Vaje 4

Barbara Nežmah

3.11.1997

Mehurčno urejanje / sortiranje števil

Ta primer bomo pokazali kot simbolični program in ne v C++.

$$\begin{array}{r} a_n \\ a_{n-1} \\ a_{n-2} \\ \vdots \\ a_2 \\ a_1 \end{array}$$

Oglejmo si to na primeru: a_n ima vrednost 1,a_{n-1} ima vrednost 3, a_{n-2} ima vrednost 5, ... a_2 ima vrednost 7 in a_1 ima vrednost 2. Ker je a_1 lažji od a_2 se morata zamenjati. Mehurčke moramo urediti po velikosti od najlažjega do najtežjega. Lažji mehurčki potujejo navzgor, težji pa ostanejo tam kjer so. Te mehurčke uredimo z n-koraki. Urejamo toliko časa, dokler kakšen mehurček še obstaja.

Prvič naredimo n primerjanj, v naslednjem koraku še $(n-1)$ ... Urejamo jih lahko po naraščajočem oz. padajočem vrstnem redu.

Koliko časa urejamo?

n nam pove koliko podatkov imamo za urediti.

1. če je $n > 1$ in če obstaja še kakšen mehurček, urejanje nadaljujemo.

2. predpostavimo, da mehurčkov za $0 < i < n$ ni (elementi so urejeni). Če je $a_i < a_{i+1}$ a_i je lažji in mehurček splava navzgor – obstaja vsaj še en mehurček, ter zato nadaljujemo z urejanjem.

Z vsako uravnavo se n krči. Če pa ne najdemo nobenega mehurčka, je že urejeno.

```
oz.  če so mehurčki že urejeni, ostane izjava mehurčkov ni.
                                sicer
                           zmanjšaj n za 1
                     in sledi konec ponavljanja
```

Deklaracijski stavek

```
int a[10000]
printf("Napiši prvo število:\n")
...
scanf("%d", &a[i]);
```

Zapis:

```
scanf("%d", &a[i]
```

postavi število na i-to mesto.

In kako naj sedaj iz niza znakov naredimo številko?

```
int l;              /* l pomeni število znakov*/
long st;            /* število, ki ga bomo dobili iz niza c */
char c[10];         /* v c-ju imamo na primer shranjene znake 17235*/
scanf("%s", c);
if((c[i] <= '9') && (c[i] >= '0'))
  printf("je število.");
else
  printf("ni število.");
l= strlen(c);       /* dolžina*/
st= 0;              /* na začetku je st 0 */
for(i=0; i<l; i++)
  st= 10*st + (c[i]-'0');
```

Če je v c-ju shranjeno na primer število 17235 kot niz. Kako dobimo iz niza število?

- Lahko ga dobimo tako, da bi začeli od zadaj: (5 + 3*10 + 2*100 + 7*1000 + 10000)
- ali pa prvo število vedno množimo z 10: (1*10+7= 17, 17*10+2= 172, 172*10+3= 1723 in 1723*10+5= 17235)

Ker je druga verzija boljša to zapišemo kot:

```
st= 10*st + (c[i]-'0')
```

Funkcijski podprogrami

Izračun faktoriele

Izračunali bomo z rekurzivno formulo $n! = n \cdot (n-1)!$.

```
#include<stdio.h>
long Fakt(int a);     /*prototip*/
void main(void) {
  int n;
  printf("Za katero naravno število naj računa faktorelo:\n");
  scanf("%d", &n);
  printf("%d! =%ld \n", n, Fakt(n));
} /* main */

long Fakt(int a) {    /* resnična funkcija */
  long c;             /* rezultat funkcije */
  if (n < 2)          /* samo za 0!=1 in 1!=1*/
    c= 1;
  else
    c= n * Fakt(n-1); /* n! = n * ((n-1)!) */
    return c;         /* vrne kar je funkcija izračunala */
} /* Fakt */
```

Funkcijo `Fakt` lahko zapišemo krajše. recimo:

```
long Fakt(int a) {    /* resnična funkcija */

  return ((n < 2)? 1: n*FAKT(n-1));

} /* Fakt */
```

Fibbonacijevo zaporedje

Tudi člen Fibbonacijevega zaporedja izračunamo s pomočjo rekurzije. Obrazec je $f_n = f_{n-1} + f_{n-2}$.

```
#include<stdio.h>
long Fib(int b);     /*prototip*/

void main(void) {
  int n;
  printf("Katero Fibbonacijevo število naj računa:\n");
```

```
  scanf("%d", &n);
  printf("%d-ti člen je %ld \n", n, Fib(n));
} /* main /*

long Fib(int n) {
  return (n<=2? 1: Fib(n-1) + Fib(n-2)
} /* Fib */
```

členi so: 1, 1, 2, 3, 5, 8, ... $f_{n+1} = f_n + f_{n-1}$. Če vzamemo za $n = 35$ računa zelo dolgo, ker je treba najprej izračunati za $n = 34$ in $n = 33$. Pri $n = 34$ je potrebno izračunani za $n = 33$ in $n = 32$. Potem pa še na drugi strani za $n = 33$ je potrebno izračunati za $n = 32$ in $n = 31$. To traja tako dolgo dokler ne računamo pri $n = 3$, $n = 2$ in $n = 1$.

Kako izračunati n-ti člen FIB zaporedja hitreje?

Fibbonacijevo zaporedje – izboljšan način

Tokrat bomo računali na nerekurziven način in sicer od prvega do zadnjega.

```
#include<stdio.h>
long Fib(int b);     /*prototip*/
void main(void) {
  int n;
  printf("Katero Fibbonaccijevo število naj računa:\n");
  scanf("%d", &n);
  printf("%d-ti člen je %ld \n", b, Fib(b));
} /* main */

long Fib(int n) {
  long a, b, c;    /* členi zaporedja */
  if (n<=2) return 1;
  else {
    a= b= 1;
    while (n>2) {
      c=a+b;
      n--;
      a= b;
      b= c;
    }
    return c;
  }
} /* Fib */
```