

## PROGRAMIRANJE 1



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

# **PROGRAMIRANJE 1**

## **Zbirka zadataka**

**Beograd  
2016.**

Autori:

*dr Milena Vujošević Jančić*, docent na Matematičkom fakultetu u Beogradu

*dr Jovana Kovačević*, docent na Matematičkom fakultetu u Beogradu

*Danijela Simić*, asistent na Matematičkom fakultetu u Beogradu

*Anđelka Zečević*, asistent na Matematičkom fakultetu u Beogradu

PROGRAMIRANJE 1

Zbirka zadataka

# Sadržaj

0.1	Strukture	v
0.2	Rešenja	xvii

## 0.1 Strukture

**Zadatak 0.1.1** Definirati strukturu kojom se predstavlja kompleksan broj. Napisati funkcije koje izračunavaju zbir, razliku, proizvod i količnik dva kompleksna broja, a zatim i program koji učitava dva kompleksna broja i ispisuje vrednost zbira, razlike, proizvoda i količnika.

### *Primer 1*

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

[Rešenje 0.1.1]

**Zadatak 0.1.2** Definirati strukturu kojom se predstavlja razlomak. Napisati funkcije koje izračunavaju zbir i proizvod dva razlomka. Unosi se broj  $n$  a potom i  $n$  razlomaka sa standardnog ulaza. Ispisati njihov zbir i proizvod na standardni izlaz.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesi broj razlomaka: 5
Uneti razlomke:
1 2
7 8
3 4
5 6
2 9
Suma svih razlomaka je 229/72.
Proizvod svih razlomaka je 35/576.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesi broj razlomaka: 10
Uneti razlomke:
4 3
12 25
3 8
1 3
8 9
2 3
5 6
-24 50
7 18
-7 19
Suma svih razlomaka je 6089/1368.
Proizvod svih razlomaka je 1568/577125.
```

[Rešenje 0.1.2]

**Zadatak 0.1.3** Zimi su prehlade česte i treba unositi više vitamina C. Struktura *Vocka* sadrži ime voćke (nisku maksimalne dužine 20 karaktera) i količinu vitamina C u miligramima (realan broj). Napisati program koji učitava podatke o voćkama sve do unosa reči KRAJ i ispisuje ime voćke sa najviše vitamina C. Pretpostaviti da broj voćki neće biti veći od 50. NAPOMENA: *Probati sa testiranjem zadataka pomoću preusmeravanja.*

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite ime voćke i njenu količinu vitamina C: jabuka 4.6
Unesite ime voćke i njenu količinu vitamina C: limun 51
Unesite ime voćke i njenu količinu vitamina C: kivi 92.7
Unesite ime voćke i njenu količinu vitamina C: banana 8.7
Unesite ime voćke i njenu količinu vitamina C: pomorandža 53.2
Unesite ime voćke i njenu količinu vitamina C: KRAJ
Voce sa najviše C vitamina je: kivi
```

[Rešenje 0.1.3]

**Zadatak 0.1.4** Definirati strukturu *Grad* u kojoj se nalazi ime grada (niska dužine 20 karaktera) i prosečna temperatura u toku decembra (realan broj). Napisati program koji učitava imena  $n$  ( $0 < n < 50$ ) gradova i njihove prosečne temperature, a zatim ispisuje one gradove koji imaju idealnu temperaturu za klizanje: od 3 do 8 stepeni. Napomena: *probati sa testiranjem zadataka pomoću preusmeravanja.*

### Primer 1

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

### Primer 2

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

[Rešenje 0.1.4]

**Zadatak 0.1.5** 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 zadaje u jednoj liniji ispisati prevod. Ako je reč u rečenici nepoznata umesto nje ispisati odgovarajući broj zvezdica. Maksimalna dužina reči je 50 karaktera, ukupan broj parova reči je maksimalno 100, a maksimalna dužina rečenice je 100 karaktera. NAPOMENA: *Probati sa testiranjem zadataka pomoću preusmeravanja.*

### Primer 1

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

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
zima winter
pas dog
sreca happiness
prijatelj friend
solja cup
covek man
Unesite recenicu za prevod:
pas je covekov najbolji prijatelj
dog is ***** best friend
```

[Rešenje 0.1.5]

**Zadatak 0.1.6** Cenoteka pomaže kupcima da pronađu najpovoljniju cenu za proizvod koji žele da kupe. Napisati program koji učitava najpre broj različitih prodavnica (ceo broj manji od 50) a zatim i podatke o ceni traženog artikla –

zadaje se naziv prodavnice (niske maksimalne dužine 20 karaktera) i cena u toj prodavnici (realan broj). Korisnik zadaje željenu cenu proizvoda, a program ispisuje imena svih onih prodavnica u kojima je cena proizvoda jednaka ili manja od željene. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj prodavnica: 5
idea 58.9
mazi 58.2
roda 55.1
tempo 54.5
intereza 57.99
Uneti zeljenu cenu: 57.0
Povoljne prodavnice su:
roda
tempo
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj prodavnica: 4
dm 43.2
lily 45.99
benu_apoteke 43.99
sephora 50.99
Uneti zeljenu cenu: 47.00
Povoljne prodavnice su:
dm
lily
benu_apoteke
```

[Rešenje 0.1.6]

**Zadatak 0.1.7** Statistički zavod Srbije istražuje kako rade obdaništa u Srbiji. Za dato obdanište dobija spisak  $n$  dece sa kolonama: pol (m ili z), broj godina (od 3 do 6) i ocena koju je dete dalo radu obdaništa (od 1 do 5). Maksimalan broj dece u obdaništu je 200. Napisati program koji za decu datog pola i broja godina ispisuje na tri decimale prosečnu ocenu obdaništa. U slučaju neispravnog unosa ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 5
Uneti podatke za svako dete, pol,
broj godina i ocenu:
m 3 5
z 3 4
m 4 2
m 5 4
m 3 4
Uneti pol i broj godina: m 3
Prosečna ocena je: 4.500.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 10
Uneti 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
Uneti pol i broj godina: z 4
Prosečna ocena je: 4.200.
```



### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 15
Uneti podatke za svako dete, pol,
broj godina i ocenu:
m 3 2
z 7 5
Neispravan broj godina.
```

### Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj dece: 2
Uneti podatke za svako dete, pol,
broj godina i ocenu:
m 3 2
z 3 5
Uneti pol i broj godina: h 5
Neispravan pol.
```

[Rešenje 0.1.7]

**Zadatak 0.1.8** Definirati strukturu kojom se opisuje student. Student je zadatak svojim imenom i prezimenom (oba su maksimalne dužine 30 karaktera), smerom (R, I, V, N, T, O) i prosečnom ocenom. Napisati program koji učitava podatke o  $n$  studenata, zatim učitava smer i ispisuje imena i prezimena onih studenta koji su sa datog smjera. Potom ispisati podatke za studenta koji ima najveći prosek. Ako ima više takvih studenata ispisati sve njih. Maksimalan broj studenata je 2000. U slučaju greške ispisati odgovarajuću poruku.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj studenata: 5
Uneti 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
Uneti smer: R
Studenti sa R smerom:
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:
Uneti broj studenata: 4
Uneti podatke o studentima:
0. student: Djordje Lazarevic N 9.05
1. student: Minja Peric W 7.70
Nekorektan smer.
```

[Rešenje 0.1.8]

**Zadatak 0.1.9** Program učitava podatke o učenicima do kraja unosa. Učenika može biti najviše 30. Za svakog učenika dato je njegovo ime (maksimalne dužine 20 karaktera) i 9 ocena (ocene su celi brojevi od 1 do 5). Ispisati:

- (a) Reč NEDOVOLJNI:, a potom imena nedovoljnih učenika. Učenik je nedovoljan ako ima barem jednu jedinicu.

- (b) Potom ispisati reč ODLICNI:, a potom imena odličnih učenika. Učenik je odličan ako ima prosek ocena veći ili jednak 4.5.

U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti podatke o djaku: Maja 4 5 2 3 4 4 3 3 4
Uneti podatke o djaku: Nikola 5 4 5 5 5 4 4 5 5
Uneti podatke o djaku: Jasmina 2 2 1 1 2 3 3 1 3
Uneti podatke o djaku: Pera 5 4 5 3 5 5 1 5 5
Uneti podatke o djaku: Pavle 4 3 2 4 3 2 4 3 2
Uneti podatke o djaku:

NEDOVOLJNI: Jasmina Pera
ODLICNI: Nikola
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti podatke o djaku: Uros 3 4 2 3 4 2 3 4 4
Uneti podatke o djaku: Nebojsa 4 5 5 5 4 5 5 5 5
Uneti podatke o djaku: Sreten 2 3 2 4 5 4 4 4 2
Uneti podatke o djaku:

NEDOVOLJNI:
ODLICNI: Nebojsa
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti podatke o djaku: Mirko 2 3 4 4 4 3 3 3 4
Uneti podatke o djaku: Mihailo 2 3 10 5 5 2 3 4 2
Neispravna ocena.
```

[Rešenje 0.1.9]

**Zadatak 0.1.10** Definisati strukturu **Osoba** kojom se opisuje jedan unos u imenik. Za svaku osobu su dati podaci: ime (maksimalne dužine 20 karaktera), prezime (maksimalne dužine 30 karaktera) i email adresa (maksimalne dužine 50 karaktera). Napisati program koji učitava ceo broj  $n$  ( $0 < n \leq 50$ ) a zatim podatke o  $n$  osoba. Ispisati imena i prezimena svih osoba koje imaju gmail adresu (čija se email adresa završava sa @gmail.com). U slučaju greške ispisati odgovarajuću poruku. Može se smatrati da je svaka email adresa dobro zadata i sadrži samo jedno pojavljivanje znaka @.

### *Primer 1*

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj osoba: 3
Uneti podatke o osobama:
ime, prezime i email.
Dusko Dugousko dusko@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:
Uneti broj osoba: 3
Uneti podatke o osobama:
ime, prezime i email.
Homer Simpson homer@yahoo.com
Mardz Simpson mardz@matf.bg.ac.rs
Vlasnici gmail naloga su:
```

[Rešenje [0.1.10](#)]

**\* Zadatak 0.1.11** 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 učitava broj potrošača  $n$  (najviše 100), 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. Možemo pretpostaviti da nijedan potrošač neće kupiti više od 20 artikala, kao i da naziv svakog artikla sadrži maksimalno 30 karaktera. U slučaju greške ispisati odgovarajuću poruku.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj potrosackih korpi: 3
Uneti podatke o korpi:
Broj artikala: 4
Unesi artikal, naziv, kolicinu i cenu: jabuke 10 22.4
Unesi artikal, naziv, kolicinu i cenu: dezodorans 1 120.99
Unesi artikal, naziv, kolicinu i cenu: C_supa 3 36.56
Unesi artikal, naziv, kolicinu i cenu: sunka 1 230.99
Uneti podatke o korpi:
Broj artikala: 2
Unesi artikal, naziv, kolicinu i cenu: Jafa_keks 55.78
Unesi artikal, naziv, kolicinu i cenu: Najlepse_zelje 62.99
Uneti podatke o korpi:
Broj artikala: 3
Unesi artikal, naziv, kolicinu i cenu: prasak_zav_1 1199.99
Unesi artikal, naziv, kolicinu i cenu: omeksivac 1 279.99
Unesi artikal, naziv, kolicinu i cenu: protiv_kamenca 1 699.99

Korpa 0:
    jabuke 10 22.40
    dezodorans 1 120.99
    C_supa 3 36.56
    sunka 1 230.99
-----
    ukupno: 685.66

Korpa 1:
    Jafa_keks 55 0.78
    Najlepse_zelje 62 0.99
-----
    ukupno: 104.28

Korpa 2:
    prasak_zav_1 1199.99
    omeksivac 1 279.99
    protiv_kamenca 1 699.99
-----
    ukupno: 2179.97

Prosečna cena potrosacke korpe: 989.97
```

[Rešenje 0.1.11]

**Zadatak 0.1.12** Uvesti tip podataka *Sifra* kojim se opisuje način šifrovanja alfanumeričkih karaktera. Svaka šifra se opisuje pozitivnom celobrojnom vrednošću  $b$  koja određuje broj pozicija pomeranja, kao i karakterom 'L' ili 'D' koji određuje smer pomeranja (levo ili desno).

- (a) Napisati funkciju `char sifruj(char c, Sifra s)` koja transformiše zadati karakter  $c$  po šifri  $s$ . Karakter se šifruje tako što se svako slovo zamenjuje slovom za  $b$  mesta levo ili desno od njega u abecedi, i to ciklično, a isto tako i za cifre. Na primer: za  $b = 2$ , i smer='D' : a se menja sa c, b sa d, ..., x

sa z, y sa a, z sa b, 1 sa 3, ..., 8 sa 0, 9 sa 1. Funkcija vraća novodobijeni karakter.

- (b) Načini šifrovanja se zadaju do kraja unosa i to u obliku 2 D 5 L. Potom se zadaju karakteri do kraja unosa. Izmeniti alfanumeričke karaktere prema svim zadatim šiframa i ispisati dobijeni rezultat. Maksimalan broj karaktera može biti 5000. Maksimalan broj šifri može biti 100. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti sifre u obliku: broj, smer:
23 D
Uneti tekst za sifrovanje:
Temperatura danas je 23 stepena Celzijusova.
Rckncpxrpx bxlqx hc 56 qrcnclx Zcjxghsqmtx.
```

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti sifre u obliku: broj, smer:
3 l 7 a
Neispravan smer.
```

#### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti sifre u obliku: broj, smer:
23 D 3 L 14 D 20 L 1 L 2 L 5 D
Uneti tekst za sifrovanje:
Temperatura danas je 23 stepena Celzijusova.
Kudguiqliq tqeqj zu 89 jkugueq Sucqyzljfmq.
```

[Rešenje 0.1.12]

**Zadatak 0.1.13** Definisati strukturu **Lopta** sa poljima poluprecnik (ceo broj u centimetrima) i **boja** (enumeracioni tip koji uključuje plavu, žutu, crvenu i zelenu boju). Zatim učitati informacije o  $n$  lopti ( $0 < n < 50$ ) i ispisati ukupnu zapreminu, kao i broj crvenih lopti. U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

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

#### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 8
Unesite dalje 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
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj lopti: 8
Unesite dalje poluprecnike i boje lopti
(1-plava, 2-zuta, 3-crvena, 4-zelena):
1. lopta: 1 2
2. lopta: 2 10
Nekorektan unos.
```

[Rešenje 0.1.13]

**Zadatak 0.1.14** Napisati program za predstavljanje poligona i izračunavanje njegovog obima i dužine stranica.

- (a) Definisati tip podataka **TACKA** pogodan za predstavljanje tačke Dekartovske ravni (čije su  $x$  i  $y$  koordinate podaci tipa **double**).
- (b) Definisati funkciju **double rastojanje(TACKA a, TACKA b)** koja izračunava rastojanje između dve tačke.
- (c) Definisati funkciju **unsigned ucitaj\_poligon(TACKA\* tacke, unsigned 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(TACKA\* poligon, unsigned n)** koja izračunava obim poligona sa  $n$  tačaka u zadatom nizu *NAPOMENA: Prilikom računanja obima ne zaboraviti stranicu koja spaja poslednje i prvo teme.*
- (e) Definisati funkciju **double maksimalna\_stranica(TACKA\* poligon, unsigned n)** koja izračunava dužinu najduže stranice poligona sa  $n$  tačaka u zadatom nizu.
- (f) Napisati funkciju **double povrsina\_trougla(TACKA A, TACKA B, TACKA C)** za računanje površine trougla.
- (g) Napisati funkciju **double povrsina(TACKA\* poligon, unsigned n)** za računanje površine konveksnog poligona. *NAPOMENA: Zadatak se može rešiti korišćenjem funkcije **povrsina\_trougla**.*
- (h) Napisati program koji učitava poligon sa maksimalno  $N$  temena ( $0 < N \leq 1000$ ) i za učitani poligon ispisuje na tri decimale obim, dužinu maksimalnu stranice i površinu. Pretpostaviti da je uneseni poligon konveksan. Poligon mora imati barem 3 temena. U slučaju greške ispisati odgovarajuću poruku.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti maksimalan broj tacaka poligona: 10
0 0
0 6
3 3
Obim poligona je 14.485.
Duzina maksimalne stranice je 6.000.
Povrsina poligona je 9.000.
```

### Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti maksimalan broj tacaka poligona: 10
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.
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti maksimalan broj tacaka poligona: 4
0 0
Neispravan broj tacaka poligona.
```

[Rešenje 0.1.14]

\* **Zadatak 0.1.15** Definirati strukturu IZRAZ kojom se opisuje numerički izraz nad celim brojevima koji se sastoji od dva celobrojna operanda, numeričke operacije (sabiranje, oduzimanje, množenje ili celobrojno deljenje) nad celim brojevima.

- Napisati funkciju koja ispituje da li je dati izraz korektno zadat i vraća 1 ako jeste a 0 u suprotnom. Podrazumevamo da je izraz korektno zadat ako operacija odgovara +, −, \* ili / i u slučaju deljenja drugi operand je različit od 0.
- Napisati funkciju koja za dati izraz određuje vrednost izraza.
- Napisati funkciju koja učitava dati izraz. Funkcija treba da učitava sa standardnog ulaza izlaz koji je zadat prefiksno — prvo operacija, a potom dva operanda. Funkcija vraća 1 ako je učitavanje bilo uspešno, tj. ako je izraz bio korektno zadat ili 0 u suprotnom.
- Napisati funkciju koja štampa dati izraz infiksno, u obliku "operand<sub>1</sub> operacija operand<sub>2</sub> = vrednost".

Napisati glavni program koji učitava prirodan broj  $n$ , ( $n < 1000$ ) 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.

### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj izraza: 4
Uneti 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 2

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj izraza: 10
Uneti 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
```

### Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Uneti broj izraza: 3
Uneti izraze u prefiksnoj notaciji:
* 1 2
/ 3 0
Deljenje nulom!
Nekorektan unos
```

[Rešenje 0.1.15]

\* **Zadatak 0.1.16** Definisati strukturu kojom se zadaje polinom. Polinom je dat svojim stepenom (može biti najviše 10) i realnim koeficijentima.

- (a) Napisati funkciju koja učitava jedan polinom dat stepenom i koeficijentima.
- (b) Napisati funkciju koja ispisuje polinom u obliku  $k_0 \pm k_1 * x \pm k_2 * x^2 \pm k_3 * x^3 \pm \dots \pm k_n * x^n$  (pri čemu je  $n$  stepen polinoma). Koeficijente ispisati na dve decimale. Ne ispisivati koeficijente koji su jednaki 0 i na mesto znaka  $\pm$  zapisati odgovarajući znak, + ili -, u zavisnosti od znaka odgovarajućeg koeficijenta.
- (c) Napisati funkciju koja za dati polinom određuje njegov integral.



- (d) Učitati polinome do kraja ulaza i za svaki učitani polinom odrediti i ispisati integral tog polinoma. Maksimalan broj polinoma je 100.

U slučaju greške ispisati odgovarajuću poruku.

#### Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Uneti stepen: 3
Uneti koeficijente polinoma:
1 0 3 1
Uneti stepen: 4
Uneti koeficijente polinoma:
7 9 4 0 4
Uneti 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:
Uneti stepen: 3
Uneti koeficijente polinoma:
1 0 -4 1
Uneti stepen: 2
Uneti koeficijente polinoma:
1 2 -3
Uneti stepen: 1
Uneti koeficijente polinoma:
0 -1
Uneti 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 0.1.16]

## 0.2 Rešenja

### Rešenje 0.1.1

```
1 #include <stdio.h>
3 /* Struktura koja opisuje kompleksni broj obuhvata polje za realni
   * i polje za imaginarni deo broja.
5 */
6 typedef struct Complex {
7     float re;
8     float im;
9 } Complex;
11
12 /* Funkcija kojom se izracunava zbir kompleksnih brojeva. */
13 Complex saberi(Complex *a, Complex *b) {
14
15     Complex c;
16     c.re = a->re + b->re;
17     c.im = a->im + b->im;
18     return c;
19 }
```

```

21  /* Funkcija kojom se izracunava razlika kompleksnih brojeva. */
    Complex oduzmi(Complex *a, Complex *b) {
23
        Complex c;
25        c.re = a->re - b->re;
        c.im = a->im - b->im;
27        return c;
    }
29
    /* Funkcija kojom se izracunava proizvod kompleksnih brojeva. */
31    Complex pomnozi(Complex *a, Complex *b) {
33
        Complex c;
        c.re = a->re * b->re - a->im * b->im;
35        c.im = b->re * a->im + a->re * b->im;
        return c;
37    }

39    /* Funkcija kojom se izracunava kolicnik kompleksnih brojeva. */
    Complex podeli(Complex *a, Complex *b) {
41
        Complex c;
43        c.re = (a->re * b->re + a->im * b->im) / (b->re*b->re + b->im*b->im
        );
        c.im = (b->re * a->im - a->re * b->im) / (b->re*b->re + b->im*b->im
        );
45        return c;
    }
47

49    int main() {
        Complex a, b;
51        Complex c;

53        /* Ucitavamo kompleksne brojeve. */
        printf("Unesite realni i imaginarni deo prvog broja: ");
55        scanf("%f%f", &a.re, &a.im);

57        printf("Unesite realni i imaginarni deo drugog broja: ");
        scanf("%f%f", &b.re, &b.im);
59

        c = saberi(&a, &b);
61        /* Ukoliko je imaginarni deo negativan,
        * njegov zapis vec ukljucuje znak,
        * te to treba proveriti.
        * Inace, broj je oblika a+b*i.
        */
63        printf("Zbir: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : '-', c.im);
67

        c = oduzmi(&a, &b);
69        printf("Razlika: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : '-', c.im)

```

```

    ;

71  c = pomnozi(&a, &b);
    printf("Proizvod: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : '-', c.im
    );
73
    if(b.re != 0 || b.im != 0) {
75        c = podeli(&a, &b);
        printf("Kolicnik: %.2f%c%.2f*i\n", c.re, c.im > 0 ? '+' : '-', c.
            im);
77    }
    /* U polju kompleksnih brojeva
79    * nije dozvoljeno deljenje nulom.
    */
81    else
        printf("Kolicnik ne postoji.\n");
83
85    return 0;
}

```

## Rešenje 0.1.2

```

1  #include <stdio.h>

3  typedef struct
    {
5      int brojilac;
        int imenilac;
7    }razlomak;

9  int nzd(int a, int b)
    {
11     int pom;

13     if (a < b)
        {
15         pom = a;
            a = b;
17         b = pom;
        }

19     while(b != 0)
        {
21         pom = a % b;
            a = b;
23         b = pom;
        }

25     return a;
27 }

```

```

29  razlomak zbir(razlomak a, razlomak b)
31  {
32      razlomak c;
33      int nzd_razlomka;

35      c.brojilac = a.brojilac * b.imenilac + b.brojilac*a.imenilac;
36      c.imenilac = a.imenilac*b.imenilac;

37      /* Brojilac i imenilac dobijenog zbira se dele najvećim zajedničkim
38       * deliocom.
39       */
40      nzd_razlomka = nzd(c.brojilac, c.imenilac);

41      c.brojilac = c.brojilac/nzd_razlomka;
42      c.imenilac = c.imenilac/nzd_razlomka;

43      return c;
44  }

45  razlomak proizvod(razlomak a, razlomak b)
46  {
47      razlomak c;
48      int nzd_razlomka;

49      c.brojilac = a.brojilac*b.brojilac;
50      c.imenilac = a.imenilac*b.imenilac;

51      /* Brojilac i imenilac dobijenog zbira se dele najvećim zajedničkim
52       * deliocom.
53       */
54      nzd_razlomka = nzd(c.brojilac, c.imenilac);

55      c.brojilac = c.brojilac/nzd_razlomka;
56      c.imenilac = c.imenilac/nzd_razlomka;

57      return c;
58  }

59  int main()
60  {
61      int n, i;

62      razlomak suma, proizvod_svih, r;

63      printf("Unesi broj razlomaka: ");
64      scanf("%d", &n);

65      suma.brojilac = 0;
66      suma.imenilac = 1;

```

```

81     proizvod_svih.brojilac = 1;
      proizvod_svih.imenilac = 1;
83
      printf("Uneti razlomke:\n");
85     for(i=0; i<n; i++)
      {
87         scanf("%d%d", &r.brojilac, &r.imenilac);
89         suma = zbir(suma, r);
          proizvod_svih = proizvod(proizvod_svih, r);
91     }
93     printf("Suma svih razlomaka je %d/%d.\n", suma.brojilac, suma.
          imenilac);
      printf("Proizvod svih razlomaka je %d/%d.\n", proizvod_svih.
          brojilac, proizvod_svih.imenilac);
95
      return 0;
97 }

```

### Rešenje 0.1.3

```

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

4  #define MAX_DUZINA 21
      #define MAX_BR_VOCKI 50
6
      typedef struct vocka
8  {
          char ime[MAX_DUZINA];
          float vitamin;
10 } VOCKA;
12
14 int main()
      {
16     VOCKA vocke[MAX_BR_VOCKI];
          int i = 0, n, max_vocka;
          char ime[MAX_DUZINA];
18
20     /*
          Program ucitava podatke o vockama i smesta ih u niz
22     sve dok se ne unese rec KRAJ ili ucita MAX_BR_VOCKI vocki.
          */
24     do
      {
26         printf("Unesite ime vocke i njenu kolicinu vitamina C: ");
          scanf("%s",ime);
28         /*
          Kada se unese rec KRAJ prekida se petlja.

```

```

30     */
31     if(strcmp(ime, "KRAJ") == 0)
32         break;
33
34     /*
35     Inace učitava se kolicina vitamina
36     i ta vrednost se smesta u vocku na poziciji "i".
37     */
38     strcpy(vocke[i].ime, ime);
39     scanf("%f", &vocke[i].vitamin);
40     i++;
41 }
42 while(i < MAX_BR_VOCKI);
43
44 n = i;
45
46 /*
47 Pretpostavka je da prva vocka ima najviše vitamina.
48 Petljom se prolazi niz vocki i ukoliko se nađje na vocku koja
49 ima više vitamina
50 od one koja trenutno ima najviše, azurira se vrednosti maksimalne
51 vocke.
52
53 Sve vreme se čuva indeks vocke sa najviše vitamina C.
54 */
55
56 max_vocka = 0;
57 for(i=1; i<n; i++)
58     if(vocke[i].vitamin > vocke[max_vocka].vitamin)
59     {
60         max_vocka = i;
61     }
62
63 printf("Voce sa najviše C vitamina je: %s\n", vocke[max_vocka].ime)
64 ;
65
66 return 0;
67 }

```

## Rešenje 0.1.4

```

1 #include <stdio.h>
2 #define MAX_DUZINA 20
3 #define MAX_BR_GRADOVA 50
4
5 typedef struct Grad{
6     char ime_grada[MAX_DUZINA+1];
7     float temperatura;
8 }Grad;
9

```

```

11 int main(){
12     int n, i;
13     Grad grad[MAX_BR_GRADOVA];

15     printf("Unesite broj n: ");
16     scanf("%d", &n);
17     if(n<0 || n>MAX_BR_GRADOVA){
18         printf("Greska: pogresan unos!\n");
19         return 0;
20     }

21     for(i=0; i<n; i++){
22         printf("Unesite grad i temperaturu: ");
23         scanf("%s %f", grad[i].ime_grada, &grad[i].temperatura);
24     }

25     printf("Gradovi sa idealnom temperaturom za klizanje u decembru:\n"
26           );
27     for(i=0; i<n; i++){
28         if(grad[i].temperatura>=3 && grad[i].temperatura<=8){
29             printf("%s\n", grad[i].ime_grada);
30         }
31     }

32     return 0;
33 }

```

## Rešenje 0.1.5

```

#include <stdio.h>
#include <string.h>
#define MAX_DUZINA 21
#define MAX_BR_REC 100

typedef struct ParReci{
    char sr[MAX_DUZINA+1];
    char en[MAX_DUZINA+1];
}ParReci;

/*
Funkcija koja u rečniku koji sadrži n reči traži prevod reči rec i
upisuje ga u prevod.
Ukoliko se rec ne nalazi u rečniku, prevod se sastoji od zvezdica
pri čemu broj zvezdica odgovara
dužini nepoznate reči.
*/

void pronadji_prevod(ParReci rečnik[], int n, char rec[], char prevod
[]){
    int i;

```

```

20  /* Pretrazuje se recnik i trazi se zadata rec. */
22  for(i=0; i<n; i++)
23  {
24      if(strcmp(recnik[i].sr, rec)==0)
25      {
26          strcpy(prevod, recnik[i].en);
27          return;
28      }
29  }
30
31  /* Ukoliko rec nije pronadjena, formira se prevod reci koji se
32     sastoji od zvezdica. */
33  for(i=0; rec[i]; i++){
34      prevod[i]='*';
35  }
36  prevod[i]='\0';
37 }
38
39 int main(){
40     ParReci recnik[MAX_BR_REC];
41     int n;
42     char sr[MAX_DUZINA+1];
43     char en[MAX_DUZINA+1];
44     int i, j, k;
45     char rec[MAX_DUZINA+1];
46     char prevod[MAX_DUZINA+1];
47     char c;
48
49     /* Ucitavaju se parovi reci sa standardnog ulaza sve do kraja ulaza
50        . */
51     i=0;
52     while(scanf("%s %s", sr, en)!=EOF){
53         if(i==MAX_BR_REC){
54             break;
55         }
56
57         strcpy(recnik[i].sr, sr);
58         strcpy(recnik[i].en, en);
59
60         i++;
61     }
62     /* Broj parova reci se cuva u promenljivoj n. */
63     n=i;
64
65     printf("Unesite recenicu za prevod: \n");
66     do
67     {
68         /* Ucitava se rec po rec date recenice i pronalazi se njen prevod
69            . */
70         scanf("%s", rec);

```



```

70     pronadji_prevod(recnik, n, rec, prevod);
    printf("%s ", prevod);

72     /* Ukoliko je karakter iza reci znak za novi red, onda se prekida
       sa unosom, a ako nije
       * učitava se sledeća rečenica.
74     */
    c = getchar();

76 }while(c != '\n');

78 putchar('\n');

80 return 0;
82 }

```

## Rešenje 0.1.6

```

1  #include <stdio.h>

3  #define MAX_PRODAVNICA 50
   #define DUZINA_RECI 21

5  typedef struct
7  {
   char prodavnica[DUZINA_RECI];
   double cena;
9  }podatak;

11

13 int main()
   {
15     podatak niz[MAX_PRODAVNICA];
   double zeljena;
   int n, i;

17     printf("Uneti broj prodavnica: ");
19     scanf("%d", &n);

21     if (n <= 0 || n > MAX_PRODAVNICA)
   {
23         printf("Neispravan broj prodavnica.\n");
       return -1;
25     }

27     for(i=0; i<n; i++)
   {
29         scanf("%s%lf", niz[i].prodavnica, &niz[i].cena);

31         if (niz[i].cena <= 0)
       {
33             printf("Neispravna cena.\n");

```

```

        return -1;
35     }
    }

37     printf("Uneti zeljenu cenu: ");
39     scanf("%lf", &zeljena);

41     printf("Povoljne prodavnice su:\n");
    for(i=0; i<n; i++)
43         if (niz[i].cena <= zeljena)
            printf("%s\n", niz[i].prodavnica);
45
    return 0;
47 }

```

### Rešenje 0.1.7

```

#include <stdio.h>
2
#define MAX_DECE 200
4
typedef struct
6 {
    char pol;
    int broj_godina;
    int ocena;
10 }dete;

12 int main()
{
14     int n, i, broj_godina;
    dete niz[MAX_DECE];
16     char blanko, pol;
    int suma, broj_dece;

18     printf("Uneti broj dece: ");
20     scanf("%d", &n);

22     if (n <= 0 || n > MAX_DECE)
    {
24         printf("Neispravan broj dece.\n");
        return -1;
26     }

28     printf("Uneti podatke za svako dete, pol, broj godina i ocenu:\n");
    for(i=0; i<n; i++)
30     {
        scanf("%c%c%d", &blanko, &niz[i].pol, &niz[i].broj_godina, &niz[i].ocena);

32         /* Ispitivanje pogresnog unosa. */
    }
}

```

```

34     if (niz[i].pol != 'm' && niz[i].pol != 'z')
35     {
36         printf("Neispravan pol.\n");
37         return -1;
38     }
39     if (niz[i].broj_godina > 6 || niz[i].broj_godina < 3)
40     {
41         printf("Neispravan broj godina.\n");
42         return -1;
43     }
44     if (niz[i].ocena < 1 || niz[i].ocena > 5)
45     {
46         printf("Neispravna ocena.\n");
47         return -1;
48     }
49 }
50
51 printf("Uneti pol i broj godina: ");
52 scanf("%c%c%d", &blanko, &pol, &broj_godina);
53
54 /* Ispitivanje ispravnosti unetih podataka. */
55 if (pol != 'm' && pol != 'z')
56 {
57     printf("Neispravan pol.\n");
58     return -1;
59 }
60 if (broj_godina > 6 || broj_godina < 3)
61 {
62     printf("Neispravan broj godina.\n");
63     return -1;
64 }
65
66 suma = 0;
67 broj_dece = 0;
68
69 for(i=0; i<n; i++)
70     if (niz[i].pol == pol && niz[i].broj_godina == broj_godina)
71     {
72         suma += niz[i].ocena;
73         broj_dece++;
74     }
75
76 if (broj_dece == 0)
77     printf("Ne postoje deca sa takvim karakteristikama.\n");
78 else
79     printf("Prosečna ocena je: %.3lf.\n", (double)suma/broj_dece);
80
81 return 0;
82 }

```

## Rešenje 0.1.8

```

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

4 #define MAXST 2000
5 #define MAX 31

6
7 typedef struct Student
8 {
9     char ime[MAX];
10    char prezime[MAX];
11    char smer;
12    float prosek;
13 } STUDENT;

14
15 void provera(char smer)
16 {
17     if (smer != 'R' && smer != 'I' && smer != 'V' && smer != 'N' &&
18         smer != 'T' && smer != 'O')
19     {
20         printf("Nekorektan smer.\n");
21         exit(EXIT_FAILURE);
22     }
23 }

24 void ucitaj(STUDENT* s)
25 {
26     scanf("%s", s->ime);

28     scanf("%s", s->prezime);

30     getchar();
31     scanf("%c", &s->smer);

32     scanf("%f", &s->prosek);
33 }

34
35 /* II */
36 /*
37
38  Kada neku promenljivu prenosimo u funkciju kao argument, obicno
39  je prenosimo po vrednosti (bez pokazivaca), ako se ona nece menjati
40  u funkciji
41  ili po adresi (preko pokazivaca), ako ce se njena vrednost
42  promeniti u funkciji.

43  Prilikom poziva funkcije, za svaki argument funkcije kreira se
44  promenljiva
45  koja predstavlja lokalnu kopiju argumenta i koja prestaje da
46  postoji po završetku
47  funkcije. S obzirom da se strukture sastoje od vise polja,
48  zauzimaju
49  vise memorije nego nestrukturane promenljive. Zbog toga je za

```

```

46     njihovo kopiranje
    potrebno vise vremena i vise memorijskih resursa nego za kopiranje
    nestrukturnih
    promenljivih.

48
    Da bismo ucinili program efikasnijim, korisno je da strukturu uvek
    kao
50 argument funkcije prenosimo po adresi (preko pokazivaca), bez
    obzira
    da li ce se struktura u toj funkciji menjati ili ne. Pokazivac na
    strukturu
52 zauzima manje memorije nego sama struktura pa je izrada njegove
    kopije
    brza a kopija pokazivaca uzima manji memorijski prostor nego kopija
    strukture.

54
    Kada prenosimo strukturnu promenljivu u funkciju po adresi (preko
    pokazivaca), tada
    imamo mogucnost da je u funkciji menjamo. Ukoliko zelimo da
    onemogucimo promenu,
58 uz argument dodajemo kljucnu rec const. Ako pokusamo da promenimo
    argument
    funkcije prenesen kao const (npr u funkciji ispisi navedemo naredbu
    s->smer='X');),
60 kompajler ce prijaviti gresku. Na ovaj nacin obezbedjujemo da
    promenljiva
    koju smo preneli po adresi ne da bismo je promenili vec radi
    povecanja
62 efikasnosti programa, ne bude, cak ni slucajno, izmenjena u
    funkciji.

64 */

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

70

72 float najveći_prosek(STUDENT studenti[], int n)
{
74     float m;
    int i;

76
    m = studenti[0].prosek;
    for(i=1;i<n;i++)
78         if(m<studenti[i].prosek)
            m=studenti[i].prosek;
80     return m;
82 }

84 /*

```

```

86      Struktura moze da bude povratna vrednost funkcije.
87  */
88  STUDENT prvi_student_sa_najvecim_prosekom(STUDENT studenti[], int n,
89      float m)
90  {
91      STUDENT s;
92      int i;
93      for(i=0;i<n;i++)
94          if(m == studenti[i].prosek)
95          {
96              /*
97               Na strukture se moze primenjivati
98               naredba dodele.
99              */
100             s = studenti[i];
101             break;
102         }
103     return s;
104 }
105
106 int main()
107 {
108     STUDENT studenti[MAXST];
109     int n;
110     int i;
111     float max_prosek;
112     STUDENT student_sa_max_prosekom;
113     int indeks;
114     char smer;
115
116     printf("Uneti broj studenata: ");
117     scanf("%d", &n);
118
119     if (n<0 || n>MAXST)
120     {
121         printf("Nekorektan unos\n");
122         return -1;
123     }
124
125     printf("Uneti podatke o studentima:\n");
126     for(i=0;i<n;i++)
127     {
128         printf("%d. student: ", i);
129         ucitaj(&studenti[i]);
130         proveraj(studenti[i].smer);
131     }
132
133     printf("Uneti smer: ");
134     getchar();
135     scanf("%c", &smer);

```

```

136     provera(smer);

138     printf("Studenti sa R smer:\n");
    for(i=0;i<n;i++)
140         if(studenti[i].smer == smer)
            printf("%s %s\n",studenti[i].ime, studenti[i].prezime);
142     printf("-----\n");

144
146     /* Stampamo podatke o svim studentima sa
    maksimalnim prosekom.
148     */

    max_prosek = najveći_prosek(studenti, n);
150     printf("Svi studenti koji imaju maksimalni prosek:\n");
    for(i=0;i<n;i++)
152         if(studenti[i].prosek == max_prosek)
            ispisi(&studenti[i]);
154
156     return 0;
}

```

## Rešenje 0.1.9

```

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

4  #define IME 21
    #define OCENE 9
6  #define MAX_DJAKA 30

8  typedef struct
    {
10     char ime[IME];
        int ocena[OCENE];
12    } _djak;

14    void provera(int ocena)
    {
16        if (ocena < 1 || ocena > 5)
            {
18            printf("Neispravna ocena.\n");
                exit(EXIT_FAILURE);
20            }
        }

22
    int main()
24    {
        _djak niz[MAX_DJAKA];
26        int i = 0, n, j;

```

```

28     int suma;
    float prosek;

30     printf("Uneti podatke o djaku: ");
    while(scanf("%s", niz[i].ime) != EOF && i < MAX_DJAKA)
32     {
        for(j=0; j<9; j++)
34         {
            scanf("%d", &niz[i].ocena[j]);
36             provera(niz[i].ocena[j]);
        }

38         i++;
40         printf("Uneti podatke o djaku: ");
    }

42     n = i;

44     printf("\n\nNEDOVOLJNI: ");
    for(i=0; i<n; i++)
46         for(j=0; j<9; j++)
48             if (niz[i].ocena[j] == 1)
            {
20             printf("%s ", niz[i].ime);
52             break;
            }
    printf("\n");

54

56     printf("ODLICNI: ");
    for(i=0; i<n; i++)
58     {
        suma = 0;
60         for(j=0; j<9; j++)
            suma += niz[i].ocena[j];

62         prosek = (float)suma/9;

64         if (prosek >= 4.5)
66             printf("%s ", niz[i].ime);
    }
68     printf("\n");

70     return 0;
}

```

## Rešenje 0.1.10

```

1 #include <stdio.h>
  #include <string.h>
3

```



```

5 #define IME 21
6 #define PREZIME 31
7 #define EMAIL 51
8
9 #define MAX_OSOBA 50
10
11 typedef struct
12 {
13     char ime[IME];
14     char prezime[PREZIME];
15     char email[EMAIL];
16 }Osoba;
17
18 int gmail(char* s)
19 {
20     char* deo = strtok(s, "@");
21     deo = strtok(NULL, "");
22
23     if (strcmp(deo, "gmail.com") == 0)
24         return 1;
25     else
26         return 0;
27 }
28
29 int main()
30 {
31     int n, i;
32     Osoba osobe[MAX_OSOBA];
33
34     printf("Uneti broj osoba: ");
35     scanf("%d", &n);
36
37     if (n < 0 || n >= MAX_OSOBA)
38     {
39         printf("Greska u broju osoba.\n");
40         return -1;
41     }
42
43     printf("Uneti podatke o osobama, ime, prezime i email.\n");
44     for(i=0; i<n; i++)
45         scanf("%s%s%s", osobe[i].ime, osobe[i].prezime, osobe[i].email);
46
47     printf("Vlasnici gmail naloga su:\n");
48     for(i=0; i<n; i++)
49         if (gmail(osobe[i].email))
50             printf("%s %s\n", osobe[i].ime, osobe[i].prezime);
51
52     return 0;
53 }

```

Rešenje [0.1.11](#)

```

1 #include <stdio.h>

3 #define MAXART 20
4 #define MAXPOT 100
5 #define MAXNAZIV 31

7 typedef struct artikal
8 {
9     char naziv[MAXNAZIV];
10    int kolicina;
11    float cena;
12 } ARTIKAL;

13 typedef struct korpa
14 {
15     int br_art;
16     ARTIKAL artikli[MAXART];
17 } KORPA;

19 /*
20  * Funkcija ucitaj_artikal ucitava podatke za jedan
21  * artikal i vraca 1 ako je ucitavanje bilo uspesno
22  * a 0 u suprotnom. Ucitavanje je neuspesno ukoliko
23  * kolicina nekog artikla ili njegova cena nisu pozitivni
24  * brojevi.
25  *
26  * S obzirom da funkcija ucitaj_artikal treba da vrati
27  * dve vrednosti (ucitanu strukturu i indikator uspesnosti),
28  * strukturu ARTIKAL prenosimo preko pokazivaca a
29  * indikator uspesnosti vracamo kao povratnu vrednost.
30  */

31 /*
32  *
33  */
34 int ucitaj_artikal(ARTIKAL* a)
35 {
36     printf("Unesi artikal, naziv, kolicinu i cenu: ");
37     scanf("%s", a->naziv);
38     scanf("%d", &a->kolicina);

39     if (a->kolicina<=0)
40     {
41         printf("Nekorektan unos za kolicinu artikla: %d\n", a->kolicina);
42     };
43     return 0;

44 }

45 scanf("%f",&a->cena);
46 if (a->cena<0)
47 {
48     printf("Nekorektan unos za cenu artikla: %f\n", a->cena);
49     return 0;

```

```

51     }

53     return 1;
54 }

55 /*
56 Funkcija izracunaj_racun izracunava racun date
57 potrosacke korpe u kojoj su inicijalizovani
58 podaci o broju artikala i o svakom pojedinacnom
59 artiklu.
60 */
61 float izracunaj_racun(const KORPA* k)
62 {
63     int i;
64     float racun=0;
65     for(i=0; i<k->br_art; i++)
66         racun+=k->artikli[i].kolicina * k->artikli[i].cena;
67     return racun;
68 }

69 /*
70 Pri ucitavanju korpe, zadaje se broj artikala a zatim
71 podaci za svaki artikal.
72
73 Funkcija ucitaj_korpu vraca 1 ako je ucitavanje uspesno
74 i 0 u suprotnom. Do neuspesnog ucitavanja moze doci
75 ako broj artikala u korpi nije pozitivan ili ako dodje
76 do neuspesnog ucitavanja nekog artikla.
77 */
78
79 int ucitaj_korpu(KORPA* k)
80 {
81     int i;
82     printf("Uneti podatke o korpi: \n");
83     printf("Broj artikala: ");
84     scanf("%d", &k->br_art);
85     if (k->br_art<=0)
86     {
87         printf("Nekorektan unos za broj artikala: %d\n", k->br_art);
88         return 0;
89     }
90     for(i=0; i<k->br_art; i++)
91         if (ucitaj_artikal(&k->artikli[i])==0)
92             return 0;
93
94     return 1;
95 }

96
97
98 /*
99 Funkcija ucitaj_niz_korpi ucitava podatke
100 za niz od n potrosackih korpi. Funkcija
101 vraca 1 ako je ucitavanje uspesno i 0 ako

```

```

103     nije. Ucitavanje je neuspesno ukoliko ne uspe
        ucitavanje jedne od korpi.
105 */
107 int ucitaj_niz_korpi(KORPA korpe[], int n)
108 {
109     int i,j;
        for(i=0; i<n; i++)
111         if(ucitaj_korpu(&korpe[i])==0)
            return 0;
113
        return 1;
115 }
117
118 /*
119 Funkcija stampaj_racun ispisuje na
        standardni izlaz racun za datu korpu
121 tako sto za svaki artikal ispise
        naziv, cenu i kolicinu i na kraju
123 ukupnu cenu za kupljene artikle.
124 */
125 void stampaj_racun(const KORPA* k)
126 {
127     int i,j;
128     for(i=0; i<k->br_art; i++)
129         printf("\t%s %d %.2f\n", k->artikli[i].naziv, k->artikli[i].
            kolicina, k->artikli[i].cena);
131     printf("-----\n");
        printf("\tukupno: %.2f\n", izracunaj_racun(k));
133 }
135
136 /*
137 Funkcija stampaj_racune_za_korpe
        ispisuje na standardni izlaz racune
139 za svaku korpu u nizu potrosackih
        korpi
140 */
141
142 void stampaj_racune_za_korpe(KORPA korpe[], int n)
143 {
144     int i;
        for (i=0; i<n; i++)
145     {
146         printf("\nKorpa %d:\n", i);
147         stampaj_racun(&korpe[i]);
148     }
149 }
151
152 /*
153

```

```

155     Funkcija prosek racuna prosecnu cenu
    potrosacke korpe za dati niz potrosackih
    korpi
157 */
    float prosek(KORPA korpe[], int n)
159 {
    int i;
    float p;

161     for(i=0;i<n;i++)
        p+=izracunaj_racun(&korpe[i]);

163     return p/n;
165 }
167
169 int main()
    {
171     int n;
    KORPA korpe[MAXPOT];

173     printf("Uneti broj potrosackih korpi:");
175     scanf("%d", &n);

177     if(n<0 || n>MAXPOT)
    {
179         printf("Nekorektan unos broja potrosackih korpi: %d\n",n);
        return -1;
181     }

183     if (ucitaj_niz_korpe(korpe, n)==0)
        return -1;

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

189     return 0;
    }

```

## Rešenje 0.1.12

```

1 #include <stdio.h>
2 #include <ctype.h>

4 #define BROJ_SIFRI 100
5 #define BROJ_KARAKTERA 5000

6 typedef struct
8 {
    int b;
10     char smer;

```

```

12 }Sifra;
13
14 char sifruj(char c, Sifra s)
15 {
16     int pomeraaj;
17
18     if (!isalnum(c))
19         return c;
20
21     if (s.smer == 'L')
22     {
23         if (isdigit(c))
24         {
25             pomeraaj = s.b % 10;
26
27             if (pomeraaj > c - '0')
28             {
29                 pomeraaj = pomeraaj - (c - '0') - 1;
30                 c = '9' - pomeraaj;
31             }
32             else
33                 c = c - pomeraaj;
34
35             return c;
36         }
37
38         pomeraaj = s.b % 26;
39
40         if (islower(c))
41         {
42             if (pomeraaj > c - 'a')
43             {
44                 pomeraaj = pomeraaj - (c - 'a') - 1;
45                 c = 'z' - pomeraaj;
46             }
47             else
48                 c = c - pomeraaj;
49
50             return c;
51         }
52
53         if (pomeraaj > c - 'A')
54         {
55             pomeraaj = pomeraaj - (c - 'A') - 1;
56             c = 'Z' - pomeraaj;
57         }
58         else
59             c = c - pomeraaj;
60
61         return c;
62     }
63     else

```

```

64 {
65     if (isdigit(c))
66     {
67         pomeraaj = s.b % 10;
68
69         if (pomeraaj > '9' - c)
70         {
71             pomeraaj = pomeraaj - ('9' - c) - 1;
72             c = '0' + pomeraaj;
73         }
74         else
75             c = c + pomeraaj;
76
77         return c;
78     }
79
80     pomeraaj = s.b % 26;
81
82     if (islower(c))
83     {
84         if (pomeraaj > 'z' - c)
85         {
86             pomeraaj = pomeraaj - ('z' - c) - 1;
87             c = 'a' + pomeraaj;
88         }
89         else
90             c = c + pomeraaj;
91
92         return c;
93     }
94
95     if (pomeraaj > 'Z' - c)
96     {
97         pomeraaj = pomeraaj - ('Z' - c) - 1;
98         c = 'A' + pomeraaj;
99     }
100     else
101         c = c + pomeraaj;
102
103     return c;
104 }
105
106 int main()
107 {
108     char linija[BROJ_KARAKTERA];
109     Sifra sifre[BROJ_SIFRI];
110     char c;
111     int n = 0, j = 0, i;
112
113     printf("Uneti sifre u obliku: broj, smer:\n");
114     while (scanf("%d %c", &sifre[n].b, &sifre[n].smer) != EOF)

```

```

116 {
117     if (sifre[n].smer != 'L' && sifre[n].smer != 'D')
118     {
119         printf("Neispravan smer.\n");
120         return -1;
121     }
122
123     if (sifre[n].b < 0)
124     {
125         printf("Neispravan broj za sifrovanje.\n");
126         return -1;
127     }
128
129     n++;
130     if (n == BROJ_SIFRI)
131         break;
132 }
133
134 printf("Uneti tekst za sifrovanje:\n");
135 while((c = getchar()) != EOF)
136 {
137     for(i=0; i<n; i++)
138         c = sifruj(c, sifre[i]);
139
140     linija[j] = c;
141     j++;
142
143     if (j == BROJ_KARAKTERA)
144         break;
145 }
146
147 linija[j] = 0;
148
149 printf("%s\n", linija);
150
151 return 0;
152 }

```

### Rešenje 0.1.13

```

1  #include <stdio.h>
2  #include <math.h>
3
4  #define MAX 50
5
6  typedef struct lopta {
7      int poluprecnik;
8      enum {plava, zuta, crvena, zelena} boja;
9  } LOPTA;
10
11 float zapremina(LOPTA l) {

```



```

13     return pow(l.poluprecnik, 3)*4/3*M_PI;
14 }
15
16 float ukupna_zapremina(LOPTA lopte[], int n) {
17     int i;
18     float z = 0;
19
20     for(i = 0; i < n; i++)
21         z += zapremina(lopte[i]);
22
23     return z;
24 }
25
26 /*
27  Funkcija je opstija od trazene i broji sve lopte odredjene boje u
28  nizu lopti.
29  U zavisnosti od prosledjene boje funkciji, funkcija vraca
30  odgovarajuci broj.
31 */
32 int broj_lopti_u_boji(LOPTA lopte[], int n, int boja) {
33     int br = 0;
34     int i;
35     for(i = 0; i < n; i++)
36         if(lopte[i].boja == boja)
37             br++;
38     return br;
39 }
40
41 int main() {
42
43     LOPTA lopte[MAX];
44     int n;
45     int i;
46     int boja;
47
48     printf("Unesite broj lopti: ");
49     scanf("%d", &n);
50
51     if(n < 1 || n > MAX) {
52
53         printf("Nekorektan unos.\n");
54         return 0;
55     }
56
57     printf("Unesite dalje poluprecnike i boje lopti (1-plava, 2-zuta,
58           3-crvena, 4-zelena):\n");
59     for(i = 0; i < n; i++) {
60
61         printf("%d. lopta: ", i+1);

```

```

61     scanf("%d%d", &lopte[i].poluprecnik, &boja);

63     /* U zavisnosti od unetog celog broja,
        bira se boja lopte.
65     */
        switch(boja) {

67         case 1: lopte[i].boja = plava; break;
69         case 2: lopte[i].boja = zuta; break;
        case 3: lopte[i].boja = crvena; break;
71         case 4: lopte[i].boja = zelena; break;
        default:
73             printf("Nekorektan unos.\n");
            return 0;
75         }
77     }

    printf("Ukupna zapremina: %.2f\n", ukupna_zapremina(lopte, n));

79     printf("Ukupno crvenih lopti: %d\n", broj_lopti_u_boji(lopte, n,
        crvena));

81     return 0;
83 }

```

### Rešenje 0.1.14

```

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

4  #define MAX_TACKA 1000

6  typedef struct
    {
8      int x, y;
    }TACKA;

10

double rastojanje(TACKA a, TACKA b)
12 {
    return sqrt(pow(a.x - b.x, 2) + pow(a.y - b.y, 2));
14 }

16 unsigned ucitaj_poligon(TACKA* tacke, unsigned n)
    {
18     int i = 0;

20     while(i < n && scanf("%d%d", &tacke[i].x, &tacke[i].y) != EOF)
        i++;

22     return i;
24 }

```

```

26 double obim(TACKA* poligon, unsigned n)
27 {
28     double o = rastojanje(poligon[0], poligon[n-1]);
29     int i;
30
31     for(i=0; i<n-1; i++)
32         o += rastojanje(poligon[i], poligon[i+1]);
33
34     return o;
35 }
36
37 double maksimalna_stranica(TACKA* poligon, unsigned n)
38 {
39     double max = rastojanje(poligon[0], poligon[n-1]);
40     double stranica;
41     int i;
42
43     for(i=0; i<n-1; i++)
44     {
45         stranica = rastojanje(poligon[i], poligon[i+1]);
46         if (stranica > max)
47             max = stranica;
48     }
49
50     return max;
51 }
52
53 double povrsina_trougla(TACKA A, TACKA B, TACKA C)
54 {
55     double a = rastojanje(B, C);
56     double b = rastojanje(A, C);
57     double c = rastojanje(A, B);
58
59     double s = (a + b + c)/2;
60
61     return sqrt(s*(s -a)*(s - b)*(s - c));
62 }
63
64 double povrsina(TACKA* poligon, unsigned n)
65 {
66     double P = 0;
67     int i;
68
69     for(i=1; i<n-1; i++)
70         P += povrsina_trougla(poligon[0], poligon[i], poligon[i+1]);
71
72     return P;
73 }
74
75 int main()
76 {

```

```

78     int N;
    unsigned m;
    TACKA poligon[MAX_TACAKA];

80
    printf("Uneti maksimalan broj tacaka poligona: ");
82     scanf("%d", &N);

84     if (N < 3 || N > MAX_TACAKA)
    {
86         printf("Neispravan broj tacaka poligona.\n");
        return -1;
88     }

90     m = ucitaj_poligon(poligon, N);

92     if (m < 3)
    {
94         printf("Neispravan broj tacaka poligona.\n");
        return -1;
96     }

98     printf("Obim poligona je %.3lf.\n", obim(poligon, m));
    printf("Duzina maksimalne stranice je %.3lf.\n",
        maksimalna_stranica(poligon, m));
100    printf("Povrsina poligona je %.3lf.\n", povrsina(poligon, m));

102    return 0;
}

```

## Rešenje 0.1.15

```

1  #include <stdio.h>
    #define MAX 1000

3
    typedef struct
5  {
        char o;
7      int x;
        int y;
9  } IZRAZ;

11
    int korektan_izraz(const IZRAZ izraz)
13  {
        if(izraz.o != '+' && izraz.o != '-' && izraz.o != '*' && izraz.o
            != '/')
15        {
            printf("Nedozvoljena operacija!\n");
17            return 0;
        }
19        if(izraz.o == '/' && izraz.y == 0)

```

```

21     {
22         printf("Deljenje nulom!\n");
23         return 0;
24     }
25     return 1;
26 }
27
28 /* Racunanje vrednosti izraza. */
29 int vrednost(const IZRAZ izraz)
30 {
31     int v;
32
33     switch (izraz.o)
34     {
35         case '+':
36             return izraz.x + izraz.y;
37         case '-':
38             return izraz.x - izraz.y;
39         case '*':
40             return izraz.x * izraz.y;
41         case '/':
42             return izraz.x / izraz.y;
43     }
44 }
45
46 /*
47  Promenljiva izraz ce se promeniti u funkciji
48  ucitaj_izraz tako sto ce njenim neinicijalizovanim
49  poljima o,x,y biti dodeljene vrednosti ucitane
50  sa ulaza. Zbog toga ovu promenljivu
51  funkciji prosledjujemo po adresi, preko pokazivaca.
52
53  S obzirom da ucitavanje karaktera nije prvo
54  ucitavanje koje se obavlja u programu, funkcijom
55  getchar() se ucita karakter kojim se razdvaja
56  unos karaktera od prethodnog unosa (najcesce blanko
57  znak ili znak za novi red).
58
59  */
60
61 int ucitaj_izraz(IZRAZ* izraz)
62 {
63     getchar();
64     scanf("%c%d%d",&izraz->o, &izraz->x, &izraz->y);
65     if (!korektan_izraz(*izraz))
66         return 0;
67     return 1;
68 }
69
70 void stampaj_izraz(const IZRAZ izraz)

```

```

73     printf("%d %c %d = %d\n", izraz.x, izraz.o, izraz.y, vrednost(
        izraz));
75 }
77 int max_vr(IZRAZ izrazi[], int n)
78 {
79     int i;
80     int max;
81
82     max=vrednost(izrazi[0]);
83
84     for(i=1; i<n; i++)
85         if(vrednost(izrazi[i])>max)
86             max=vrednost(izrazi[i]);
87
88     return max;
89 }
90
91 int main()
92 {
93     int n;
94     IZRAZ izrazi[MAX];
95     int max;
96     int i;
97
98     printf("Uneti broj izraza: ");
99     scanf("%d", &n);
100     if(n<0 || n>MAX)
101     {
102         printf("Nekorektna vrednost broja n!\n");
103         return -1;
104     }
105
106     printf("Uneti izraze u prefiksnoj notaciji:\n");
107     for(i=0; i<n; i++)
108         if(ucitaj_izraz(&izrazi[i])==0)
109         {
110             printf("Nekorektan unos\n");
111             return -1;
112         }
113
114
115     max = max_vr(izrazi, n);
116     printf("Maksimalna vrednost izraza:%d\n", max);
117
118     printf("Izrazi cija je vrednost manja od polovine maksimalne
        vrednosti:\n");
119
120     for(i=0; i<n; i++)
121         if(vrednost(izrazi[i])<max/2)

```

```

123         stampaj_izraz(izrazi[i]);
125     return 0;
}

```

## Rešenje 0.1.16

```

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

#define STEPEN 10
#define MAX_POLINOMA 100

typedef struct
{
    int stepen;
    float koef[STEPEN+1];
}_polinom;

int učitaj(_polinom* p)
{
    int i;

    printf("Uneti stepen: ");
    if (scanf("%d", &(p->stepen)) == EOF)
        return 0;

    if (p->stepen > STEPEN || p->stepen < 0)
    {
        printf("Greska u unosu stepena.\n");
        exit(EXIT_FAILURE);
    }

    printf("Uneti koeficijente polinoma:\n");
    for(i=0; i<=p->stepen; i++)
        scanf("%f", &(p->koef[i]));

    return 1;
}

int ispis_prvog_monoma(float koef, int stepen)
{
    if (koef != 0)
    {
        printf("%.2f", koef);

        if (stepen == 1)
            printf("x ");
        else if (stepen > 1)
            printf("x^%d ", stepen);
    }
}

```

```

    return 1;
46 }
    else
48     return 0;
}

50 void ispis_monoma(float koef, int stepen)
52 {
    if (koef != 0)
54     {
        if (koef > 0)
56         printf("+ ");

58         printf("%.2f", koef);

60         if (stepen == 1)
            printf("*x ");
62         else if (stepen > 1)
            printf("*x^%d ", stepen);
64     }
}

66 void ispis(const _polinom *p)
68 {
    int prvi = 1;
70     int i;

72     for(i=0; i <= p->stepen; i++)
        if (prvi)
74         {
            prvi = !ispis_prvog_monoma(p->koef[i], i);
76         }
        else
78         ispis_monoma(p->koef[i], i);

80     printf("\n");
}

82 void integral(const _polinom* p, _polinom* integ)
84 {
    int i;

86     integ->stepen = p->stepen + 1;

88     integ->koef[0] = 0;

90     for(i=1; i <= integ->stepen; i++)
        integ->koef[i] = (float)p->koef[i-1]/i;
92     }

94 }

96 int main()

```



```
98 {
100     _polinom p[MAX_POLINOMA], integ;
102     int i = 0, j;

104     while(ucitaj(&p[i]))
106         i++;

108     printf("\n\nIntegrali su:\n");
110     for(j=0; j<i; j++)
112     {
        integral(&p[j], &integ);
        ispis(&integ);
    }

    return 0;
}
```