

PROGRAMIRANJE 1

**Milena Vujošević Jančić, 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	Funkcije	v
0.2	Rešenja	xxi

0.1 Funkcije

Zadatak 0.1.1 Napisati funkciju `int kvadrat(int x)` koja računa kvadrat datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 15  
| Kvadrat broja 15 je 225
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: -89  
| kvadrat broja -89 je 7921
```

[Rešenje 0.1.1]

Zadatak 0.1.2 Napisati funkciju `int kub(int x)` koja računa kub datog broja. Napisati program koji učitava ceo broj i ispuje rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: 15  
| Kub broja 15 je 3375
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
| Unesite broj: -89  
| Kub broja -89 je -704969
```

[Rešenje 0.1.2]

Zadatak 0.1.3 Napisati funkciju `unsigned int apsolutna_vrednost(int x)` koja izračunava apsolutnu vrednost broja x . Napisati program koji učitava jedan ceo broj i ispisuje rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -34  
|| Apsolutna vrednost: 34
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 5  
|| Apsolutna vrednost: 5
```

[Rešenje 0.1.3]

Zadatak 0.1.4 Napisati funkciju `int min(int x, int y, int z)` koja izračunava minimum tri broja. Napisati program koji učitava tri cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: 19 8 14  
|| Minimum je: 8
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite brojeve: -6 11 -12  
|| Minimum je: -12
```

[Rešenje 0.1.4]

Zadatak 0.1.5 Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja x . Napisati program koji učitava jedan realan broj i ispisuje rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 8.235  
|| Razlomljeni deo: 0.235000
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -5.11  
|| Razlomljeni deo: 0.110000
```

[Rešenje 0.1.5]

Zadatak 0.1.6 Napisati funkciju `float stepen(float x, int n)` koja računa vrednost n -tog stepena realnog broja x . Napisati program koji učitava realan broj x i ceo broj n i ispisuje rezultat rada funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan i jedan ceo broj:  
|| 4.5 3  
|| 4.5000003=91.125000
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite jedan realan i jedan ceo broj:  
|| -33.2 5  
|| -33.2000015=-40335776.000000
```

[Rešenje 0.1.6]

Zadatak 0.1.7 Napisati funkciju `int faktorijel(int n)` koja računa faktorijel broja n . Napisati i program koji učitava dva cela broja x i y iz intervala $[0, 12]$ i ispisuje vrednost zbira $x! + y!$.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 0.1.7]

Zadatak 0.1.8 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 rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva cela broja: 1024 832  
|| Najveci zajednicki delilac je 64
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva cela broja: -900 112  
|| Najveci zajednicki delilac je -4
```

[Rešenje 0.1.8]

Zadatak 0.1.9 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 broj i ispisuje rezultat rada funkcije zaokružen na dve decimale.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesi jedan pozitivan ceo broj: 10  
|| Zbir reciprocnih je 2.93
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesi jedan pozitivan ceo broj: 100  
|| Zbir reciprocnih je 5.19
```

[Rešenje 0.1.9]

Zadatak 0.1.10 Napisati funkciju `aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati program koji učitava ceo broj i ispisuje rezultat rada funkcije zaokružen na tri decimale.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 461  
|| 3.667
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 1001  
|| 0.500
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -84723  
|| 4.800
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 0  
|| 0.000
```

[Rešenje 0.1.10]

Zadatak 0.1.11 Napisati funkciju `void ispis(float x, float y, unsigned 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 odgovarajuće vrednosti i testira rad ove funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva realna broja: 7 32  
|| Unesite jedan prirodan broj: 10  
|| 0.6570 -0.3457 -0.0108 0.3659 -0.6731  
|| 0.8922 -0.9945 0.9666 -0.8122
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva realna broja: 20.5 -8.32  
|| Unesite jedan prirodan broj: 5  
|| -0.8934 -0.8979 -0.1920 0.6658 0.9968
```

[Rešenje 0.1.11]

Zadatak 0.1.12 Napisati funkciju `int broj_neparnih_cifara(int x)` koja određuje broj neparnih cifre u zapisu datog celog broja. Testirati rad ove funkcije u programu koji učitava cele brojeve dok se ne unese nula i ispisuje broj neparnih cifara svakog unetog broja.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite cele brojeve:
2341
Broj neparnih cifara je 2
78
Broj neparnih cifara je 1
800
Broj neparnih cifara je 0
-99761
Broj neparnih cifara je 4
0
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite cele brojeve:
987611
Broj neparnih cifara je 4
135
Broj neparnih cifara je 3
-701
Broj neparnih cifara je 2
602
Broj neparnih cifara je 0
-884
Broj neparnih cifara je 0
79901
Broj neparnih cifara je 4
0
```

[Rešenje 0.1.12]

Zadatak 0.1.13 Napisati funkciju `int zbir_delilaca(int n)` koja izračunava zbir delilaca broja n . Napisati program koji učitava ceo broj k i ispisuje zbir delilaca svakog broja od 1 do k .

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj k: 6
1 3 4 7 6 12
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj k: -2
Greska: pogresan unos!
```

[Rešenje 0.1.13]

Zadatak 0.1.14 Napisati funkciju `int ukloni_stotine(int n)` koja modifikuje zadati broj tako što iz njegovog zapisa uklanja cifru stotina (ako postoji). Napisati program koji za brojeve koji se unose sve do pojave broja 0 ispisuje rezultat poziva funkcije.

Primer 1

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: 1210
110
Unesite broj: 18
18
Unesite broj: 3856
356
Unesite broj: 0
```

Primer 2

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj: -9632
-932
Unesite broj: 246
46
Unesite broj: -52
-52
Unesite broj: 0
```

[Rešenje 0.1.14]

Zadatak 0.1.15 Napisati funkciju `int rotacija(int n)` koja rotira cifre zadatog broja za jednu poziciju u levo. Napisati program koji za brojeve koji se unose sve do pojave broja 0 ispisuje rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 146
|| 461
|| Unesite broj: 18
|| 81
|| Unesite broj: 3856
|| 8563
|| Unesite broj: 7
|| 7
|| Unesite broj: 0
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 89
|| 98
|| Unesite broj: -369
|| -693
|| Unesite broj: -55281
|| -52815
|| Unesite broj: 0
```

[Rešenje 0.1.15]

Zadatak 0.1.16 Broj je prost ako je deljiv samo sa 1 i sa samim sobom. Napisati funkciju `int prost (int x)` koja ispituje da li je dati ceo broj prost. Funkcija treba da vrati 1 ako je broj prost i 0 u suprotnom. Testirati rad funkcije u programu koji za uneti ceo broj n ispisuje prvih n prostih brojeva.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 17
|| Broj je prost!
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 24
|| Broj nije prost!
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: -11
|| Broj nije prost!
```

[Rešenje 0.1.16]

Zadatak 0.1.17 Napisati funkciju `void prosti_brojevi(int m)` koja ispisuje sve proste brojeve manje od broja m . Napisati program koji učitava ceo broj veći od 1 i ispisati rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci. NAPOMENA: *Koristiti funkciju `int prost(int x)`.*

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 15
|| 2 3 5 7 11 13
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj: 9
|| 2 3 5 7
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 1  
|| Greska: pogresan unos!
```

Zadatak 0.1.18 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 1 ako se cifra nalazi u broju, a 0 inače. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj i cifru: 17890 7  
|| Cifra se nalazi u broju.
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj i cifru: 1982 6  
|| Cifra se ne nalazi u broju.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj i cifru: 17890 26  
|| Neispravan unos.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj i cifru: -1982 9  
|| Cifra se nalazi u broju.
```

[Rešenje [0.1.18](#)]

Zadatak 0.1.19 Napisati program za ispitivanje svojstava cifara datog celog broja.

- (a) Napisati funkciju `sve_parne_cifre` koja ispituje da li se dati ceo broj sastoji isključivo iz parnih cifara. Funkcija treba da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.
- (b) Napisati funkciju `sve_cifre_jednake` koja ispituje da li su sve cifre datog celog broja jednake. Funkcija treba da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

Napisati program koji učitava ceo broj i ispisuje da li su sve cifre parne i da li su sve cifre jednake.

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  
|| Sve cifre broja nisu parne.  
|| 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  
|| Sve cifre broja nisu parne.  
|| Cifre broja nisu jednake.
```

[Rešenje 0.1.19]

Zadatak 0.1.20 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 cela pozitivna broja i ispisuje rezultat poziva funkcije. Pretpostaviti da je unos korektan.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 81 3  
|| Jeste: 81 = 34
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 162 11  
|| Broj 162 nije stepen broja 11.
```

[Rešenje 0.1.20]

Zadatak 0.1.21 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 vrši dok je razlika sabiraka u redu po apsolutnoj vrednosti manja od ε . 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 0.1.21]

Zadatak 0.1.22 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 srećan(int x)` koja vraća 1 ako je broj srećan, a 0 u suprotnom. Napisati program koji za uneti prirodan broj n ispisuje sve srećne brojeve od 1 do n . Pretpostaviti da je unos korektan.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 100  
|| Srecni brojevi:  
|| 1 10 19 28 37 46 55 64 73 82 91 100
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 0  
|| Nema srečnih brojeva.
```

[Rešenje 0.1.22]

Zadatak 0.1.23 Napisati funkciju `float aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati i program koji učitava ceo broj i ispisuje rezultat na tri decimale.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 461  
|| 3.667
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 1001  
|| 0.500
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: -84723  
|| 4.800
```

[Rešenje 0.1.23]

Zadatak 0.1.24 Napisati funkciju `int zapis(int x, int y)` koja proverava da li se brojevi x i y zapisuju pomoću istih cifara. Funkcija treba da vrati vrednost 1 ako je uslov ispunjen, a 0 ako nije. Napisati i program koji učitava dva cela broja i ispisuje rezultat primene funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 251 125  
|| Uslov je ispunjen!
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: 8898 9988  
|| Uslov nije ispunjen!
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite dva broja: -7391 1397  
|| Uslov je ispunjen!
```

[Rešenje 0.1.24]

Zadatak 0.1.25 Napisati funkciju `int rastuce(int n)` koja ispituje da li su cifre datog celog broja u rastućem poretku. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i ispisuje poruku da li su cifre unetog broja u rastućem poretku.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 2689  
|| Cifre su u rastucem poretku!
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 559  
|| Cifre su u rastucem poretku!
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite broj: 628  
|| Cifre nisu u rastucem poretku!
```

[Rešenje 0.1.25]

Zadatak 0.1.26

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

- (a) Napisati funkciju `int stepen(int x, int n)` koja izračunava n -ti stepen broja x .
- (b) Napisati funkciju `int armstrong(int x)` koja vraća 1 ako je broj Armstrongov, odnosno 0 ako nije.

Napisati program koji učitava ceo broj i proverava da li je Armstrongov.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 0.1.26]

Zadatak 0.1.27 Napisati funkciju `int par_nepar(int n)` koja ispituje da li su cifre datog celog broja naizmenično parne i neparne. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i testira rad funkcije.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje [0.1.27](#)]

Zadatak 0.1.28 Napisati funkciju `int prebrojavanje(float x)` koja prebrojava koliko puta se broj x pojavljuje u nizu brojeva koji se unose sve do pojave nule. Napisati program koji učitava vrednost broja x i testira rad napisane funkcije.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj x: 2.84
|| Unesite brojeve: 8.13 2.84 5 21.6 2.84 11.5 0
|| Broj pojavljivanja broja 2.84 je: 2
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj x: -1.17
|| Unesite brojeve: -128.35 8.965 8.968 89.36 0
|| Broj pojavljivanja broja -1.17 je: 0
```

[Rešenje [0.1.28](#)]

Zadatak 0.1.29 Napisati funkciju `long unsigned fibonacci(int n)` koja računa n -ti element Fibonačijevog niza. Napisati i program koji učitava ceo broj n ($0 \leq n \leq 50$) i ispisuje traženi Fibonačijev broj.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:
|| Unesite broj n: 7
|| 21
```

Primer 2

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

[Rešenje [0.1.29](#)]

Zadatak 0.1.30 Napisati funkciju `int konverzija (int c)` koja prebacuje veliko slovo u ekvivalentno malo i obrnuto. Napisati program koji testira ovu funkciju na karakterima koji se unose do pojave znaka EOF.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: ZDRAVO  
|| zdravo
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite karaktere: Dobro jutro, R2D2!  
|| dOBRO JUTRO, r2d2!
```

[Rešenje 0.1.30]

Zadatak 0.1.31 Napisati funkciju `char sifra(char c, int k)` koja za dati karakter `c` određuje šifru na sledeći način: ukoliko je `c` slovo, šifra je karakter koji se nalazi `k` pozicija ispred njega u abecedi. Karakteri koji nisu slova se ne šifruju. Šifrovanje treba da bude kružno, što znači da je, na primer, šifra za karakter `b` i pomeraj 2 karakter `z`. Napisati program koji učitava karakter po karakter do kraja ulaza i ispisuje šifrovani tekst.

Primer 1

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

[Rešenje 0.1.31]

Zadatak 0.1.32 Napisati funkciju `int prestupna(int godina)` koja za zadatu godinu proverava da li je prestupna. Funkcija treba da vrati 1 ako je godina prestupna ili 0 ako nije. Napisati program koji učitava dva cela broja `g1` i `g2` i ispisuje sve godine iz intervala $[g1, g2]$ koje su prestupne.

Primer 1

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

Primer 2

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


Primer 3

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

Primer 4

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

[Rešenje 0.1.32]

Zadatak 0.1.33 Napisati funkciju `int broj_dana(int mesec, int godina)` koja za dati mesec i godinu vraća broj dana u datom mesecu. Napisati program koji učitava dva cela broja (mesec i godinu) i ispisuje broj dana u datom mesecu. U slučaju nekorektnog unosa ispisati odgovarajuću poruku o grešci.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite mesec i godinu: 8  
|| 1998  
|| Broj dana je: 31
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite mesec i godinu: 2  
|| 2004  
|| Broj dana je: 29
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite mesec i godinu: 24  
|| 2004  
|| Neispravan unos.
```

Zadatak 0.1.34 Napisati funkciju `int ispravan(int dan, int mesec, int godina)` koja za dati datum proverava da li je ispravan. Napisati program koji učitava tri cela broja (dan, mesec, godinu) i ispisuje da li je datum ispravan ili ne.

Primer 1

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 24.8.1998.  
|| Datum je ispravan.
```

Primer 2

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.1789.  
|| Datum nije ispravan.
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 29.2.2004.  
|| Datum je ispravan.
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 29.14.2004.  
|| Datum nije ispravan.
```

Zadatak 0.1.35 Napisati funkciju `void sledeci_dan(int dan, int mesec, int godina)` koja za dati datum određuje datum sledećeg dana. Napisati program koji učitava tri cela broja i ispisuje datum sledećeg dana.

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.  
|| Datum nije ispravan.
```

Zadatak 0.1.36 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 i ispisuje koliko dana je proteklo od Nove godine.

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: 28.2.2004.  
|| Broj dana od Nove godine je: 58
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Datum nije ispravan.
```

Zadatak 0.1.37 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 i ispisuje broj dana do kraja godine.

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 od Nove godine je: 0
```

Primer 3

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 28.2.2004.  
|| Broj dana od Nove godine je: 307
```

Primer 4

```
|| INTERAKCIJA SA PROGRAMOM:  
|| Unesite datum: 31.4.2004.  
|| Datum nije ispravan.
```

[Rešenje 0.1.38]

Zadatak 0.1.38 Napisati funkciju `int broj_dana_između(int dan1, int mesec1, int godina1, int dan2, int mesec2, int godina2)` koja određuje broj dana između dva datuma. Napisati program koji učitava šest celih brojeva, koji označavaju dva datuma, i na standardni izlaz ispisuje broj dana između ta dva datuma. U slučaju pogrešnog unosa ispisati 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: 104319
```

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 24.8.1998.
Unesite drugi datum: 12.10.2010.
Broj dana izmedju dva datuma je: 4432
```

Primer 4

```
INTERAKCIJA SA PROGRAMOM:
Unesite prvi datum: 24.8.1998.
Unesite drugi datum: 31.4.2004.
Ulaz nije ispravan.
```

[Rešenje 0.1.38]

Zadatak 0.1.39 Napisati funkciju `void romb(int n)` koja iscrtava romb čija je stranica dužine n . Napisati program koji učitava ceo pozitivan broj i ispisuje rezultat poziva funkcije.

Primer 1

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

Primer 2

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

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite broj n: -5
Greska: pogresna dimenzija!
```

[Rešenje 0.1.39]

Zadatak 0.1.40 Napisati funkciju `void grafikon_h(int a, int b, int c, int d)` koja iscrtava horizontalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i prikazuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

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

[Rešenje 0.1.40]

Zadatak 0.1.41 Napisati funkciju `void grafikon_v(int a, int b, int c, int d)` koja iscrtava vertikalni prikaz zadatih vrednosti. Napisati program koji učitava četiri pozitivna cela broja i ispisuje rezultat poziva funkcije. U slučaju pogrešnog unosa, ispisati poruku o grešci.

Primer 1

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

Primer 2

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

Primer 3

```
INTERAKCIJA SA PROGRAMOM:
Unesite vrednosti: 5 2 2 4
*
* *
* *
****
****
```

[Rešenje 0.1.41]

0.2 Rešenja

Rešenje 0.1.1

```
1 #include <stdio.h>
3 int kvadrat(int x)
4 {
5     /* promenljive u listi argumenata funkcije, kao i one
6        deklarisanе u samoj funkciji, lokalne su za tu funkciju
7        sto znaci da se promenljive x i y neće "videti" nigde izvan
8        funkcije kvadrat (ni u funkciji main ni u funkciji kub)
9        */
10
11     int y;
12     y = x*x;
13     return y;
14 }
15
16 int main()
17 {
18     int a, kv, kb;
19     printf("Unesi ceo broj:");
20     scanf("%d", &a);
21
22     kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
23        funkcije kvadrat */
24
25     printf("Kvadrat broja %d je %d\n", a, kv);
26     return 0;
27 }
```

Rešenje 0.1.2

```
1 #include <stdio.h>
3 int kub(int a)
4 {
5     /*
6        u listi argumenata funkcije mozemo, a ne moramo, imati
7        promenljivu
8        istog naziva kao promenljiva koja je deklarisanа u main
9        funkciji
10        (u ovom slucaju promenljiva a); ova promenljiva se razlikuje
11        od promenljive a deklarisanе u main funkciji i vidljiva je
12        samo unutar funkcije kub
13        */
14     return a*a*a;
15 }
```

```

15 int main()
16 {
17     int a, kb;
18     printf("Unesi ceo broj:");
19     scanf("%d",&a);

21     kb = kub(a);      /* promenljivoj kb dodeljujemo povratnu vrednost
                          funkcije kub */

23     printf("Kub broja %d je %d\n", a, kb);
24     return 0;
25 }

```

Rešenje 0.1.3

```

1  #include <stdio.h>

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

9  int main(){
10     int n;

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

14     /* Ispisujemo njegovu apsolutnu vrednost */
15     printf("Apsolutna vrednost: %u\n", apsolutna_vrednost(n));

17     return 0;
18 }

```

Rešenje 0.1.4

```

1  #include <stdio.h>

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

9      min=x;

```

```

11     if(min>y)
12         min=y;
13
14     if(min>z)
15         min=z;
16
17     return min;
18 }
19
20 int main(){
21     int x,y,z;
22
23     /* Ucitavamo brojeve */
24     printf("Unesite brojeve: ");
25     scanf("%d%d%d", &x, &y, &z);
26
27     /* Pozivamo funkciju i ispisujemo rezultat */
28     printf("Minimum je: %d\n", min(x,y,z));
29
30     return 0;
31 }

```

Rešenje 0.1.5

```

1  #include<stdio.h>
2  #include<math.h>
3
4  /* Funkcija koja vraca razlomljeni deo prosledjenog broja */
5  float razlomljeni_deo(float x){
6
7      /* Funkcija fabs vraca apsolutnu vrednost realnog broja
8       * NAPOMENA: funkcija fabs se nalazi u zaglavlju math.h
9       * NAPOMENA2: funkcija abs se nalazi u zaglavlju stdlib.h, ali se
10        koristi samo za cele brojeve!
11       */
12      x = fabs(x);
13
14      /* Razlomljeni deo broja dobijamo tako sto od samog broja oduzmemo
15       njegov ceo deo*/
16      return x - (int)x;
17 }
18
19 int main(){
20     float n;
21
22     /* Ucitavamo broj */
23     printf("Unesite broj:");
24     scanf("%f", &n);
25
26     /* Ispisujemo rezultat */

```

```

25     printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));
27     return 0;
}

```

Rešenje 0.1.6

```

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

/* Funkcija za realan broj x i ceo broj n odredjuje vrednost stepena
   x^n. */
float stepen(float a, int b)
{
    float s=1;
    int i, abs_b = abs(b);

    for(i=0; i<abs_b; i++)
        s=s*a;

    /* ukoliko je izlozilac b negativan, izracunamo a^|b| i vracamo
       reciprocnu vrednost izracunatog stepena */
    return b>0 ? s : 1/s;
}

int main()
{
    int n;
    float x;
    float s;

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

    s = stepen(x,n);

    printf("%f^%d=%f\n",x,n,s);

    return 0;
}

```

Rešenje 0.1.7

```

#include <stdio.h>

/* Funkcija racuna faktoriyel broja.
   * Faktoriyel formiramo mnozenjem sa trenutnom vrednoscu broja x,
   * a zatim smanjujuci tu vrednost za 1.
   * Ukoliko je x = 5, f = 5 * 4 * 3 * 2 * 1

```



```

7  */
   int faktorijel(int x) {
9
   int f = 1;
11  while(x) {
      f *= x;
13      x--;
   }
15  return f;
   }
17
18  int main() {
19
   int x, y;
21
   printf("Unesite dva broja: ");
23   scanf("%d%d", &x, &y);

24   /* Provera uslova.
   *
25   * Faktorijel je veoma brza funkcija, tj.
26   * s povecanjem broja x, drasticno brzo uvecava se i vrednost x!.
27   * Tip podatka int ima ogranicenje u velicini broja koji moze da
28   * sadrzi.
29   * Za 13! i vece, int ne bi mogao da sacuva sve cifre potrebne za
30   * zapis tako velikog broja,
31   * te bi doslo do prekoracenja.
32   *
33   * Slicno, faktorijel nije definisan nad skupom negativnih celih
34   * brojeva.
35   */
36   if(x < 0 || y < 0 || x > 12 || y > 12) {
37       printf("Greska: pogresan unos!\n");
   }
   else{
39       printf("%d\n", faktorijel(x) + faktorijel(y));
   }
41   return 0;
   }

```

Rešenje 0.1.8

```

1  #include <stdio.h>

3  int euklid(int x, int y)
   {
5      int r;
      /* Euklidov algoritam */
7      while(y) /* algoritam se zaustavlja kada vrednost */
      { /* promenljive y postane nula */
9          r=x%y;
      }
   }

```

```

11     x=y;
    y=r;
}

13
15     return x; /* nzd je sacuvan u promenljivoj x */
}

17 int main()
{
19     int a,b;
    int nzd;

21
    printf("Unesite dva cela broja:");
23     scanf("%d%d", &a,&b);

25     nzd = euklid(a,b); /* promenljivoj nzd dodeljujemo povratnu
        vrednost funkcije euklid */

27     printf("Najveci zajednicki delilac je %d\n", nzd);

29     return 0;
}

```

Rešenje 0.1.9

```

1 #include <stdio.h>

3 float zbir_reciprocnih(int n)
{
5     float z=0;
    int i;
7     for(i=1;i<=n;i++)
        z+=1.0/i; /* da bismo dobili reciprocnu vrednost broja, vazno je
            da izbegnemo celobrojno deljenje dva cela broja */
9     return z; /* tako sto ce npr deljenik biti 1.0 umesto 1 */
}

11
13 int main()
{
15     int n;
    printf("Unesi jedan pozitivan ceo broj:\n");
    scanf("%d", &n);
17     printf("Zbir je %.2f\n", zbir_reciprocnih(n));
    /* povratna vrednost funkcije zbir_reciprocnih je float; funkciju
        mozemo pozvati u okviru
19         naredbe printf i umesto specifikatora %.2f bice ispisana
        povratna vrednost funkcije
        zbir_reciprocnih zaokruzena na dve decimale */
21     return 0;
}

```

Rešenje 0.1.10

```
1  #include<stdio.h>
   #include<stdlib.h>
3
   float aritmeticka_sredina(int x)
5  {
       int zbir_cifara=0;
       int broj_cifara=0;
       char cifra;
9
       if (x==0)      /* u slucaju da je uneta 0 */
11      return 0;     /* aritmeticka sredina cifara iznosi 0 i tu vrednost
                       vracamo */

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

15      while(x)
17      {
          cifra=x%10;

19          broj_cifara++;
          zbir_cifara+=cifra;

21          x/=10;
23      }

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

29  }

31  int main()
   {
33      int x;
       printf("Unesi jedan ceo broj:");
35      scanf("%d",&x);
       printf("Aritmeticka sredina cifara broja %d iznosi %.2f\n", x,
           aritmeticka_sredina(x));
37      return 0;
   }
```

Rešenje 0.1.11

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

```

2  #include <math.h>

4  void ispis(float x, float y, int n) /* funkcija nema povratnu
    vrednost; zbog toga je povratni tip void */
{
6     float i;
    float korak=(y-x)/(n-1);

8     for(i=x;i<=y;i+=korak)
10        printf("%.4f ", sin(i));

12    printf("\n");

14 }

16 int main()
{
18     float a,b;
    int n;
20     float t;
    printf("Unesi dva realna broja:");
22     scanf("%f%f",&a,&b);
    printf("Unesi jedan prirodan broj:");
24     scanf("%u",&n);

26     if (n<=1 || a==b)
    {
28         printf("Nekorektan unos\n");
        return -1;
30     }
    if (b<a) /* u slucaju da je desni kraj intervala manji od levog */
32     {
        /* zamenimo im mesta */
34         t=a;
        a=b;
        b=t;
36     }

38

40     ispis(a,b,n);

42     return 0;

44 }

```

Rešenje 0.1.12

```

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

3
    int broj_ncifara(int x)

```

```

5 {
6     int s=0;
7     char cifra;
8     x = abs(x);
9
10    while(x)
11    {
12        cifra = x%10;
13        s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
14                       neparna,
15                       a 0 kada je cifra parna */
16        x/=10;
17    }
18    return s;
19 }
20
21 int main()
22 {
23     int x;
24     do
25     {
26         scanf("%d",&x);
27         printf("%d\n", broj_ncifara(x));
28     } while(x!=0);
29
30     return 0;
31 }

```

Rešenje 0.1.13

```

1  #include <stdio.h>
2
3  /* Funkcija koja racuna zbir delilaca broja x */
4  int zbir_delilaca(int x){
5      int i=0;
6
7      /* Na pocetku zbir inicijalizujemo na 0 */
8      int zbir = 0;
9
10     /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
11     for(i=1; i<=x; i++){
12         if(x % i == 0)
13             zbir += i;
14     }
15
16     /* Vracamo dobijeni zbir */
17     return zbir;
18 }
19
20 int main(){

```

```

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

```

Rešenje 0.1.14

```

1  #include <stdio.h>
2
3  /* Funkcija koja uklanja broj stotina iz broja n */
4  int ukloni_stotine(int n){
5
6      /* Ako je broj izmedju -100 i 100 nema cifru desetica pa onda
7         vratimo isti taj broj */
8      if(n>-100 && n<100)
9          return n;
10     else
11     {
12         /* U suprotnom vratimo broj sa uklonjenom cifrom stotina */
13
14         /* Odredjujemo znak broja */
15         int znak=(n<0)? -1 : 1;
16
17         /* I nadalje radimo sa apsolutnom vrednoscu broja */
18         n=abs(n);
19
20         return znak*((n/1000)*100 + n%100);
21     }
22 }
23
24 /* Funkcija koja vraca znak broja */
25 int znak(int broj){
26     return broj<0?-1:1;
27 }

```

```

26 }
28 int main(){
30     int broj;
32     while(1){
34         /* Ucitavamo broj sa standardnog ulaza */
35         printf("Unesite broj: ");
36         scanf("%d", &broj);
38         /* Broj 0 oznacava kraj rada */
39         if(broj == 0)
40             break;
42         /* Ispisujemo rezultat, vodeci racuna da program treba da radi
43            ispravno i za negativne brojeve */
44         printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
45     }
46
47     return 0;
48 }

```

Rešenje 0.1.15

```

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

4 int rotacija(int n){
6     /* U promenljivoj broj pamtimo originalnu vrednost n */
7     int broj, br = 0, znak;
8
9     /* Odredjujemo znak broja */
10    znak=(n<0) ? -1: 1;
12
13    /* I nadalje radimo sa apsolutnom vrednoscu broja */
14    n=abs(n);
15
16    /* U promenljivoj broj cuvamo kopiju broja n */
17    broj=n;
18
19    /* Ako je broj jednocifren, nema potrebe da ga rotiramo. */
20    if(n>-10 && n < 10)
21        return n;
22
23    /* Petljom izdvajamo cifru po cifru, kako bismo dosli do krajnje
24       leve cifre broja

```

```

24     (one koja treba da postane krajnje desna), npr za n = 1234, treba
        da dobijemo 1,
        zatim da "pomerimo" 234 u levo i da na kraj nalepimo 1 = 2341 */
26     /* Na kraju ove petlje, u n se nalazi najlevlja cifra broja (koja
        treba da postane krajnje desna),
        dok se u br nalazi broj cifara unetog broja */
28     while(n >=10){
        n/=10;
30         br++;
    }
32
    /*
34     Levi deo (234) dobijamo kao n%(10^broj_cifara)
    Zatim levi deo pomnozimo sa 10, da bi dobili 2340
36     Zatim na levi deo dodamo desni deo (1) koja se nalazi u
        promenljivoj n
    */
38
    return znak* ((broj%(int)pow(10, br))*10 + n);
40 }

42 int main(){

44     int n;
    while(1){
46
        /* Ucitavamo broj */
48         printf("Unesite broj: ");
        scanf("%d", &n);
50
        /* Ako je uneta 0, izlazimo iz petlje */
52         if(n == 0)
            break;
54
        /* Stampamo broj rotiran za jedno mesto u levo */
56         printf("%d\n", rotacija(n));
    }
58

60     return 0;
}

```

Rešenje 0.1.16

```

1  #include <stdio.h>
    #include <math.h>
3
    /* Funkcija vraca 1 ako je broj prost i 0 u suprotnom. */
5  int prost (int x) /* 1-broj je prost, 0-broj nije prost */
    {

```



```

7   int i;

9   if (x==2 || x==3) /* brojevi 2 i 3 su prosti */
       return 1;

11  if (x%2==0)          /* parni brojevi nisu prosti */
13      return 0;

15  for (i=3; i<=sqrt(x);i+=2) /* trazimo delioca */
       if (x%i==0) /* ako je pronadjen, to znaci da broj nije prost */
17          return 0; /* završavamo funkciju */

19  /* ukoliko izvršavanje funkcije dodje do poslednje naredbe return,
       to znaci da broj nije ispunio nijedan od prethodnih uslova
       (nije ni 2, ni 3, ni paran, niti ima ijednog delioca), odakle
       sledi da je prost i zbog toga vracamo 1
21  */
23  return 1;
25 }

27 int main()
{
29     int n;
       scanf("%d",&n);
31     int i,j;

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

45     return 0;
}

```

Rešenje 0.1.18

```

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

3

   /* Funkcija ispituje da li se cifra c nalazi u zapisu celog broja x.
       */
5  int sadrzi(int x, int c)
   {

```

```

7      char cifra;
      x=abs(x);
9      while(x)
      {
11         cifra = x%10;
         if (cifra==c)
13             return 1;
         x/=10;
15     }
     return 0;
17 }
int main()
19 {
    int x;
21    int c;
    printf("Unesi jedan ceo broj i jednu cifru:");
23    scanf("%d%d",&x,&c);
    if (sadrzi(x,c))
25        printf("Cifra %d se nalazi u zapisu broja %d\n",c,x);
    else
27        printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
    return 0;
29 }

```

Rešenje 0.1.19

```

/*
2
a) Napisati funkciju sve_parne_cifre koja ispituje da li se dati ceo
    broj sastoji isključivo iz parnih cifara. Funkcija treba
4    da vrati 1 ako su sve cifre broja parne i 0 u suprotnom.

6    b) Napisati funkciju sve_cifre_jednake koja ispituje da li su sve
    cifre datog celog broja jednake. Funkcija treba
    da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.
8
c) Napisati potom glavni program koji na uneti ceo broj primenjuje
    napisane funkcije i ispisuje odgovarajuće poruke.
10
Na primer, za uneti broj 222, program treba da ispise:
12 Sve cifre broja su parne.
    Sve cifre broja su jednake.
14
A za uneti broj -284:
16 Sve cifre broja su parne.
    Broj sadrži različite cifre
18
*/
20 #include <stdio.h>
    #include <stdlib.h>
22

```

```

24 int sve_parne_cifre(int x) /* funkcija vraca 1 ako su sve cifre broja
    parne i 0 u suprotnom*/
{
26     char d;
    x=abs(x); /* uzimamo apsolutnu vrednost broja za slucaj da je
        broj negativan */
    while (x>0)
28     {
        d=x%10; /* izdvajamo cifru broja */

30         if (d%2==1) /* u slucaju da je neparna, to znaci da nisu sve
            cifre broja parne */
32             return 0; /* vracamo 0 */

34         x/=10; /* "uklanjamo" poslednju cifru broja celobrojnim
            deljenjem sa 10 */
    }

36     return 1; /* ukoliko se while petlja zavrсила, to znaci da
        uslov d%2==1 nije
38         nijednom bio ispunjen i da su sve cifre broja
            parne; zbog toga
                vracamo 1
40         */
42 }

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

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

56         if (d!=prva_cifra)
            return 0;

58         x/=10;
60     }

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

```

```

68     int d;

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

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

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

82 }

```

Rešenje 0.1.20

```

1  #include <stdio.h>

3  /* Funkcija za dva uneta neoznacena broja x i n utvrđuje da li je x
    neki stepen
    broja n. Ukoliko jeste, funkcija vraca izlozilac stepena, a u
    suprotnom vraca -1. */
5  int je_stepen(unsigned x, unsigned n)
    {
7      int i=1;
        int s=n;

9

        while(s<x)
11         {
            s=s*n;
13             i++;
        }

15         if (s==x)
17             return i;

19         return -1;
    }

21
22 int main()
23 {
24     unsigned x;
25     unsigned n;
26     int st;

27     scanf("%u%u",&x,&n);

29     st = je_stepen(x,n);

```

```

31     if (st!=-1)
32         printf("%u=%u^%d\n",x,n,st);
33     else
34         printf("%u nije stepen broja %u\n",x,n);
35
36     return 0;
37 }

```

Rešenje 0.1.21

```

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

/* Funkcija racuna vrednost e^x kao parcijalnu sumu reda
suma(x^n/n!), gde indeks n ide od
od 0 do beskonacno, pri cemu se sumiranje vrši dok
je razlika sabiraka u redu po apsolutnoj vrednosti
manja od eps. */
double e_na_x(double x, double eps)
{
    double s=1;
    double clan=1;
    int n=1;

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

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

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

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

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

    return s;
}

```

```

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

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

```

Rešenje 0.1.22

```

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

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

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

#include<stdio.h>

int zbir_cifara(int x)
{
    int s=0;
    char cifra;
    while(x)
    {
        cifra = x%10;
        s+=cifra;
    }
}

```

```

29     x/=10;
30 }
31 return s;
32 }
33
34 int srecan(int x)
35 {
36     int s; /* promenljiva s sadrzi sumu cifara */
37
38     do
39     {
40         s=zbir_cifara(x);
41         x=s; /* kada izracunamo sumu cifara, dodeljujemo je promenljivoj
42             x jer iz te promenljive izdvajamo cifre u funkciji zbir_cifara
43             */
44     } while(x>=10);
45
46     return (x==1);
47 }
48
49 int main()
50 {
51     unsigned n;
52     int i;
53     printf("Unesi jedan neoznaceni broj:");
54     scanf("%u",&n);
55
56     for(i=1;i<=n;i++)
57         if (srecan(i))
58             printf("%d je srecan\n", i);
59
60     return 0;
61 }

```

Rešenje 0.1.23

```

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
7      /* Proveravamo slucaj broja 0 */
8      if(x==0)
9          return 0;
10
11     /* Brojaci sa svojim pocetnim vrednostima */
12     int zbir_cifara = 0;
13     int broj_cifara = 0;

```

```

15  /* U slucaju da je broj negativan,
16     * ostaci pri deljenju sa 10 bi takodje bili negativni
17     * sto bi se moralo dodatno proveravati.
18     * Stoga je zgodnije posmatrati samo apsolutnu vrednost broja.
19     */
20  x = abs(x);
21
22  /* Izdvajamo cifru po cifru s kraja zapisa broja x.
23     * Uslov while(x > 0) ekvivalentan je uslovu while(x)
24     * Broj x je pozitivan (apsolutna vrednost), pa cifre izdvajati
25     * sve dok ima cifara izdvajanje, tj. broj x nije deljenjem sa 10
26     * postao 0
27     */
28  while(x) {
29
30      zbir_cifara += x % 10;
31      broj_cifara++;
32
33      x /= 10;
34  }
35
36  /* Da nije izvršena implicitna konverzija (kastovanje),
37     * operator / obavljao bi celobrojno deljenje.
38     * Zato je potrebno da bar jedan operand bude ceo broj,
39     * sto je u ovom slucaju deljenik.
40     * Operator kastovanja je unaran, pa ima veci prioritet od /
41     */
42  return (float) zbir_cifara / broj_cifara;
43 }
44
45 int main() {
46
47     int x;
48
49     printf("Unesite ceo broj: ");
50     scanf("%d", &x);
51
52     printf("Aritmeticka sredina cifara broja %d je %.3f\n", x,
53         aritmeticka_sredina(x));
54
55     /* Iako smo u funkciji aritmeticka_sredina menjali broj x,
56        * prosledjivanjem argumenata funkciji pravi se lokalna kopija
57        * promenljive x za tu funkciju,
58        * tako da je ona menjana unutar funkcije aritmeticka_sredina,
59        * a promenljiva x iz funkcije main() ostaje netaknuta.
60        */
61
62     return 0;
63 }

```

Rešenje 0.1.24


```

1  #include <stdio.h>

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

17  int izbaci_cifru(int y, int cifra) {
19
21     int novo_y = 0;
     int indikator = 0;
     int izdvojena_cifra;
23
     while(y) {
25
         izdvojena_cifra = y % 10;
27         /* U slucaju da se cifra razlikuje od one koju treba eliminisati,
            * ili ukoliko je jedna cifra vec eliminisana =>
29             * tekucu cifru ukljuciti prilikom formiranja novog y
            * */
31         if(izdvojena_cifra != cifra || indikator)

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

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

47         y /= 10;

```

```

    }
49     return novo_y;
51 }

53 int zapis(int x, int y) {

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

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

61     while(x) {

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

67         y = izbaci_cifru(y, cifra);

69         /* otkomentarisati donju liniju radi lakseg pracenja rada
        programa: */
        // printf("Iz x izdvojeno: %d\n\tx = %d, y = %d\n\n", cifra, x, y
        );
71     }

73     return (x == 0 && y == 0);
}

75 int main() {

77     int x, y;
79     printf("Unesite dva cela broja: ");
    scanf("%d%d", &x, &y);

81     if(zapis(x, y))
83         printf("Uslov je ispunjen!\n");
    else
85         printf("Uslov nije ispunjen!\n");

87     return 0;
}

```

Rešenje 0.1.25

```

1 #include <stdio.h>
  #include <stdlib.h>
3
  /* Funkcija proverava da li se
5  * cifre u zapisu broja nalaze u rastucem poretku.

```

```

7  *
  * Situacija od interesa je kada za dve uzastopne cifre to nije
    slucaj.
  * Tada ne treba proveravati i za ostale cifre,
9  * vec odmah prekinuti izvršavanje funkcije.
  *
11 * Ukoliko funkcija nije ranije prekinuta,
  * to znaci da cifre jesu u rastucem poretku
13 * (odnosno, kako izdvajamo cifre od nazad, u stvari proveravamo
    opadajuci poredak),
  * te treba vratiti 1.
15 */

17 int rastuce(int n) {
19     int tekuca_cifra;
    int prethodna_cifra;
21
    n = abs(n);
23
    /* Prvu cifru (odnosno, poslednju u zapisu broja)
25     * izdvajamo izvan petlje
    * kako bismo mogli da je poredimo sa narednom
27     */
    tekuca_cifra = n % 10;
29     n /= 10;

31     while(n) {

33         /* Cifra koja je bila tekuca u prethodnoj iteraciji petlje,
        * u novoj iteraciji postaje prethodna.
35         *
        * Novoizdvojena cifra je tekuca.
37         */
        prethodna_cifra = tekuca_cifra;
39         tekuca_cifra = n % 10;

41         /* Ukoliko smo naisli na cifre koje kvare rastuci poredak,
        * prekidamo izvršavanje funkcije sa odgovarajucom povratnom
        vrednoscu 0.
43         */
        if(prethodna_cifra < tekuca_cifra)
45             return 0;

47         /* Inace, nastavljamo sa izdvajanjem cifara */
        n /= 10;
49     }

51     return 1;
53 }

int main() {

```

```

55     int x;
57     printf("Unesite broj: ");
58     scanf("%d", &x);
59
60     if(rastuce(x))
61         printf("Cifre su u rastucem poretku!\n");
62     else
63         printf("Cifre nisu u rastucem poretku!\n");
64
65     return 0;
66 }

```

Rešenje 0.1.26

```

1  #include <stdio.h>
2
3  /* Funkcija racuna broj x na n-ti stepen */
4  int stepen(int x, int n) {
5
6      int i;
7      /* Promenljiva u kojoj se cuva proizvod broja x sa samim sobom, n
8       puta */
9      int st = 1;
10
11     for(i = 1; i <= n; i++)
12         st *= x;
13
14     return st;
15 }
16
17 /* Funkcija proverava da li je broj Armstrongov. */
18 int armstrong(int x) {
19     /* u y se cuva zbir i-tih stepena cifara */
20     int y;
21     /* stepen za koji se proverava */
22     int i = 1;
23     /* prilikom izdvajanja cifara, broj x se menja,
24      * te treba imati promenlju koja cuva pravu vrednost x
25      */
26     int original = x;
27
28     do {
29
30         y = 0;
31         /* Racunamo i-te stepene za svaku cifru,
32          * i istovremeno te stepen sabiramo.
33          * Rezultat pamtimo u promenljivoj y.
34          */
35         while(x) {

```

```

37     y += stepen(x % 10, i);
38     x /= 10;
39 }

41 /* x je sada promenjen, pa ga treba vratiti na pravu vrednost. */
42 x = original;
43 i++;

45 } while(y < x); /* Petlju vrtimo sve dok je zbir stepena cifara
46    manji od datog broja. */

47 /* Ukoliko smo nasli i, takvo da je zbir i-tih stepena cifara
48    * jednak upravo broju x, takav broj je Armstrongov,
49    * te izraz x == y vraca 1.
50    *
51    * Inace, vraca 0, tj. broj nije Armstrongov.
52    */
53 return x == y;
54 }

55 int main() {
56
57     int x;
58     printf("Unesite broj: ");
59     scanf("%d", &x);

61     if(armstrong(x))
62         printf("Broj je Armstrongov!\n");
63     else
64         printf("Broj nije Armstrongov!\n");

65     return 0;
66 }

```

Rešenje 0.1.27

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  /* Funkcija proverava da li su dve uzastopne cifre
5     * razlicite parnosti.
6     *
7     * Interesantna situacija je ukoliko su dve uzastopne cifre
8     * obe parne, odnosno obe neparne.
9     * Ovaj uslov svodimo na poredjenje njihovih ostataka pri deljenju sa
10     * 2:
11     * ukoliko su ostaci isti, cifre su iste parnosti,
12     * te ne treba dalje proverati da li je uslov zadovoljen,
13     * vec odmah prekinuti sa izvršavanjem funkcije.
14     */

```

```

15  * Ukoliko dve uzastopne cifre ni u jednom slucaju nisu bile iste
    parnosti,
16  * a izdvojene su sve cifre iz broja x,
    * uslov je ispunjen, pa funkcija vraca 1.
17  */
    int par_nepar(int x) {
18
19      int prethodna_cifra;
20      int tekuca_cifra;
21
22      /* u slucaju da je uneti broj negativan */
23      x = abs(x);
24
25      /* jednu cifru izdvajamo van petlje
26       * kako bismo mogli da je odmah u petlji poredimo sa narednom
27       */
28      prethodna_cifra = x % 10;
29      x /= 10;
30
31      while(x) {
32
33          tekuca_cifra = x % 10;
34
35          if(tekuca_cifra % 2 == prethodna_cifra % 2)
36              return 0;
37
38          /* tekuca cifra postaje prethodna cifra za narednu iteraciju */
39          prethodna_cifra = tekuca_cifra;
40          x /= 10;
41      }
42
43      return 1;
44  }
45
46  int main() {
47
48      int x;
49      printf("Unesite broj: ");
50      scanf("%d", &x);
51
52      if(par_nepar(x))
53          printf("Broj ispunjava uslov!\n");
54      else
55          printf("Broj ne ispunjava uslov!\n");
56
57      return 0;
58  }
59

```

Rešenje 0.1.28

```

2  #include <stdio.h>
4  /* Funkcija broji koliko puta se realan broj x
   * javlja u nizu unetih brojeva sa tastature.
   */
6  * Brojevi se unose sve do pojave 0,
   * pa treba koristiti do..while petlju,
8  * kako bi bar jedan broj bio unet (makar bio i 0).
   */
10 int prebrojavanje(float x) {
12     /* y prihvata uneti broj sa tastature */
    float y;
14     /* br_pojavljivanja je brojac koji broji koliko puta se broj x
       * javlja u unetom nizu brojeva */
    int br_pojavljivanja = 0;

    printf("Unesite brojeve: ");
18     do {

20         /* Unosimo broj. */
        scanf("%f", &y);

22         /* Poredimo uneti broj sa datim brojem.
           * Ukoliko je unet bas trazeni broj,
           * uvecavamo brojac.
           */
24         /* */
        if(x == y)
26             br_pojavljivanja++;

30     } while(y); /* Sve dok nije uneta 0 */

32     return br_pojavljivanja;
}

34 int main() {
36     float x;
38     int br_pojavljivanja;

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

42     br_pojavljivanja = prebrojavanje(x);
44     printf("Broj pojavljivanja broja %.2f je: %d\n", x,
           br_pojavljivanja);

46     return 0;
}

```

Rešenje **0.1.29**

```

1 #include <stdio.h>

3 /* Funkcija racuna n-ti clan Fibonaccijevog niza.
   * Clanovi ovog niza zadaju se rekurzivno tj. u zavisnosti od
   * prethodnih clanova.
5  * Fibonaccijevi brojevi od 0. do 47. se mogu smestiti u tip int, a
   * kako n moze uzimati vrednosti
   * od 1 do 50, povratni tip funkcije je long int.
7  */
long int fibonaci(int n) {
9
11     int i;

13     /* f0 i f1 su prva dva clana niza */
    int f0 = 1;
    int f1 = 1;
15     /* promenljiva u kojoj se cuvaju opsti clanovi: n+2, n+1. i n-ti
       clan */
    long int fn2, fn1, fn;

17     /* ukoliko treba vratiti nulti ili prvi clan,
       * njih ne treba racunati
       * jer su vec dati.
21    */
    if(n == 0 || n == 1)
23        return 1;

25     /* postavljamo prethodne clanove niza */
    fn = f0;
27     fn1 = f1;
    /* racunamo od drugog clana, pa dok ne dodjemo do n-tog */
29     for(i = 2; i <= n; i++) {

31         /* izracunamo n+2-i clan niza sabiranjem prethodna dva clana */
        fn2 = fn1 + fn;
33         /* promenimo prethodne clanove niza, zbog naredne iteracije */
        fn = fn1;
35         fn1 = fn2;
    }

37     return fn2;
39 }

41 int main() {

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

47     /* Provera vrednosti za broj n */
    if(n < 0 || n > 50) {

```



```

49     printf("Greska: nedozvoljena vrednost!\n");
    }
51     else{
        printf("%ld\n", fibonaci(n));
53     }

55     return 0;
}

```

Rešenje 0.1.30

```

1  /*
   . a) Napisati funkciju
3
   int konverzija (int c)
5
   koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.
7
   b) Napisati program koji omogućava korisniku da unese niz karaktera
9   sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
   Na primer, za uneti tekst "Kolokvijum iz Prog1 je 1.12." program
11  treba da ispise "kOLOVKIJUM IZ pROG1 JE 1.12."

13  */
#include <stdio.h>
15
16  int konverzija(int c)
17  {
    /* ključna rec return vraća povratnu vrednost funkcije (ako je ima)
       */
19    /* i završava izvršavanje funkcije */

21    if (c>='A' && c<='Z')
        return c+'a'-'A';
23
24    if (c>='a' && c<='z')
25        return c-'a'+'A';

27    return c;
    }
29
30  int main()
31  {
    int c;
33

34    while((c=getchar())!=EOF) /* korisnik unosi karakter po karakter
       do konstante EOF */
35        putchar(konverzija(c)); /* funkcija putchar ispisuje jedan
       karakter na standardni izlaz */
37

    return 0;
}

```

39 }

Rešenje 0.1.31

```
1 #include <stdio.h>
3 /* Funkcija vraca karakter koji se u abecedi
   * nalazi k mesta pre datog karaktera c
   */
5 char sifra(char c, int k) {
7     /* Ukoliko je uneto malo slovo ... */
9     if(c >= 'a' && c <= 'z')
10         /* Pri tome karakter koji je k pozicija pre datog karaktera
11            ispada iz opsega malih slova ... */
12         if(c-k < 'a')
13             /* Treba krenuti s drugog kraja abecede, racunajuci i
14                preskocena slova.
15                *
16                * Na primer, ukoliko je c = 'b' i k = 2
17                * Jedan karakter pre 'b' je 'a'.
18                * Dva karaktera pre 'b' je 'z' (kruzno).
19                *
20                * Karakter iz prvog dela abecede, koji je preskocen, je 'a'.
21                * Broj preskocenih karaktera iz prvog dela abecede
22                * racunamo tako sto izracunamo c - 'a' (rastojanje od datog
23                * karaktera do malog slova a)
24                * sto je u ovom slucaju 'b' - 'a' = 1.
25                *
26                * Ostatak karaktera do k ispisujemo, ali gledavsi unazad od z.
27                * Zato racunamo k - (c - 'a') - 1.
28                *
29                * Od k oduzimamo rastojanje izmedju c i 'a',
30                * kako bismo dobili preostali broj karaktera koji treba
31                * preskociti.
32                */
33             return 'z' - (k - (c - 'a') - 1);
34     else
35         /* U suprotnom, karakter ne ispada iz opsega malih slova, te je
36            dovoljno bas njega i vratiti */
37         return c-k;
38
39     /* Ukoliko je uneto veliko slovo ... */
40     else if(c >= 'A' && c <= 'Z')
41         if(c-k < 'A')
42             return 'Z' - (k - (c - 'A') - 1);
43         else
44             return c-k;
45
46     return c;
```

```

43 }

45 int main() {

47     int k;
48     char c;

49     printf("Unesite broj k: ");
51     scanf("%d", &k);

53     printf("Unesite tekst (CTRL + D za prekid): ");
54     while((c = getchar()) != EOF)
55         putchar(sifra(c, k));

57     return 0;
}

```

Rešenje 0.1.32

```

1  #include<stdio.h>

3  /* Funkcija koja proverava da li je godina prestupna */
4  int prestupna(int godina){
5      if((godina %100 != 0 && godina%4 == 0) || godina%400 == 0)
6          return 1;
7      else
8          return 0;
9  }

11 /* Funkcija koja proverava da li postoji prestupna godina u datom
    intervalu */
12 int postoji_prestupna(int g1, int g2){
13     for(; g1<=g2; g1++){
14         if(prestupna(g1))
15             return 1;
16     }
17     return 0;
18 }

19
20 int main(){
21
22     int g1, g2;

23     /* Ucitavamo godine */
24     printf("Unesite dve godine: ");
25     scanf("%d%d", &g1, &g2);

26
27     /* Proveravamo korektnost ulaza */
28     if(g1 < 0 || g2 < 0 || g1>g2){
29         printf("Greska: pogresan unos!\n");
30     }
31 }

```

```

33     else{
34         /* Proveravamo da li uopste postoji prestupna godina u datom
           intervalu */
35     if(postoji_prestupna(g1,g2)){
36         /* Ako postoje, ispisujemo ih */
37         printf("Prestupne godine su: ");
38         for(; g1<=g2; g1++){
39             if(prestupna(g1))
40                 printf("%d ", g1);
41         }
42         printf("\n");
43     }else{
44         /* U suprotnom, stampamo odgovarajucu poruku */
45         printf("Nema prestupnih godina u ovom intervalu!\n");
46     }
47 }
48 return 0;
49 }

```

Rešenje 0.1.38

Rešenje 0.1.38

Rešenje 0.1.39

```

1  #include<stdio.h>
2
3  /* Funkcija koja iscrtava romb */
4  void romb(int n){
5      int i, j;
6
7      /* U svakoj liniji */
8      for(i=0; i<n; i++){
9
10         /* Prvo ispisujemo n-i-1 razmaka */
11         for(j=0; j<n-i-1; j++){
12             printf(" ");
13         }
14
15         /* Zatim ispisujemo n zvezdica */
16         for(j=0; j<n; j++){
17             printf("*");
18         }
19         /* Na kraju svake linije stoji oznaka za novi red */
20         printf("\n");
21     }
22 }
23

```

```

25 int main(){
26     int n;
27
28     /* Ucitavamo broj n */
29     printf("Unesite broj n: ");
30     scanf("%d", &n);
31
32     /* Proveravamo korektnost ulaza i ispisujemo rezultat */
33     if(n<=0)
34         printf("Greska: pogresna dimenzija!\n");
35     else
36         romb(n);
37
38     return 0;
39 }

```

Rešenje 0.1.40

```

#include<stdio.h>
2
/* Funkcija koja 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 koja crta grafikon */
13 void grafikon_h(int a, int b, int c, int d)
14 {
15     int i;
16
17     /* Prvo ispisujemo a zvezdica */
18     stampaj_zvezdice(a);
19
20     /* Zatim u sledecem redu b zvezdica */
21     stampaj_zvezdice(b);
22
23     /* Zatim u sledecem redu c zvezdica */
24     stampaj_zvezdice(c);
25
26     /* Zatim u poslednjem redu d zvezdica */
27     stampaj_zvezdice(d);
28 }
29
30 int main(){
31     int a,b,c,d;
32

```

```

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

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

44
    return 0;
46 }

```

Rešenje 0.1.41

```

1  #include<stdio.h>

3  int maksimum(int a, int b, int c, int d) {
    int max;

5
    max=a;
7    if(b>max)
        max=b;
9    if(c>max)
        max=c;
11   if(d>max)
        max=d;

13
    return max;
15 }

17 void stampaj_znak(int polje, int granica) {
    if(polje<granica)
19     printf(" ");
    else
21     printf("*");
}

23
/* Funkcija koja iscrtava vertikalni grafikon */
25 void grafikon_v(int a, int b, int c, int d){
    int i, max;

27
    /* Na pocetku je potrebno pronaci najvecu od ove cetiri vrednosti
       */
29    max=maksimum(a, b, c, d);

31    /* Grafikon ukupno ima max horizontalnih linija */
    for(i=0; i<max; i++){
33

```

```

35     /* U svakoj od horizontalnih linija se nalazi po 4 polja:
36     polje za a,b,c i d uspravnu liniju.
37     U svako od polja treba da se upise ili * ili belina,
38     u zavisnosti od vrednosti i toga u kojoj liniji se trenutno
39     nalazimo
40     */
41
42     /* Stampamo znak za polje a */
43     stampaj_znak(i, max-a);
44
45     /* Proveravamo uslov za polje b */
46     stampaj_znak(i, max-b);
47
48     /* Proveravamo uslov za polje c */
49     stampaj_znak(i, max-c);
50
51     /* Proveravamo uslov za polje d */
52     stampaj_znak(i, max-d);
53
54     /* Na kraju svake horizontalne linije stampamo novi red */
55     printf("\n");
56 }
57
58 int main(){
59     int a,b,c,d;
60
61     /* Ucitavamo vrednosti a,b,c,d */
62     printf("Unesite vrednosti: ");
63     scanf("%d%d%d%d", &a, &b, &c, &d);
64
65     /* Proveravamo korektnost ulaza i stampamo grafikon */
66     if(a <0 || b<0 || c<0 || d<0)
67         printf("Greska: pogresan unos!\n");
68     else
69         grafikon_v(a,b,c,d);
70
71     return 0;
72 }

```