILURE);
29      }
30
31      if (a->cena < 0) {
32          printf("Greska: neispravan unos cene (%f).\n", a->cena);
33          exit(EXIT_FAILURE);
34      }
35  }
36
37 /* Funkcija ucitava podatke o jednoj potrosackoj korpi. */
38 void ucitaj_korpu(Korpa *k) {
39     int i;
40     printf("Unesite podatke o korpi: \n");
41
42     /* Ucitavanje broja artikala u korpi. */
43     printf("Broj artikala: ");
44     scanf("%d", &k->broj_artikala);
45     if (k->broj_artikala <= 0) {
46         printf("Greska: neispravan unos broja artikala (%d).\n",
47                k->broj_artikala);
48         exit(EXIT_FAILURE);
49     }
50
51     /* Ucitavanje podataka o svakom artiklu. */
52     for (i = 0; i < k->broj_artikala; i++)
53         ucitaj_artikal(&k->artikli[i]);
54 }
55
56 /* Funkcija ucitava podatke o n potrosackih korpi. */
57 void ucitaj_niz_korpi(Korpa korpe[], int n) {
58     int i;
59     for (i = 0; i < n; i++)
60         ucitaj_korpu(&korpe[i]);
61 }
62
63 /* Funkcija racuna ukupan racun za datu korpu. */
64 float izracunaj_racun(const Korpa *k) {
65     int i;
66     float racun = 0;
67
68     for (i = 0; i < k->broj_artikala; i++)
69         racun += k->artikli[i].kolicina * k->artikli[i].cena;
70 }
```

2 Napredni tipovi podataka

```
72     return racun;
73 }
74
75 /* Funkcija ispisuje racun za datu korpu. */
76 void ispisi_racun(const Korpa *k) {
77     int i;
78     for (i = 0; i < k->broj_artikala; i++)
79         printf("\t%s %d %.2f\n", k->artikli[i].naziv,
80               k->artikli[i].kolicina, k->artikli[i].cena);
81     printf("-----\n");
82     printf("\tukupno: %.2f\n", izracunaj_racun(k));
83 }
84
85 /* Funkcija ispisuje racune za sve potrosacke korpe u nizu. */
86 void ispisi_racune_za_korpe(Korpa korpe[], int n) {
87     int i;
88     for (i = 0; i < n; i++) {
89         printf("\nKorpa %d:\n", i);
90         ispisi_racun(&korpe[i]);
91     }
92 }
93
94 /* Funkcija racuna prosecnu cenu potrosacke korpe za dati niz
95    potrosackih korpi. */
96 float prosek(Korpa korpe[], int n) {
97     int i;
98     float prosecna_cena = 0;
99
100    for (i = 0; i < n; i++)
101        prosecna_cena += izracunaj_racun(&korpe[i]);
102
103    return prosecna_cena / n;
104 }
105
106 int main() {
107     /* Deklaracije potrebnih promenljivih. */
108     int n;
109     Korpa korpe[MAKS_KORPI];
110
111     /* Ucitavanje broja potrosackih korpi i provera ispravnosti
112        ulaza. */
113     printf("Unesite broj potrosackih korpi:");
114     scanf("%d", &n);
115     if (n < 0 || n > MAKS_KORPI) {
116         printf("Greska: neispravan unos.\n");
117         exit(EXIT_FAILURE);
118     }
119
120     /* Ucitavanje podataka o potrosackim korpama. */
121     ucitaj_niz_korpi(korpe, n);
122
123     /* Ispis svih racuna. */
```

```

124     ispisi_racune_za_korpe(korpe, n);

126     /* Ispis prosecne cene potrosacke korpe. */
127     printf("Prosecna cena potrosacke korpe: %.2f\n",
128            prosek(korpe, n));

130    exit(EXIT_SUCCESS);
}

```

Rešenje 2.9.11

```

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

4 #define MAKS 50

6 /* Struktura koja opisuje loptu. */
8 typedef struct {
    int poluprecnik;
10   enum { plava = 1, zuta, crvena, zelena } boja;
} Lopta;

12 /* Funkcija racuna zapreminu loptu. */
14 float zapremina(const Lopta *l) {
    return pow(l->poluprecnik, 3) * 4 / 3 * M_PI;
}

18 /* Funkcija racuna zbir zapremina svih lopti u nizu. */
float ukupna_zapremina(Lopta lopte[], int n) {
20   int i;
    float ukupno = 0;

22   for (i = 0; i < n; i++)
    ukupno += zapremina(&lopte[i]);

26   return ukupno;
}

28 /* Funkcija broji lopte cija je boja jednaka boji koja je
30   prosledjena kao argument funkcije. */
int broj_lopti_u_boji(Lopta lopte[], int n, unsigned boja) {
32   int brojac = 0, i;

34   for (i = 0; i < n; i++)
    if (lopte[i].boja == boja)
      brojac++;

38   return brojac;
}

```

2 Napredni tipovi podataka

```
int main() {
    /* Deklaracije potrebnih promenljivih. */
    Lopta lopte[MAKS];
    int i, n;
    unsigned boja;

    /* Ucitavanje broja lopti i provera ispravnosti ulaza. */
    printf("Unesite broj lopti: ");
    scanf("%d", &n);
    if (n < 0 || n > MAKS) {
        printf("Greska: neispravan unos.\n");
        exit(EXIT_FAILURE);
    }

    /* Ucitavanje lopti u niz. */
    printf("Unesite poluprecnike i boje lopti "
           "(1-plava, 2-zuta, 3-crvena, 4-zelena):\n");
    for (i = 0; i < n; i++) {
        printf("%d. lopta: ", i + 1);
        scanf("%d%u", &lopte[i].poluprecnik, &boja);
        if (boja < 1 || boja > 4) {
            printf("Greska: neispravan unos.\n");
            exit(EXIT_FAILURE);
        }
        lopte[i].boja = boja;
    }

    /* Ispis rezultata. */
    printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));
    printf("Ukupno crvenih lopti: %d\n",
           broj_lopti_u_boji(lopte, n, crvena));
    exit(EXIT_SUCCESS);
}
```

Rešenje 2.9.12

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

#define MAKS_TEMENA 1000

/* Struktura koja opisuje tacku. */
typedef struct {
    int x, y;
} Tacka;

/* Funkcija racuna rastojanje izmedju dve tacke. */
double rastojanje(const Tacka *a, const Tacka *b) {
    return sqrt(pow(a->x - b->x, 2) + pow(a->y - b->y, 2));
}
```

```

16 }
17 /* Funkcija ucitava temena poligona. */
18 int ucitaj_poligon(Tacka poligon[], int maks_temena) {
19     int i = 0;
20
21     printf("Unesite temena poligona:\n");
22     while (scanf("%d%d", &poligon[i].x, &poligon[i].y) != EOF) {
23         i++;
24         if (i >= maks_temena)
25             break;
26     }
27
28     return i;
29 }
30
31 /* Funkcija racuna obim poligona. */
32 double obim_poligona(Tacka poligon[], int n) {
33     double obim = 0;
34     int i;
35
36     for (i = 0; i < n - 1; i++)
37         obim += rastojanje(&poligon[i], &poligon[i + 1]);
38
39     obim += rastojanje(&poligon[n - 1], &poligon[0]);
40
41     return obim;
42 }
43
44 /* Funkcija racuna najduzu stranicu poligona. */
45 double maksimalna_stranica(Tacka poligon[], int n) {
46     double maks = rastojanje(&poligon[0], &poligon[n - 1]);
47     double stranica;
48     int i;
49
50     for (i = 0; i < n - 1; i++) {
51         stranica = rastojanje(&poligon[i], &poligon[i + 1]);
52         if (stranica > maks)
53             maks = stranica;
54     }
55
56     return maks;
57 }
58
59 /* Funkcija racuna povrsinu trougla cija su temena A, B i C. */
60 double povrsina_trougla(const Tacka *A, const Tacka *B,
61                         const Tacka *C) {
62     double a = rastojanje(B, C);
63     double b = rastojanje(A, C);
64     double c = rastojanje(A, B);
65     double s = (a + b + c) / 2;
66 }
```

2 Napredni tipovi podataka

```
    return sqrt(s * (s - a) * (s - b) * (s - c));
68 }

70 /* Funkcija racuna povrsinu poligona. */
71 double povrsina_poligona(Tacka *poligon, int n) {
72     double povrsina = 0;
73     int i;

74     for (i = 1; i < n - 1; i++)
75         povrsina += povrsina_trouglja(&poligon[0], &poligon[i],
76                                         &poligon[i + 1]);

77     return povrsina;
78 }

79 int main() {
80     /* Deklaracije potrebnih promenljivih. */
81     int maks_temena, n;
82     Tacka poligon[MAKS_TEMENA];

83     /* Ucitavanje maksimalnog broja temena i provera ispravnosti. */
84     printf("Unesite maksimalan broj temena poligona: ");
85     scanf("%d", &maks_temena);
86     if (maks_temena < 3 || maks_temena > MAKS_TEMENA) {
87         printf("Greska: neispravan unos.\n");
88         exit(EXIT_FAILURE);
89     }

90     /* Ucitavanje poligona. */
91     n = ucitaj_poligon(poligon, maks_temena);
92     if (n < 3) {
93         printf("Greska: poligon mora imati bar tri temena.\n");
94         exit(EXIT_FAILURE);
95     }

96     /* Ispis rezultata. */
97     printf("Obim poligona je %.3lf.\n", obim_poligona(poligon, n));
98     printf("Duzina maksimalne stranice je %.3lf.\n",
99           maksimalna_stranica(poligon, n));
100    printf("Povrsina poligona je %.3lf.\n",
101          povrsina_poligona(poligon, n));

102    exit(EXIT_SUCCESS);
103 }
```

Rešenje 2.9.13

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

4 #define MAKS 1000
```

```

6  /* Struktura koja opisuje izraz. */
7  typedef struct {
8      char o;
9      int x;
10     int y;
11 } Izraz;
12
13 /* Funkcija proverava da li je izraz ispravno zadat. */
14 int korektan_izraz(const Izraz *izraz) {
15     if (izraz->o != '+' && izraz->o != '-' &&
16         izraz->o != '*' && izraz->o != '/') {
17         printf("Greska: neispravna operacija.\n");
18         return 0;
19     }
20
21     if (izraz->o == '/' && izraz->y == 0) {
22         printf("Greska: deljenje nulom.\n");
23         return 0;
24     }
25
26     return 1;
27 }
28
29 /* Funkcija ucitava n izraza sa standardnog ulaza. */
30 void ucitaj(Izraz izrazi[], int n) {
31     int i;
32
33     printf("Unesite izraze u prefiksnoj notaciji:\n");
34     for (i = 0; i < n; i++) {
35         scanf("%c%d%d", &izrazi[i].o, &izrazi[i].x, &izrazi[i].y);
36         /* Preskace se novi red koji se nalazi nakon izraza, kako bi
37            naredni izraz bio ispravno ucitan. */
38         getchar();
39
40         /* Provera ispravnosti ucitanog izraza. */
41         if (!korektan_izraz(&izrazi[i])) {
42             printf("Greska: neispravan unos.\n");
43             exit(EXIT_FAILURE);
44         }
45     }
46 }
47
48 /* Funkcija racuna vrednost izraza. */
49 int vrednost(const Izraz *izraz) {
50     switch (izraz->o) {
51     case '+':
52         return izraz->x + izraz->y;
53     case '-':
54         return izraz->x - izraz->y;
55     case '*':
56         return izraz->x * izraz->y;

```

2 Napredni tipovi podataka

```
58     case '/':
59         return izraz->x / izraz->y;
60     default:
61         printf("Greska: neispravna operacija.\n");
62         exit(EXIT_FAILURE);
63     }
64
65     /* Funkcija racuna najvecu vrednost izraza. */
66     int najveca_vrednost(Izraz izrazi[], int n) {
67         int i, maks_vrednost, trenutna_vrednost;
68
69         maks_vrednost = vrednost(&izrazi[0]);
70
71         for (i = 1; i < n; i++) {
72             trenutna_vrednost = vrednost(&izrazi[i]);
73             if (trenutna_vrednost > maks_vrednost)
74                 maks_vrednost = trenutna_vrednost;
75         }
76
77         return maks_vrednost;
78     }
79
80     int main() {
81         /* Deklaracije potrebnih promenljivih. */
82         int i, n;
83         Izraz izrazi[MAKS];
84         int maks, trenutna_vrednost;
85         float polovina;
86
87         /* Ucitavanje broja izraza i provera ispravnosti ulaza. */
88         printf("Unesite broj izraza: ");
89         scanf("%d", &n);
90         if (n < 0 || n > MAKS) {
91             printf("Greska: neispravan unos.\n");
92             exit(EXIT_FAILURE);
93         }
94
95         /* Preskace se belina koja se unosi nakon broja izraza. Ovaj
96            korak je neophodan jer se izraz zadaje u formatu:
97            <operacija> <operand> <operand>
98            Kako je <operacija> tipa char, izostavljanjem ovog koraka,
99            ta belina bi bila ucitana kao <operacija> za prvi izraz. */
100        getchar();
101        ucitaj(izrazi, n);
102
103        /* Pronalazak polovine maksimalne vrednosti. */
104        maks = najveca_vrednost(izrazi, n);
105        printf("Maksimalna vrednost izraza: %d\n", maks);
106        polovina = maks / 2.0;
107
108        /* Ispis rezultata. */
```

```

110     printf("Izrazi cija je vrednost manja od polovine maksimalne "
111           "vrednosti:\n");
112     for (i = 0; i < n; i++) {
113         trenutna_vrednost = vrednost(&izrazi[i]);
114         if (trenutna_vrednost < polovina) {
115             printf("%d %c %d = %d\n", izrazi[i].x, izrazi[i].o,
116                   izrazi[i].y, trenutna_vrednost);
117         }
118     }
119     exit(EXIT_SUCCESS);
120 }
```

Rešenje 2.9.14

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 #define MAKS_STEPEN 10
6 #define MAKS_POLINOMA 100
7
8 /* Struktura koja opisuje polinom. Polinom stepena MAKS_STEPEN
9    može imati najviše MAKS_STEPEN+1 koeficijenata, a njegov
10   integral onda može imati najviše MAKS_STEPEN+2 koeficijenata. */
11 typedef struct {
12     int stepen;
13     float koef[MAKS_STEPEN + 2];
14 } Polinom;
15
16 /* Funkcija ucitava podatke o polinomima. */
17 int ucitaj(Polinom niz[]) {
18     int i = 0, j;
19
20     while (i < MAKS_POLINOMA) {
21         printf("Unesite stepen: ");
22         if (scanf("%d", &(niz[i].stepen)) == EOF)
23             break;
24
25         if (niz[i].stepen > MAKS_STEPEN || niz[i].stepen < 0) {
26             printf("Greska: neispravan unos stepena.\n");
27             exit(EXIT_FAILURE);
28         }
29
30         printf("Unesite koeficijente polinoma:\n");
31         for (j = 0; j <= niz[i].stepen; j++)
32             scanf("%f", &(niz[i].koef[j]));
33
34         i++;
35     }
36 }
```

```
    return i;
38 }

40 /* Prvi monom je specijalan jer se ispred njega ne vrsi eksplicitan
   ispis znaka. Na primer, za polinom x + 3*x^2, prvi monom je x.
   Svakom sledecem monomu (u ovom slucaju samo 3*x^2) u ispisu
   prethodi znak (+ ili -). Funkcija ispisuje prvi monom. */
42 void ispis_prvog_monom(a float koef, int stepen) {
43     printf("%.2f", koef);

44     if (stepen == 1)
45         printf("*x ");
46     else if (stepen > 1)
47         printf("*x^%d ", stepen);
48 }

49 /* Funkcija ispisuje monom koji nije prvi. */
50 void ispis_monom(a float koef, int stepen) {
51     /* Monomi ciji je koeficijent nula se ne ispisuju. */
52     if (koef != 0) {
53         /* Ispis znaka. */
54         if (koef > 0)
55             printf("+ ");
56         else
57             printf("- ");

58         /* Ispis koeficijenta. */
59         printf("%.2f", fabs(koef));

60         /* Ispis ostatka. */
61         if (stepen == 1)
62             printf("*x ");
63         else if (stepen > 1)
64             printf("*x^%d ", stepen);
65     }
66 }

67 /* Funkcija ispisuje ceo polinom p. */
68 void ispis(const Polinom *p) {
69     int i;

70     /* Vrsi se ispis prvog monoma. Posto je moguce da prvi monom ima
       koeficijent 0, trazi se prvi monom sa koeficijentom razlicitim
       od nule. */
71     for (i = 0; i <= p->stepen; i++)
72         if (p->koef[i] != 0) {
73             ispis_prvog_monom(p->koef[i], i);
74             i++;
75             break;
76         }

77     /* Ispis ostalih monoma. Nastavlja se od mesta gde se stalo u
```

```

90     prethodnoj petlji i iz tog razloga je preskocen korak
91     inicijalizacije brojaca i. */
92     for ( ; i <= p->stopen; i++)
93         ispis_monomia(p->koef[i], i);
94
95     printf("\n");
96 }
97
98 /* Funkcija racuna integral polinoma p. */
99 void integral(const Polinom *p, Polinom *tekuci_integral) {
100    int i;
101
102    tekuci_integral->stopen = p->stopen + 1;
103    tekuci_integral->koef[0] = 0;
104
105    for (i = 1; i <= tekuci_integral->stopen; i++)
106        tekuci_integral->koef[i] = (float) p->koef[i - 1] / i;
107
108 int main() {
109    /* Deklaracija potrebnih promenljivih. */
110    Polinom polinomi[MAKS_POLINOMA], tekuci_integral;
111    int n, i;
112
113    /* Ucitavanje polinoma. */
114    n = ucitaj(polinomi);
115
116    /* Ispis integrala. */
117    printf("\n\nIntegrali su:\n");
118    for (i = 0; i < n; i++) {
119        integral(&polinomi[i], &tekuci_integral);
120        ispis(&tekuci_integral);
121    }
122
123    exit(EXIT_SUCCESS);
124 }
```

Elektronsko izdanie (2019)

3

Ulaz i izlaz programa

3.1 Argumenti komandne linije

Zadatak 3.1.1 Napisati program koji ispisuje broj navedenih argumenata komandne linije, a zatim i same argumente i njihove redne brojeve.

Primer 1

```
POKRETANJE: ./a.out d1.txt 10 13.5 d2.txt
IZLAZ:
Broj argumenata je 5.
0: ./a.out
1: d1.txt
2: 10
3: 13.5
4: d2.txt
```

Primer 2

```
POKRETANJE: ./a.out
IZLAZ:
Broj argumenata je 1.
0: ./a.out
```

[Rešenje 3.1.1]

Zadatak 3.1.2 Napisati program koji ispisuje zbir celobrojnih argumenata komandne linije. UPUTSTVO: Koristiti funkciju `atoi`.

Primer 1

```
POKRETANJE:
./a.out 5 ana 9 -2 11 4 +2
IZLAZ:
Zbir celobrojnih argumenata: 29
```

Primer 2

```
POKRETANJE:
./a.out a1 b1 1a 1b
IZLAZ:
Zbir celobrojnih argumenata: 0
```

Primer 3

```
POKRETANJE:
./a.out 33 1 @matf 44 22.56
IZLAZ:
Zbir celobrojnih argumenata: 78
```

[Rešenje 3.1.2]

3 Ulaz i izlaz programa

Zadatak 3.1.3 Napisati program koji na osnovu broja n koji se zadaje kao argument komandne linije, ispisuje cele brojeve iz intervala $[-n, n]$. U slučaju neispravnog pokretanja programa ispisati odgovarajuću poruku o grešci.

Primer 1

POKRETANJE: ./a.out 2
IZLAZ:
-2 -1 0 1 2

Primer 2

POKRETANJE: ./a.out
IZLAZ:
Greska: neispravan poziv.

Primer 3

POKRETANJE: ./a.out 0
IZLAZ:
0

[Rešenje 3.1.3]

Zadatak 3.1.4 Napisati program koji ispisuje argumemente komandne linije koji počinju karakterom @.

Primer 1

POKRETANJE:
./a.out @ana @aca #zvezda
IZLAZ:
Argumenti koji pocinju sa @: @ana @aca

Primer 2

POKRETANJE:
./a.out sanke @zapad zujanje
IZLAZ:
Argumenti koji pocinju sa @: @zapad

Primer 3

POKRETANJE:
./a.out bundeva pomorandza
IZLAZ:
Nema argumenata koji pocinju sa @.

[Rešenje 3.1.4]

Zadatak 3.1.5 Napisati program koji ispisuje broj argumenata komandne linije koji sadrže karakter @.

Primer 1

POKRETANJE:
./a.out pera@gmail.com @
IZLAZ:
Rezultat: 2

Primer 2

POKRETANJE:
./a.out japan caj
IZLAZ:
0

Primer 3

POKRETANJE:
./a.out
IZLAZ:
Rezultat: 0

[Rešenje 3.1.5]

Zadatak 3.1.6 Napisati program koji proverava da li se među zadatim argumentima komandne linije nalaze barem dva ista. NAPOMENA: Uzeti u obzir i naziv programa koji se pokreće.

3.1 Argumenti komandne linije

Primer 1

POKRETANJE:
./a.out ulaz.txt izlaz.txt ulaz.txt

IZLAZ:
Medju argumentima ima istih.

Primer 2

POKRETANJE:
./a.out srce pik tref tref

IZLAZ:
Medju argumentima ima istih.

Primer 3

POKRETANJE:
./a.out Riba ribi grize rep.

IZLAZ:
Medju argumentima nema istih.

Primer 4

POKRETANJE:
./a.out

IZLAZ:
Medju argumentima nema istih.

[Rešenje 3.1.6]

Zadatak 3.1.7 Napisati program koji ispisuje sve opcije koje su navedene u komandnoj liniji. Opcije su karakteri argumenata komandne linije koji za koje važi da počinju karakterom -.

Primer 1

POKRETANJE:
./a.out -rf in.txt

IZLAZ:
Opcije su: r f

Primer 2

POKRETANJE:
./a.out

IZLAZ:
Medju argumentima nema opcija.

Primer 3

POKRETANJE:
./a.out ulaz.txt

IZLAZ:
Medju argumentima nema opcija.

Primer 4

POKRETANJE:
./a.out in.txt -l -n 10 -fi out.txt

IZLAZ:
Opcije su: l n f i

[Rešenje 3.1.7]

3.2 Rešenja

Rešenje 3.1.1

```
1 #include <stdio.h>
2
3 /* Argumenti komandne linije cuvaju se u nizu niski. Svaki element
4    tog niza odgovara jednom argumentu komandne linije, pri cemu
5    prvi element predstavlja naziv programa koji se pokreće.
6    Celobrojna promenljiva argc predstavlja ukupan broj argumenata
7    komandne linije uključujući i argument koji odgovara nazivu
8    programa, a promenljiva argv pomenuti niz niski koji sadrži same
9    argumente. */
10   int main(int argc, char *argv[]) {
11       /* Deklaracija potrebne promenljive. */
12       int i;
13
14       /* Ispis broja argumenata komandne linije. */
15       printf("Broj argumenata je %d.\n", argc);
16
17       /* Ispis svakog od navedenih argumenata. */
18       for (i = 0; i < argc; i++)
19           printf("%d: %s\n", i, argv[i]);
20
21   return 0;
22 }
```

Rešenje 3.1.2

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 /* Funkcija proverava da li prosledjenu nisku cine samo karakteri
6    koji su cifre. */
7   int samo_cifre(char arg[]) {
8       int i;
9
10      /* Prvi karakter mora biti ili cifra ili znak broja. */
11      if (!isdigit(arg[0]) && arg[0] != '+' && arg[0] != '-')
12          return 0;
13
14      /* Ostali karakteri moraju biti cifre. */
15      for (i = 1; arg[i]; i++)
16          if (!isdigit(arg[i]))
17              return 0;
18
19      return 1;
20 }
```

```

22 int main(int argc, char *argv[]) {
23     /* Deklaracija potrebnih promenljivih. */
24     int i, suma = 0;
25
26     /* Kako su argumenti komandne linije niske, potrebno ih je
27      konvertovati u brojeve. Za ovo je moguce koristiti funkciju
28      atoi. Npr. atoi("567") ima vrednost 567. Treba voditi racuna:
29      atoi("abc") ima vrednost 0, ali atoi("12abc") ima vrednost 12.
30      Dakle ova funkcija se zaustavlja u trenutku kada se u okviru
31      niske naidje na prvi karakter koji nije cifra. Iz tog razloga
32      je potrebno proveriti da li dati argument sadrzi samo cifre. */
33     for (i = 1; i < argc; i++)
34         if (samo_cifre(argv[i]))
35             suma += atoi(argv[i]);
36
37     /* Ispis rezultata. */
38     printf("Zbir celobrojnih argumenata: %d\n", suma);
39
40     return 0;
41 }
```

Rešenje 3.1.3

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main(int argc, char *argv[]) {
5     /* Deklaracija potrebnih promenljivih. */
6     int n, i;
7
8     /* Provera broja argumenata komandne linije. */
9     if (argc != 2) {
10         printf("Greska: neispravan poziv.\n");
11         exit(EXIT_FAILURE);
12     }
13
14     /* Ucitavanje broja n i cuvanje njegove absolutne vrednosti. */
15     n = atoi(argv[1]);
16     n = abs(n);
17
18     /* Ispis rezultata. */
19     for (i = -n; i <= n; i++)
20         printf("%d ", i);
21     printf("\n");
22
23     exit(EXIT_SUCCESS);
24 }
```

3 Ulaz i izlaz programa

Rešenje 3.1.4

```
1 #include <stdio.h>
2
3 int main(int argc, char *argv[]) {
4     /* Deklaracija potrebnih promenljivih. */
5     int i, prikazi_poruku = 0;
6
7     /* Ispis svih argumenata komandne linije ciji je prvi karakter
8      znak '@'. Ako se program pokrene sa:
9      ./a.out @pera mika @zika
10     argv[0] je "./a.out" i on se preskace.
11     argv[1] je "@pera", a prvi karakter je onda argv[1][0].
12     Dakle, za argv[i] treba proveravati da li je argv[i][0] jednak
13     karakteru '@'. */
14     for (i = 1; i < argc; i++) {
15         if (argv[i][0] == '@') {
16             /* Promenljiva prikazi_poruku sluzi da detektuje da li
17              postoji bar jedna niska koja pocinje sa '@'. Ukoliko se
18              naidje na prvu takvu nisku, ispisuje se trazena poruka i
19              prikazi_poruku se postavlja na 1. */
20             if (!prikazi_poruku) {
21                 printf("Argumenti koji pocinju sa @:\n");
22                 prikazi_poruku = 1;
23             }
24             printf("%s ", argv[i]);
25         }
26     }
27
28     /* Ukoliko je vrednost promenljive prikazi_poruku i dalje 0,
29      znam da nijedan argument ne pocinje karakterom '@'. */
30     if (!prikazi_poruku)
31         printf("Nema argumenata koji pocinju sa @.");
32     printf("\n");
33
34     return 0;
35 }
```

Rešenje 3.1.5

```
1 #include <stdio.h>
2 #include <string.h>
3
4 int main(int argc, char *argv[]) {
5     /* Deklaracija potrebnih promenljivih. */
6     int i, brojac = 0;
7
8     /* Prebrojavanje argumenata koji sadrze karakter @. */
9     for (i = 1; i < argc; i++)
10         if (strchr(argv[i], '@') != NULL)
11             brojac++;
```

```

13  /* Ispis rezultata. */
14  printf("Rezultat: %d\n", brojac);
15
16  return 0;
17 }

```

Rešenje 3.1.6

```

1 #include <stdio.h>
2 #include <string.h>
3
4 int main(int argc, char *argv[]) {
5     /* Deklaracije potrebnih promenljivih. */
6     int i, j;
7
8     /* Ukoliko je naveden jedan ili nijedan argument, onda ne moze da
9      bude duplikata. */
10    if (argc < 2) {
11        printf("Medju argumentima nema istih.\n");
12        return 0;
13    }
14
15    /* Za svaki argument komandne linije se proverava da li postoji
16       neki od argumenata koji mu je jednak. */
17    for (i = 0; i < argc; i++) {
18        /* Za fiksirano argv[i] se vrsti provera svih argumenata koji se
19           nalaze nakon njega. */
20        for (j = i + 1; j < argc; j++)
21            if (strcmp(argv[i], argv[j]) == 0) {
22                printf("Medju argumentima ima istih.\n");
23                return 0;
24            }
25    }
26
27    /* Ukoliko se prethodna petlja zavrsila, a nije se izaslo iz
28       programa, znaci da medju argumentima nema istih. */
29    printf("Medju argumentima nema istih.\n");
30
31    return 0;
32 }

```

Rešenje 3.1.7

```

1 #include <stdio.h>
2
3 int main(int argc, char *argv[]) {
4     /* Deklaracija potrebnih promenljivih. */
5     int i, j, prikazi_poruku = 0;

```

3 Ulaz i izlaz programa

```
7  /* Prolazi se kroz sve argumente komandne linije. */
8  for (i = 1; i < argc; i++) {
9      /* Ukoliko argument pocinje karakterom '-', znaci da se navode
10         opcije. */
11     if (argv[i][0] == '-') {
12         /* Ukoliko je u pitanju prvi niz opcija, ispisuje se
13            odgovarajuca poruka i vrednost promenljive prikazi_poruku
14            se postavlja na 1. */
15         if (!prikazi_poruku) {
16             printf("Opcije su: ");
17             prikazi_poruku = 1;
18         }
19
20         /* Ispisuju se sve opcije, tj. svi karakteri argumenta
21            argv[i] koji se nalaze nakon '-'. */
22         for (j = 1; argv[i][j]; j++)
23             printf("%c ", argv[i][j]);
24     }
25
26
27     /* Ukoliko je vrednost promenljive prikazi_poruku nakon petlje 0,
28        znaci da nije navedena nijedna opcija. */
29     if (!prikazi_poruku)
30         printf("Medju argumentima nema opcija.\n");
31     else
32         printf("\n");
33
34     return 0;
35 }
```

3.3 Datoteke

Zadatak 3.3.1 Napisati program koji prepisuje sadržaj datoteke *ulaz.txt* u datoteku *izlaz.txt* karakter po karakter. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
||| ULAZ.TXT
||| Danas je 21. mart.
||| To je prvi dan proleća.

||| IZLAZ.TXT
||| Danas je 21. mart.
||| To je prvi dan proleća.
```

Primer 2

```
||| ULAZ.TXT
||| Ispit iz Programiranja 1 je
||| zakazan za 10. jun.

||| IZLAZ.TXT
||| Ispit iz Programiranja 1 je
||| zakazan za 10. jun.
```

Primer 3

```
||| ULAZ.TXT NE POSTOJI
||| IZLAZ ZA GREŠKE:
||| Greska: neuspesno otvaranje
||| datoteke ulaz.txt
```

[Rešenje 3.3.1]

Zadatak 3.3.2 Napisati program koji prepisuje svaki treći karakter datoteke *ulaz.txt* u datoteku *izlaz.txt*. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
||| ULAZ.TXT
||| Volim programiranje.

||| IZLAZ.TXT
||| Vipgmae
```

Primer 2

```
||| ULAZ.TXT
||| Zivot je lep!

||| IZLAZ.TXT
||| Zojl!
```

Primer 3

```
||| ULAZ.TXT
||| 1234567890

||| IZLAZ.TXT
||| 1470
```

Primer 4

```
||| ULAZ.TXT
||| Ova datoteka
||| sadrzi tekst
||| u vise
||| linija.

||| IZLAZ.TXT
||| O te
||| diet sli.
```

Primer 5

```
||| ULAZ.TXT
||| U Beogradu ce biti
||| suncan i lep
||| dan.

||| IZLAZ.TXT
||| Ueruei
||| nn pa
```

[Rešenje 3.3.2]

Zadatak 3.3.3 Napisati program koji šifruje sadržaj datoteke *podaci.txt* tako što svako slovo ciklično zamenjuje njegovim prethodnikom suprotne veličine i upisuje u datoteku *sifra.txt*. Na primer, slovo b se zamenjuje slovom A, slovo

3 Ulaz i izlaz programa

B slovom **a**, slovo **a** slovom **Z**, slovo **A** slovom **z**, itd. Ostali karakteri ostaju ne-promenjeni. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
PODACI.TXT
Matematicki fakultet
Studentski trg 16
Beograd

SIFRA.TXT
1ZSDLZSHBJH EZJTKSDS
rSTCDMSRJH SQF 16
aDNFQZC
```

Primer 2

```
PODACI.TXT
a=x+y;
x=b+5;

SIFRA.TXT
Z=W+X;
W=A+5;
```

Primer 3

```
PODACI.TXT NE POSTOJI
IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
datoteke podaci.txt.
```

[Rešenje 3.3.3]

Zadatak 3.3.4 Napisati program koji za dve datoteke čija se imena unose sa standardnog ulaza, radi sledeće:

- za svaku cifru u prvoj datoteci, u drugu datoteku upisuje 0
- za svako slovo u prvoj datoteci, u drugu datoteku upisuje 1
- za sve ostale karaktere u prvoj datoteci, u drugu datoteku upisuje 2

Prepostaviti da je maksimalna dužina naziva datoteka 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime prve datoteke:
kilometraze.txt
Unesite ime druge datoteke:
sifra.txt

KILOMETRAZE.TXT
Beograd - Nis 230km
Uzice - Cacak 56.3km
Subotica - Ruma 139km

SIFRA.TXT
111111122211120001121111122
21111120020112111111122211
11200011
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime prve datoteke:
ulaz.txt
Unesite ime druge datoteke:
izlaz.txt

ULAZ.TXT
18. februar 2019.

IZLAZ.TXT
00221111111200002
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime prve datoteke:
in.dat
Unesite ime druge datoteke:
out.dat

IN.DAT NE POSTOJI
IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
prve datoteke.
```

[Rešenje 3.3.4]

Zadatak 3.3.5 Sa standardnog ulaza učitavaju se imena dveju datoteka i jedan karakter koji označava opciju. Napisati program koji prepisuje sadržaj prve datoteke u drugu tako što u slučaju da je navedena opcija u, sva mala slova zamenjuje velikim slovima, a u slučaju da je navedena opcija l, sva velika slova zamenjuje malim slovima. Prepostaviti da je maksimalna dužina naziva datoteke 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

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

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite imena datoteka i opciju:  
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:  
Unesite imena datoteka i opciju:  
primer.c prazna.txt V  
  
PRIMER.C  
#include <stdio.h>  
int main()  
{  
}  
  
PRAZNA.TXT  
  
IZLAC ZA GREŠKE:  
Greska: neuspesno otvaranje
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:  
Unesite imena datoteka i opciju:  
primer.c prazna.txt V  
  
PRIMER.C NE POSTOJI  
  
IZLAC ZA GREŠKE:  
Greska: neuspesno otvaranje  
prve datoteke.
```

[Rešenje 3.3.5]

Zadatak 3.3.6 Napisati program koji prebrojava mala slova u datoteci *podaci.txt* i dobijeni rezultat ispisuje na standardni izlaz. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

3 Ulaz i izlaz programa

Primer 1

```
PODACI.TXT
Matematicki fakultet
Studentski trg 16
Beograd
IZLAZ:
Broj malih slova je: 36
```

Primer 2

```
PODACI.TXT
PrograMiranje
IZLAZ:
Broj malih slova je: 11
```

Primer 3

```
PODACI.TXT
MATEMATIKA
12+34=46
IZLAZ:
Broj malih slova je: 0
```

[Rešenje 3.3.6]

Zadatak 3.3.7 Napisati program koji u datoteci čije se ime unosi sa standardnog ulaza prebrojava koliko se puta pojavljuje svaka od cifara i na standardni izlaz ispisuje cifru sa najvećim brojem pojavljivanja. Ukoliko ima više takvih cifara, ispisati sve. Ukoliko datoteka ne sadrži nijednu cifru, ispisati odgovarajuću poruku. Prepostaviti da je maksimalna dužina naziva datoteke 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
ulaz.txt

ULAZ.TXT
danas je lep dan
i ja zelim
da postanem programer

IZLAZ:
Datoteka ne sadrzi cifre.
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
prva.dat

PRVA.DAT
Cena soka je 30
Cena vina je 150
Cena limunade je 200
Cena sendvica je 120

IZLAZ:
Najcesce cifre: 0
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
primer.c

PRIMER.C
1 22 333.444

IZLAZ:
Najcesce cifre: 3 4
```

[Rešenje 3.3.7]

Zadatak 3.3.8 Napisati program koji u datoteci čije je ime dano kao argument komandne linije proverava da li su zagrade pravilno uparene. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1	Primer 2	Primer 3
<pre>POKRETANJE: ./a.out zgrade.dat ZGRADE.TXT ab(cd) .. ((3+4)*5+1)*9 IZLAZ: Zgrade jesu uparene.</pre>	<pre>POKRETANJE: ./a.out primer2.dat PRIMER2.DAT (7+8 nisu(uparene IZLAZ: Zgrade nisu uparene.</pre>	<pre>POKRETANJE: ./a.out primer3.dat PRIMER3.DAT)) 7 + 6 ((IZLAZ: Zgrade nisu uparene.</pre>

[Rešenje 3.3.8]

Zadatak 3.3.9 Napisati program koji prebrojava slova i cifre u datoteci.

- (a) Napisati funkciju `int ucitaj_karaktere(char s[], FILE *f)` kojom se učitavaju karakteri iz datoteke `f` u niz karaktera `s`. Dozvoljeni karakteri za učitavanje su mala i velika slova engleske abecede, kao i cifre. Učitavanje se prekida kada se nađe na znak za novi red ili nedozvoljeni karakter. Funkcija vraća broj elemenata niza uspešno učitanih karaktera.
- (b) Napisati funkciju `void prebroj(char s[], int n, int *broj_slova, int *broj_cifara)` kojom se određuje broj slovnih elemenata niza karaktera (velikih ili malih slova) kao i broj cifara.

Napisati program koji koristeći prethodne funkcije prebrojava cifre i slova u datoteci čije se ime zadaje kao argument komandne linije, a zatim ispisuje dobijene vrednosti na standardni izlaz. Prepostaviti da je maksimalni broj karaktera datoteke 1000. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1	Primer 2	Primer 3
<pre>POKRETANJE: ./a.out skup.txt SKUP.TXT OvoJeSkupKaraktera.123 IZLAZ: Broj slova: 18 Broj cifara: 0</pre>	<pre>POKRETANJE: ./a.out skup2.txt SKUP2.TXT ovdeimamo\$dolar IZLAZ: Broj slova: 9 Broj cifara: 0</pre>	<pre>POKRETANJE: ./a.out skup3.txt SKUP3.TXT broj3 broj5 IZLAZ: Broj slova: 4 Broj cifara: 1</pre>

3 Ulaz i izlaz programa

Primer 4

```
POKRETANJE:  
./a.out skup4.txt  
  
SKUP4.TXT  
11.2.2019.  
  
IZLAZ:  
Broj slova: 0  
Broj cifara: 2
```

Primer 5

```
POKRETANJE:  
./a.out skup5.txt  
  
SKUP5.TXT NE POSTOJI  
  
IZLAZ ZA GREŠKE:  
Greska: neuspesno otvaranje  
ulazne datoteke.
```

Primer 6

```
POKRETANJE:  
./a.out  
  
IZLAZ ZA GREŠKE:  
Greska: neispravan poziv.
```

[Rešenje 3.3.9]

Zadatak 3.3.10 Napisati program koji sa standardnog ulaza učitava reč s i u datoteku *rotacije.txt* upisuje sve njene rotacije. Pretpostaviti da je maksimalna dužina reči 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite rec: abcde  
  
ROTACIJE.TXT  
abcde  
bcdea  
cdeab  
deabc  
eabcd
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite rec: 1234  
  
ROTACIJE.TXT  
1234  
2341  
3412  
4123
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite rec: a=3*x+5;  
  
ROTACIJE.TXT  
a=3*x+5;  
=3*x+5;a=  
3*x+5;a=  
*x+5;a=3  
x+5;a=3*  
+5;a=3*x  
5;a=3*x+  
;a=3*x+5
```

[Rešenje 3.3.10]

Zadatak 3.3.11 Sa standarnog ulaza se učitava ime datoteke i nenegativan ceo broj k . Napisati program koji učitava reči iz datoteke (reč je niz karaktera između blanko simbola) i svaku pročitanu reč rotira za k mesta u levo i tako dobijenu reč upisuje u datoteku čije je ime *rotirano.txt*. Pretpostaviti da je maksimalna dužina naziva datoteke 20 karaktera, da datoteka sadrži samo slova i beline i da je maksimalna dužina jedne reči u datoteci 100 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
ulaz.txt
Unesite broj k: 3

ULAZ.TXT
jedan dva
tri cetiri

ROTIRANO.TXT
anjed dva tri iricet
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
in.dat
Unesite broj k: 5

IN.DAT
Popodne ce biti kise

ROTIRANO.TXT
nePopod ec itib isek
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime datoteke:
input.txt
Unesite broj k: 0

INPUT.TXT
Popodne ce
biti kise

ROTIRANO.TXT
Popodne ce biti kise
```

[Rešenje 3.3.11]

Zadatak 3.3.12 Napisati program koji iz datoteke *razno.txt* u datoteku *palindromi.txt* prepisuje sve palindrome. Reč je palindrom ako se isto čita sa leve i desne strane bez obzira na veličinu slova. Pretpostaviti da je maksimalna dužina reči 200 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

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 Oko
kapAk radar
```

Primer 3

```
RAZNO.TXT
ovde nema palindroma

PALINDROMI.TXT:
```

[Rešenje 3.3.12]

Zadatak 3.3.13 U datoteci čije se ime zadaje sa standardnog ulaza nalazi se broj n ($n \leq 256$), a zatim i n reči. Napisati program koji učitava reči iz datoteke u niz i iz niza uklanja sve duplike i upisuje izmenjeni niz u datoteku *bez_duplikata.txt*. Pretpostaviti da je maksimalna dužina naziva datoteke 20 karaktera, a maksimalna dužina jedne reči u datoteci 50 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

3 Ulaz i izlaz programa

Primer 1

```
INTERAKCIJA SA PROGRAMOM:  
Unesite ime datoteke:  
imena.txt  
  
IMENA.TXT  
12  
Ana Milos Ana Marko  
Petar Filip Jovana Ana  
Petar Ivan Nikola Filip  
  
BEZ_DUPLIKATA.TXT:  
Ana Milos Marko Petar  
Filip Jovana Ivan Nikola
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:  
Unesite ime datoteke:  
gradovi.txt  
  
GRADOVI.TXT  
10  
Sombor Beograd  
Nis Beograd  
Beograd Indjija  
Nis Ruma  
Ruma Sombor  
  
BEZ_DUPLIKATA.TXT:  
Sombor Beograd Nis  
Indjija Ruma
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:  
Unesite ime datoteke:  
gradovi.txt  
  
GRADOVI.TXT NE POSTOJI  
  
IZLAZ ZA GREŠKE:  
Greska: neuspesno otvaranje  
ulazne datoteke.
```

[Rešenje 3.3.13]

Zadatak 3.3.14 U datoteci čije se ime zadaje kao prvi argument komandne linije nalazi se ceo pozitivan broj n , a zatim i n celih brojeva. Napisati program koji na standardni izlaz ispisuje koliko k -tocifrenih brojeva postoji u datoteci, pri čemu se pozitivan ceo broj k zadaje kao drugi argument komandne linije. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
POKRETANJE:  
. ./a.out ulaz.txt 2  
  
ULAZ.TXT  
6  
15 193 -27 9790 35 1  
  
IZLAZ:  
Broj 2-cifrenih brojeva: 3
```

Primer 2

```
POKRETANJE:  
. ./a.out ulaz.txt 5  
  
ULAZ.TXT  
4  
15 193 -27 9790  
  
IZLAZ:  
Broj 5-cifrenih brojeva: 0
```

Primer 3

```
POKRETANJE:  
. ./a.out ulaz.txt  
  
IZLAZ ZA GREŠKE:  
Greska: neispravan poziv.
```

[Rešenje 3.3.14]

Zadatak 3.3.15 Napisati program koji na standardni izlaz ispisuje maksimum brojeva iz datoteke *brojevi.txt*. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

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

IZLAZ:
Najveci broj je: 54.96
```

Primer 2

```
BROJEVI.TXT
10.5 183.111 -90.2 3.167

IZLAZ:
Najveci broj je: 183.111
```

Primer 3

```
BROJEVI.TXT
-62.7 -190.2 -2.3 -1000
-198.25 -8

IZLAZ:
Najveci broj je: -2.3
```

[Rešenje 3.3.15]

Zadatak 3.3.16 Prvi red datoteke *matrica.txt* sadrži dva 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 pronađe sve elemente matrice a koji su jednaki zbiru svih svojih susednih elemenata i štampa ih u obliku (*broj vrste*, *broj kolone*, *vrednost elementa*). Prepostaviti da je sadržaj datoteke ispravan. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku. UPUTSTVO: *U zadatku 2.7.6 je dato objašnjenje koji elementi matrice su susedni.*

Primer 1

```
MATRICA.TXT
3 4
1 2 3 4
7 2 15 -3
-1 3 1 3

IZLAZ:
(1, 0, 7)
(1, 2, 15)
```

Primer 2

```
MATRICA.TXT
2 2
1 1
-2 2

IZLAZ:
(0, 0, 1)
(0, 1, 1)
```

Primer 3

```
MATRICA.TXT
1 4
9 3 5 2

IZLAZ:
(0, 2, 5)
```

[Rešenje 3.3.16]

Zadatak 3.3.17 Prvi red datoteke *ulaz.txt* sadrži dva cela broja između 2 i 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. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

3 Ulaz i izlaz programa

Primer 1

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

Primer 2

```
ULAZ.TXT  
1 4  
9 3 5 2  
  
IZLAZ ZA GREŠKE:  
Greska: neispravna  
dimenzija.
```

Primer 3

```
ULAZ.TXT  
2 2  
1 1  
-2 2  
  
IZLAZ:
```

[Rešenje 3.3.17]

Zadatak 3.3.18 U datoteci *tacke.txt* se nalazi broj tačaka, a zatim u posebnim redovima za svaku tačku njene x i y koordinate. Napisati program koji u datoteku *rastojanja.txt* upisuje rastojanje svake od učitanih tačaka od koordinatnog početka, a na standardni izlaz koordinate tačke koja je od njega najudaljenija. Ukoliko ima više takvih tačaka, ispisati koordinate prve. Koristiti strukturu *Tacka* sa poljima x i y , kao i funkciju kojom se računa rastojanje tačke od koordinatnog početka. Prepostaviti da je maksimalan broj tačaka u datoteci 50. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
TACKE.TXT  
4  
11 -2  
3 5  
8 -8  
0 4  
  
RASTOJANJA.TXT  
11.18  
5.83  
11.31  
4.00  
  
IZLAZ:  
Najudaljenija tačka: (8, -8)
```

Primer 2

```
TACKE.TXT  
-2  
0 0  
9 -8  
  
IZLAZ ZA GREŠKE:  
Greska: neispravan broj tacaka.
```

[Rešenje 3.3.18]

Zadatak 3.3.19 Definisati strukturu kojom se opisuje trodimenzioni vektor sa celobrojnim koordinatama x , y i z . U datoteci *vektori.txt* nalazi se nepoznati broj vektora. Napisati program koji učitava vektore iz ove datoteke i na standardni izlaz ispisuje koordinate vektora sa najvećom dužinom. Ukoliko ima

više takvih vektora, ispisati koordinate prvog. Dužina vektora se izračunava po formuli: $|v| = \sqrt{x^2 + y^2 + z^2}$. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

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

```
IZLAZ:
4 -1 7
```

Primer 2

```
VEKTORI.TXT
4 -4 4
-4 -4 -4
```

```
IZLAZ:
4 -4 4
```

Primer 3

```
VEKTORI.TXT
0 0 0
0 1 0
1 0 0
```

```
IZLAZ:
0 1 0
```

Primer 4

```
VEKTORI.TXT
3 0 1
4 5 2
1 0 0
2 -1 2
```

```
IZLAZ:
4 5 2
```

Primer 5

```
VEKTORI.TXT NE POSTOJI
```

```
IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

Primer 6

```
VEKTORI.TXT
1 1 1
```

```
IZLAZ:
1 1 1
```

[Rešenje 3.3.19]

Zadatak 3.3.20 Definisati strukturu Pravougaonik koja sadrži dužine stranica i ime pravougaonika. Napisati program koji iz datoteke čije ime se zadaje kao argument komandne linije učitava podatke o pravougaonicima (nije poznato koliko), a zatim ispisuje imena onih pravougaonika koji su kvadrati i vrednost najveće površine među pravougaonicima koji nisu kvadrati. Pretpostaviti da je maksimalan broj pravougaonika 200, a maksimalna dužina imena pravougaonika 4. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

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
```

[Rešenje 3.3.20]

3 Ulaz i izlaz programa

Zadatak 3.3.21 U datoteci *studenti.txt* se nalaze podaci o studentima. Za svakog studenta je dato korisničko ime na Alas serveru i poslednjih pet ocena koje je dobio. Napisati program koji pronalazi studenta koji je ostvario najbolji uspeh i ispisuje njegove podatke. Ukoliko ima više takvih studenata, ispisati informacije o svima. Prepostaviti da je maksimalni broj studenta 100. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
STUDENTI.TXT
mr15239 10 9 9 8 10
mi14005 8 8 9 8 10
ml15112 9 8 8 7 10
mr15007 10 10 10 10 10
mn13208 7 7 9 6 10
```

IZLAZ:

```
Studenti sa najvećim prosekom:
Korisnicko ime: mr15007
Prosek ocena: 10.00
```

Primer 2

```
STUDENTI.TXT
mr16156 10 9 9 10 10
mi17234 9 9 10 10 10
ml17084 9 8 8 8 8
```

IZLAZ:

```
Studenti sa najvećim prosekom:
Korisnicko ime: mr16156
Prosek ocena: 9.6

Korisnicko ime: mi17234
Prosek ocena: 9.6
```

[Rešenje 3.3.21]

Zadatak 3.3.22 Definisati strukturu **Student** koja sadrži puno ime studenta, niz njegovih ocena, broj ocena i prosečnu ocenu. U datoteci čije se ime zadaje kao argument komandne linije se nalaze podaci o studentima. Za svakog studenta dato je ime, prezime i niz ocena koji se završava nulom. Svi podaci su razdvojeni razmacima. Napisati program koji učitava podatke o studentima i na standardni izlaz ispisuje podatke za studenta sa najvećim prosekom (prosek ispisati na 2 decimale). Ukoliko ima više takvih studenata, ispisati informacije o prvom studentu. Prepostaviti da je maksimalni broj ocena 10 i maksimalna dužina punog imena 100 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku. UPUTSTVO: *Ime i prezime studenta se mogu pročitati pomoću specifikatora %s a potom se za kreiranje niske **puno_ime** u traženom formatu može iskoristiti funkcija **strcat**.*

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 0
```

IZLAZ:

```
Jelena Jankovic 10 10 10 10.00
```

Primer 2

```
POKRETANJE: ./a.out
```

```
IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

[Rešenje 3.3.22]

Zadatak 3.3.23 Imena ulazne i izlazne datoteke se redom navode kao argumenti komandne linije. U ulaznoj datoteci se nalaze podaci o razlomcima: u prvom redu se nalazi broj razlomaka, a u svakom sledećem redu brojilac i imenilac po jednog razlomka. Definisati strukturu koja opisuje razlomak i napisati program koji učitava niz razlomaka iz datoteke, a potom:

- (a) ukoliko je prilikom pokretanja programa navedena opcija `x`, upisati u izlaznu datoteku recipročni razlomak za svaki razlomak iz niza
- (b) ukoliko je prilikom pokretanja programa navedena opcija `y`, upisati u izlaznu datoteku realnu vrednost recipročnog razlomka svakog razlomka iz niza

Pretpostaviti da se u ulaznoj datoteci nalazi najviše 100 razlomaka. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
POKRETANJE:  
./a.out d1.txt d2.txt -x  
  
D1.TXT  
4  
1 5  
19 3  
-2 7  
97 90
```

Primer 1 (nastavak)

```
D2.TXT  
5/1  
3/19  
-7/2  
90/97
```

Primer 2

```
POKRETANJE:  
./a.out ulaz.txt izlaz.txt  
  
IZLAZ ZA GREŠKE:  
Greska: neispravan poziv.
```

Primer 3

```
POKRETANJE:  
./a.out d1.txt d2.txt -y  
  
D1.TXT  
4  
1 5  
19 3  
-2 7  
97 90
```

Primer 3 (nastavak)

```
D2.TXT  
5.000000  
0.157894  
-3.500000  
0.927835
```

[Rešenje 3.3.23]

Zadatak 3.3.24 Definisati strukturu `Automobil` koja sadrži marku, model i cenu. Napisati program koji iz datoteke čije se ime zadaje sa standardnog ulaza učitava broj automobila i podatke za svaki automobil i zatim:

- (a) ispisuje prosečnu cenu po marki automobila
- (b) za maksimalnu cenu koju je kupac spreman da plati, a koja se zadaje kao argument komandne linije, ispisuje automobile u tom cenovnom rangu

3 Ulaz i izlaz programa

Prepostaviti da se model i marka sastoje od jedne reči, da svaka od njih sadrži najviše 30 karaktera i da se u datoteci nalaze podaci za najviše 100 automobila. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
POKRETANJE: ./a.out 5000
INTERAKCIJA SA PROGRAMOM:
Unesite naziv datoteke:
dat1.txt

DAT1.TXT NE POSTOJI

IZLAZ ZA GREŠKE:
Greska: neuspesno otvaranje
ulazne datoteke.
```

Primer 2

```
POKRETANJE: ./a.out
IZLAZ ZA GREŠKE:
Greska: neispravan poziv.
```

Primer 3

```
POKRETANJE: ./a.out 4000
INTERAKCIJA SA PROGRAMOM:
Unesite naziv datoteke:
dat1.txt

DAT1.TXT
7
renault twingo 2900
renault megan 6250
renault clio 3650
dacia logan 5400
dacia sandero 7800
```

Primer 3 (nastavak)

```
fiat bravo 4900
fiat linea 4290

IZLAZ:
Informacije o prosecnoj
ceni po markama:
renault 4266.67
dacia 6600.00
fiat 4595.00

Kola u Vasem cenovnom rangu:
renault twingo 2900
renault clio 3650
```

[Rešenje 3.3.24]

Zadatak 3.3.25 Kao argumenti komandne linije zadaju se ime datoteke i ceo broj k . Napisati program koji na standardni izlaz ispisuje sve linije zadate datoteke čija je dužina veća od k . Prepostaviti da je maksimalna dužina linije 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
POKRETANJE: ./a.out test.txt 7
```

```
TEST.TXT  
Teme koje su obradjivane:  
Petlje  
Funkcije  
Nizovi  
Strukture
```

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

Primer 2

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

```
IZLAZ ZA GREŠKE:  
Greska: neispravan poziv.
```

[Rešenje 3.3.25]

Zadatak 3.3.26 Napisati program koji u datoteci čije se ime navodi kao argument komandne linije određuje liniju maksimalne dužine i ispisuje je na standardni izlaz. Ukoliko ima više takvih linija, ispisati onu koja je leksikografski prva. Prepostaviti da je maksimalna dužina linije 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

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

```
TEST.TXT  
Danas je veoma hladno decembarsko  
popodne. Ne pada sneg, kazu mozda  
ce sutra.
```

```
IZLAZ:  
Danas je veoma hladno decembarsko
```

Primer 2

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

```
IN.TXT NE POSTOJI  
IZLAZ ZA GREŠKE:  
Greska: neuspesno otvaranje  
ulazne datoteke.
```

[Rešenje 3.3.26]

Zadatak 3.3.27 U datoteci čije se ime navodi kao prvi argument komandne linije navedena je reč *r* i niz linija. Napisati program koji u datoteku čije se ime navodi kao drugi argument komandne linije upisuje sve linije prve datoteke u kojima se reč *r* pojavljuje bar *n* puta gde je *n* pozitivan ceo broj koji se unosi sa standardnog ulaza. Prilikom prebrojavanja, računaju se i samostalna pojavljivanja reči *r* i pojavljivanja u okviru neke druge reči. Ispis treba da bude u formatu *broj_pojavljivanja:linija*. Prepostaviti da je maksimalna dužina reči 100 karaktera, a linije 500 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

3 Ulaz i izlaz programa

Primer 1

```
POKRETANJE: ./a.out input.txt output.txt  
  
INTERAKCIJA SA PROGRAMOM:  
Unesite broj n: 2  
  
INPUT.TXT  
sto  
stolica lampa  
postotak Stopiranje stopa  
presto Ostaja stotina prostorija  
  
OUTPUT.TXT  
2: postotak Stopiranje stopa  
4: presto Ostaja stotina prostorija
```

Primer 2

```
POKRETANJE: ./a.out input.txt output.txt  
  
INTERAKCIJA SA PROGRAMOM:  
Unesite broj n: 3  
  
INPUT.TXT  
red  
redar za ovu nedelju  
redosled ured  
odrediti raspored  
  
OUTPUT.TXT
```

Primer 3

```
POKRETANJE: ./a.out in.txt out.txt  
  
IN.TXT NE POSTOJI  
  
IZLAZ ZA GREŠKE:  
Greska: neuspesno otvaranje  
ulazne datoteke.
```

Primer 4

```
POKRETANJE: ./a.out in.txt  
  
IZLAZ ZA GREŠKE:  
Greska: neispravan poziv.
```

[Rešenje 3.3.27]

Zadatak 3.3.28 Napisati program koji prebrojava koliko se linija datoteke *ulaz.txt* završava niskom *s* koja se učitava sa standardnog ulaza. Prepostaviti da je maksimalna dužina linije 80 karaktera, a niske *s* 20 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

Primer 1

```
ULAZ.TXT  
/home/korisnik/imena.txt  
/home/korisnik/a.out  
/home/cv.pdf  
/home/korisnik/ulaz.txt  
/home/rezultati.xlsx  
/var/log/apache2/error.log  
  
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku s: .txt  
Broj linija: 2
```

Primer 2

```
ULAZ.TXT  
/var/log/apache2/error.log  
/var/log/dpkg.log  
moj_log.log  
/home/korisnik.login  
/home/korisnik.log.txt  
  
INTERAKCIJA SA PROGRAMOM:  
Unesite nisku s: .log  
Broj linija: 3
```

[Rešenje 3.3.28]

Zadatak 3.3.29 Napisati program koji linije koje se učitavaju sa standardnog ulaza sve do kraja ulaza prepisuje u datoteku *izlaz.txt* i to, ako je prilikom

pokretanja zadata opcija `-v` ili `-V` samo one linije koje počinju velikim slovom, ako je zadata opcija `-m` ili `-M` samo one linije koje počinju malim slovom, a ako je opcija izostavljena sve linije. Pretpostaviti da je maksimalna dužina linije 80 karaktera. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

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
```

IZLAZ ZA GREŠKE:

Greska: neispravna opcija.

Primer 4

```
POKRETANJE: ./a.out
```

IZLAZ ZA GREŠKE:

Greska: neispravan poziv.

[Rešenje 3.3.29]

Zadatak 3.3.30 Napisati program koji poredi dve datoteke i ispisuje redni broj linija u kojima se datoteke razlikuju. Imena datoteka se zadaju kao argumenti komandne linije. Pretpostaviti da je maksimalna dužina linije 200 karaktera. Linije brojati počevši od 1. U slučaju greške, na standardni izlaz za greške ispisati odgovarajuću poruku.

3 Ulaz i izlaz programa

Primer 1

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

Primer 2

```
POKRETANJE:  
./a.out u1.dat u2.dat  
  
U1.DAT:  
danasm vezbamo  
analizu  
ovo je primer kad  
su datoteke razlicite  
  
U2.DAT:  
danasm vezbamo  
programiranje  
ovo je primer kad su  
datoteke razlicite  
  
IZLAC:  
2 3 4
```

Primer 3

```
POKRETANJE:  
./a.out prva.dat druga.dat  
  
PRVA.DAT:  
ovo je primer  
kada su  
datoteke  
razlicite  
duzine  
  
DRUGA.DAT:  
ovo je primer kada  
su  
datoteke  
razlicite  
duzine  
i kada treba ispisati broj  
tih redova  
  
IZLAC:  
1 2 4 5 6 7
```

[Rešenje 3.3.30]

3.4 Rešenja

Rešenje 3.3.1

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracije potrebnih promenljivih. */
6     int c;
7
8     /* Promenljive ulaz i izlaz predstavljaju pokazivace na ugradjenu
9      strukturu FILE. Unutar ove strukture se nalaze polja neophodna
10     za rad sa datotekama. */
11    FILE *ulaz, *izlaz;
12
13    /* Funkcija fopen sluzi da otvori datoteku. Prvi argument je
14      putanja do datoteke koja se otvara, a drugi argument je niska
15      koja moze imati vrednosti "r", "r+", "w", "w+", "a", "a+".
16      Kada ovaj argument ima vrednost "r" datoteka se otvara za
17      citanje. Ukoliko datoteka ne postoji, funkcija fopen kao
18      povratnu vrednost vraca NULL. */
19    ulaz = fopen("ulaz.txt", "r");
20    if (ulaz == NULL) {
21        /* Funkcija fprintf vrsti ispis u datoteku. Funkcionise isto kao
22           i funkcija printf - razlika je sto se kao prvi argument
23           prosledjuje datoteka u koju se ispisuje izlaz.
24
25           Ukoliko je izlaz potrebno ispisati na standardni izlaz za
26           greske, kao prvi argument se navodi stderr. */
27        fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
28                "ulaz.txt.\n");
29        exit(EXIT_FAILURE);
30    }
31
32    /* Ukoliko je drugi argument funkcije fopen "w", tada se
33       prosledjena datoteka otvara za pisanje. */
34    izlaz = fopen("izlaz.txt", "w");
35    if (izlaz == NULL) {
36        fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
37                "izlaz.txt. \n");
38        exit(EXIT_FAILURE);
39    }
40
41    /* Funkcija fgetc ucitava jedan karakter iz datoteke ulaz.
42       Povratna vrednost ove funkcije je ASCII kod unetog karaktera.
43       Funkcija fputc ispisuje karakter c u datoteku izlaz. */
44    while ((c = fgetc(ulaz)) != EOF)
45        fputc(c, izlaz);
46
47    /* Nakon zavrsetka rada sa datotekama, neophodno ih je zatvoriti

```

3 Ulaz i izlaz programa

```
    pomocu ugradjene funkcije fclose. */
49    fclose(ulaz);
50    fclose(izlaz);
51
52    exit(EXIT_SUCCESS);
53 }
```

Rešenje 3.3.2

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

4 int main() {
/* Deklaracije potrebnih promenljivih. */
6     FILE *ulaz, *izlaz;
8     int c;
10
12     /* Otvaranje datoteke ulaz.txt za citanje i provera uspeha. */
14     ulaz = fopen("ulaz.txt", "r");
16     if (ulaz == NULL){
18         fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
20             "ulaz.txt.\n");
22         exit(EXIT_FAILURE);
24
26     /* Otvaranje datoteke izlaz.txt za pisanje i provera uspeha. */
28     izlaz = fopen("izlaz.txt", "w");
30     if (izlaz == NULL){
32         fprintf(stderr, "Greska: neuspesno otvaranje datoteke "
34             "izlaz.txt. \n");
36         exit(EXIT_FAILURE);
38
40     /* Citanje karaktera iz ulazne datoteke. */
42     while ((c = fgetc(ulaz)) != EOF){
44         /* Upisivanje procitanog karaktera u izlaznu datoteku. */
46         fputc(c, izlaz);

48         /* Preskakanje naredna dva karaktera. */
50         fgetc(ulaz);
52         fgetc(ulaz);

54         /* Ovakvo resenje ce raditi i u slucaju kada broj karaktera u
56             datoteci nije deljiv sa 3 jer kada se dodje do kraja
58             datoteke svaki sledeci poziv funkcije fgetc vraca EOF. */
60     }
62
64     /* Zatvaranje otvorenih datoteka. */
66     fclose(izlaz);
68     fclose(ulaz);
70 }
```

```
44 } exit(EXIT_SUCCESS);
```

Rešenje 3.3.3

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>

5 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i prekida izvrsavanje programa. */
7 void greska(char *poruka) {
8     fprintf(stderr, "%s\n", poruka);
9     exit(EXIT_FAILURE);
10 }

11 int main() {
12     /* Deklaracije potrebnih promenljivih. */
13     FILE *ulaz, *izlaz;
14     char c;

17     /* Otvaranje datoteke podaci.txt za citanje i provera uspeha. */
18     ulaz = fopen("podaci.txt", "r");
19     if (ulaz == NULL)
20         greska("Greska: neuspesno otvaranje datoteke podaci.txt.");
21
22     /* Otvaranje datoteke sifra.txt za pisanje i provera uspeha. */
23     izlaz = fopen("sifra.txt", "w");
24     if (izlaz == NULL)
25         greska("Greska: neuspesno otvaranje datoteke sifra.txt.");

27     /* Citanje karaktera iz ulazne datoteke. */
28     while ((c = fgetc(ulaz)) != EOF) {
29         /* Sifrovanje procitanog karaktera na trazeni nacin. */
30         if (islower(c)) {
31             c = toupper(c);
32             if (c == 'A')
33                 c = 'Z';
34             else
35                 c = c - 1;
36         } else if (isupper(c)) {
37             c = tolower(c);
38             if (c == 'a')
39                 c = 'z';
40             else
41                 c = c - 1;
42         }
43
44         /* Upisivanje izmenjenog karaktera u izlaznu datoteku. */
45         fputc(c, izlaz);
46     }
}
```

3 Ulaz i izlaz programa

```
47     /* Zatvaranje datoteka. */
48     fclose(ulaz);
49     fclose(izlaz);
50
51     exit(EXIT_SUCCESS);
52 }
```

Rešenje 3.3.4

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_IME 21
6
7 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
8    izlaz za greske i prekida izvrsavanje programa. */
9 void greska(char *poruka) {
10     fprintf(stderr, "%s\n", poruka);
11     exit(EXIT_FAILURE);
12 }
13
14 int main() {
15     /* Deklaracije potrebnih promenljivih. */
16     FILE *ulaz, *izlaz;
17     char c;
18     char ime_datoteke1[MAKS_IME], ime_datoteke2[MAKS_IME];
19
20     /* Ucitavanje imena datoteka. */
21     printf("Unesite ime prve datoteke: ");
22     scanf("%s", ime_datoteke1);
23     printf("Unesite ime druge datoteke: ");
24     scanf("%s", ime_datoteke2);
25
26     /* Otvaranje prve datoteke za citanje i provera uspeha. */
27     ulaz = fopen(ime_datoteke1, "r");
28     if (ulaz == NULL)
29         greska("Greska: neuspesno otvaranje prve datoteke.");
30
31     /* Otvaranje druge datoteke za pisanje i provera uspeha. */
32     izlaz = fopen(ime_datoteke2, "w");
33     if (izlaz == NULL)
34         greska("Greska: neuspesno otvaranje druge datoteke.");
35
36     /* Iz datoteke se cita karakter po karakter i za svaku procitanu
37        cifru u izlaznu datoteku se upisuje 0, za svako slovo 1, a za
38        ostale karaktere 2. */
39     while ((c = fgetc(ulaz)) != EOF) {
40         if (isdigit(c))
41             fprintf(izlaz, "0");
```

```

43     else if (isalpha(c))
44         fprintf(izlaz, "1");
45     else
46         fprintf(izlaz, "2");
47
48     /* Zatvaranje datoteka. */
49     fclose(ulaz);
50     fclose(izlaz);
51
52     exit(EXIT_SUCCESS);
53 }

```

Rešenje 3.3.5 Pogledajte zadatke 3.3.3 i 3.3.4.

Rešenje 3.3.6 Pogledajte zadatke 3.3.3 i 3.3.4.

Rešenje 3.3.7

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_IME 21
6 #define BROJ_CIFARA 10
7
8 int main() {
9     /* Deklaracije potrebnih promenljivih. */
10    FILE *ulaz;
11    char c;
12    char ime_datoteke[MAKS_IME];
13    int brojaci[BROJ_CIFARA];
14    int i, maks;
15
16    /* Ucitavanje imena ulazne datoteke. */
17    printf("Unesite ime datoteke: ");
18    scanf("%s", ime_datoteke);
19
20    /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
21    ulaz = fopen(ime_datoteke, "r");
22    if (ulaz == NULL) {
23        fprintf(stderr, "Greska: neuspesno otvaranje datoteke.\n");
24        exit(EXIT_FAILURE);
25    }
26
27    /* Brojaci za cifre se inicijalizuju na nule. Indeks niza brojaci
28     oznacava cifru (brojaci[0] se koristi za prebrojavanje cifre
29     0, brojaci[1] za 1, ..., brojaci[9] za cifru 9). */
30    for (i = 0; i < BROJ_CIFARA; i++)
31        brojaci[i] = 0;

```

3 Ulaz i izlaz programa

```
33  /* Citanje karaktera i uvecavanje odgovarajucih brojaca. */
34  while ((c = fgetc(ulaz)) != EOF) {
35      if (isdigit(c))
36          brojaci[c - '0']++;
37  }
38
39  /* Pronalazak cifre koja se najvise puta pojavljuje u datoteci. */
40  maks = brojaci[0];
41  for (i = 1; i < BROJ_CIFARA; i++)
42      if (brojaci[i] > maks)
43          maks = brojaci[i];
44
45  /* Ispis rezultata. */
46  if(maks == 0)
47      printf("Datoteka ne sadrzi cifre.\n");
48  else {
49      printf("Najcesce cifre: ");
50      for (i = 0; i < BROJ_CIFARA; i++)
51          if (brojaci[i] == maks)
52              printf("%d ", i);
53      printf("\n");
54  }
55
56  /* Zatvaranje datoteke. */
57  fclose(ulaz);
58
59  exit(EXIT_SUCCESS);
}
```

Rešenje 3.3.8

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

/* Funkcija ispisuje prosledjenu poruku o gresci na standardni
izlaz za greske i prekida izvrsavanje programa. */
void greska(char *poruka) {
    fprintf(stderr, "%s\n", poruka);
    exit(EXIT_FAILURE);
}

int main(int argc, char **argv) {
    /* Deklaracije potrebnih promenljivih. */
    FILE *ulaz;
    char c;
    int broj_zagrada = 0, nisu_uparene = 0;

    /* Provera broja argumenata komandne linije. */
    if (argc != 2)
        greska("Greska: neispravan poziv.");
```

```

1  /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
2  ulaz = fopen(argv[1], "r");
3  if (ulaz == NULL)
4      greska("Greska: neuspesno otvaranje datoteke.");
5
6  /* Cita se karakter po karakter i proverava se da li je procitana
7   zagrada. Ako se naidje na otvorenu zagrada, brojac se uvecava.
8   Ako se naidje na zatvorenu zagrada, brojac se smanjuje. Zgrade
9   su ispravno uparene ukoliko je ovaj brojac na kraju 0. Dodatno,
10  ukoliko brojac u bilo kom trenutku postane negativan, to znaci
11  da je zatvorena zagrada procitana pre otvorene, tako da ni u
12  tom slučaju zgrade nisu uparene. */
13
14  while ((c = fgetc(ulaz)) != EOF) {
15      if (c == '(')
16          broj_zagrada++;
17      else if (c == ')')
18          broj_zagrada--;
19
20      if (broj_zagrada < 0) {
21          nisu_uparene = 1;
22          break;
23      }
24  }
25
26  /* Ispis rezultata. */
27  if (broj_zagrada != 0 || nisu_uparene)
28      printf("Zgrade nisu uparene.\n");
29  else
30      printf("Zgrade jesu uparene.\n");
31
32  /* Zatvaranje datoteke. */
33  fclose(ulaz);
34
35  exit(EXIT_SUCCESS);
36 }
```

Rešenje 3.3.9

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4
5 #define MAKS_ELEMENATA 1000
6
7 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
8   izlaz za greske i prekida izvršavanje programa. */
9 void greska(char *poruka) {
10     fprintf(stderr, "%s\n", poruka);
11     exit(EXIT_FAILURE);
12 }
13 }
```

3 Ulaz i izlaz programa

```
15  /* Funkcija ucitava karaktere iz datoteke i smesta ih u niz s. */
16  int ucitaj_karaktere(char s[], FILE *f) {
17      char c;
18      int n = 0;
19
20      /* Citanje karaktera do kraja datoteke ili dok se ne ucita
21         MAKS_ELEMENATA ili dok se ne dodje do karaktera koji nije ni
22         slovo ni cifra. */
23      while((c = fgetc(f)) != EOF && n < MAKS_ELEMENATA) {
24          if(isalpha(c) || isdigit(c))
25              s[n] = c;
26          else
27              break;
28
29          n++;
30      }
31
32      return n;
33  }
34
35  /* Funkcija racuna koliko slova i koliko cifara se nalazi u nizu
36     s.*/
37  void prebroj(char s[], int n, int *broj_slova, int *broj_cifara) {
38      int i;
39
40      /* Inicijalizacija brojaca. */
41      *broj_slova = *broj_cifara = 0;
42
43      /* Prebrojavanje slova i cifara. */
44      for(i=0; i<n; i++)
45          if(isalpha(s[i]))
46              (*broj_slova)++;
47          else
48              (*broj_cifara)++;
49
50  int main(int argc, char* argv[]) {
51      /* Deklaracije potrebnih promenljivih. */
52      FILE* ulaz;
53      char karakteri[MAKS_ELEMENATA];
54      int broj_elemenata, broj_slova, broj_cifara;
55
56      /* Provera broja argumenata komandne linije. */
57      if (argc != 2)
58          greska("Greska: neispravan poziv.");
59
60      /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
61      ulaz = fopen(argv[1], "r");
62      if (ulaz == NULL)
63          greska("Greska: neuspesno otvaranje datoteke.");
64
65      /* Ucitavanje karaktera datoteke. */
```

```

67     broj_elemenata = ucitaj_karaktere(karakteri, ulaz);
68
69     /* Racunanje i ispis rezultata. */
70     prebroj(karakteri, broj_elemenata, &broj_slova, &broj_cifara);
71     printf("Broj slova: %d\n", broj_slova);
72     printf("Broj cifara: %d\n", broj_cifara);
73
74     /* Zatvaranje datoteke. */
75     fclose(ulaz);
76
77     exit(EXIT_SUCCESS);
}

```

Rešenje 3.3.10

```

1 #include <stdio.h>
2 #include <string.h>
3 #include <stdlib.h>
4
5 #define MAKS_NISKA 21
6
7 /* Funkcija rotira nisku s duzine n za jedno mesto u levo. */
8 void rotiraj_za_1(char *s, int n) {
9     int i;
10    char c = s[0];
11
12    for (i = 0; i < n - 1; i++)
13        s[i] = s[i + 1];
14
15    s[n - 1] = c;
16}
17
18 int main() {
19     /* Deklaracije potrebnih promenljivih. */
20     char s[MAKS_NISKA];
21     int n, i;
22     FILE *izlaz;
23
24     /* Otvaranje datoteke rotacije.txt za pisanje i provera uspeha. */
25     izlaz = fopen("rotacije.txt", "w");
26     if (izlaz == NULL) {
27         fprintf(stderr, "Greska: neuspesno otvaranje datoteke.");
28         exit(EXIT_FAILURE);
29     }
30
31     /* Ucitavanje reci koju je potrebno rotirati. */
32     printf("Unesite rec: ");
33     scanf("%s", s);
34
35     /* Racunanje duzine reci. */
36     n = strlen(s);
}

```

3 Ulaz i izlaz programa

```
38  /* U petlji se uneta rec rotira za 1 i upisuje u datoteku.
   Postupak se ponavlja n puta. */
40  for (i = 0; i < n; i++) {
41      fprintf(izlaz, "%s\n", s);
42      rotiraj_za_1(s, n);
43  }
44
45  /* Zatvaranje datoteke. */
46  fclose(izlaz);
47
48  exit(EXIT_SUCCESS);
49 }
```

Rešenje 3.3.11

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_REC 101
6 #define MAKSIME 21
7
8 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i prekida izvrsavanje programa. */
9 void greska(char *poruka) {
10     fprintf(stderr, "%s\n", poruka);
11     exit(EXIT_FAILURE);
12 }
13
14 /* Funkcija u nisku rezultat smesta nisku rec rotiranu za k mesta u
   desno. */
15 void rotiraj(char *rec, int k, char *rezultat) {
16     int i, n;
17
18     /* Racunanje duzine reci. */
19     n = strlen(rec);
20
21     /* Ako je duzina reci npr. 5, a k ima vrednost 13, onda je
       zapravo potrebno izvrsiti rotaciju za 3 mesta (nema potrebe da
       se vrte dva cela kruga pre toga). */
22     k = k % n;
23
24     /* Karakteri koji se u pocetnoj reci nalaze na pozicijama od 0 do
       k-1, u rezultujucoj reci treba da budu na pozicijama od n-k do
       n-1. */
25     for (i = 0; i < k; i++)
26         rezultat[n - k + i] = rec[i];
27
28     /* Slicno, karakteri koji se u pocetnoj reci nalaze na pozicijama
       od k do n-1, u rezultujucoj reci treba da budu na pozicijama od
```

```

    0 do n-k-1. */
37   for (i = k; i < n; i++)
38     rezultat[i - k] = rec[i];
39
40   /* Na kraj rezultujuće niske se upisuje terminirajuća nula. */
41   rezultat[n] = '\0';
42 }
43
44 int main() {
45   /* Deklaracije potrebnih promenljivih. */
46   FILE *ulaz, *izlaz;
47   char ime_datoteke[MAKS_IME];
48   char rec[MAKS_REC], rezultat[MAKS_REC];
49   int k;
50
51   /* Ucitavanje imena ulazne datoteke. */
52   printf("Unesite ime datoteke: ");
53   scanf("%s", ime_datoteke);
54
55   /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
56   ulaz = fopen(ime_datoteke, "r");
57   if (ulaz == NULL)
58     greska("Greska: neuspesno otvaranje ulazne datoteke.");
59
60   /* Otvaranje datoteke rotirano.txt za citanje i provera uspeha. */
61   izlaz = fopen("rotirano.txt", "w");
62   if (izlaz == NULL)
63     greska("Greska: neuspesno otvaranje izlazne datoteke.");
64
65   /* Ucitavanje broja k. */
66   printf("Unesite broj k: ");
67   scanf("%d", &k);
68   if (k < 0)
69     greska("Greska: neispravan unos broja k.");
70
71   /* Citanje reci iz ulazne datoteke sve do kraja datoteke. */
72   while (fscanf(ulaz, "%s", rec) != EOF) {
73     /* Rotiranje procitane reci i upisivanje u izlaznu datoteku. */
74     rotiraj(rec, k, rezultat);
75     fprintf(izlaz, "%s ", rezultat);
76   }
77
78   /* Zatvaranje datoteka. */
79   fclose(ulaz);
80   fclose(izlaz);
81
82   exit(EXIT_SUCCESS);
83 }

```

Rešenje 3.3.12

Pogledajte zadatke 3.3.11 i 2.1.18.

Rešenje 3.3.13

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>

5 #define MAKS_BROJ_REC 256
6 #define MAKS_DUZINA_REC 51
7 #define MAKS_IME 21

9 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
10    izlaz za greske i prekida izvrsavanje programa. */
11 void greska(char *poruka) {
12     fprintf(stderr, "%s\n", poruka);
13     exit(EXIT_FAILURE);
14 }

15 int main() {
16     /* Deklaracije potrebnih promenljivih. */
17     char ime_datoteke[MAKS_IME];
18     char niz_reci[MAKS_BROJ_REC][MAKS_DUZINA_REC];
19     FILE *ulaz, *izlaz;
20     int n, i, k, indikator;

23     /* Ucitavanje imena ulazne datoteke. */
24     printf("Unesite ime datoteke: ");
25     scanf("%s", ime_datoteke);

27     /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
28     ulaz = fopen(ime_datoteke, "r");
29     if (ulaz == NULL)
30         greska("Greska: neuspesno otvaranje ulazne datoteke.");
31

32     /* Iz datoteke se ucitava broj reci. */
33     fscanf(ulaz, "%d", &n);
34     if (n < 0 || n > MAKS_BROJ_REC)
35         greska("Greska: neispravna vrednost broja reci.");

37     /* Ucitavanje reci u niz. */
38     for (i = 0; i < n; i++)
39         fscanf(ulaz, "%s", niz_reci[i]);

41     /* Otvaranje izlazne datoteke za pisanje i provera uspeha. */
42     izlaz = fopen("bez_duplikata.txt", "w");
43     if (izlaz == NULL)
44         greska("Greska: neuspesno otvaranje izlazne datoteke.");
45

46     /* U izlaznu datoteku se upisuju reci, izostavljajuci duplike. */
47     for (i = 0; i < n; i++) {
48         /* Za rec na poziciji i se proverava da li se ona nalazi negde
49             na pozicijama od 0 do i. Ukoliko se nalazi, to znači da je
50             vec upisana u datoteku i da je treba preskociti. U tom
```

```

51     slucaju vrednost promenljive indikator ce biti postavljena
52     na 1. */
53     indikator = 0;
54     for (k = 0; k < i; k++) {
55         if (strcmp(niz_reci[k], niz_reci[i]) == 0) {
56             indikator = 1;
57             break;
58         }
59
60         /* Ako indikator ima vrednost 0, znaci da je u pitanju prvo
61            pojavljivanje reci i da je treba upisati u izlaznu
62            datoteku. */
63         if (!indikator)
64             fprintf(izlaz, "%s\n", niz_reci[i]);
65     }
66
67     /* Zatvaranje datoteka. */
68     fclose(ulaz);
69     fclose(izlaz);
70
71     exit(EXIT_SUCCESS);
72 }
```

Rešenje 3.3.14

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
6    izlaz za greske i prekida izvrsavanje programa. */
7 void greska(char *poruka) {
8     fprintf(stderr, "%s\n", poruka);
9     exit(EXIT_FAILURE);
10 }
11
12 /* Funkcija racuna broj cifara broja x. */
13 int broj_cifara(int x) {
14     int brojac = 0;
15
16     do {
17         brojac++;
18         x /= 10;
19     } while (x);
20
21     return brojac;
22 }
23
24 /* Funkcija broji koliko ima k-tocifrenih brojeva u datoteci f. */
25 int prebrojavanje(FILE *f, int k) {
26     int n, broj, i, brojac;
```

3 Ulaz i izlaz programa

```
27     /* Ucitavanje broja brojeva u datoteci. */
29     fscanf(f, "%d", &n);
30     if (n <= 0)
31         greska("Greska: neispravna vrednost broja n.");
32
33     /* Cita se broj po broj i za svaki procitani broj se racuna broj
34      cifara. Ukoliko je on jednak k, uvecava se odgovarajuci
35      brojac. */
36     brojac = 0;
37     for (i = 0; i < n; i++) {
38         fscanf(f, "%d", &broj);
39         if (broj_cifara(broj) == k)
40             brojac++;
41     }
42
43     /* Povratna vrednost funkcije je broj k-tocifrenih brojeva. */
44     return brojac;
45 }
46
47 int main(int argc, char *argv[]) {
48     /* Deklaracije potrebnih promenljivih. */
49     int k;
50     FILE *ulaz;
51
52     /* Provera broja argumenata komandne linije. */
53     if (argc != 3)
54         greska("Greska: neispravan poziv.");
55
56     /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
57     ulaz = fopen(argv[1], "r");
58     if (ulaz == NULL)
59         greska("Greska: neuspesno otvaranje ulazne datoteke.");
60
61     /* Citanje broja k i provera ispravnosti. */
62     k = atoi(argv[2]);
63     if (k <= 0)
64         greska("Greska: neispravna vrednost broja k.");
65
66     /* Ispis rezultata. */
67     printf("Broj %d-cifrenih brojeva: %d\n", k,
68           prebrojavanje(ulaz, k));
69
70     /* Zatvaranje datoteke. */
71     fclose(ulaz);
72
73     exit(EXIT_SUCCESS);
74 }
```

Rešenje 3.3.15

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i prekida izvršavanje programa. */
5 void greska(char *poruka) {
6     fprintf(stderr, "%s\n", poruka);
7     exit(EXIT_FAILURE);
8 }
9
10 int main() {
11     /* Deklaracije potrebnih promenljivih. */
12     FILE *ulaz;
13     float broj, najveci_broj;
14
15     /* Otvaranje datoteke brojevi.txt za citanje i provera uspeha. */
16     ulaz = fopen("brojevi.txt", "r");
17     if (ulaz == NULL)
18         greska("Greska: neuspesno otvaranje ulazne datoteke.");
19
20     /* Promenljiva u koju se smesta najveci broj se inicijalizuje na
       prvi broj iz datoteke. Ukoliko se pri prvom citanju dodje do
       kraja datoteke, ispisuje se odgovarajuća poruka. */
21     if (fscanf(ulaz, "%f", &najveci_broj) == EOF)
22         greska("Greska: datoteka je prazna.");
23
24     /* Iz datoteke se cita broj po broj, sve dok se ne dodje do kraja
       datoteke i trazi se najveci procitani broj. */
25     while (fscanf(ulaz, "%f", &broj) != EOF)
26         if (broj > najveci_broj)
27             najveci_broj = broj;
28
29     /* Ispis rezultata. */
30     printf("Najveci broj je: %g\n", najveci_broj);
31
32     /* Zatvaranje datoteke. */
33     fclose(ulaz);
34
35     exit(EXIT_SUCCESS);
36 }

```

Rešenje 3.3.16

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAKS_DIM 50
5
6 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni

```

3 Ulaz i izlaz programa

```
    izlaz za greske i prekida izvrsavanje programa. */
8 void greska(char *poruka) {
9     fprintf(stderr, "%s\n", poruka);
10    exit(EXIT_FAILURE);
11}
12
13 int main() {
14     /* Deklaracije potrebnih promenljivih. */
15     FILE *ulaz;
16     float a[MAKS_DIM][MAKS_DIM];
17     int i, j, n, m, k, l, suma = 0;
18
19     /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
20     ulaz = fopen("matrica.txt", "r");
21     if (ulaz == NULL)
22         greska("Greska: neuspesno otvaranje ulazne datoteke.");
23
24     /* Ucitavanje dimenzija matrice i provera ispravnosti. */
25     fscanf(ulaz, "%d%d", &n, &m);
26     if (n <= 0 || n > MAKS_DIM || m <= 0 || m > MAKS_DIM)
27         greska("Greska: neispravne dimenzije matrice.");
28
29     /* Ucitavanje elemenata matrice. */
30     for (i = 0; i < n; i++)
31         for (j = 0; j < m; j++)
32             fscanf(ulaz, "%f", &a[i][j]);
33
34     /* Za svaku poziciju (i,j) vrsti se provera trazenog uslova. */
35     for (i = 0; i < n; i++) {
36         for (j = 0; j < m; j++) {
37             /* Za poziciju (i,j) racuna se suma suseda. Ona se moze
38                 dobiti kao suma podmatrice 3*3 ciji je gornji levi ugao
39                 (i-1, j-1), a donji desni (i+1, j+1). Pri racunanju ove
40                 sume treba voditi racuna da se ne izadje iz granica
41                 originalne matrice. */
42             suma = 0;
43             for (k = i - 1; k <= i + 1; k++) {
44                 for (l = j - 1; l <= j + 1; l++) {
45                     /* Ako se nije izaslo iz granica originalne matrice,
46                         vrednost a[k][l] se dodaje na sumu. */
47                     if (k >= 0 && k < n && l >= 0 && l < m)
48                         suma += a[k][l];
49                 }
50             }
51
52             /* Kako suma uključuje i centralni element (i,j), njega je
53                 potrebno oduzeti jer je potrebno sumirati samo njegove
54                 susede. */
55             suma -= a[i][j];
56
57             /* Ako je suma suseda elementa a[i][j] jednaka njegovoj
58                 vrednosti, odgovarajuce pozicije i vrednost elementa se
```

```

    ispisuju na standardni izlaz. */
60   if (a[i][j] == suma)
61     printf("(%d, %d, %g)\n", i, j, a[i][j]);
62 }
63
64 /* Zatvaranje datoteke. */
65 fclose(ulaz);
66
67 exit(EXIT_SUCCESS);
68 }
```

Rešenje 3.3.17 Pogledajte zadatak 3.3.16.

Rešenje 3.3.18

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <math.h>
4
5 #define MAKS_TACAKA 50
6
7 /* Struktura koja opisuje tacku. */
8 typedef struct {
9   int x, y;
10 } Tacka;
11
12 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
13 izlaz za greske i prekida izvrsavanje programa. */
14 void greska(char *poruka) {
15   fprintf(stderr, "%s\n", poruka);
16   exit(EXIT_FAILURE);
17 }
18
19 /* Funkcija racuna rastojanje tacke od koordinatnog pocetka. */
20 double rastojanje_od_koordinatnog_pocetka(const Tacka *a) {
21   return sqrt(pow(a->x, 2) + pow(a->y, 2));
22 }
23
24 int main() {
25   /* Deklaracije potrebnih promenljivih. */
26   FILE *ulaz, *izlaz;
27   int n, i;
28   Tacka tacka, maks_tacka;
29   double rastojanje, maks_rastojanje = -1;
30
31   /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
32   ulaz = fopen("tacke.txt", "r");
33   if (ulaz == NULL)
34     greska("Greska: neuspesno otvaranje ulazne datoteke.");
35 }
```

3 Ulaz i izlaz programa

```
37  /* Otvaranje izlazne datoteke za pisanje i provera uspeha. */
38  izlaz = fopen("rastojanja.txt", "w");
39  if (izlaz == NULL)
40      greska("Greska: neuspesno otvaranje izlazne datoteke.");
41
42  /* Ucitavanje broja tacaka i provera ispravnosti. */
43  fscanf(ulaz, "%d", &n);
44  if (n < 0 || n > MAKS_TACAKA)
45      greska("Greska: neispravan broj tacaka.");
46
47  /* Citanje tacaka iz datoteke. */
48  for (i = 0; i < n; i++) {
49      fscanf(ulaz, "%d%d", &tacka.x, &tacka.y);
50
51  /* Racunanje rastojanja tacke t od koordinatnog pocetka. */
52  rastojanje = rastojanje_od_koordinatnog_pocetka(&tacka);
53
54  /* Upisivanje izracunatog rastojanja u datoteku. */
55  fprintf(izlaz, "%.2lf\n", rastojanje);
56
57  /* Azuriranje maksimalnog rastojanja i odgovarajuce tacke. */
58  if (rastojanje > maks_rastojanje) {
59      maks_rastojanje = rastojanje;
60      maks_tacka = tacka;
61  }
62
63  /* Ispis rezultata. */
64  printf("Najdaljenija tacka: (%d, %d)\n", maks_tacka.x, maks_tacka.
65          y);
66
67  /* Zatvaranje datoteke. */
68  fclose(ulaz);
69
70  exit(EXIT_SUCCESS);
71 }
```

Rešenje 3.3.19 Pogledajte zadatak 3.3.18.

Rešenje 3.3.20

```
2  #include <stdio.h>
3  #include <stdlib.h>
4  #include <string.h>
5
6  #define MAKS_IME 5
7
8  /* Struktura koja opisuje pravougaonik. */
9  typedef struct {
10      unsigned int a, b;
11      char ime[MAKS_IME];
```

```

} Pravougaonik;

/* Funkcija ispisuje posledjenu poruku o gresci na standardni
izlaz za greske i prekida izvrsavanje programa. */
void greska(char *poruka) {
    fprintf(stderr, "%s\n", poruka);
    exit(EXIT_FAILURE);
}

int main(int argc, char *argv[]) {
    /* Deklaracije potrebnih promenljivih. */
    unsigned int maksimalna_povrsina = 0;
    Pravougaonik p;

    /* Provera broja argumenata komandne linije. */
    if (argc != 2)
        greska("Greska: neispravan poziv.");

    /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
    FILE *ulaz = fopen(argv[1], "r");
    if (ulaz == NULL)
        greska("Greska: neuspesno otvaranje ulazne datoteke.");

    /* Citanje podataka o pravougaonicima. */
    while (fscanf(ulaz, "%u%u%s", &p.a, &p.b, p.ime) == 3) {
        /* Provera ispravnosti duzina stranica. */
        if (p.a == 0 || p.b == 0)
            greska("Greska: duzina stranice ne moze biti 0.");

        /* U slucaju da je ucitan kvardat, njegovo ime se ispisuje na
        standardni izlaz. */
        if (p.a == p.b)
            printf("%s ", p.ime);
        else { /* Ako je u pitanju pravougaonik, njegova povrsina
                se poredi sa maksimalnom. */
            if (p.a * p.b > maksimalna_povrsina)
                maksimalna_povrsina = p.a * p.b;
        }
    }

    /* Ukoliko je bilo ucitanih pravougaonika, ispisuje se povrsina
    najveceg. */
    if (maksimalna_povrsina != 0)
        printf("%u\n", maksimalna_povrsina);
    else
        printf("\n");

    /* Zatvaranje datoteke. */
    fclose(ulaz);

    exit(EXIT_SUCCESS);
}

```

Rešenje 3.3.21

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

4 #define MAKS_STUDENATA 100

6 /* Struktura koja opisuje studenta. */
7 typedef struct {
8     char korisnicko_ime[8];
9     float prosek;
10 } Student;

12 int main() {
13     /* Deklaracije potrebnih promenljivih. */
14     FILE *ulaz;
15     Student studenti[MAKS_STUDENATA];
16     int ocena1, ocena2, ocena3, ocena4, ocena5, zbir_ocena;
17     int i = 0, n;
18     float maksimalni_prosek = 0;

20     /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
21     ulaz = fopen("studenti.txt", "r");
22     if (ulaz == NULL) {
23         fprintf(stderr, "Greska: neuspesno otvaranje "
24                 "ulazne datoteke.\n");
25         exit(EXIT_FAILURE);
26     }

28     /* Citanje podataka o studentima sve dok se ne dodje do kraja
29      datoteke. */
30     while (fscanf(ulaz, "%s%d%d%d%d", studenti[i].korisnicko_ime,
31                   &ocena1, &ocena2, &ocena3, &ocena4, &ocena5) != EOF) {
32         /* Racunanje proseka trenutnog studenta. */
33         zbir_ocena = ocena1 + ocena2 + ocena3 + ocena4 + ocena5;
34         studenti[i].prosek = zbir_ocena / 5.0;

36         /* Azuriranje maksimalnog proseka. */
37         if (studenti[i].prosek > maksimalni_prosek)
38             maksimalni_prosek = studenti[i].prosek;

40         /* Prelazak na sledeceg studenta. */
41         i++;
42     }

44     /* Promenljiva n cuva ukupan broj studenata. */
45     n = i;
46
47     /* Ispis svih studenata sa maksimalnim prosekom. */
48     printf("Studenti sa najvecim prosekom:\n");
49     for (i = 0; i < n; i++)
50         if (studenti[i].prosek == maksimalni_prosek)
```

```

52     printf("Korisnicko ime: %s\nProsek ocena: %.2f\n\n",
53             studenti[i].korisnicko_ime, studenti[i].prosek);

54 /* Zatvaranje datoteke. */
55 fclose(ulaz);

56 exit(EXIT_SUCCESS);
57 }

```

Rešenje 3.3.22

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_PUNO_IME 101
6 #define MAKS_OCENA 10

7 /* Struktura koja opisuje studenta. */
8 typedef struct {
9     char puno_ime[MAKS_PUNO_IME];
10    int ocene[MAKS_OCENA];
11    int broj_ocena;
12    float prosek;
13} Student;

14 /* Funkcija ispisuje prosledjenu poruku o greski na standardni
15 izlaz za greske i prekida izvršavanje programa. */
16 void greska(char *poruka) {
17     fprintf(stderr, "%s\n", poruka);
18     exit(EXIT_FAILURE);
19 }

20 int main(int argc, char **argv) {
21     /* Deklaracije potrebnih promenljivih. */
22     FILE *ulaz;
23     char ime[MAKS_PUNO_IME], prezime[MAKS_PUNO_IME];
24     int i = 0, j, ocena, suma_ocena;
25     Student student, maks_student;
26     float maks_prosek;

27     /* Provera broja argumenata komandne linije. */
28     if (argc != 2)
29         greska("Greska: neispravan poziv.");
30
31     /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
32     ulaz = fopen(argv[1], "r");
33     if (ulaz == NULL)
34         greska("Greska: neuspesno otvaranje ulazne datoteke.");
35
36     /* Iz datoteke se ucitavaju podaci o studentima sve dok se ne
37 
```

3 Ulaz i izlaz programa

```
    dodje do kraja datoteke. */
42 while (fscanf(ulaz, "%s%s", ime, prezime) != EOF) {
43     /* Od imena i prezimena se formira puno ime. */
44     strcpy(student.puno_ime, ime);
45     strcat(student.puno_ime, " ");
46     strcat(student.puno_ime, prezime);

47     /* Ucitavanje ocena sve dok se ne ucita broj 0. */
48     j = 0;
49     suma_ocena = 0;
50     while (1) {
51         fscanf(ulaz, "%d", &ocena);
52         if (ocena == 0)
53             break;

54         student.ocene[j] = ocena;
55         suma_ocena += ocena;
56         j++;
57     }

58     /* Racunanje proseka ocena. */
59     student.broj_ocena = j;
60     student.prosek = (float) suma_ocena / j;

61     /* Ukoliko je u pitanju student ciji je prosek veci od trenutno
62      najveceg proseka, pamte se njegovi podaci i azurira se
63      vrednost najveceg proseka. */
64     if (student.prosek > maks_prosek) {
65         maks_prosek = student.prosek;
66         maks_student = student;
67     }
68 }

69 /* Ispis podataka o studentu sa najvecim prosekom. */
70 printf("%s ", maks_student.puno_ime);
71 for (i = 0; i < maks_student.broj_ocena; i++)
72     printf("%d ", maks_student.ocene[i]);
73 printf("%.2f\n", maks_student.prosek);

74 /* Zatvaranje datoteke. */
75 fclose(ulaz);

76 exit(EXIT_SUCCESS);
77 }
```

Rešenje 3.3.23

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
# include <string.h>
```

```

5 #define MAKS_RAZLOMAKA 100
7
8 /* Struktura koja opisuje razlomak. */
9 typedef struct {
10     int brojilac;
11     int imenilac;
12 } Razlomak;
13
14 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
15    izlaz za greske i prekida izvršavanje programa. */
16 void greska(char *poruka) {
17     fprintf(stderr, "%s\n", poruka);
18     exit(EXIT_FAILURE);
19 }
20
21 /* Funkcija ucitava razlomke u niz razlomaka i kao povratnu
22    vrednost vraca broj ucitanih razlomaka. */
23 int ucitaj_razlomke(Razlomak niz[], FILE *f) {
24     int i, n;
25     fscanf(f, "%d", &n);
26
27     for (i = 0; i < n; i++) {
28         fscanf(f, "%d %d", &niz[i].brojilac, &niz[i].imenilac);
29         if (niz[i].imenilac == 0)
30             greska("Greska: Imenilac ne moze biti 0.");
31     }
32     return n;
33 }
34
35 /* Funkcija racuna razlomak reciprocan razlomku r. */
36 Razlomak reciprocni(const Razlomak *r) {
37     if (r->brojilac == 0)
38         greska("Greska: nije moguce izracunati reciprocni razlomak.");
39
40     Razlomak r2;
41     r2.imenilac = r->brojilac;
42     r2.brojilac = r->imenilac;
43     return r2;
44 }
45
46 /* Funkcija racuna brojevnu vrednost razlomka r. */
47 float vrednost(const Razlomak *r) {
48     return 1.0 * r->brojilac / r->imenilac;
49 }
50
51 int main(int argc, char *argv[]) {
52     /* Deklaracije potrebnih promenljivih. */
53     FILE *ulaz, *izlaz;
54     int i, n, opcija_x = 0, opcija_y = 0;
55     Razlomak razlomci[MAKS_RAZLOMAKA];
56     Razlomak r;

```

3 Ulaz i izlaz programa

```
57  /* Provera broja argumenata komandne linije. */
59  if (argc != 4)
60      greska("Greska: neispravan poziv.");
61
62  /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
63  ulaz = fopen(argv[1], "r");
64  if (ulaz == NULL)
65      greska("Greska: neuspesno otvaranje ulazne datoteke.");
66
67  /* Otvaranje izlazne datoteke za pisanje i provera uspeha. */
68  izlaz = fopen(argv[2], "w");
69  if (izlaz == NULL)
70      greska("Greska: neuspesno otvaranje izlazne datoteke.");
71
72  /* Ucitavanje zadate opcije i postavljanje vrednosti
73   odgovarajuceg indikatora. */
74  if (strcmp(argv[3], "-x") == 0)
75      opcija_x = 1;
76  else if (strcmp(argv[3], "-y") == 0)
77      opcija_y = 1;
78  else if (strcmp(argv[3], "-xy") == 0
79          || strcmp(argv[3], "-yx") == 0)
80      opcija_x = opcija_y = 1;
81  else
82      greska("Greska: neispravna opcija.");
83
84  /* Ucitavanje podataka o razlomcima. */
85  n = ucitaj_razlomke(razlomci, ulaz);
86
87  /* Prolazak kroz niz razlomaka. */
88  for (i = 0; i < n; i++) {
89      /* Racunanje reciprocnog razlomka. */
90      r = reciprojni(&razlomci[i]);
91
92      /* Ispis rezultata u zavisnosti od navedenih opcija. */
93      if (opcija_x)
94          fprintf(izlaz, "%d/%d ", r.brojilac, r.imenilac);
95      if (opcija_y)
96          fprintf(izlaz, "%f ", vrednost(&r));
97      fprintf(izlaz, "\n");
98  }
99
100 /* Zatvaranje datoteka. */
101 fclose(ulaz);
102 fclose(izlaz);
103
104 exit(EXIT_SUCCESS);
105 }
```

Rešenje 3.3.24

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>

5 #define MAKS_IME 31
6 #define MAKS_AUTOMOBILA 100
7
8 /* Struktura koja opisuje automobil. */
9 typedef struct {
10     char marka[MAKS_IME];
11     char model[MAKS_IME];
12     float cena;
13 } Automobil;

15 /*
16     Struktura Info sadrži naziv marke automobila, prosek cena za tu
17     marku i broj automobila te marke */
18 typedef struct {
19     char marka[MAKS_IME];
20     float prosecna_cena;
21     int n;
22 } Info;

23 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
24 izlaz za greske i prekida izvršavanje programa. */
25 void greska(char *poruka) {
26     fprintf(stderr, "%s\n", poruka);
27     exit(EXIT_FAILURE);
28 }

31 /* Funkcija ucitava informacije o automobilima iz fajla i smesta ih
32 u niz struktura. Kao povratnu vrednost funkcija vraca broj
33 ucitanih automobila. */
34 int ucitaj(FILE *f, Automobil a[]) {
35     int i, n;

37     fscanf(f, "%d", &n);
38     if (n <= 0 || n > MAKS_AUTOMOBILA)
39         greska("Greska: neispravan broj automobila.");

41     for (i = 0; i < n; i++)
42         fscanf(f, "%s %s %f", a[i].marka, a[i].model, &a[i].cena);

43     return n;
44 }

47 /* Funkcija proverava da li se u nizu sa informacijama o markama
48 nalazi prosledjena marka. Ukoliko se nalazi, vraca odgovarajuću
49 poziciju, a u suprotnom vraca -1. */
50 int sadrzi(Info info[], int n, char marka[]) {

```

3 Ulaz i izlaz programa

```
51     int i;
52     for (i = 0; i < n; i++)
53         if (strcmp(info[i].marka, marka) == 0)
54             return i;
55
56     return -1;
57 }

58 /* Funkcija popunjava niz sa informacijama o markama na osnovu
59    podataka datih u nizu automobila. */
60 void izracunaj_proseke(Automobil a[], int automobili_n,
61                         Info info[], int *n) {
62     int i, pozicija, j = 0;
63     for (i = 0; i < automobili_n; i++) {
64         pozicija = sadrzi(info, j, a[i].marka);
65         if (pozicija == -1) {
66             strcpy(info[j].marka, a[i].marka);
67             info[j].prosecna_cena = a[i].cena;
68             info[j].n = 1;
69             j++;
70         } else {
71             info[pozicija].prosecna_cena += a[i].cena;
72             info[pozicija].n += 1;
73         }
74     }

75     for (i = 0; i < j; i++)
76         info[i].prosecna_cena /= info[i].n;
77
78     *n = j;
79 }
80

81 /* Funkcija ispisuje informacije o prosecnim cenama za svaku
82    marku. */
83 void ispisi_informacije(Info info[], int n) {
84     int i;
85     printf("Informacije o prosecnoj ceni po markama:\n");
86     for (i = 0; i < n; i++)
87         printf("%s %.2f\n", info[i].marka, info[i].prosecna_cena);
88 }

89 /* Funkcija ispisuje podatke o automobilima cija je cena manja ili
90    jednaka budzetu kojim korisnik raspolaze. */
91 void ispisi_kandidate(Automobil a[], int automobili_n,
92                         float budzet) {
93     int i;
94     printf("Kola u Vasem cenovnom rangu:\n");
95     for (i = 0; i < automobili_n; i++)
96         if (a[i].cena < budzet)
97             printf("%s %s %g\n", a[i].marka, a[i].model, a[i].cena);
98 }
99
100 }
```

```

103 int main(int argc, char *argv[]) {
104     /* Deklaracije potrebnih promenljivih. */
105     Automobil automobili[MAKS_AUTOMOBILA];
106     FILE *ulaz;
107     char ime_datoteke[MAKS_IME];
108     float budzet;
109     Info info[MAKS_AUTOMOBILA];
110     int automobili_n, info_n;
111
112     /* Provera broja argumenata komandne linije. */
113     if (argc != 2)
114         greska("Greska: neispravan poziv.");
115
116     /* Ucitavanje budzeta. */
117     budzet = atof(argv[1]);
118
119     /* Ucitavanje naziva datoteke. */
120     printf("Unesite naziv datoteke: ");
121     scanf("%s", ime_datoteke);
122
123     /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
124     ulaz = fopen(ime_datoteke, "r");
125     if (ulaz == NULL)
126         greska("Greska: neuspesno otvaranje ulazne datoteke.");
127
128     /* Ucitavanje podataka o automobilima. */
129     automobili_n = ucitaj(ulaz, automobili);
130
131     /* Racunanje proseka za svaku marku. */
132     izracunaj_proseke(automobili, automobili_n, info, &info_n);
133
134     /* Ispis podataka za sve marke automobila. */
135     ispisi_informacije(info, info_n);
136
137     /* Ispis podataka o automobilima cija je cena manja ili
138      jednaka granici koju je korisnik uneo. */
139     ispisi_kandidate(automobili, automobili_n, budzet);
140
141     /* Zatvaranje datoteke. */
142     fclose(ulaz);
143
144     exit(EXIT_SUCCESS);
145 }

```

Rešenje 3.3.25

```

2 #include <stdio.h>
3 #include <string.h>
4 #include <stdlib.h>
5
6 #define MAKS_LINIJA 81

```

3 Ulaz i izlaz programa

```
6  /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
8   izlaz za greske i prekida izvrsavanje programa. */
10  void greska(char *poruka) {
11      fprintf(stderr, "%s\n", poruka);
12      exit(EXIT_FAILURE);
13  }
14
15  int main(int argc, char *argv[]) {
16      /* Deklaracije potrebnih promenljivih. */
17      FILE *ulaz;
18      char linija[MAKS_LINIJA];
19      int k;
20
21      /* Provera broja argumenata komandne linije. */
22      if (argc != 3)
23          greska("Greska: neispravan poziv.");
24
25      /* Otvaranje datoteke cije se ime zadaje kao prvi argument
26       komandne linije i provera uspeha. */
27      ulaz = fopen(argv[1], "r");
28      if (ulaz == NULL)
29          greska("Greska: neuspesno otvaranje ulazne datoteke.");
30
31      /* Racunanje vrednosti drugog argumenta komandne linije. */
32      k = atoi(argv[2]);
33
34      /* Funkcija fgets cita jednu liniju iz datoteke. Njeni
35       argumenti su:
36          1. Niska u koju ce biti smestena procitana linija
37          2. Maksimalna duzina linije
38          3. Datoteka iz koje se cita.
39          Kada dodje do kraja datoteke, kao povratnu vrednost
40          funkcija vraca NULL. */
41      while (fgets(linija, MAKS_LINIJA, ulaz) != NULL) {
42          /* Ispis svih linija cija je duzina veca od k. */
43          if (strlen(linija) > k)
44              printf("%s", linija);
45      }
46      printf("\n");
47
48      /* Zatvaranje datoteke. */
49      fclose(ulaz);
50
51      exit(EXIT_SUCCESS);
52 }
```

Rešenje 3.3.26

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_LINIJA 81
6
7 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i prekida izvršavanje programa. */
8 void greska(char *poruka) {
9     fprintf(stderr, "%s\n", poruka);
10    exit(EXIT_FAILURE);
11}
12
13 int main(int argc, char *argv[]) {
14     /* Deklaracije potrebnih promenljivih. */
15     char linija[MAKS_LINIJA], najduza_linija[MAKS_LINIJA];
16     int duzina, maks_duzina;
17     FILE *ulaz;
18
19     /* Provera broja argumenata komandne linije. */
20     if (argc != 2)
21         greska("Greska: neispravan poziv.");
22
23     /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
24     ulaz = fopen(argv[1], "r");
25     if (ulaz == NULL)
26         greska("Greska: neuspesno otvaranje ulazne datoteke.");
27
28     /* Pronalazak najduze linija u datoteci. */
29     maks_duzina = 0;
30     while (fgets(linija, MAKS_LINIJA, ulaz) != NULL) {
31         duzina = strlen(linija);
32
33         if (duzina > maks_duzina ||
34             (duzina == maks_duzina &&
35              strcmp(linija, najduza_linija) < 0)) {
36             strcpy(najduza_linija, linija);
37             maks_duzina = duzina;
38         }
39     }
40
41     /* Ispis najduze linije na standardni izlaz. */
42     printf("%s", najduza_linija);
43
44     /* Zatvaranje datoteke. */
45     fclose(ulaz);
46
47     exit(EXIT_SUCCESS);
48}

```

Rešenje 3.3.27

```
#include <stdio.h>
2 #include <stdlib.h>
# include <string.h>
4
#define MAKS_LINIJA 81
6 #define MAKS_REC 31
8 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i prekida izvrsavanje programa. */
10 void greska(char *poruka) {
    fprintf(stderr, "%s\n", poruka);
    exit(EXIT_FAILURE);
}
14
/* Funkcija broji koliko puta se niska t javlja u okviru niske s. */
16 int broj_pojavljivanja(char s[], char t[]) {
    int brojac = 0, i;
    int tn = strlen(t);
    int sn = strlen(s);
20
    /* Funkcija strncmp(s,t,n) poredi prvih n karaktera niski s i t.
       U petlji se vrsti poredjenje niske t sa svim podniskama niske s
       cija je duzina tn.
24    Na primer, ako je s = "abcab", a t = "ab", tada je sn = 5,
       a tn = 2.
26    Za i = 0, zove se strncmp("abcab", "ab", 2) i na taj nacin se
       porede "ab" i "ab".
28    Za i = 1, zove se strncmp("bcab", "ab", 2) i na taj nacin se
       porede "bc" i "ab".
30    ...
32    Za i = sn - st = 5 - 2 = 3, zove se strncmp("ab", "ab", 2) i
       na taj nacin se porede "ab" i "ab". */
34    for (i = 0; i <= sn - tn; i++)
        if (strncmp(s + i, t, tn) == 0)
            brojac++;
36
    return brojac;
38 }
40 int main(int argc, char *argv[]) {
    /* Deklaracije potrebnih promenljivih. */
42    char rec[MAKS_REC];
    char linija[MAKS_LINIJA];
FILE *ulaz, *izlaz;
44    int n, brojac;
46
    /* Provera broja argumenata komandne linije. */
48    if (argc != 3)
        greska("Greska: neispravan poziv.");
50 }
```

```

52  /* Otvaranje ulazne datoteke za citanje i provera uspeha. */
53  ulaz = fopen(argv[1], "r");
54  if (ulaz == NULL)
55      greska("Greska: neuspesno otvaranje ulazne datoteke.");
56
57  /* Otvaranje izlazne datoteke za pisanje i provera uspeha. */
58  izlaz = fopen(argv[2], "w");
59  if (izlaz == NULL)
60      greska("Greska: neuspesno otvaranje izlazne datoteke.");
61
62  /* Ucitavanje broja n i provera ispravnosti unosa. */
63  printf("Unesite broj n: ");
64  scanf("%d", &n);
65  if (n <= 0)
66      greska("Greska: neispravan unos.");
67
68  /* Ucitavanje trazene reci. */
69  fscanf(ulaz, "%s", rec);
70
71  /* Iz ulazne datoteke se cita linija po linija i u izlaznu
72   datoteku se upisuju sve linije koje trazenu rec sadrze bar n
73   puta. */
74  while (fgets(linija, MAKS_LINIJA, ulaz) != NULL) {
75      brojac = broj_pojavljivanja(linija, rec);
76      if (brojac >= n)
77          fprintf(izlaz, "%d: %s", brojac, linija);
78  }
79
80  /* Zatvaranje datoteka. */
81  fclose(ulaz);
82  fclose(izlaz);
83
84  exit(EXIT_SUCCESS);
}

```

Rešenje 3.3.28

```

1 #include <stdio.h>
2 #include <string.h>
3 #include <stdlib.h>
4
5 #define MAKS_LINIJA 81
6 #define MAKS_NISKA 21
7
8 /* Funkcija prebrojava koliko linija datoteke ulaz se zavrsava
9  niskom s. */
10 int broj_linija(FILE *ulaz, char *s) {
11     char linija[MAKS_LINIJA];
12     int brojac = 0, duzina_linije;
13     int duzina_s = strlen(s);
14

```

3 Ulaz i izlaz programa

```
16    /* Citanje linija iz datoteke sve do kraja datoteke. */
17    while (fgets(linija, MAKS_LINIJA, ulaz) != NULL) {
18        /* Racunanje duzine procitane linije. */
19        duzina_liniije = strlen(linija);
20
21        /* Uklanjanje znaka za novi red sa kraja linije. */
22        if (linija[duzina_liniije - 1] == '\n') {
23            linija[duzina_liniije - 1] = '\0';
24            duzina_liniije--;
25        }
26
27        /* Poredjenje kraja linije sa niskom s. Kraj linije se moze
28         * dobiti tako sto se izvrsi 'pomeranje' u desno do kraja
29         * linije, a zatim 'pomeranje' u levo onoliko mesta koliko je
30         * dugacka niska s.
31         * Na primer, ako je linija "abcdefghijklm", a niska s "ab",
32         * onda se sa linija + duzina_liniije vrsti pomeranje na karakter
33         * iza karaktera 'k' (odnosno null-terminator), a sa
34         * linija + duzina_liniije - duzina_s
35         * na karakter 'j'. Ukoliko se funkcija strcmp pozove sa
36         * strcmp(linija + duzina_liniije - duzina_s, s),
37         * vrsice se poredjenje niske "jk" i "ab", sto je i bio cilj. */
38        if (strcmp(linija + duzina_liniije - duzina_s, s) == 0)
39            brojac++;
40    }
41
42    return brojac;
43}
44
45int main() {
46    /* Deklaracije potrebnih promenljivih. */
47    FILE *ulaz;
48    char s[MAKS_NISKA];
49
50    /* Otvaranje datoteke ulaz.txt za citanje i provera uspeha. */
51    ulaz = fopen("ulaz.txt", "r");
52    if (ulaz == NULL) {
53        fprintf(stderr, "Greska: neuspesno otvaranje ulazne "
54                "datoteke.\n");
55        exit(EXIT_FAILURE);
56    }
57
58    /* Ucitavanje niske s. */
59    printf("Unesite nisku s: ");
60    scanf("%s", s);
61
62    /* Ispis rezultata. */
63    printf("Broj linija: %d\n", broj_liniija(ulaz, s));
64
65    /* Zatvaranje datoteke. */
66    fclose(ulaz);
```

```

68     exit(EXIT_SUCCESS);
}

```

Rešenje 3.3.29

```

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

#define MAKS_LINIJA 81

/* Funkcija ispisuje prosledjenu poruku o gresci na standardni
   izlaz za greske i prekida izvršavanje programa. */
void greska(char *poruka) {
    fprintf(stderr, "%s\n", poruka);
    exit(EXIT_FAILURE);
}

int main(int argc, char *argv[]) {
    /* Deklaracije potrebnih promenljivih. */
    char linija[MAKS_LINIJA];
    FILE *izlaz;
    int ispis_velika_slova = 0, ispis_mala_slova = 0;

    /* Provera broja argumenata komandne linije. */
    if (argc > 2)
        greska("Greska: neispravan poziv.");

    /* Postavljanje vrednosti indikatora za ispis u zavisnosti od
       navedene opcije. */
    if (argc == 1)
        ispis_velika_slova = ispis_mala_slova = 1;
    else {
        /* Funkcija strcasecmp poredi niske ignorisuci razliku izmedju
           malih i velikih slova. */
        if (strcasecmp(argv[1], "-v") == 0)
            ispis_velika_slova = 1;
        else if (strcasecmp(argv[1], "-m") == 0)
            ispis_mala_slova = 1;
        else
            greska("Greska: neispravna opcija.");
    }

    /* Otvaranje datoteke izlaz.txt za pisanje i provera uspeha. */
    izlaz = fopen("izlaz.txt", "w");
    if (izlaz == NULL)
        greska("Greska: neuspesno otvaranje izlazne datoteke.");

    /* Citanje linija sa standardnog ulaza i ispis odgovarajucih
       slova. */
    while (fgets(linija, MAKS_LINIJA, stdin) != NULL) {
        if (ispis_mala_slova)
            for (int i = 0; linija[i] != '\0'; i++)
                if (isupper(linija[i]))
                    linija[i] = tolower(linija[i]);
        if (ispis_velika_slova)
            for (int i = 0; linija[i] != '\0'; i++)
                if (islower(linija[i]))
                    linija[i] = toupper(linija[i]);
        if (fputs(linija, izlaz) == EOF)
            greska("Greska: neuspesno pisanje u izlaznu datoteku.");
    }
}

```

3 Ulaz i izlaz programa

```
46     linija u izlaznu datoteku. */
47     printf("Unesite recenice: \n");
48     while (fgets(linija, MAKS_LINIJA, stdin) != NULL) {
49         if ((ispis_mala_slova && islower(linija[0])) ||
50             (ispis_velika_slova && isupper(linija[0])) ||
51             (ispis_mala_slova && ispis_velika_slova))
52             fputs(linija, izlaz);
53     }
54
55     /* Zatvaranje datoteke. */
56     fclose(izlaz);
57
58     exit(EXIT_SUCCESS);
59 }
```

Rešenje 3.3.30

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_LINIJA 201
6
7 /* Funkcija ispisuje prosledjenu poruku o gresci na standardni
8    izlaz za greske i prekida izvrsavanje programa. */
9 void greska(char *poruka) {
10     fprintf(stderr, "%s\n", poruka);
11     exit(EXIT_FAILURE);
12 }
13
14 int main(int argc, char **argv) {
15     /* Deklaracije potrebnih promenljivih. */
16     int i = 1;
17     char *d1, *d2;
18     FILE *ulazi1, *ulaz2;
19     char linija1[MAKS_LINIJA], linija2[MAKS_LINIJA];
20
21     /* Provera broja argumenata komandne linije. */
22     if (argc != 3)
23         greska("Greska: neispravan poziv.");
24
25     /* Otvaranje ulaznih datoteka za citanje i provera uspeha. */
26     ulazi1 = fopen(argv[1], "r");
27     ulaz2 = fopen(argv[2], "r");
28     if (ulazi1 == NULL || ulaz2 == NULL)
29         greska("Greska: neuspesno otvaranje datoteke.");
30
31     /* Citanje prve linije iz obe datoteke. */
32     d1 = fgets(linija1, MAKS_LINIJA, ulazi1);
33     d2 = fgets(linija2, MAKS_LINIJA, ulaz2);
```

```

 36  /* Citanje preostalih linija dok se ne dodje do kraja bar jedne
 37   * datoteke. */
 38   while (d1 != NULL && d2 != NULL) {
 39     /* Poredjenje ucitanih linija. */
 40     if (strcmp(linija1, linija2) != 0)
 41       printf("%d ", i);

 42     /* Prelazak na sledece linije. */
 43     d1 = fgets(linija1, MAKS_LINIJA, ulaz1);
 44     d2 = fgets(linija2, MAKS_LINIJA, ulaz2);

 45     i++;
 46   }

 47   /* Iz prethodne petlje je moglo da se izadje u 3 slucaja:
 48    1. Doslo se do kraja prve datoteke.
 49    2. Doslo se do kraja druge datoteke.
 50    3. Doslo se do kraja obeju datoteka.
 51    Ako se desio treci slucaj, nijedna od naredne dve
 52    petlje se nece izvrsiti. U prvom slucaju ce se izvrsiti samo
 53    prva petlja, a u drugom slucaju druga. */

 54
 55   /* Ispis preostalih rednih brojeva linija prve datoteke. */
 56   while (d1 != NULL) {
 57     printf("%d ", i);
 58     d1 = fgets(linija1, MAKS_LINIJA, ulaz1);
 59     i++;
 60   }

 61   /* Ispis preostalih rednih brojeva linija druge datoteke. */
 62   while (d2 != NULL) {
 63     printf("%d ", i);
 64     d2 = fgets(linija2, MAKS_LINIJA, ulaz2);
 65     i++;
 66   }

 67   /* Zatvaranje datoteka. */
 68   fclose(ulaz1);
 69   fclose(ulaz2);

 70   exit(EXIT_SUCCESS);
 71 }

```

3 Ulaz i izlaz programa

Elektronsko izdanie (2019)

Dodatak A

Ispitni rokovи

A.1 Modul Matematika

A.1.1 Praktični deo ispita, januar 2019.

Zadatak A.1.1 Napisati program koji učitava četvorocifrene brojeve do unosa broja 0, a zatim ispisuje one brojeve kojima je cifra desetica najveća cifra u zapisu. Ukoliko nema takvih brojeva među unetima, ispisati broj 0. U slučaju greške, ispisati -1 na standardni izlaz za greške.

Test 1

```
||| ULAZ:  
9523 -8542 3232 -9999 -1121 1576 0  
IZLAZ:  
3232 -9999 -1121 1576
```

Test 2

```
||| ULAZ:  
4596 1234 9631 -120 0  
IZLAZ:  
4596  
IZLAZ ZA GREŠKE:  
-1
```

Test 3

```
||| ULAZ:  
9876 2258 -4579 4689 -5567 6630 1200 5204 0  
IZLAZ:  
0
```

[Rešenje A.1.1]

Zadatak A.1.2 Napisati program koji pomaže korisniku da "šifruje" svoju elektronsku adresu kako ne bi dobijao nepoželjne poruke. "Šifrovanje" adrese

A Ispitni rokovi

se vrši tako što se znak @ zameni sa [AT]. Elektronska adresa se učitava kao niska maksimalne dužine 100 karaktera sa standardnog ulaza, a šifrovana adresa se ispisuje na standardni izlaz. U slučaju da elektronska adresa nije ispravno zadata ispisati -1 na standardni izlaz za greške.

Test 1	Test 2	Test 3
ULAZ: korisnik@gmail.com IZLAZ: korisnik[AT]gmail.com	ULAZ: student@matf.bg.ac.rs IZLAZ: student[AT]matf.bg.ac.rs	ULAZ: pogresnaadresayahoo.com IZLAZ ZA GREŠKE: -1

[Rešenje A.1.2]

Zadatak A.1.3 Definisati strukturu *Hemijski_element* koja sadrži naziv elementa (nisku dužine najviše 20 karaktera), oznaku elementa (nisku dužine najviše 2 karaktera) i broj neutrona (ceo broj). Napisati program koji učitava podatke o hemijskim elementima do unosa reči **kraj**, a potom još jedan naziv elementa i na standardni izlaz ispisuje oznaku i broj neutrona tog elementa. Ukoliko element nije pronađen među učitanim podacima, ispisati -1.

NAPOMENA: Pretpostaviti da neće biti uneto više od 120 elemenata, kao i da su podaci o hemijskim elementima ispravno zadati.

Test 1	Test 2	Test 3
ULAZ: kalcijum Ca 20 cink Zn 35 fosfor P 16 kraj fosfor IZLAZ: P 16	ULAZ: nikl Ni 31 bor B 6 kripton Kr 48 natrijum Na 12 kraj hrom IZLAZ ZA GREŠKE: -1	ULAZ: litijum Li 4 ugljenik C 6 aluminijum Al 14 srebro Ag 61 guozdje Fe 40 brom Br 45 kraj ugljenik IZLAZ: C 6

[Rešenje A.1.3]

Zadatak A.1.4 U datoteci *pesme.txt* dat je ceo broj *n* koji označava broj pesama, a potom i *n* redova sa podacima o pesmama. U svakom redu naveden je naziv pesme i njen žanr (niske bez belina, dužine najviše 30 karaktera). Napisati program koji učitava podatke iz datoteke, a zatim, u zavisnosti od opcije koja se zadaje kao argument komandne linije, obrađuje podatke na sledeći način:

- ukoliko je zadata opcija **-p**, učitava se sa standardnog ulaza jedan karakter i na standardni izlaz ispisuju svi nazivi pesama koji počinju zadatim karakterom;
- ukoliko je zadata opcija **-z**, učitava se sa standardnog ulaza niska koja predstavlja žanr pesme i na standardni izlaz ispisuju nazivi svih pesama odabranog žanra.

Prilikom odabira pesama za ispis, zanemariti veličinu slova. U slučaju greške, ispisati **-1** na standardni izlaz za greške.

Test 1

```
POKRETANJE: ./a.out -p
PESME.TXT
7
BohemianRhapsody rock
RollingInTheDeep pop
StairwayToHeaven rock
BeatIt pop
SoWhat jazz
MyFunnyValentine jazz
Smooth pop
ULAZ:
S
IZLAZ:
StairwayToHeaven
SoWhat
Smooth
```

Test 2

```
POKRETANJE: ./a.out -z
PESME.TXT
7
BohemianRhapsody rock
RollingInTheDeep pop
StairwayToHeaven rock
BeatIt pop
SoWhat jazz
MyFunnyValentine jazz
Smooth pop
ULAZ:
pop
IZLAZ:
RollingInTheDeep
BeatIt
Smooth
```

Test 3

```
POKRETANJE: ./a.out -x
IZLAZ ZA GREŠKE:
-1
```

Test 4

```
POKRETANJE: ./a.out -p -z
IZLAZ ZA GREŠKE:
-1
```

Test 5

```
POKRETANJE: ./a.out
IZLAZ ZA GREŠKE:
-1
```

[Rešenje A.1.4]

A.1.2 Praktični deo ispita, februar 2019.

Zadatak A.1.5 Napisati program koji učitava pozitivan četvorocifren broj n , a zatim na standardni izlaz ispisuje zbir onih cifara broja n koje su po vrednosti veće od aritmetičke sredine svih cifara broja n . U slučaju greške, ispisati **-1** na standardni izlaz za greške.

Test 1

```
ULAZ:
1234
IZLAZ:
7
```

Test 2

```
ULAZ:
6745
IZLAZ:
13
```

Test 3

```
ULAZ:
100
IZLAZ ZA GREŠKE:
-1
```

Test 4

```
ULAZ:
-1234
IZLAZ ZA GREŠKE:
-1
```

[Rešenje A.1.5]

A Ispitni rokovi

Zadatak A.1.6 Napisati program koji učitava nisku s parne dužine od najviše 20 karaktera i na standardni izlaz ispisuje nisku koja se dobija nadovezivanjem karaktera prve polovine niske s na drugu polovinu niske s . U slučaju greške, ispisati -1 na standardni izlaz za greške.

Test 1	Test 2	Test 3	Test 4
ULAZ: Beograde IZLAZ: radeBeog	ULAZ: matematika IZLAZ: atikamatem	ULAZ: 1234 IZLAZ: 3412	ULAZ: abc1234 IZLAZ ZA GREŠKE: -1

[Rešenje A.1.6]

Zadatak A.1.7 Napisati program koji čita sadržaj datoteke *ulaz.txt* i ispisuje na standardni izlaz sve niske datoteke koje predstavljaju cele brojeve. U slučaju greške, ispisati -1 na standardni izlaz za greške.

Test 1	Test 2	Test 3
POKRETANJE: ./a.out ULAZ.TXT 123 ab1 2ab -23 IZLAZ: 123 -23	POKRETANJE: ./a.out ULAZ.TXT 145as 25gf 265 478 65 -96 IZLAZ: 265 478 65 -96	POKRETANJE: ./a.out ULAZ.TXT Onde nema brojeva IZLAZ:
Test 4		
POKRETANJE: ./a.out ULAZ.TXT NE POSTOJI! IZLAZ ZA GREŠKE: -1		

[Rešenje A.1.7]

Zadatak A.1.8 Napisati program koji sa standardnog ulaza učitava podatke o osvajačima takmičenja. Za svako takmičenje se redom zadaju godina takmičenja (pozitivan ceo broj) i ime osvajača (niska od najviše 30 karaktera bez belina). Program treba da ispiše:

- ako je navedena opcija $-y$ kao prvi argument komandne linije, ime osvajača takmičenja za godinu koja se navodi kao drugi argument

- ako je navedena opcija -w kao prvi argument komandne linije, sve godine u kojima je takmičar čije se ime navodi kao drugi argument komande linije osvajač takmičenje.

U slučaju greške, ispisati -1 na standardni izlaz za greške.

NAPOMENA: Podrazumevati da su ulazni podaci o takmičenjima ispravni.
Broj osvajača nije unapred poznat.

Test 1

```
POKRETANJE: ./a.out -y 2016
ULAZ:
2011 ManUtd
2012 ManCity
2013 ManUtd
2014 ManCity
2015 Chelsea
2016 Leicester
2017 Chelsea
2018 ManCity
IZLAZ:
Leicester
```

Test 2

```
POKRETANJE: ./a.out -w RealMadrid
ULAZ:
2011 Barcelona
2012 Chelsea
2013 BayernMunich
2014 RealMadrid
2015 Barcelona
2016 RealMadrid
2017 RealMadrid
2018 RealMadrid
IZLAZ:
2014 2016 2017 2018
```

Test 3

```
POKRETANJE: ./a.out -s 2001
IZLAZ ZA GREŠKE:
-1
```

Test 4

```
POKRETANJE: ./a.out -x
IZLAZ ZA GREŠKE:
-1
```

Test 5

```
POKRETANJE: ./a.out -s 2012 2000
IZLAZ ZA GREŠKE:
-1
```

Test 6

```
POKRETANJE: ./a.out -y 2005 -w RealMadrid
IZLAZ ZA GREŠKE:
-1
```

[Rešenje A.1.8]

A.2 Modul Informatika

A.2.1 Praktični deo ispita, januar 2019.

Zadatak A.2.1 Napisati program koji učitava cele trocifrene brojeve sve do kraja ulaza i na standardni izlaz ispisuje one čije su cifre uređene strogo rastuće (cifre se čitaju sa leva na desno). U slučaju greške, ispisati -1 i prekinuti izvršavanje programa.

A Ispitni rokovi

Test 1	Test 2	Test 3	Test 4
ULAZ: -532 236 100 -555 546 IZLAZ: 236	ULAZ: 123 -123 321 -321 IZLAZ: 123 -123	ULAZ: 258 695 -1234 IZLAZ: 258 -1	ULAZ: 14 IZLAZ: -1

[Rešenje A.2.1]

Zadatak A.2.2 Napisati program koji sa standardnog ulaza učitava reč s maksimalne dužine 20 karaktera (bez belina), a zatim karakter koji predstavlja način modifikacije učitane niske:

- ukoliko je učitan karakter m , sve karaktere reči s koji su mala slova, pretvoriti u odgovarajuća velika
- ukoliko je učitan karakter v , sve karaktere reči s koji su velika slova, pretvoriti u odgovarajuća mala
- ukoliko je učitan karakter o , ne menjati karaktere reči s

Na standardni izlaz ispisati nisku nakon modifikacije. U slučaju greške, ispisati -1 na standardni izlaz i prekinuti izvršavanje programa.

Test 1	Test 2	Test 3	Test 4
ULAZ: sreca m IZLAZ: SRECA	ULAZ: IspiT v IZLAZ: ispit	ULAZ: Rec o IZLAZ: Rec	ULAZ: PROgram x IZLAZ: -1

[Rešenje A.2.2]

Zadatak A.2.3 Napisati program za praćenje rezultata automobilske trke. Na takmičenju učestvuje n ($n \geq 3$) takmičara u m ($m \geq 2$) trka. Program prvo učitava broj takmičara i trka, a zatim za svakog od n takmičara vreme u sekundama u svakoj od m trka. Pretpostaviti da neće biti više od 100 takmičara i 100 trka. Vremena čuvati u matrici dimenzije $n \times m$ tako da element (i, j) predstavlja vreme koje je takmičar i postigao u j -toj trci. Na standardni izlaz ispisati redne brojeve takmičara (brojeći ih od 0) koji su pobedili u trkama (bili najbrži), redom za svaku trku. Pretpostaviti da neće biti više takmičara sa istim prolaznim vremenom po trci. U slučaju greške, ispisati -1 na standardni izlaz i prekinuti izvršavanje programa.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>	<i>Test 4</i>
ULAZ: 3 3 192.9 87.8 109.102 181.2 92.1 102.4 151.1 87.9 118.9 IZLAZ: 2 0 1	ULAZ: 3 4 51.3 184.94 121.7 99.51 50.9 182.71 119.2 99.2 51.2 192.11 122.9 100.1 IZLAZ: 1 1 1 1	ULAZ: 4 3 113.5 145.2 -14.5 IZLAZ: -1	ULAZ: 4 -3 IZLAZ: -1

[Rešenje A.2.3]

Zadatak A.2.4 Definisati strukturu sa nazivom *Kutija* koja sadrži dužinu, širinu i visinu kutije u centimetrima (pozitivni celi brojevi). Napisati program koji učitava pozitivan ceo broj n ($n \leq 100$), a zatim i podatke o n kutija. Nakon toga, program treba da ispiše zapreminu kutije u koju se može smestiti svaka od preostalih $n - 1$ kutija pojedinačno. Prepostaviti da neće biti više takvih kutija, a ukoliko takva kutija ne postoji, ispisati 0. U slučaju greške, ispisati -1 na standardni izlaz i prekinuti izvršavanje programa.

NAPOMENA: Da bi jedna kutija (sa celobrojnim dimenzijama) stala u drugu, svaka od dimenzija te kutije (dužina, širina i visina redom) mora biti manja barem 1 centimetar od odgovarajućih dimenzija druge kutije. Prilikom smeštanja jedne kutije u drugu nema obrtanja kutije.

<i>Test 1</i>	<i>Test 2</i>	<i>Test 3</i>	<i>Test 4</i>
ULAZ: 4 15 2 9 185 27 12 16 21 10 120 12 3 IZLAZ: 59940	ULAZ: 3 9 18 2 21 5 3 3 15 5 IZLAZ: 0	ULAZ: -3 IZLAZ: -1	ULAZ: 3 1 2 3 8 9 -5 IZLAZ: -1

[Rešenje A.2.4]

A.2.2 Praktični deo ispita, februar 2019.

Zadatak A.2.5 Napisati program koji učitava cele trocifrene brojeve sve do kraja ulaza i na standardni izlaz ispisuje one brojeve čija je cifra desetica jednaka aritmetičkoj sredini cifara stotina i jedinica. U slučaju greške, ispisati -1 na standardni izlaz i prekinuti izvršavanje programa.

A Ispitni rokovi

Test 1

ULAZ:	543 236 100 -555 546
IZLAZ:	543 -555

Test 2

ULAZ:	402 -402 103 -103
IZLAZ:	

Test 3

ULAZ:	-1234
IZLAZ:	-1

Test 4

ULAZ:	14
IZLAZ:	-1

[Rešenje A.2.5]

Zadatak A.2.6 Sa standardnog ulaza se učitava niska s maksimalne dužine 30 karaktera. Napisati program koji na standardni izlaz ispisuje dužinu najduže podniske niske s čiji su karakteri uređeni strogo rastuće po ASCII kodovima čitajući sa leva na desno.

Test 1

ULAZ:	stolica
IZLAZ:	2

Test 2

ULAZ:	a12bcABC
IZLAZ:	4

Test 3

ULAZ:	PPPPPPPP
IZLAZ:	1

Test 4

ULAZ:	abcdefw
IZLAZ:	7

[Rešenje A.2.6]

Zadatak A.2.7 Sa standardnog ulaza se učitava neparan prirodan broj n ($n \leq 101$), a zatim n^2 celih brojeva koje treba sačuvati u odgovarajućoj kvadratnoj matrici. Proveriti da li je suma elemenata na glavnoj dijagonali matrice neparna, i ako jeste, na standardni izlaz ispisati vrednost maksimalnog elementa glavne dijagonale. Ako to nije slučaj, ispisati vrednost minimalnog elementa glavne dijagonale. U slučaju greške, ispisati -1 na standardni izlaz i prekinuti izvršavanje programa.

Test 1

ULAZ:	3
	15 6 7
	2 -4 -2
	3 2 6
IZLAZ:	15

Test 2

ULAZ:	5
	12 6 7 1 2
	2 -4 -2 2 0
	3 2 6 10 7
	3 2 6 12 5
	12 6 7 1 2
IZLAZ:	-4

Test 3

ULAZ:	4
IZLAZ:	-1

Test 4

ULAZ:	-7
IZLAZ:	-1

[Rešenje A.2.7]

Zadatak A.2.8 Definisati strukturu sa nazivom *Student* koja sadrži podatke o studentu: indeks studenta (pozitivan ceo broj), broj poena ostvaren na ispitu (nenegativan realan broj dvostrukе tačnosti iz intervala [0, 100]) i oznaku učionice u kojoj je student polagao ispit (niska iz skupa "704", "718", "rlab" i "bim"). Napisati program koji sa standardnog ulaza učitava prirodan broj n , a zatim podatke o n studenata koji su polagali ispit iz Programiranja 1, redom, indeks, broj poena i oznaku učionice. Nakon podataka o studentima se učitava oznaka učionice za koju treba ispisati broj studenata iz te učionice koji su položili ispit. Oznaka učionice se zadaje kao niska od najviše 10 karaktera. Pretpostaviti da su podaci o studentima ispravni i da neće biti više od 100 studenata. U slučaju greške ispisati -1 na standardni izlaz i prekinuti izvršavanje programa. Student je položio ispit ako je na istom ostvario bar 51 poen.

Test 1	Test 2	Test 3	Test 4
<p>ULAZ: 9 20180001 98 704 20180002 33 704 20180003 7 718 20180005 61.8 rlab 20180006 50.5 bim 20180007 55.6 718 20180008 51 704 20180009 30 rlab 20180010 40.4 rlab 704</p> <p>IZLAZ: 2</p>	<p>ULAZ: 7 20180003 73 718 20180005 60.8 rlab 20180006 40.5 bim 20180007 45.6 718 20180008 19.9 704 20180009 31.4 rlab 20180010 49.4 rlab</p> <p>IZLAZ: 1</p>	<p>ULAZ: 4 20180001 98 704 20180002 33 704 20180003 73.5 718 20180005 60.8 rlab bim</p> <p>IZLAZ: 0</p>	<p>ULAZ: -4 IZLAZ: -1</p>

[Rešenje A.2.8]

A.3 Rešenja

Rešenje A.1.1

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

int main() {
    /* Deklaracija potrebnih promenljivih. */
    int broj, broj_kopija;
    char cifra_jedinica, cifra_desetica, cifra_stotina, cifra_hiljada;
    char postoji_broj = 0;

    while (1) {
        /* Ucitavanje korisnickog unosa. */
        scanf("%d", &broj);

        /* Provera da li se doslo do kraja unosa. */
        if (broj == 0) {
            break;
        }

        /* Cuvanje kopije broja. */
        broj_kopija = broj;

        /* Provera ispravnosti ulaza. */
        broj = abs(broj);
        if (broj < 1000 || broj > 9999) {
            fprintf(stderr, "-1\n");
            exit(EXIT_FAILURE);
        }

        /* Izdvajanje cifara zadatog broja. */
        cifra_jedinica = broj % 10;
        broj /= 10;

        cifra_desetica = broj % 10;
        broj /= 10;

        cifra_stotina = broj % 10;
        broj /= 10;

        cifra_hiljada = broj;

        /* Proverava da li je cifra desetica najveca cifra. */
        if (cifra_desetica >= cifra_jedinica
            && cifra_desetica >= cifra_stotina
            && cifra_desetica >= cifra_hiljada) {
            /* Ako jeste, ispisuje se ucitani broj. */
            printf("%d\n", broj_kopija);
        }
    }
}
```

```

48     /* Pamti se informacija da je broj sa ovim svojstvom
50      pronadjen. */
51     postoji_broj = 1;
52 }
53
54 /* Ako broj sa traženim svojstvom nije pronadjen, ispisuje se
55   odgovarajuća poruka. */
56 if (!postoji_broj) {
57     printf("0\n");
58 }
59
60 exit(EXIT_SUCCESS);
61 }
```

Rešenje A.1.2

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_DUZINA 101
6
7 int main() {
8     /* Deklaracija potrebnih promenljivih. */
9     char email[MAKS_DUZINA];
10    char sifrovani_email[MAKS_DUZINA];
11    char *at_pozicija;
12
13    /* Ucitavanje elektronske adrese. */
14    scanf("%s", email);
15
16    /* Odredjivanje pozicije @ karaktera. */
17    at_pozicija = strchr(email, '@');
18
19    /* Ukoliko elektronska adresa ne sadrzi @ karakter, ispisuje se
20       tražena poruka. */
21    if (at_pozicija == NULL) {
22        fprintf(stderr, "-1\n");
23        exit(EXIT_FAILURE);
24    }
25
26    /* Sifrovana adresa inicijalno sadrzi samo terminirajući nulu. */
27    sifrovani_email[0] = '\0';
28
29    /* U sifrovani adresu se kopira deo originalne adrese koji
30       prethodi @ karakteru. */
31    *at_pozicija = '\0';
32    strcpy(sifrovani_email, email);
```

A Ispitni rokovi

```
34  /* Zatim se sifrovana adresa nadovezuje sa [AT] zamenom. */
35  strcat(sifrovani_email, "[AT]");
36
37  /* Na kraju se sifrovana adresa nadovezuje sa delom originalne
38  adresе koji se nalazi posle @ karaktera. */
39  strcat(sifrovani_email, at_pozicija + 1);
40
41  /* Ispis rezultata. */
42  printf("%s\n", sifrovani_email);
43
44  exit(EXIT_SUCCESS);
}
```

Rešenje A.1.3

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAKS_DUZINA 21
6 #define MAKS_DUZINA_OZNAKE 3
7 #define MAKS_BROJ_ELEMENATA 120
8
9 /* Struktura koja opisuje hemijski element. */
10 typedef struct Hemijski_element {
11     char naziv[MAKS_DUZINA];
12     char oznaka[MAKS_DUZINA_OZNAKE];
13     int broj_neutrona;
14 } Hemijski_element;
15
16 int main() {
17     /* Deklaracija potrebnih promenljivih. */
18     Hemijski_element elementi[MAKS_BROJ_ELEMENATA];
19     int i, n;
20     char naziv_trazenog_elementa[MAKS_DUZINA];
21     char nadjen;
22
23     /* Ucitavanje hemijskih elemenata. */
24     for (i = 0; i < n; i++) {
25         /* Prvo se ucitava naziv elementa. */
26         scanf("%s", elementi[i].naziv);
27
28         /* Ako je u pitanju rec "kraj", ucitavanje hemijskih elemenata se
29         prekida. */
30         if (strcmp(elementi[i].naziv, "kraj") == 0) {
31             break;
32         }
33
34         /* U suprotnom, ucitava se oznaka elementa i broj neutrona. */
35         scanf("%s%d", elementi[i].oznaka, &elementi[i].broj_neutrona);
36     }
37 }
```

```

37  /* Poslednja vrednost brojaca i odgovara broju elemenata ucitanog
38  * niza. */
39  n = i;
40
41  /* Ucitavanje naziva trazenog elementa. */
42  scanf("%s", naziv_trazenog_elementa);
43
44  /* Provera da li se trazeni element nalazi u nizu elemenata. */
45  /* Informacija da li se element nalazi u nizu ili ne bice upisana
46  * kao vrednost 1 ili 0 u promenljivu nadjen. */
47  nadjen = 0;
48  for (i = 0; i < n; i++) {
49      if (strcmp(elementi[i].naziv, naziv_trazenog_elementa) == 0) {
50          nadjen = 1;
51          printf("%s %d\n", elementi[i].oznaka,
52                 elementi[i].broj_neutrona);
53          break;
54      }
55  }
56
57  /* Ukoliko se trazeni element ne nalazi u nizu elemenata,
58  * ispisuje se odgovarajuca poruka. */
59  if (!nadjen) {
60      fprintf(stderr, "-1\n");
61  }
62
63  return 0;
64 }
```

Rešenje A.1.4

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <ctype.h>
5
6 #define MAKS_DUZINA 31
7
8 /* Funkcija ispisuje odgovarajucu poruku na standardni izlaz za
9  * greske i prekida izvrsavanje programa. */
10 void greska() {
11     fprintf(stderr, "-1\n");
12     exit(EXIT_FAILURE);
13 }
14
15 int main(int argc, char *argv[]) {
16     /* Deklaracija potrebnih promenljivih. */
17     FILE *ulaz;
18     int n, i;
19     char karakter;
```

A Ispitni rokovi

```
17
18     char opcija;
19     char zanr[MAKS_DUZINA], tmp_pesma[MAKS_DUZINA],
20           tmp_zanr[MAKS_DUZINA];
21
22     /* Proverava broja argumenata komandne linije. */
23     if (argc != 2) {
24         greska();
25     }
26
27     /* Proverava da li je opcija ispravno zadata tj. da li pocinje
28      karakterom -. */
29     if (argv[1][0] != '-') {
30         greska();
31     }
32
33     /* Ako je opcija ispravno zadata, cuva se u promenljivoj opcija. */
34     opcija = argv[1][1];
35
36     /* Otvaranje datoteke za citanje i proverava uspesnosti
37      otvaranja. */
38     ulaz = fopen("pesme.txt", "r");
39     if (ulaz == NULL) {
40         greska();
41     }
42
43     /* Ucitavanje broja pesama. */
44     fscanf(ulaz, "%d", &n);
45
46     /* Analiza zadate opcije. */
47     switch (opcija) {
48
49         case 'p':
50             /* 1) cita se karakter za pretragu */
51             scanf("%c", &karakter);
52
53             /* 2) za svaku pesmu */
54             for (i = 0; i < n; i++) {
55                 /* 3) citaju se ime pesme i zanr pesme */
56                 fscanf(ulaz, "%s", tmp_pesma);
57                 fscanf(ulaz, "%s", tmp_zanr);
58
59                 /* 4) proverava se da li ime pesme pocinje procitanim
60                  karakterom */
61                 if (toupper(tmp_pesma[0]) == toupper(karakter)) {
62                     /* 5) ispisuje se ime pesme */
63                     printf("%s\n", tmp_pesma);
64                 }
65             }
66             break;
67
68         case 'z':
69             /* 1) ucitava se zanr */
70
71     }
```

```

    scanf("%s", zanr);

73     /* 2) za svaku pesmu */
75     for (i = 0; i < n; i++) {
76         /* 3) citaju se ime pesme i zanr pesme */
77         fscanf(ulaz, "%s", tmp_pesma);
78         fscanf(ulaz, "%s", tmp_zanr);

79         /* 4) proverava se da li zanr pesme odgovara procitanom zanru
80         */
81         if (strcmp(tmp_zanr, zanr) == 0) {
82             /* 5) ispisuje se ime pesme */
83             printf("%s\n", tmp_pesma);
84         }
85     }
86     break;

87 default:
88     /* Ako je zadata pogresna opcija, prekida se izvrsavanje
89      programa. */
90     greska();
91 }
92

93 /* Zatvaranje datoteke. */
94 fclose(ulaz);
95
96 exit(EXIT_SUCCESS);
97 }
```

Rešenje A.1.5

```

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

4 /* Funkcija ispisuje odgovarajucu poruku na standardni izlaz za
   greske i prekida izvrsavanje programa. */
6 void greska() {
7     fprintf(stderr, "-1\n");
8     exit(EXIT_FAILURE);
9 }

10 int main() {
11     /* Deklaracije potrebnih promenljivih. */
12     int n;
13     char cifra_jedinica, cifra_desetica, cifra_stotina, cifra_hiljada;
14     float aritmeticka_sredina;
15     char suma_cifara;

16     /* Ucitavanje i provera ispravnosti ulaza. */
17     scanf("%d", &n);
18     if (n < 1000 || n > 9999) {
```

A Ispitni rokovi

```
22     greska();
23 }
24 /* Izdvajanje cifara unetog broja. */
25 cifra_jedinica = n % 10;
26 cifra_desetica = (n / 10) % 10;
27 cifra_stotina = (n / 100) % 10;
28 cifra_hiljada = n / 1000;
29
30 /* Izracunavanje aritmeticke sredine cifara. */
31 aritmeticka_sredina =
32     (cifra_hiljada + cifra_desetica + cifra_jedinica +
33      cifra_stotina) / 4.0;
34
35 /* Izracunavanje sume onih cifara koje su vece od aritmeticke
36    sredine. */
37 suma_cifara = 0;
38
39 if (cifra_jedinica > aritmeticka_sredina)
40     suma_cifara += cifra_jedinica;
41
42 if (cifra_desetica > aritmeticka_sredina)
43     suma_cifara += cifra_desetica;
44
45 if (cifra_stotina > aritmeticka_sredina)
46     suma_cifara += cifra_stotina;
47
48 if (cifra_hiljada > aritmeticka_sredina)
49     suma_cifara += cifra_hiljada;
50
51 /* Ispis rezultata. */
52 printf("%d\n", suma_cifara);
53
54 exit(EXIT_SUCCESS);
55 }
```

Rešenje A.1.6

```
2 #include <stdio.h>
3 #include <stdlib.h>
4 #include <string.h>
5
6 #define MAX 21
7
8 /* Funkcija ispisuje odgovarajucu poruku na standardni izlaz za
9    greske i prekida izvrsavanje programa. */
10 void greska() {
11     fprintf(stderr, "-1\n");
12     exit(EXIT_FAILURE);
13 }
```

```

14 int main() {
15     /* Deklaracije potrebnih promenljivih. */
16     char s[MAX];
17     char novo_s[MAX];
18     int n;
19
20     /* Ucitavanje niske. */
21     scanf("%s", s);
22
23     /* Provera duzine ucitane niske. */
24     n = strlen(s);
25     if (n % 2 != 0) {
26         greska();
27     }
28
29     /* Popunjavanje nove niske nulama. */
30     memset(novo_s, '\0', sizeof(n));
31
32     /* Kopiranje druge polovine niske s u novu nisku. */
33     strcpy(novo_s, s + n / 2);
34
35     /* Kopiranje prve polovine niske s u novu nisku. */
36     s[n / 2] = 0;
37     strcpy(novo_s + n / 2, s);
38
39     /* Ispis rezultata. */
40     printf("%s\n", novo_s);
41
42     exit(EXIT_SUCCESS);
43 }
```

Rešenje A.1.7

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <ctype.h>
4 #include <string.h>
5
6 #define MAX 21
7
8 /* Funkcija ispisuje odgovarajucu poruku na standardni izlaz za
9    greske i prekida izvrsavanje programa. */
10 void greska() {
11     fprintf(stderr, "-1\n");
12     exit(EXIT_FAILURE);
13 }
14
15 /* Funkcija proverava da li je niska s zapisana samo pomocu
16    cifara. Povratna vrednost funkcije je 1 ako je uslov ispunjen,
17    dok je u suprotnom 0. */
18 int sve_cifre(const char *s) {
```

A Ispitni rokovi

```
19    int i;

21    /* Provera da je pocetni karakter niske znak -. */
22    if (s[0] == '-') {
23        if (strlen(s) == 1)
24            return 0;
25        else
26            s += 1;
27    }

29    /* Provera da li su karakteri niske cifre: cim se pronadje
30     * karakter koji nije cifra, izvrsavanje funkcije se prekida. */
31    i = 0;
32    while (s[i]) {
33        if (!isdigit(s[i]))
34            return 0;
35
36        i++;
37    }

38    return 1;
39}
40

41 int main() {
42    /* Deklaracije potrebnih promenljivih. */
43    FILE *ulaz = NULL;
44    char s[MAX];

45    /* Otvaranje datoteke za citanje i proverava uspesnosti
46     * otvaranja. */
47    if ((ulaz = fopen("ulaz.txt", "r")) == NULL)
48        greska();

49    /* Citaju se niske datoteke sve do kraja ulaza. */
50    while (fscanf(ulaz, "%s", s) != EOF)
51        /* Ako se procitana niska sastoji samo od brojeva, ispisuje se
52         * na standardni izlaz. */
53        if (sve_cifre(s))
54            printf("%s ", s);

55        putchar('\n');

56    /* Zatvaranje datoteke. */
57    fclose(ulaz);

58    exit(EXIT_SUCCESS);
59}
```

Rešenje A.1.8

```

1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4
5 #define MAX 31
6
7 /* Funkcija ispisuje odgovarajucu poruku na standardni izlaz za
8    greske i prekida izvrsavanje programa. */
9 void greska() {
10    fprintf(stderr, "-1\n");
11    exit(EXIT_FAILURE);
12}
13
14 int main(int argc, char *argv[]) {
15    /* Deklaracije potrebnih promenljivih. */
16    int godina;
17    int tekuca_godina;
18    char ime[MAX];
19    char tekuce_ime[MAX];
20
21    /* Proverava broja argumenata komandne linije. */
22    if (argc != 3) {
23        greska();
24    }
25
26    /* Provera da li je prvi argument komandne linije -y. */
27    if (!strcmp(argv[1], "-y")) {
28        /* Ako jeste, citava se godina koja se očekuje kao drugi
29           argument. */
30        godina = atoi(argv[2]);
31
32        /* Sve do kraja unosa ucitavaju se podaci o osvajacima. */
33        while (scanf("%d %s", &tekuca_godina, tekuce_ime) == 2) {
34            /* Ako uneta godina odgovara trazenoj godini, ispisuje se ime
35               osvajaca. */
36            if (tekuca_godina == godina) {
37                printf("%s\n", tekuce_ime);
38            }
39        }
40
41        exit(EXIT_SUCCESS);
42    }
43
44    /* Provera da li je prvi argument komandne linije -w. */
45    if (!strcmp(argv[1], "-w")) {
46        /* Ako jeste, citava se ime osvajaca koje se očekuje kao drugi
47           argument. */
48        strcpy(ime, argv[2]);
49
50        /* Sve do kraja unosa ucitavaju se podaci o osvajacima. */

```

```
52     while (scanf("%d %s", &tekuca_godina, tekuce_ime) == 2) {
53         /* Ako uneto ime odgovara imenu osvajaca, ispisuje se godina.
54         */
55         if (!strcmp(ime, tekuce_ime)) {
56             printf("%d ", tekuca_godina);
57         }
58         putchar('\n');
59
60         exit(EXIT_SUCCESS);
61     }
62
63     /* Ako prvi argument komandne linije nije ni -y ni -w, program
64     nije korektno pozvan. */
65     greska();
66 }
```

Rešenje A.2.1

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

4 int main() {
5     /* Deklaracija potrebnih promenljivih. */
6     int x, abs_x;
7     int cifra_jedinica, cifra_desetica, cifra_stotina;
8
9     /* Ucitavanje brojeva sve do kraja ulaza. */
10    while (scanf("%d", &x) != EOF) {
11
12        /* Izracunavanje absolutne vrednosti tekuceg broja. */
13        abs_x = x < 0 ? -x : x;
14
15        /* Provera da li je u pitanju trocifren broj. */
16        if (abs_x < 100 || abs_x > 999) {
17            printf("-1\n");
18            exit(EXIT_FAILURE);
19        }
20
21        /* Izdvajanje cifara broja. */
22        cifra_jedinica = abs_x % 10;
23        cifra_desetica = (abs_x / 10) % 10;
24        cifra_stotina = abs_x / 100;
25
26        /* Provera da li su cifre broja uredjene rastuce. */
27        if (cifra_jedinica > cifra_desetica
28            && cifra_desetica > cifra_stotina) {
29            printf("%d ", x);
30        }
31    }
32 }
```

```

32     printf("\n");
33
34 }  


```

Rešenje A.2.2

```

1 #include <stdio.h>
2 #include <string.h>
3 #include <ctype.h>
4 #include <stdlib.h>
5
6 #define MAX 21
7
8 int main() {
9     /* Deklaracija potrebnih promenljivih. */
10    char s[MAX], c;
11    int i, n;
12
13    /* Ucitavanje reci i karaktera koji određuje tip transformacije.
14     */
15    scanf("%s %c", s, &c);
16
17    /* Odredjivanje duzine reci. */
18    n = strlen(s);
19
20    /* Analiza procitanog karaktera. */
21    switch (c) {
22        case 'm':
23            /* Zamena svih malih slova reci odgovarajucim velikim slovima. */
24            for (i = 0; i < n; i++) {
25                if (islower(s[i])) {
26                    s[i] = toupper(s[i]);
27                }
28            }
29            break;
30
31        case 'v':
32            /* Zamena svih velikih slova reci odgovarajucim malim slovima. */
33            for (i = 0; i < n; i++) {
34                if (isupper(s[i])) {
35                    s[i] = tolower(s[i]);
36                }
37            }
38            break;
39
40        case 'o':
41            /* Rec se ne menja. */
42            break;
43        default:
44            /* Transformacija nije definisana pa se ispisuje poruka o

```

A Ispitni rokovi

```
        gresci i prekida izvrsavanje programa. */
46    printf("-1\n");
47    exit(EXIT_FAILURE);
48}

/* Ispis rezultata. */
50    printf("%s\n", s);
51
52    exit(EXIT_SUCCESS);
53}
```

Rešenje A.2.3

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

#define MAX 100

int main() {
    /* Deklaracija potrebnih promenljivih. */
    int n, m, i, j, redni_broj_pobednika;
    float trke[MAX][MAX], pobednik;

    /* Ucitavanje broja takmicara i broja trka. */
    scanf("%d%d", &n, &m);

    /* Provera ispravnosti ucitanih vrednosti. */
    if (n < 3 || n > MAX || m < 2 || m > MAX) {
        printf("-1\n");
        exit(EXIT_FAILURE);
    }

    /* Ucitavanje vremena takmicara po trkama. */
    for (i = 0; i < n; i++) {
        for (j = 0; j < m; j++) {
            scanf("%f", &trke[i][j]);

            /* Provera da li je zadato korektno vreme. */
            if (trke[i][j] <= 0.0) {
                printf("-1\n");
                exit(EXIT_FAILURE);
            }
        }
    }

    /* Odredjivanje pobednika u trkama. */
    for (j = 0; j < m; j++) {

        /* Odredjivanje pobednika j-te trke se svodi na problem
           pronalazenje minimuma j-te kolone. */
        pobednik = trke[0][j];
```

```

    redni_broj_pobednika = 0;
40
41   for (i = 1; i < n; i++) {
42     if (pobednik > trke[i][j]) {
43       pobednik = trke[i][j];
44       redni_broj_pobednika = i;
45     }
46   }
47
48   printf("%d ", redni_broj_pobednika);
49 }
50
51   printf("\n");
52
53   exit(EXIT_SUCCESS);
54 }
```

Rešenje A.2.4

```

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

4 #define MAX 100

6 /* Struktura koja opisuje kutiju. */
7 typedef struct {
8   int sirina, duzina, visina;
9 } Kutija;
10
11 /* Funkcija proverava da li se u zadatu kutiju koja se nalazi
12   u nizu kutija na poziciji j mogu smestiti preostale kutije.
13   Povratna vrednost funkcije je 1 ako kutije mogu da se smeste, a
14   u suprotnom 0. */
15 int smesti(Kutija kutije[], int n, Kutija * kutija, int j) {
16   int i;
17
18   /* Uporedjuju se dimenzije zadate j-te kutije sa dimenzijama svih
19   preostalih kutija. */
20   for (i = 0; i < n; i++) {
21     if (i != j
22         && (kutije[i].sirina >= kutija->sirina
23              || kutije[i].duzina >= kutija->duzina
24              || kutije[i].visina >= kutija->visina)) {
25       return 0;
26     }
27   }
28
29   return 1;
30 }
31
32 int main() {
```

A Ispitni rokovi

```
/* Deklaracija potrebnih promenljivih. */
34 Kutija kutije[MAX];
35     int i, n;
36
37     /* Ucitavanje broja kutija i provera ispravnosti ulaza. */
38     scanf("%d", &n);
39     if (n <= 0 || n > MAX) {
40         printf("-1\n");
41         exit(EXIT_FAILURE);
42     }
43
44     /* Ucitavanje dimenzija kutija uz proveru ispravnosti ulaza. */
45     for (i = 0; i < n; i++) {
46         scanf("%d%d%d", &kutije[i].sirina, &kutije[i].duzina,
47               &kutije[i].visina);
48         if (kutije[i].sirina <= 0 || kutije[i].duzina <= 0
49             || kutije[i].visina <= 0) {
50             printf("-1\n");
51             exit(EXIT_FAILURE);
52         }
53
54     /* Za svaku kutiju se proverava trazeno svojstvo. */
55     for (i = 0; i < n; i++) {
56         /* Ukoliko u i-tu kutiju mogu da se smeste preostale kutije,
57          izracunava se i ispisuje njena zapremina. */
58         if (smesti(kutije, n, &kutije[i], i)) {
59             printf("%d\n",
60                   kutije[i].sirina * kutije[i].duzina *
61                   kutije[i].visina);
62             exit(EXIT_SUCCESS);
63         }
64     }
65
66     /* U suprotnom, zaključujemo da ne postoji kutija sa traženim
67      svojstvom. */
68     printf("0\n");
69
70     exit(EXIT_SUCCESS);
71 }
```

Rešenje A.2.5

```
#include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     /* Deklaracija potrebnih promenljivih. */
6     int x, abs_x;
7     int cifra_jedinica, cifra_desetica, cifra_stotina;
8 }
```

```

10  /* Ucitavanje brojeva sve do kraja ulaza. */
11  while (scanf("%d", &x) != EOF) {
12
13      /* Izracunavanje absolutne vrednosti tekuceg broja. */
14      abs_x = x < 0 ? -x : x;
15
16      /* Provera da li je u pitanju trocifren broj. */
17      if (abs_x < 100 || abs_x > 999) {
18          printf("-1\n");
19          exit(EXIT_FAILURE);
20      }
21
22      /* Izdvajanje cifara broja. */
23      cifra_jedinica = abs_x % 10;
24      cifra_desetica = (abs_x / 10) % 10;
25      cifra_stotina = abs_x / 100;
26
27      /* Provera da li je cifra desetica jednaka aritmetickoj sredini
28         cifara stotine i jedinice. */
29      if (cifra_desetica == (cifra_jedinica + cifra_stotina) / 2.0) {
30          printf("%d ", x);
31      }
32  }
33  printf("\n");
34
35  exit(EXIT_SUCCESS);
36 }
```

Rešenje A.2.6

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 #define MAX 31
5
6 int main() {
7     /* Deklaracija potrebnih promenljivih. */
8     char s[MAX];
9     int i, max_duzina, trenutna_duzina;
10
11     /* Ucitavanje niske. */
12     scanf("%s", s);
13
14     /* Odredjivanje najduze podniske karaktera koji su uredjeni
15        rastuce. */
16     max_duzina = 1;
17     trenutna_duzina = 1;
18
19     for (i = 1; s[i]; i++) {
20         /* Ako je ASCII kod tekuceg karaktera veci od ASCII koda
            prethodnog karaktera, podniska je rastuca pa se njena
```

```
22     trenutna_duzina_uvecava. */
23     if (s[i - 1] < s[i]) {
24         trenutna_duzina++;
25     } else {
26         /* Ako se naislo na par karaktera koji nisu uredjeni rastuce,
27            azurira se, po potrebi, maksimalna duzina trazene podniske
28            i resetuje se trenutna duzina. */
29         if (max_duzina < trenutna_duzina) {
30             max_duzina = trenutna_duzina;
31         }
32         trenutna_duzina = 1;
33     }
34 }

35 /* Postupak azuriranja maksimalne duzine se, po potrebi, vrsti i
36    kada se stigne do kraja niske. */
37 if (max_duzina < trenutna_duzina) {
38     max_duzina = trenutna_duzina;
39 }
40

41 /* Ispis rezultata. */
42 printf("%d\n", max_duzina);
43
44 exit(EXIT_SUCCESS);
45 }
```

Rešenje A.2.7

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

4 #define MAX 101

6 /* Funkcija izracunava zbir elemenata na glavnoj dijagonali
7   kvadratne matrice dimenzije n. */
8 int suma(int m[][][MAX], int n) {
9     int i, s = 0;
10
11     for (i = 0; i < n; i++) {
12         s += m[i][i];
13     }
14
15     return s;
16 }

17 /* Funkcija izracunava vrednost najmanjeg elementa glavne
18   dijagonale kvadratne matrice dimenzije n. */
19 int minimum(int m[][][MAX], int n) {
20     int i;
21     int min = m[0][0];
```

```

24     for (i = 1; i < n; i++) {
25         if (min > m[i][i]) {
26             min = m[i][i];
27         }
28     }
29
30     return min;
31 }
32
33 /* Funkcija izracunava vrednost najveceg elementa glavne
34    dijagonale kvadratne matrice dimenzije n. */
35 int maximum(int m[][][MAX], int n) {
36     int i;
37     int max = m[0][0];
38
39     for (i = 1; i < n; i++) {
40         if (max < m[i][i]) {
41             max = m[i][i];
42         }
43     }
44
45     return max;
46 }
47
48 int main() {
49     /* Deklaracija potrebnih promenljivih. */
50     int m[MAX][MAX], n, i, j;
51
52     /* Ucitavanje dimenzije matrice i provera ispravnosti ulaza. */
53     scanf("%d", &n);
54     if (n <= 0 || n > MAX || n % 2 == 0) {
55         printf("-1\n");
56         exit(EXIT_FAILURE);
57     }
58
59     /* Ucitavanje elemenata matrice. */
60     for (i = 0; i < n; i++) {
61         for (j = 0; j < n; j++) {
62             scanf("%d", &m[i][j]);
63         }
64     }
65
66     /* Provera da li je suma elemenata na glavnoj dijagonali matrice
67        parna. */
68     if (suma(m, n) % 2) {
69         /* Ako jeste, ispisuje se vrednost maksimalnog elementa
70            dijagonale. */
71         printf("%d\n", maximum(m, n));
72     } else {
73         /* U suprotnom se ispisuje vrednost minimalnog elementa
74            dijagonale. */
75         printf("%d\n", minimum(m, n));
76     }
77 }

```

```
76     }
77
78     exit(EXIT_SUCCESS);
79 }
```

Rešenje A.2.8

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

#define MAX_BROJ_STUDENATA 100
#define MAX_UCIONICA 11

/* Struktura koja opisuje studenta. */
typedef struct {
    int indeks;
    double poeni;
    char ucionica[MAX_UCIONICA];
} STUDENT;

int main() {
    /* Deklaracija potrebnih promenljivih. */
    STUDENT studenti[MAX_BROJ_STUDENATA];
    int i, n, broj_studenata;
    char ucionica[MAX_UCIONICA];

    /* Ucitavanje broja studenata i provera ispravnosti ulaza. */
    scanf("%d", &n);
    if (n <= 0 || n > MAX_BROJ_STUDENATA) {
        printf("-1\n");
        exit(EXIT_FAILURE);
    }

    /* Ucitavanje podataka o studentima. */
    for (i = 0; i < n; i++) {
        scanf("%d%lf%s", &studenti[i].indeks, &studenti[i].poeni,
              studenti[i].ucionica);
    }

    /* Ucitavanje oznake ucionice i provera ispravnosti ulaza. */
    scanf("%s", ucionica);
    if (strcmp(ucionica, "704") && strcmp(ucionica, "718") &&
        strcmp(ucionica, "bim") && strcmp(ucionica, "rlab")) {
        printf("-1\n");
        exit(EXIT_FAILURE);
    }

    /* Odredjivanje broja studenata koji su polagali ispit u zadatoj
       ucionici i polozili ga. */
    broj_studenata = 0;
```

```
46     for (i = 0; i < n; i++) {
47         if (!strcmp(ucionica, studenti[i].ucionica)
48             && studenti[i].poeni >= 51.0) {
49             broj_studenata++;
50         }
51     }
52
53     /* Ispis rezultata. */
54     printf("%d\n", broj_studenata);
55
56     exit(EXIT_SUCCESS);
57 }
```

Elektronsko izdanie (2019)

