C#

Zanke – zanka for

Zanka for - sintaksa

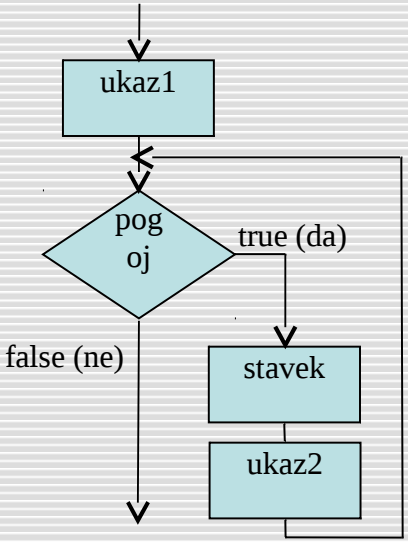
- Je najpogosteje uporabljena zanka, zelo podobna zanki *while*
- za besedo *for* v oklepajih zapišemo 3 dele: *ukaz1*, *pogoj* in *ukaz2,* ločene z podpičjem. Sledi mu blok stavkov (oz. stavek), ki jih želimo izvajati večkrat – toliko časa, dokler je pogoj izpolnjen.

```
for(predZanko/ukaz1/; pogoj; poKoraku/ukaz2/)
{
        stavek1;
        stavek2;
        …
        stavekn;
}
```

```
/*Oblika zanke pomeni: "Izvedi stavek predZanko. Nato preveri pogoj.
    Če je izpolnjen, izvedi stavke stavek1, stavek2 … stavekn in
    nato še stavek poKoraku. Ponovno preveri pogoj. Če je še vedno
    izpolnjen, ponovno izvedi stavke stavek1, stavek2 … stavekn in
    še stavek poKoraku. Ponovno preveri pogoj. Ko pogoj ni
    izpolnjen, se zanka konča." */
```

Zanka for – diagram poteka

- Diagram poteka



- Katero zanko bomo uporabili, je odvisno od naše odločitve. Zanko for običajno uporabljamo, ko moramo šteti ponovitve. Stavek predZanko takrat običajno uporabimo za nastavitev števca, v pogoju preverjamo, če je števec že dosegel določeno mejo, v stavku poKoraku pa povečujemo (ali zmanjšujemo) števec.

Prednosti zanke for - uporaba

- Zanke *for* so torej programske strukture, za katere so značilni naslednji elementi:
  - števec – spremenljivka, katere vrednost se spreminja med izvajanjem zanke,
  - začetna vrednost je vrednost, ki določa začetno stanje števca,
  - končna vrednost je vrednost, pri kateri se izvajanje zanke konča
  - korak zanke je vrednost, ki se prišteje števcu v eni ponovitvi zanke.

Zgled – izpis števil

- Izpisati želimo števila od *1* do *10*, vsakega v svoji vrstici

```
for(int stevilo = 1; stevilo <= 10; stevilo++)
{
   Console.WriteLine (stevilo);
}

//namesto stevilo++ bi seveda lahko zapisali daljšo obliko:
   stevilo=stevilo + 1;
```

Zgled – izačun vsote vrste

- Napiši program, ki izračuna naslednjo vsoto za poljubno število sumandov!

$$\frac{2}{2} + \frac{2}{3} + \frac{2}{4} + \frac{2}{5} + \ldots$$

Izpisuj tudi vmesne vsote!

```
Console.Write("Vnesi število členov(1 ali več) : ");
//pretvorba v celo število: lahko tudi Convert.ToInt32()
int clenov = int.Parse(Console.ReadLine());
double suma = 0;
for (int i=0;i<clenov;i++)
{
    suma = suma + (double)2 / (2 + i);
    Console.WriteLine(i + 1 + " : " + suma); /*izpisujemo
                                    vmesne rezultate*/
}
Console.WriteLine("\nVsota " + clenov + ". členov tega
   zaporedja je " + suma);
```

Zgled – dvojna for zanka

- Napiši program, ki izpiše naslednji vzorec. Primer: za n = 5 je izpis takle:

```
5
4 4
3 3 3
2 2 2 2
1 1 1 1 1
```

```
Console.Write("Velikost vzorca (1 - 9): ");
int n = Convert.Toint32(Console.ReadLine());
for(int i = n; i > 0; i--)
{   //  z zunanjo zanko izpisujemo vrstice
    for(int j = n; j >= i; j--) /*  izpis posameznih
                                    števil v vrstici*/
       Console.Write(i);
    Console.WriteLine();  //  zaključimo tekočo vrstico
}
```

Zanka for – problemi, napake

- Paziti moramo, da se zanka konča. V primeru nepravilne uporabe tudi zanka *for* teče v neskončnost (se zacikla).

```
for(int stevilo = 1; stevilo > 0; stevilo++)
{
   //NAPAČEN pogoj – neskončna zanka
   Console.WriteLine (stevilo);
}
```

- Pogosta napaka: podpičje na začetku!

```
for(int i = 1; i < 100; i++);
{
   //PRAZEN for stavek, zaradi podpičja
   ... Stavki...
}
```

Zanka for - posebnosti

- V vsakega izmed treh glavnih delov stavka *for* lahko združimo več stavkov, ki morajo biti ločeni z vejico. Pomembno pa je, da se tako zapisani stavki izvajajo od **leve proti desni**.

```
for (int i=0,j=10; i<10 && j>0; i++,j--)
{
    ...stavki...
}
```

- Ker je korak zanke najpogosteje 1, bomo v tretjem delu najpogosteje srečali obliko *stevec++*, ki, kot že vemo, pomeni *stevec = stevec + 1, j-- pa pomeni j=j-1;*.

Zgled – zanimiva števila

- Sestavi program ki poišče vsa naravna števila med 0 in 10000, ki so enaka vsoti kubov svojih števk. Eno od števil, ki ima zahtevano lastnost, je npr. 153: 153 = 13 + 53 + 33.

```
int i, j ,k, l;  //števke
double st, vk;    //število in vsota kubov števk
for (i = 0; i < 10; i++)
 for (j = 0; j < 10; j++)
   for (k = 0; k < 10; k++)
     for (l = 0; l < 10; l++)
     {
       st = 1000*i + 100*j + 10*k + l; // sestavimo število
       vk = Math.Pow(i,3) + Math.Pow(j,3) + Math.Pow(k,3)
            + Math.Pow(l, 3); // vsota kubov
       if (st == vk)
          Console.Write(st+",   ");
     }
```

Vaje – točke na premici

- Dana je premica y = 2x+3. Napiši program, ki bo izpisal koordinate poljubnih petih točk, ki ležijo na dani premici. Program naj te vpraša za x koordinate točk.
- Napiši program, ki za vsa naravna števila med 1 in 20 izpiše tabelo njegovih kvadratov, kubov in kvadratnih korenov. Izpis primerno oblikuj
- Napiši program, ki bo prebral dve celi števili, ugotovil, katero je večje in izpisal:
  - vsa števila med najmanjšim in največjim številom.
  - vsa števila med najmanjšim in največjim številom, ki delijo največje število.
  - vsa števila med najmanjšim in največjim številom, ki so soda in delijo največje število.
  - vsa števila med najmanjšim in največjim številom, ki so soda ali delijo največje število.