Osnovi programiranja

Beleške sa vežbi

Smer *Računarstvo i informatika*
Matematički fakultet, Beograd

Jelena Tomašević

March 15, 2006

Sadržaj

1

[1]

1.1 Rekurzija

C funkcije se mogu rekurzivno koristiti, što znači da funkcija može pozvati samu sebe direktno ili indirektno.

Primer 1 *Štampanje celog broja.*

```
#include<stdio.h>
void printb(long int n)
{
   if(n<0)
   {
      putchar('-');
      n=-n;
   }
   if(n/10)
      printb(n/10);
   putchar(n % 10 + '0');
}

int main()
{
long int b=-1234;
printb(b);
putchar('\n');
return 0;
}
```

Kad funkcija rekurzivno pozove sebe, svakim pozivom pojavljuje se novi skup svih automatskih promenljivih, koji je nezavisan od prethodonog skupa. Prva funkcija printb kao argument dobija broj -12345, ona prenosi 1234 u drugu printb funkciju, koja dalje prenosi 123 u treću, i tako redom do poslednje koja prima 1 kao argument. Ta funkcija štampa 1 i završava sa radom tako da se vraća na prethodni nivo, na kome se štampa dva i tako redom.

Primer 2 *Računanje sume prvih n prirodnih brojeva.*

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

[1] Zasnovano na primerima sa sajta `http://www.matf.bg.ac.yu/~milena`

```
int suma(int n)
{
   if(n!=0)
      return( n + suma(n-1) );
   else return n;
}

main()
{
int S,n;
printf("Unesite n\n");
scanf("%d",&n);
S=suma(n);
printf("S=%d",S);
putchar('\n');
}
```

Primer 3 *Računanje faktorijela prirodnog broja.*

```
#include<stdio.h>
unsigned long fakt(int n)
{
   if(n!=0)
      return( n*fakt(n-1) );
   else return 1;
}

main()
{
int n;
unsigned long f;
printf("Unesite n\n");
scanf("%d",&n);
f=fakt(n);
printf("f=%d",f);
putchar('\n');
}
```

Primer 4 *Fibonačijevi brojevi.*

```
#include<stdio.h>
int fibr(int n)
{
   if((n==1)||(n==2))
      return 1;
   else return(fibr(n-1)+fibr(n-2));
}

int main()
{
int Fn,n;
printf("Unesite n\n");
scanf("%d",&n);
```

```
Fn=fibr(n);
printf("F[%d]=%d",n,Fn);
putchar('\n');
return 0;
}
```

Primer 5 *Iterativna i rekurzivna varijanta računanja sume niza.*

```
int suma_niza_iterativno(int a[], int n)
{
    int suma = 0;
    int i;
    for (i = 0; i<n;  i++)
        suma+=a[i];
    return suma;
}

int suma_niza(int a[], int n)
{
    if (n == 1)
        return a[0];
    else
        return suma_niza(a, n-1)+a[n-1];
}
```

Primer 6 *Stepenovanje prirodnog broja*

```
int stepenuj (int n, int k)
{
    if (k == 0)
        return 1;
    else
        return n*stepenuj(n, k-1);
}
```

1.2 Životni vek i oblast važenja promenjivih, statičke promenljive

Primer 7 *Demonstracija zivotnog veka i oblasti vazenja promenjivih (scope).*

```
#include <stdio.h>

/* Globalna promenjiva */
int a = 0;

/* Uvecava se globalna promenjiva a */
void increase()
{
    a++;
    printf("increase::a = %d\n", a);
}

/* Umanjuje se lokalna promenjiva a. Globalna promenjiva zadrzava svoju vrednost. */
void decrease()
```

```
{
    /* Ovo a je nezavisna promenjiva u odnosu na globalno a */
    int a = 0;
    a--;
    printf("decrease::a = %d\n", a);

}

void nonstatic_var()
{
    /* Nestaticke promenjive ne cuvaju vrednosti kroz pozive funkcije */
    int s=0;
    s++;
    printf("nonstatic::s=%d\n",s);
}

void static_var()
{
    /* Staticke promenjive cuvaju vrednosti kroz pozive funkcije.
       Inicijalizacija se odvija samo u okviru prvog poziva. */
    static int s=0;
    s++;
    printf("static::s=%d\n",s);
}

main()
{
    /* Promenjive lokalne za funkciju main */
    int i;
    int x = 3;

    printf("main::x = %d\n", x);


    for (i = 0; i<3; i++)
    {
        /* Promenjiva u okviru bloka je nezavisna od spoljne promenjive.
           Ovde se koristi promenjiva x lokalna za blok petlje koja ima
           vrednost 5, dok originalno x i dalje ima vrednost 3*/
        int x = 5;
        printf("for::x = %d\n", x);
    }

    /* U ovom bloku x ima vrednost 3 */
    printf("main::x = %d\n", x);

    increase();
    decrease();

    /* Globalna promenjiva a */
    printf("main::a = %d\n", a);
```

```
    /* Demonstracija nestatickih promenjivih */
    for (i = 0; i<3; i++)
        nonstatic_var();

    /* Demonstracija statickih promenjivih */
    for (i = 0; i<3; i++)
        static_var();
}

Izlaz iz programa:
main::x = 3
for::x = 5
for::x = 5
for::x = 5
main::x = 3
increase::a = 1
decrease::a = -1
main::a = 1
nonstatic::s=1
nonstatic::s=1
nonstatic::s=1
static::s=1
static::s=2
static::s=3
```

Primer 8 *Ilustracija statičkih promenljivih.*

```
#include <stdio.h>

void f()
{
    static int a;
    a++;
    printf("%d\n",a);
}

main()
{
int i;
   for (i=0; i<=10; i++)
          f();
}

/* Izlaz iz programa 1 2 3 4 5 6 7 8 9 10 11*/
```

Primer 9 *Ilustruje vidljivost imena*

```
#include <stdio.h>

int i=10;

void main() {
```

```
    {
        int i=3;
        {
            int i=1;
            printf("%d\n", i);
        }
        printf("%d\n",i);
    }
     printf("%d\n",i);
}
```

1.3 Pokazivači

Pokazivač je promenljiva koja sadrži adresu promenljive.

```
int x=1, y=1, z[10];
int *ip;    /* ip je pokazivac na int,
               odnosno *ip je tipa int*/

ip = &x;    /* ip sada pokazuje na x */
y=*ip;      /* y je sada 1 */
*ip = 0;    /* x je sada 0 */

*ip+=10;    /* x je sada 10*/
++*ip;      /* x je sada 11*/
(*ip)++;    /* x je sada 12,
            zagrada neophodna zbog prioriteta
            operatora*/

ip = &z[0]; /* ip sada pokazuje na z[0]*/
```

Primer 10 *Ilustracija rada sa pokazivačkim promenljivim.*

```
#include <stdio.h>
main() {
    int x = 3;

    /* Adresu promenjive x zapamticemo u novoj promeljivoj.
       Nova promenljiva je tipa pokazivaca na int (int*) */
    int* px;

    printf("Adresa promenljive x je : %p\n", &x);
    printf("Vrednost promenljive x je : %d\n", x);


    px = &x;
    printf("Vrednost promenljive px je (tj. px) : %p\n", px);
    printf("Vrednost promenljive na koju ukazuje px (tj. *px) je : %d\n", *px);

    /* Menjamo vrednost promenljive na koju ukazuje px */
    *px = 6;
    printf("Vrednost promenljive na koju ukazuje px (tj. *px) je : %d\n", *px);
```

```
    /* Posto px sadrzi adresu promenljive x, ona ukazuje na x tako da je
       posredno promenjena i vrednost promenljive x */
    printf("Vrednost promenljive x je : %d\n", x);


}

Izlaz (u konkretnom slucaju):
Adresa promenljive x je : 0012FF88
Vrednost promenljive x je : 3
Vrednost promenljive px je (tj. px) : 0012FF88
Vrednost promenljive na koju ukazuje px (tj. *px) je : 3
Vrednost promenljive na koju ukazuje px (tj. *px) je : 6
Vrednost promenljive x je : 6
```

Pored pokazivača na osnovne tipove, postoji i pokazivač na prazan tip (void).

```
void *pp;
```

Njemu može da se dodeli da pokazuje na int, ili na char ili na proizvoljan tip ali je to neophodno eksplicitno naglasiti svaki put kada želimo da koristimo ono na šta on pokazuje.

Primer 11 *Upotreba pokazivača na prazan tip.*

```
#include<stdio.h>

main()
{
void *pp;
int x=2;
char c='a';

pp = &x;
*(int *)pp = 17;    /* x postaje 17*/
printf("\n adresa od x je %p", &x);
printf("\n%d i %p",*(int*)pp,(int * )pp);

pp = &c;
printf("\n adresa od c je %p", &c);
printf("\n%c i %p",*(char*)pp,(char * )pp);

}

/*
 adresa od x je 0012FF78
17 i 0012FF78
 adresa od c je 0012FF74
a i 0012FF74

*/
```

Posebna konstanta koja se koristi da se označi da pokazivač ne pokazuje na neko mesto u memoriji je NULL.

1.4 Pokazivači i argumenti funkcija

C prosleđuje argumente u funkcije pomoću vrednosti. To znači da sledeća funkcija neće uraditi ono što želimo:

```
void swap (int x, int y) /* POGRESNO!!!!!!!!*/
{
int temp;
temp = x;
x=y;
y=temp;
}
```

Zbog prenosa parametara preko vrednosti swap ne može da utiče na argumente a i b u funkciji koja je pozvala swap. Ova swap funkcija samo zamenjuje kopije od a i b.

Da bi se dobio željeni efekat, potrebno je da se proslede pokazivači:

```
/* Zameni *px i *py */
void swap (int *px, int *py)
{
int temp;
temp =*px;
*px = *py;
*py = temp;
}
```

a poziv funkcije swap izlgeda sada ovako

```
swap(&a, &b);
```

Primer 12 *Demonstracija više povratnih vrednosti funkcije koristeći prenos preko pokazivača.*

```
/* Funkcija istovremeno vraca dve vrednosti - kolicnik i ostatak
dva data broja.
   Ovo se postize tako sto se funkciji predaju vrednosti dva broja (x i y) koji se dele
   i adrese dve promenljive na koje ce se smestiti rezultati */
void div_and_mod(int x, int y, int* div, int* mod) {
    printf("Kolicnik postavljam na adresu : %p\n", div);
    printf("Ostatak postavljam na adresu : %p\n", mod);
    *div = x / y;
    *mod = x % y;
}

main() {
    int div, mod;
    printf("Adresa promenljive div je %p\n", &div);
    printf("Adresa promenljive mod je %p\n", &mod);

    /* Pozivamo funkciju tako sto joj saljemo vrednosti dva broja (5 i 2)
       i adrese promenljvih div i mod na koje ce se postaviti rezultati */
    div_and_mod(5, 2, &div, &mod);

    printf("Vrednost promenljive div je %d\n", div);
    printf("Vrednost promenljive mod je %d\n", mod);
```

```
}

Izlaz u konkretnom slucaju:
Adresa promenljive div je 0012FF88
Adresa promenljive mod je 0012FF84
Kolicnik postavljam na adresu : 0012FF88
Ostatak postavljam na adresu : 0012FF84
Vrednost promenljive div je 2
Vrednost promenljive mod je 1
```

1.5 Zadaci za vežbu

Zadatak 1 *Napisati program koji (a) iterativno (b) rekurzivno računa n-ti Fibonačijev broj, pri čemu se broj n zadaje sa standardnog ulaza. Uporediti brzine izvršavanja ova dva programa za n=5, n=55 i n=95.*

Zadatak 2 *Napisati program u kome se korišćenjem rekurzivne funkcije izračunava NZD brojeva x i y.*

$$nzd(x,y) = \begin{cases} y, & x = 0 \\ nzd(y\%x, x), & x! = 0 \end{cases}$$

Zadatak 3 *Broj je Armstrongov ako je jednak sumi n-tih stepena svojih cifara. Ispitati da li je broj koji se unosi sa standardnog ulaza Armstrongov.*

Zadatak 4 *Napisati program u C-u koji prikazuje sve proste brojeve u datom intervalu kojima je zbir cifara složen broj. Interval se zadaje učitavanjem gornje i donje granice (dva prirodna broja). Brojeve prikazati u opadajućem poretku.*

Zadatak 5 *(a) Napisati funkciju int palindrom(int broj) koja proverava da li je broj palindrom i vraća vrednost 1 ako jeste, 0 ako nije. Na primer, brojevi 1, 44, 121, 112211, 12321 i 5665 jesu palindromi, a brojevi 123, 67, 8908 nisu.*
(b) Napisati program koji proverava da li je uneti broj palindrom.

Zadatak 6 *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 sa jedinicom. Napisati program koji za uneti broj određuje da li je srećan.*

Zadatak 7 *Sa ulaza se unosi broj u osnovi deset i osnova <= 10. Odštampati vrednost datog broja u datoj osnovi.*

Zadatak 8 *Sa ulaza se unosi osnova <= 10 i broj. Proveriti da li je taj broj ispravan broj za datu osnovu i ako jeste izračunati njegovu vrednost u osnovi 10.*

Zadatak 9 *Broj je* **Nivenov** *ako je deljiv sumom svojih cifara.*

1. *Napsati funkciju koja računa sumu cifara broja* **a**. *Na primer, za broj* `121` *funkcija treba da vrati* `4`.
2. *Napisati funkciju koja proverava da li je broj* Nivenov *i vraća* `1` *ako jeste a* `0` *ako nije.*
3. *Napisati program koji za uneto* `n` *ispisuje prvih* `n` Nivenovih *brojeva.*
4. *Napisati program koji za uneto* `n` *ispisuje sve* Nivenove *brojeve manje od* `n`.

Zadatak 10 *Napisati program koji izračunava vrednost polinoma u tački x:*

1. *Napisati funkciju koja računa k-ti stepen prirodnog broja n.*
2. *Napisati program koji za uneti niz koeficijenata $a[i]$ i uneti broj x računa vrednost polinoma* $a_n * x^n + a_{n-1} * x^{n-1} + ... + a_1 * x + a_0$