PROGRAMIRANJE 1

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

PROGRAMIRANJE 1
Zbirka zadataka

Beograd
2016.

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

PROGRAMIRANJE 1
Zbirka zadataka

Sadržaj

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
```

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

Zadatak 0.1.3 Napisati funkciju `float stepen(float x, int n)` koja računa vrednost n-tog stepena realnog broja x. Napisati program koji učitava relan 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
```

Zadatak 0.1.4 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
```

Zadatak 0.1.5 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 funkcje 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
```

Zadatak 0.1.6 Napisati funkciju `aritmeticka_sredina(int n)` koja računa aritmetičku sredinu cifara datog broja. Napisati program koji učitava ceo broj i ispisuje rezultat rada funkcje 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
```

Zadatak 0.1.7 Napisati funkciju `void ispis(float x, float y, unsigned n)` koja za dva realna broja x i y i jedan neoznačeni ceo broj n ispisuje vrednosti

sinusne funkcije u n ravnomerno raspoređenih tačaka intervala $[x, y]$. Napisati program koji učitava dva realna broja i jedan ceo pozitivan broj i ispisuje rezultat rada 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
```

Zadatak 0.1.8 Napisati funkciju `int broj_ncifara(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 broja:
 2341 78 800 -99761 0
 Broj neparnih cifara je 2
 Broj neparnih cifara je 1
 Broj neparnih cifara je 0
 Broj neparnih cifara je 4
```

Primer 2

```
Interakcija sa programom:
 Unesite cele broja:
 987611 135 -701 602 -884 79901 0
 Broj neparnih cifara je 4
 Broj neparnih cifara je 3
 Broj neparnih cifara je 2
 Broj neparnih cifara je 0
 Broj neparnih cifara je 0
 Broj neparnih cifara je 4
```

Zadatak 0.1.9 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
```

Zadatak 0.1.10 Napisati funkciju `unsigned int apsolutna_vrednost(int x)` koja izračunava apsolutnu vrednost broja x. Napisati program koji sa 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
```

Zadatak 0.1.11 Napisati funkciju `float razlomljeni_deo(float x)` koja izračunava razlomljeni deo broja x. Napisati program koji sa 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
```

Zadatak 0.1.12 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!
```

Zadatak 0.1.13 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
 *****
 **
 **
 **********
```

Zadatak 0.1.14 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
 *
 *  *
 *  *
 ****
 ****
```

Zadatak 0.1.15 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!
```

Zadatak 0.1.16 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.

Zadatak 0.1.17 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.

Zadatak 0.1.18 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.

Zadatak 0.1.19 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.

Zadatak 0.1.20 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 krja godine.

Zadatak 0.1.21 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 standarni izlaz ispisuje broj dana između ta dva datuma. U slučaju pogrešnog unosa ispisati poruku o grešci.

Zadatak 0.1.22 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!
```

Zadatak 0.1.23 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
```

Zadatak 0.1.24 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
```

Zadatak 0.1.25 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!
```

Zadatak 0.1.26 Napisati funkciju `int sadrzi(int x, int c)` koja ispituje da li se cifra c nalazi u zapisu celog broja x. Napisati program koji učitava dva cela broja i ispisuje rezultat poziva funkcije.

Zadatak 0.1.27 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.

Zadatak 0.1.28 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.

Zadatak 0.1.29 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.

Zadatak 0.1.30 Za dati broj može se formirati niz tako da je svaki sledeći član niza dobijen kao suma cifara prethodnog člana niza. Broj je *srećan* ako se dati niz završava jedinicom. Napisati funkciju `int srecan(int x)` koja vraća 1 ako je broj srećan, a 0 u suprotnom. Napisati program koji za uneti prirodan broj n ispisuje sve srećne brojeve od 1 do n.

Zadatak 0.1.31 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`.

Zadatak 0.1.32 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: *Napomena: koristiti funkciju* ***`int prost(int x`*** napisanu u prethodnom zadatku.

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 0.1.33 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
```

Zadatak 0.1.34 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!
```

Zadatak 0.1.35 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
```

Zadatak 0.1.36 Napisati funkciju `int rastuce(int n)` koja ispituje da li su cifre datog celog broja u rastućem poretku. Funkcija treba da vrati vrednost 1 ako cifre ispunjavaju uslov, odnosno 0 ako ne ispunjavaju uslov. Napisati i program koji učitava ceo broj i ispisuje rezultat poziva funkcije.

Primer 1

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

Primer 2

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

Primer 3

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

Zadatak 0.1.37 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!
```

Zadatak 0.1.38 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!
```

Zadatak 0.1.39 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
```

Zadatak 0.1.40 Napisati funkciju `long int fibonaci(int n)` koja računa n-ti element Fibonačijevog niza. Napisati i program koji učitava ceo broj n ($0 \leq n \leq 50$) i ispisuje traženi Fibonačijev broj.

Primer 1

```
Interakcija sa programom:
 Unesite broj n: 7
 21
```

Primer 2

```
Interakcija sa programom:
 Unesite broj n: 65
 Greska: nedozvoljena vrednost!
```

Zadatak 0.1.41 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
```

0.2 Rešenja

Rešenje 0.1.1

```c
#include <stdio.h>

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

   int y;
   y = x*x;
   return y;
}

int main()
```

```c
{
    int a,kv, kb;
    printf("Unesi ceo broj:");
    scanf("%d",&a);

    kv = kvadrat(a); /* promenljivoj kv dodeljujemo povratnu vrednost
     funkcije kvadrat */

    printf("Kvadrat broja %d je %d\n", a, kv);
    return 0;
}
```

Rešenje 0.1.2

```c
#include <stdio.h>

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

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

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

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

Rešenje 0.1.3

```c
/*
Napisati program koji za uneti realan broj x i ceo broj n ispisuje
vrednost stepena x^n. Unosenje promenljivih, racunanje stepena i
ispis promenljivih realizovati u posebnim funkcijama.
```

```c
*/

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

float stepen(float a, int b)
{
  float s=1;
  int i;

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

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

}

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


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

  s = stepen(x,n);


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

  return 0;
}
```

Rešenje 0.1.4

```c
#include <stdio.h>

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

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

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 0.1.5

```
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
int main()
13 {
  int n;
15   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 0.1.6

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

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

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


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

   while(x)
   {
      cifra=x%10;

      broj_cifara++;
      zbir_cifara+=cifra;

      x/=10;
   }

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

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

Rešenje 0.1.7

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

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

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

  printf("\n");

}

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

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



  ispis(a,b,n);


  return 0;
}
```

Rešenje 0.1.8

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

int broj_ncifara(int x)
{
   int s=0;
```

```
    char cifra;
    x = abs(x);

    while(x)
    {
       cifra = x%10;
       s+=(cifra%2); /* izraz cifra%2 ima vrednost 1 kada je cifra
     neparna,
                        a 0 kada je cifra parna */
       x/=10;
    }

    return s;
}

int main()
{
  int x;
  do
  {
     scanf("%d",&x);
     printf("%d\n", broj_ncifara(x));
  } while(x!=0);

  return 0;
}
```

Rešenje 0.1.9

```
#include <stdio.h>

/*
  Funkcija koja racuna minimum tri cela broja
*/
int min(int x, int y, int z){
  int min;

  min=x;

  if(min>y)
    min=y;

  if(min>z)
    min=z;

  return min;
}

int main(){
  int x,y,z;
```

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

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

Rešenje 0.1.10

```
1   #include <stdio.h>

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

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

19    return 0;
    }
```

Rešenje 0.1.11

```
1   #include<stdio.h>
    #include<math.h>
3
    /* Funkcija koja vraca razlomljeni deo prosledjenog broja */
5   float razlomljeni_deo(float x){

7     /* Funkcija fabs vraca apsolutnu vrednost realnog broja
       * NAPOMENA: funkcija fabs se nalazi u zaglavlju math.h
9      * NAPOMENA2: funkcija abs se nalazi u zaglavlju stdlib.h, ali se
        koristi samo za cele brojeve!
      */
11    x = fabs(x);
```

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

int main(){
  float n;

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

  /* Ispisujemo rezultat */
  printf("Razlomljeni deo: %.6f\n", razlomljeni_deo(n));

  return 0;
}
```

Rešenje 0.1.12

```c
#include<stdio.h>

/* Funkcija koja iscrtava romb */
void romb(int n){
  int i, j;

  /* U svakoj liniji */
  for(i=0; i<n; i++){

  /* Prvo ispisujemo n-i-1 razmaka */
  for(j=0; j<n-i-1; j++)
    printf(" ");

  /* Zatim ispisujemo n zvezdica */
  for(j=0; j<n; j++)
    printf("*");

  /* Na kraju svake linije stoji oznaka za novi red */
  printf("\n");
  }

}

int main(){
  int n;

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

  /* Proveravamo korektnost ulaza i ispisujemo rezultat */
```

```
  if(n<=0)
33  printf("Greska: pogresna dimenzija!\n");
  else
35  romb(n);

37  return 0;
}
```

Rešenje 0.1.13

```
#include<stdio.h>
2
/* Funkcija koja stampa n zvezdica za kojima sledi znak za novi red
    */
4 void stampaj_zvezdice(int n){
  int i;
6 for(i=0; i<n; i++)
    printf("*");
8
  printf("\n");
10 }

12 /* Funkcija koja crta grafikon */
void grafikon_h(int a, int b, int c, int d)
14 {
  int i;
16
  /* Prvo ispisujemo a zvezdica */
18 stampaj_zvezdice(a);

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

26 /* Zatim u poslednjem redu d zvezdica */
  stampaj_zvezdice(d);
28
}
30
int main(){
32 int a,b,c,d;

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

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

```
  }else{
  grafikon_h(a,b,c,d);
  }

  return 0;
}
```

Rešenje 0.1.14

```
#include<stdio.h>

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

  max=a;
  if(b>max)
    max=b;
  if(c>max)
    max=c;
  if(d>max)
    max=d;

  return max;
}

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

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

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

    /* U svakoj od horizontalnih linija se nalazi po 4 polja:
    polje za a,b,c i d uspravnu liniju.
    U svako od polja treba da se upise ili * ili belina,
    u zavisnosti od vrednosti a i toga u kojoj liniji se trenutno
    nalazimo
   */

  /* Proveravamo uslov za polje a */
  if(i<max-a)
    printf(" ");
  else
    printf("*");

  /* Proveravamo uslov za polje b */
  if(i<max-b)
```

```
41      printf(" ");
    else
43      printf("*");

45    /* Proveravamo uslov za polje c */
    if(i<max-c)
47      printf(" ");
    else
49      printf("*");

51    /* Proveravamo uslov za polje d */
    if(i<max-d)
53      printf(" ");
    else
55      printf("*");

57    /* Na kraju svake horizontalne linije stampamo novi red */
    printf("\n");
59    }
}
61
int main(){
63    int a,b,c,d;

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

69    /* Proveravamo korektnost ulaza i stampamo grafikon */
    if(a <0 || b<0 || c<0 || d<0)
71    printf("Greska: pogresan unos!\n");
    else
73    grafikon_v(a,b,c,d);

75    return 0;
}
```

Rešenje 0.1.15

```
1 #include<stdio.h>

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

11 /* Funkcija koja proverava da li postoji prestupna godina u datom
      intervalu */
```

```c
int postoji_prestupna(int g1, int g2){
  for(; g1<=g2; g1++){
  if(prestupna(g1))
    return 1;
  }
  return 0;
}

int main(){

  int g1, g2;

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

  /* Proveravamo korektnost ulaza */
  if(g1 < 0 || g2 < 0 || g1>g2){
  printf("Greska: pogresan unos!\n");
  }
  else{

  /* Proveravamo da li uopste postoji prestupna godina u datom
     intervalu */
  if(postoji_prestupna(g1,g2)){
    /* Ako postoje, ispisujemo ih */
    printf("Prestupne godine su: ");
    for(; g1<=g2; g1++){
    if(prestupna(g1))
      printf("%d ", g1);
    }
    printf("\n");
  }else{
    /* U suprotnom, stampamo odgovarajucu poruku */
    printf("Nema prestupnih godina u ovom intervalu!\n");
  }
  }
  return 0;
}
```

Rešenje 0.1.32

```c
#include <stdio.h>

/* Funkcija koja racuna zbir delilaca broja x */
int zbir_delilaca(int x){
  int i=0;

  /* Na pocetku zbir inicijalizujemo na 0 */
  int zbir = 0;

```

```
  /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
  for(i=1; i<=x; i++){
  if(x % i == 0)
    zbir += i;
  }

  /* Vracamo dobijeni zbir */
  return zbir;
}

int main(){

  int k, i;

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

  /* Proveravamo korektnost ulaza */
  if(k <= 0)
    printf("Greska: pogresan unos!\n");
  else{

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

    printf("\n");
  }


  return 0;
}
```

Rešenje 0.1.32

Rešenje 0.1.32

Rešenje 0.1.32

Rešenje 0.1.32

Rešenje 0.1.32

Rešenje 0.1.22

```c
#include <stdio.h>

/* Funkcija koja racuna zbir delilaca broja x */
int zbir_delilaca(int x){
  int i=0;

  /* Na pocetku zbir inicijalizujemo na 0 */
  int zbir = 0;

  /* Svaki broj izmedju 1 i x koji deli broj x dodajemo u zbir. */
  for(i=1; i<=x; i++){
  if(x % i == 0)
    zbir += i;
  }

  /* Vracamo dobijeni zbir */
  return zbir;
}

int main(){

  int k, i;

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

  /* Proveravamo korektnost ulaza */
  if(k <= 0)
    printf("Greska: pogresan unos!\n");
  else{

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

    printf("\n");
  }


  return 0;
}
```

Rešenje 0.1.23

```c
#include <stdio.h>

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

```
  /* Ako je broj izmedju -100 i 100 nema cifru desetica pa onda
     vracamo isti taj broj */
  if(n>-100 && n<100)
  return n;
  else
  {
  /* U suprotnom vracamo broj sa uklonjenom cifrom stotina */

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

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

  return znak*((n/1000)*100 + n%100);
  }
}

/* Funkcija koja vraca znak broja */
int znak(int broj){
  return broj<0?-1:1;
}

int main(){

  int broj;

  while(1){

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

  /* Broj 0 oznacava kraj rada */
  if(broj == 0)
    break;

  /* Ispisujemo rezultat, vodeci racuna da program treba da radi
     ispravno i za negativne brojeve */
  printf("%d\n", znak(broj)*ukloni_stotine(abs(broj)));
  }


  return 0;
}
```

Rešenje 0.1.24

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

```c
int rotacija(int n){

  /* U promenljivoj broj pamtimo originalnu vrednost n */
  int broj, br = 0, znak;

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

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

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

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

  /* Petljom izdvajamo cifru po cifru, kako bismo dosli do krajnje
    leve cifre broja
   (one koja treba da postane krajnje desna), npr za n = 1234, treba
    da dobijemo 1,
   zatim da "pomerimo" 234 u levo i da na kraj nalepimo 1 = 2341 */

  /* Na kraju ove petlje, u n se nalazi najlevlja cifra broja (koja
    treba da postane krajnje desna),
   dok se u br nalazi broj cifara unetog broja */
  while(n >=10){
    n/=10;
    br++;
  }

  /*
   Levi deo (234) dobijamo kao n%(10^broj_cifara)
   Zatim levi deo pomnozimo sa 10, da bi dobili 2340
   Zatim na levi deo dodamo desni deo (1) koja se nalazi u
    promenljivoj n
   */

  return znak* ((broj%(int)pow(10, br))*10 + n);
}

int main(){

  int n;
  while(1){

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

  /* Ako je uneta 0, izlazimo iz petlje */
```

```
  if(n == 0)
    break;

  /* Stampamo broj rotiran za jedno mesto u levo */
  printf("%d\n", rotacija(n));
  }


  return 0;
}
```

Rešenje 0.1.25

```
/*
Napisati funkciju koja ispituje da li je dati ceo broj prost.
    Funkcija treba
da vrati 1 ako je broj prost i 0 u suprotnom. Napisati potom glavni
    program
koji za uneti ceo broj n ispisuje prvih n prostih brojeva.
*/

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

int prost (int x) /* 1-broj je prost, 0-broj nije prost */
{
  int i;

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

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

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

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

int main()
{
  int n;
  scanf("%d",&n);
  int i,j;
```

```
37
   i=1; /* kandidat za prost broj */
39 j=0; /* brojac prostih brojeva */
   while(j<n)
41 {
      if (prost(i))          /* ako je broj prost */
43    {
         printf("%d\n", i); /* stampamo ga i */
45       j++;               /* uvecavamo brojac prostih brojeva */
      }
47    i++; /* bilo da je i prost ili ne, uvecavamo ga za 1 i
     nastavljamo sa sledecom iteracijom */
   }
49
   return 0;
51 }
```

Rešenje 0.1.26

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

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

```
  else
     printf("Cifra %d se ne nalazi u zapisu broja %d\n",c,x);
  return 0;
}
```

Rešenje 0.1.27

```
/*

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

b) Napisati funkciju sve_cifre_jednake koja ispituje da li su sve
    cifre datog celog broja jednake. Funkcija treba
da vrati 1 ako su sve cifre broja jednake i 0 u suprotnom.

c) Napisati potom glavni program koji na uneti ceo broj primenjuje
    napisane funkcije i ispisuje odgovarajuce poruke.

Na primer, za uneti broj 222, program treba da ispise:
Sve cifre broja su parne.
Sve cifre broja su jednake.

A za uneti broj -284:
Sve cifre broja su parne.
Broj sadrzi razlicite cifre

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

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

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

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

```

```c
  return 1;      /* ukoliko se while petlja zavrsila, to znaci da
    uslov d%2==1 nije
                    nijednom bio ispunjen i da su sve cifre broja
    parne; zbog toga
                    vracamo 1
                */

}

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

  while(x)
  {
     d = x%10;

     if (d!=prva_cifra)
        return 0;

     x/=10;
  }

  return 1;
}
main()
{
  int x;
  int d;

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

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

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

}
```

Rešenje 0.1.28

```c
/*
Napisati funkciju koja za dva uneta neoznacena broja x i n utvrdjuje da li je x neki stepen
broja n. Ukoliko jeste, funkcija vraca izlozilac stepena, a u suprotnom vraca -1. Napisati
potom glavni program koji testira ovu funkciju.
*/

#include <stdio.h>

int je_stepen(unsigned x, unsigned n) /* funkcija vraca izlozilac stepena ukoliko broj x jeste neki stepen broja n */
{
   int i=1;
   int s=n;

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

   if (s==x)
      return i;

   return -1;
}

int main()
{
  unsigned x;
  unsigned n;
  int st;

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

  st = je_stepen(x,n);

  if (st!=-1)
    printf("%u=%u^%d\n",x,n,st);
  else
    printf("%u nije stepen broja %u\n",x,n);

  return 0;
}
```

Rešenje 0.1.29

```c
/*

 Napisati funkciju

 double e_na_x(double x, double eps)

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

*/

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

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()
```

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

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

Rešenje 0.1.30

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

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

14 Napisati program koji omogucava korisnuku da unese prirodan broj,
     poziva funkciju
i ispisuje da li je dati broj srecan. Potom traziti od korisnika da
     unese prirodan
16 broj n i ispisati sve srecne brojeve od 1 do n.
*/
18

#include<stdio.h>
20

int zbir_cifara(int x)
22 {
    int s=0;
24    char cifra;
    while(x)
26    {
       cifra = x%10;
28       s+=cifra;
       x/=10;
```

```
    }
    return s;
}

int srecan(int x)
{
    int s; /* promenljiva s sadrzi sumu cifara */

    do
    {
      s=zbir_cifara(x);
      x=s; /* kada izracunamo sumu cifara, dodeljujemo je promenljivoj
      x jer iz te promenljive izdvajamo cifre u funkciji zbir_cifara
     */
    } while(x>=10);

    return (x==1);

}

int main()
{
    unsigned n;
    int i;
    printf("Unesi jedan neoznacen broj:");
    scanf("%u",&n);

    for(i=1;i<=n;i++)
       if (srecan(i))
          printf("%d je srecan\n", i);

  return 0;
}
```

Rešenje 0.1.31

```
/*
. a) Napisati funkciju

  int konverzija (int c)

koja prebacuje veliko slovo u ekvivalentno malo i obrnuto.

b) Napisati program koji omogucava korisniku da unese niz karaktera
sa tastature, a potom ispisuje uneseni niz konvertovanih karaktera.
Na primer, za uneti tekst "Kolokvijum iz Prog1 je 1.12." program
treba da ispise "kOLOVKIJUM IZ pROG1 JE 1.12."

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

```c
int konverzija(int c)
{
  /* kljucna rec return vraca povratnu vrednost funkcije (ako je ima)
     */
  /* i zavrsava izvrsavanje funkcije */

  if (c>='A' && c<='Z')
    return c+'a'-'A';

  if (c>='a' && c<='z')
    return c-'a'+'A';

  return c;
}

int main()
{
  int c;

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

  return 0;
}
```

Rešenje 0.1.33

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

/* Funkcija racuna aritmeticku sredinu cifara datog celog broja */
float aritmeticka_sredina(int x) {

  /* Proveravamo slucaj broja 0 */
  if(x==0)
    return 0;

  /* Brojaci sa svojim pocetnim vrednostima */
  int zbir_cifara = 0;
  int broj_cifara = 0;

  /* U slucaju da je broj negativan,
   * ostaci pri deljenju sa 10 bi takodje bili negativni
   * sto bi se moralo dodatno proveravati.
   * Stoga je zgodnije posmatrati samo apsolutnu vrednost broja.
   */
  x = abs(x);

  /* Izdvajamo cifru po cifru s kraja zapisa broja x.
```

```
    * Uslov while(x > 0) ekvivalentan je uslovu while(x)
    * Broj x je pozitivan (apsolutna vrednost), pa cifre izdvajati
    * sve dok ima cifara izdvajanje, tj. broj x nije deljenjem sa 10
      postao 0
    */
  while(x) {

    zbir_cifara += x % 10;
    broj_cifara++;

    x /= 10;
  }

  /* Da nije izvrsena implicitna konverzija (kastovanje),
   * operator / obavljao bi celobrojno deljenje.
   * Zato je potrebno da bar jedan operand bude ceo broj,
   * sto je u ovom slucaju deljenik.
   * Operator kastovanja je unaran, pa ima veci prioritet od /
   */
  return (float) zbir_cifara / broj_cifara;
}

int main() {

  int x;

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

  printf("Arimeticka sredina cifara broja %d je %.3f\n", x,
    aritmeticka_sredina(x));

  /* Iako smo u funkciji aritmeticka_sredina menjali broj x,
   * prosledjivanjem argumenata funkciji pravi se lokalna kopija
     promenljive x za tu funkciju,
   * tako da je ona menjana unutar funkcije aritmeticka_sredina,
   * a promenljiva x iz funkcije main() ostaje netaknuta.
   */

  return 0;
}
```

Rešenje 0.1.34

```
#include <stdio.h>

/* 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
 * isti broj puta.
 * Ideja je sledeca:
```

```
 *  iz broja x izdvajaju se redom cifra po cifra s kraja,
 *  a zatim se svaka takva cifra trazi i u broju y.
 *  Ukoliko postoji, eliminise se prvi put kada se pojavi (dakle,
    samo jednom).
 *  Ukoliko su sve cifre iste (***redosled nije bitan***),
 *  na kraju ce i iz x i iz y biti sve cifre eliminisane",
 *  te ostaju nule u oba broja.
 *
 *  Broj novo_y formira se, zbog jednostavnosti, pomocu Heronovog
    obrasca.
 *  Ovaj postupak obradjen je u okviru funkcije int izbaci_cifru(int
    y, int cifra).
 */

int izbaci_cifru(int y, int cifra) {

  int novo_y = 0;
  int indikator = 0;
  int izdvojena_cifra;

  while(y) {

    izdvojena_cifra = y % 10;
    /* U slucaju da se cifra razlikuje od one koju treba eliminisati,
     * ili ukoliko je jedna cifra vec elimisana =>
     * tekucu cifru ukljuciti prilikom formiranja novog y
     * */
    if(izdvojena_cifra != cifra || indikator)

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

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

    y /= 10;
  }

  return novo_y;
}

int zapis(int x, int y) {

  /* Cifra koja se izdvaja iz x, a onda eliminise iz y */
```

```
  int cifra;

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

  while(x) {

    cifra = x % 10;
    x /= 10;

    y = izbaci_cifru(y, cifra);

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

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

int main() {

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

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

  return 0;
}
```

Rešenje 0.1.35

```
#include <stdio.h>

/* Funkcija racuna faktorijel broja.
 * Faktorijel formiramo mnozenjem sa trenutnom vrednoscu broja x,
 * a zatim smanjujuci tu vrednost za 1.
 * Ukoliko je x = 5, f = 5 * 4 * 3 * 2 * 1
 */
int faktorijel(int x) {

  int f = 1;
  while(x) {
    f *= x;
    x--;
```

```
  }
  return f;
}

int main() {

  int x, y;

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

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

Rešenje 0.1.36

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

/* Funkcija proverava da li se
 * cifre u zapisu broja nalaze u rastucem poretku.
 *
 * Situacija od interesa je kada za dve uzastopne cifre to nije
    slucaj.
 * Tada ne treba proveravati i za ostale cifre,
 * vec odmah prekinuti izvrsavanje funkcije.
 *
 * Ukoliko funkcija nije ranije prekinuta,
 * to znaci da cifre jesu u rastucem poretku
 * (odnosno, kako izdvajamo cifre od nazad, u stvari proveravamo
    opadajuci poredak),
 * te treba vratiti 1.
```

```c
 */

int rastuce(int n) {

  int tekuca_cifra;
  int prethodna_cifra;

  n = abs(n);

  /* Prvu cifru (odnosno, poslednju u zapisu broja)
   * izdvajamo izvan petlje
   * kako bismo mogli da je poredimo sa narednom
   */
  tekuca_cifra = n % 10;
  n /= 10;

  while(n) {

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

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

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

  return 1;
}

int main() {

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

  if(rastuce(x))
    printf("Cifre su u rastucem poretku!\n");
  else
    printf("Cifre nisu u rastucem poretku!\n");

  return 0;
```

```
}
```

Rešenje 0.1.37

```
#include <stdio.h>

/* Funkcija racuna broj x na n-ti stepen */
int stepen(int x, int n) {

  int i;
  /* Promenljiva u kojoj se cuva proizvod broja x sa samim sobom, n
     puta */
  int st = 1;

  for(i = 1; i <= n; i++)
    st *= x;

  return st;
}

/* Funkcija proverava da li je broj Armstrongov. */
int armstrong(int x) {

  /* u y se cuva zbir i-tih stepena cifara */
  int y;
  /* stepen za koji se proverava */
  int i = 1;
  /* prilikom izdvajanja cifara, broj x se menja,
   * te treba imati promenlju koja cuva pravu vrednost x
   */
  int original = x;

  do {

    y = 0;
    /* Racunamo i-te stepene za svaku cifru,
     * i istovremeno te stepen sabiramo.
     * Rezultat pamtimo u promenljivoj y.
     */
    while(x) {

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

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

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

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

int main() {

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

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

  return 0;
}
```

Rešenje 0.1.38

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

/* Funkcija proverava da li su dve uzastopne cifre
 * razlicite parnosti.
 *
 * Interesantna situacija je ukoliko su dve uzastopne cifre
 * obe parne, odnosno obe neparne.
 * Ovaj uslov svodimo na poredjenje njihovih ostataka pri deljenju sa
     2:
 * ukoliko su ostaci isti, cifre su iste parnosti,
 * te ne treba dalje proverati da li je uslov zadovoljen,
 * vec odmah prekinuti sa izvrsavanjem funkcije.
 *
 * Ukoliko dve uzastopne cifre ni u jednom slucaju nisu bile iste
    parnosti,
 * a izdvojene su sve cifre iz broja x,
 * uslov je ispunjen, pa funkcija vraca 1.
 */
int par_nepar(int x) {

  int prethodna_cifra;
  int tekuca_cifra;

  /* u slucaju da je uneti broj negativan */
```

```
  x = abs(x);

  /* jednu cifru izdvajamo van petlje
   * kako bismo mogli da je odmah u petlji poredimo sa narednom
   */
  prethodna_cifra = x % 10;
  x /= 10;

  while(x) {

    tekuca_cifra = x % 10;

    if(tekuca_cifra % 2 == prethodna_cifra % 2)
      return 0;

    /* tekuca cifra postaje prethodna cifra za narednu iteraciju */
    prethodna_cifra = tekuca_cifra;
    x /= 10;
  }

  return 1;
}

int main() {

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

  if(par_nepar(x))
    printf("Broj ispunjava uslov!\n");
  else
    printf("Broj ne ispunjava uslov!\n");

  return 0;
}
```

Rešenje 0.1.39

```
#include <stdio.h>

/* Funkcija broji koliko puta se realan broj x
 * javlja u nizu unetih brojeva sa tastature.
 *
 * Brojevi se unose sve do pojave 0,
 * pa treba koristiti do..while petlju,
 * kako bi bar jedan broj bio unet (makar bio i 0).
 */
int prebrojavanje(float x) {

  /* y prihvata uneti broj sa tastature */
```

```
  float y;
  /* br_pojavljivanja je brojac koji broji koliko puta se broj x
     javlja u unetom nizu brojeva */
  int br_pojavljivanja = 0;

  printf("Unesite brojeve: ");
  do {

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

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

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

  return br_pojavljivanja;
}

int main() {

  float x;
  int br_pojavljivanja;

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

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

  return 0;
}
```

Rešenje 0.1.40

```
#include <stdio.h>

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

```

```
  int i;

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

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

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

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

  return fn2;
}

int main() {

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

  /* Provera vrednosti za broj n */
  if(n <0 || n > 50) {
    printf("Greska: nedozvoljena vrednost!\n");
  }
  else{
    printf("%ld\n", fibonaci(n));
  }

  return 0;
}
```

Rešenje 0.1.41

```c
#include <stdio.h>

/* Funkcija vraca karakter koji se u abecedi
 * nalazi k mesta pre datog karaktera c
 */
char sifra(char c, int k) {

  /* Ukoliko je uneto malo slovo ... */
  if(c >= 'a' && c <= 'z')
    /* Pri tome karakter koji je k pozicija pre datog karaktera
    ispada iz opsega malih slova ... */
    if(c-k < 'a')
      /* Treba krenuti s drugog kraja abecede, racunajuci i
    preskocena slova.
       *
       * Na primer, ukoliko je c = 'b' i k = 2
       * Jedan karakter pre 'b' je 'a'.
       * Dva karaktera pre 'b' je 'z' (kruzno).
       *
       * Karakter iz prvog dela abecede, koji je preskocen, je 'a'.
       * Broj preskocenih karaktera iz prvog dela abecede
       * racunamo tako sto izracunamo c - 'a' (rastojanje od datog
    karaktera do malog slova a)
       * sto je u ovom slucaju 'b' - 'a' = 1.
       *
       * Ostatak karaktera do k ispisujemo, ali gledavsi unazad od z.
       * Zato racunamo k - (c - 'a') - 1.
       *
       * Od k oduzimamo rastojanje izmedju c i 'a',
       * kako bismo dobili preostali broj karaktera koji treba
    preskociti.
       */
      return 'z' - (k - (c -  'a') - 1);
    else
      /* U suprotnom, karakter ne ispada iz opsega malih slova, te je
     dovoljno bas njega i vratiti */
      return c-k;

  /* Ukoliko je uneto veliko slovo ... */
  else if(c >= 'A' && c <= 'Z')
    if(c-k < 'A')
      return 'Z' - (k - (c -  'A') - 1);
    else
      return c-k;


  return c;
}

int main() {
```

```
47    int k;
      char c;
49
      printf("Unesite broj k: ");
51    scanf("%d", &k);

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

57    return 0;
  }
```