Programiranje I

Beleške sa vežbi

Smer *Informatika*

Matematički fakultet, Beograd

Sana Stojanović

December 19, 2007

Sadržaj

1 For petlja, primeri **3**

2 *while* i *do-while* petlja **4**

3 *Define* direktiva, primeri **6**

4 Izdvajanje cifara celog broja **7**

5 Naredbe *break* i *continue* **9**

6 Prosti brojevi **12**

7 Umetanje jedne petlje u drugu **15**

1 For petlja, primeri

1. Napisati program koji sa standardnog ulaza učitava ceo broj n i izračunava prvi stepen dvojke koji je veći (ili jednak) od njega. Na primer, za $n = 7$ program treba da nam vrati 3 (jer je $2^3 > 7$).

```
#include <stdio.h>

main()
{
    int n;
    int m;       /* Broj kojim stepenujemo dvojku */
    int stepen;  /* Promenljiva u kojoj cuvamo vrednost 2^m */

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

    /* Inicijalizacija pomocnih promenljivih
       u svakom trenutku vazice: stepen = 2^m */
    m = 0;

    /* U petlji povecavamo m za 1 i stepen mnozimo sa 2
       da bi i dalje vazilo: stepen = 2^m */
    for(stepen = 1; stepen<n; stepen *= 2)
        m++;

    printf("Prvi stepen 2 koji je veci od %d je %d\n", n, m);
}
```

2. Prethodna *for* petlja može da se zapise i sa višestrukom inicijalizacijom i višestrukim naredbama dodele na sledeći način:

```
    for(m=0, stepen=1; stepen<n; stepen *= 2, m++)
        ;
```

ili ako izdvojimo obe inicijalizacije van for petlje i obe naredbe dodele ubacimo u telo for petlje možemo dobiti sledeći kod:

```
    m = 0;
    stepen = 1;
    for(; stepen<n; )
    {
        m++;
        stepen *=2;
    }
```

2 *while* i *do-while* petlja

1. ```
   /* Program ilustruje while petlju */

   #include <stdio.h>

   int main()
   {
       int x;

       x = 1;
       while (x<=10)
       {
           printf("x = %d\n",x);
           x++;
       }

   }

   Izlaz:
   x = 1
   x = 2
   x = 3
   x = 4
   x = 5
   x = 6
   x = 7
   x = 8
   x = 9
   x = 10
   ```

2. ```
   /* Program ilustruje do-while petlju */

   #include <stdio.h>

   int main()
   {
       int x;

       x = 1;
       do
       {
           printf("x = %d\n",x);
           x++;
       }
   ```

```
    while (x<=10);

}

Izlaz:
x = 1
x = 2
x = 3
x = 4
x = 5
x = 6
x = 7
x = 8
x = 9
x = 10
```

3. Napisati program koji sa standardnog ulaza učitava ceo broj n i izračunava prvi stepen dvojke koji je veći od njega korišćenjem *while* petlje.

```
#include <stdio.h>

main()
{
    int n;
    int m;       /* Broj kojim stepenujemo dvojku */
    int stepen;  /* Promenljiva u kojoj cuvamo vrednost 2^m */

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

    /* Inicijalizacija pomocnih promenljivih
       u svakom trenutku vazice: stepen = 2^m */
    m = 0;
    stepen = 1;

    /* U petlji povecavamo m za 1 i stepen mnozimo sa 2
       da bi i dalje vazilo: stepen = 2^m */
    while(stepen<n)
    {
        m++;
        stepen *= 2;
    }

    printf("Prvi stepen 2 koji je veci od %d je %d\n", n, m);
}
```

3 *Define* direktiva, primeri

1.
```
/* Konverzija centimetara u ince - while petlja */

#include <stdio.h>

/* Definicija simbolickih konstanti preko #define direktiva */
/* U fazi pretprocesiranja se vrsi doslovna zamena konstanti
   njihovim vrednostima */

#define POCETAK 0
#define KRAJ 20
#define KORAK 10

main()
{
    int a;
    a = POCETAK;
    while (a <= KRAJ)
    {
        printf("%d cm = %f in\n", a, a/2.54);
        a += KORAK; /* isto sto i a = a + KORAK; */
    }
}

Izlaz:
0 cm = 0.000000 in
10 cm = 3.937008 in
20 cm = 7.874016 in
```

2.
```
/* Konverzija centimetara u ince - for petlja */

#include <stdio.h>
#define POCETAK 0
#define KRAJ 20
#define KORAK 10

main()
{
    int a;
    for  (a = POCETAK; a <= KRAJ; a += KORAK)
        printf("%d cm = %f in\n", a, a/2.54);

}
```

```
Izlaz:
0 cm = 0.000000 in
10 cm = 3.937008 in
20 cm = 7.874016 in
```

4 Izdvajanje cifara celog broja

1. Napisati program koji za uneti ceo broj n sa standardnog ulaza izdvaja i ispisuje njegove cifre (počev od cifre najmanje težine) na standardni izlaz.

   Program treba da izgleda ovako:

   ```
   Unesite ceo broj: 123
   Cifre broja 123 su:
   3
   2
   1
   ```

   ```
   #inlcude <stdio.h>

   main()
   {
       int n;      //Ceo broj cije cifre izdvajamo
       int c;      //Pomocna promenljiva u kojoj cuvamo tekucu cifru

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

       printf("Cifre broja %d su: \n", n);
       /* U petlji izdvajamo jednu po jednu cifru broja */
       while(n > 0)
       {
           /* Izdvajamo poslednju cifru broja n (dobijamo je kao
              ostatak pri deljenju sa 10 broja n) */
           c = n % 10;

           /* Ispisujemo cifru na standardni izlaz */
           printf("%d\n", c);

           /* Modifikujemo vrednost broja n da bismo u sledecoj iteraciji
              izdvojili sledecu cifru */
           n = n / 10;
       }
   }
   ```

2. Napisati program za uneti ceo broj sa standardnog ulaza štampa zbir njegovih cifara.

Program bi trebalo da izgleda ovako:

```
Unesite ceo broj: 123
Zbir cifara broja 123 je: 6
```

```
#inlcude <stdio.h>

main()
{
    int n;        //Ceo broj cije cifre izdvajamo
    int c, zbir;  //Pomocne promenljive u kojima cuvamo tekucu cifru i sumu cifara

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

    /* Inicijalizujemo sumu cifara na nulu */
    zbir = 0;

    /* U petlji izdvajamo jednu po jednu cifru broja */
    while(n > 0)
    {
        /* Izdvajamo poslednju cifru broja n (dobijamo je kao
           ostatak pri deljenju sa 10 broja n) */
        c = n % 10;

        /* Dodajemo tekucu cifru na zbir */
        zbir = zbir + c;

        /* Modifikujemo vrednost broja n da bismo u sledecoj iteraciji
           izdvojili sledecu cifru */
        n = n / 10;
    }

    printf("Zbir cifara broja %d je: %d\n", zbir);
}
```

3.
```
/* Program koji menja mesta ciframa u broju. */
#include <stdio.h>

/* Funkcija koja obrce cifre broja */
int reverse(int n)
{
  int m = 0;   //pomocna promenljiva

  /* Dokle god imamo jos cifara u broju n, dodajemo ih u broj m */
```

```
  while (n > 0)
  {
    /* Izdvajamo sledecu cifru i dodajemo je u m */
    m = m * 10 + n % 10;

    /* Menjamo vrednost broja n da bismo u sledecoj iteraciji izdvojili
       sledecu cifru */
    n = n / 10;
  }

  return m;
}

main()
{
  int n;

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

  printf("Obrnut broj je %d\n", reverse(n));
}

Izlaz:
Unesite broj
1234
Obrnut broj je 4321
```

5 Naredbe *break* i *continue*

1. Napisati program koji sa standardnog ulaza unosi i sabira cele brojeve dok se ne unese 0 pomoću *while* petlje.

```
#include <stdio.h>

main()
{
    int x, zbir;

    printf("Unesite niz cifara pri cemu je 0 oznaka za kraj\n");

    /* Inicijalizujemo zbir na 0 */
    zbir = 0;

    /* Citamo prvi element van petlje da bismo mogli da imamo uslov
```

```
       while (x != 0) */
    scanf("%d", &x);

    /* Dokle god ne procitamo nulu dodajemo brojeve u sumu */
    while(x != 0)
    {
        zbir = zbir + x;

        /* citamo sledeci broj */
        scanf("%d", &x);
    }

    printf("Suma brojeva je %d\n", zbir);
}
```

2. Napisati program koji sa standardnog ulaza unosi i sabira cele brojeve dok se ne unese 0 pomoću *do-while* petlje.

```
#include <stdio.h>

main()
{
    int x, zbir;

    printf("Unesite niz cifara pri cemu je 0 oznaka za kraj\n");

    /* Inicijalizujemo zbir na 0 */
    zbir = 0;

    /* U petlji ucitavamo broj, dodajemo ga na sumu i izlazimo iz
       petlje kad ucitamo nulu */
    do
    {
        /* citamo sledeci broj */
        scanf("%d", &x);
        zbir = zbir + x;
    }
    while(x != 0);

    printf("Suma brojeva je %d\n", zbir);
}
```

3. Napisati program koji sa standardnog ulaza unosi i sabira cele brojeve dok se ne unese 0 pomoću naredbe *break*.

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

```
main()
{
    int x, zbir;

    printf("Unesite niz cifara pri cemu je 0 oznaka za kraj\n");

    /* Inicijalizujemo zbir na 0 */
    zbir = 0;

    /* Beskonacna while petlja */
    while(1)
    {
        /* Citamo sledeci element... */
        scanf("%d", &x);

        /* ako smo procitali 0 znaci da smo stigli do kraja... */
        if (x == 0)
            /* i izlazimo iz petlje */
            break;
        /* u suprotnom dodajemo procitani broj u zbir */
        else
            zbir = zbir + x;
    }

    printf("Suma brojeva je %d\n", zbir);
}
```

4. Napisati program koji sabira samo pozitivne brojeve niza cifara koji se završava nulom.

```
#include<stdio.h>

main()
{
 int x, zbir;

 printf("Unesite niz cifara pri cemu je 0 oznaka za kraj\n");

 /* Inicijalizujemo zbir na 0 */
 zbir = 0;

 /* Beskonacna while petlja. */
 while( 1 ) {
    /* Citamo sledeci element... */
    scanf("%d", &x );
```

```
    /* ako smo procitali 0 znaci da smo stigli do kraja... */
    if( x == 0 )
        /* i izlazimo iz petlje */
        break;
    /* Ako je broj negativan preskacemo ga... */
    else if( x < 0 )
        /* i idemo na sledeci */
        continue;
    /* inace, broj je pozitivan i dodajemo ga u zbir. */
    else zbir = zbir + x;
 }

 printf("Suma pozitivnih je %d\n", zbir);
}
```

6 Prosti brojevi

1. Funkcija koja proverava da li je broj prost.

```
/* funkcija vraca 1 ako je dati broj prost, odnosno 0 ako nije prost */
int prost(int n)
{
    int i;    /* Delioc */

    if (n <= 3)
        /* ako je u pitanju 1, 2 ili 3
           vracamo 1 kao indikator da je broj prost */
        return 1;
    else
        /* ako je broj veci od 3 proveravamo da li ima delioce */
        for(i=2; i<n; i++)
            /* ako smo nasli delioca za nas broj to znaci da
               broj nije prost i vracamo 0 */
            if (n%i == 0)
                return 0;
    /* ako smo dosli dovde to znaci da nas broj nema delioca
       tako da vracamo 1 */
    return 1;
}
```

2. Program koji proverava da li je zadati broj sa ulaza prost pozivom prethodne funkcije.

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

```
int prost(int n);

main()
{
    int n;

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

    /* poziv funkcije prost(n) */
    if (prost(n))
        printf("Broj %d je prost.\n", n);
    else
        printf("Broj %d nije prost. \n", n);
}

int prost(int n)
{
    int i;    /* Delioc */

    if (n <= 3)
        /* ako je u pitanju 1, 2 ili 3
           vracamo 1 kao indikator da je broj prost */
        return 1;
    else
        /* ako je broj veci od 3 proveravamo da li ima delioce */
        for(i=2; i<n; i++)
            /* ako smo nasli delioca za nas broj to znaci da
               broj nije prost i vracamo 0 */
            if (n%i == 0)
                return 0;
    /* ako smo dosli dovde to znaci da nas broj nema delioca
       tako da vracamo 1 */
    return 1;
}
```

3. II način - optimizovana funkcija za proveru da li je broj prost

```
int prost1(int n)
{
    int i;

    if (n <= 3)
        return 1;
    else if (n % 2 == 0)
        /* ako je broj paran i veci od 2 onda nije prost */
```

```
        return 0;
        /* inace je u pitanju neparan broj, pa delioci mogu biti
           samo neparni brojevi */
     else for(i=3; i*i <= n; i+=2)
              if (n%i == 0)
                   return 0;
     return 1;
}
```

4. Program koji ispisuje prvih n prostih brojeva

```
#include<stdio.h>

int prost1(int n)
{
    int i;

    if (n <= 3)
        return 1;
    else if (n % 2 == 0)
        /* ako je broj paran i veci od 2 onda nije prost */
        return 0;
        /* inace je u pitanju neparan broj, pa delioci mogu biti
           samo neparni brojevi */
     else for(i=3; i*i <= n; i+=2)
              if (n%i == 0)
                   return 0;
     return 1;
}

main()
{
 int i;     /* Brojac u kome cuvamo podatak o broju do sada pronadjenih
               prostih brojeva */
 int n;
 int p;     /* Kandidat za proveru da li je prost */

 printf("Unesite koliko prostih brojeva zelite da dobijete: \n");
 scanf("%d", &n);

 /* Inicijalizujemo brojac i - koliko smo prostih brojeva nasli do sad */
 i = 0;

 /* Pocetni broj za koji proveravamo da li je prost */
 p = 1;
```

```
 /* Trazimo i-ti prost broj. Sve dok ne nadjemo n prostih brojeva... */
 while(i < n)
 {
    /* proveravamo da li je tekuci kandidat prost */
    if (prost1(p))
        {
            /* i ako jeste stampamo ga... */
            printf("%d ", n);

            /* i uvecavamo broj pronadjenih prostih brojeva. */
            i++;
        }
     /* U svakom slucaju prelazimo na proveru da li je sledeci broj prost */
     p++;
 }
}
```

7 Umetanje jedne petlje u drugu

1. Napisati program za sabiranje dva broja na osnovu algoritma koji je korišćen u *URM* mašinama.

```
#include <stdio.h>

main()
{
    int x, y, zbir;   //brojevi koje sabiramo i zbir
    int i;         //brojaci u petljama

    printf("Unesite brojeve x i y: ");
    scanf("%d%d", &x, &y);

    /* kako zbir brojeva racunamo po formuli
       x + y = x + (1 + 1 + ... + 1), gde se 1 pojavljuje y puta
       pocetnu vrednost zbira postavljamo na x... */
    zbir = x;

    /* a nakon toga u petlji dodajemo y jedinica na zbir */
    for(i=0; i<y; i++)
        zbir++;

    printf("Zbir brojeva %d i %d je %d\n", x, y, zbir);
}
```

2. Napisati program za množenje dva broja na osnovu algoritma koji je korišćen u *URM* mašinama.

```
#include <stdio.h>

main()
{
    int x, y, pr;     //brojevi koje mnozimo i proizvod
    int i, j;         //brojaci u petljama

    printf("Unesite brojeve x i y: ");
    scanf("%d%d", &x, &y);

    /* kako proizvod brojeva racunamo po formuli
       x * y = x + ((1+1+...+1) + ... + (1+1+...+1)),
       gde se 1 u zagradi pojavljuje x puta a mala zagrada (sa jedinicama)
       se pojavljuje y-1 puta (da bi x bilo dodato y puta u proizvod),
       pocetnu vrednost proizvoda postavljamo na x... */
    zbir = x;

    /* a nakon toga u petlji dodajemo y-1 puta */
    for(i=1; i<y; i++)
        /* po x jedinica */
        for(j=0; j<x; j++)
            pr++;

    printf("Proizvod brojeva %d i %d je %d\n", x, y, pr);
}
```

3. Šta će biti izlaz iz sledećeg programa?

```
#include<stdio.h>
int main()
{
int i,j;

for(i=1; i<=3; i++)
    {
    for(j=1; j<=3; j++)
        printf("%d * %d = %d\t", i, j, i*j);
    printf("\n");
    }
}

Izlaz:
1 * 1 = 1       1 * 2 = 2       1 * 3 = 3
```

```
2 * 1 = 2       2 * 2 = 4       2 * 3 = 6
3 * 1 = 3       3 * 2 = 6       3 * 3 = 9
```

4.
```
#include <stdio.h>
void Zbir_stepena (int n, int granica);
main()
{
    Zbir_stepena(5,2);
    Zbir_stepena(5,3);
    Zbir_stepena(10,4);
    return 0;
}

void Zbir_stepena (int n, int granica)
{

    int i,j;   /*brojaci u for petljama */
    long Zbir=0  ,  stepenovan ;

     /*spoljasnji for ciklus obavlja sumiranja*/
    for (i=1; i<=n; Zbir +=stepenovan, ++i)
        /*unutrasnji for ciklus obavlja stepenovanje */
         for( stepenovan=1,j=1;  j<=granica;  stepenovan*= (long) i, ++j) ;
    printf(" Zbir %d. stepena od 1 do %d jeste %ld\n", granica,n,Zbir);
}

Izlaz:
 Zbir 2. stepena od 1 do 5 jeste 55
 Zbir 3. stepena od 1 do 5 jeste 225
 Zbir 4. stepena od 1 do 10 jeste 25333
```