3 A FOR ciklus

- a **FOR** ciklus
- a szám ill. szöveg kiírása előre megadott hosszúságú helyre
- a **BEGIN** ... **END;** kulcsszavak használata

3.1 A FOR ciklus

Gyakran előfordul, hogy a programunkban valamit többször meg szeretnénk ismételni. Ilyenkor ciklusokat használunk. Ezeknek több fajtájuk van, most a **for** ciklussal fogunk foglalkozni, melynek a következő szerkezete van:

```
for ciklusváltozó := kifejezés1 to kifejezés2 do utasítás ;
```

A *ciklusváltozó* felveszi először a *kifejezés1* értékét. Végrehajtja az *utasítás*-t, majd a *ciklusváltozó* növekszik eggyel és ismét végrehajtja az *utasítás*-t. Ezután ismét növekszik eggyel és végrehajtja az *utasítás*-t. Mindezt addig ismétli, amíg a *ciklusváltozó* nem lesz egyenlő a *kifejezés2* értékével. Ekkor még utoljára végrehajtja az utasítást.

Ha a *kifejezés2* értéke kisebb, mint a *kifejezés1* értéke, akkor az utasítást egyszer sem hajtja végre.

Ha a *kifejezés1* értéke egyenlő a *kifejezés2* értékével, akkor az utasítást csak egyszer hajtja végre.

Példaként készítsünk egy programot, amely 1-től 10-ig kiírja az összes egész számot és mellé a szám négyzetét is. Ehhez szükségünk lesz egy ciklusváltozóra - jelöljük ezt most i-vel. Mivel ez a változó egész számértékeket fog felvenni, ezért ezt **integer** típusú változónak deklaráljuk. Ennek a változónak az értéke először 1 lesz, majd 2, 3,... egészen 10-ig. Ezt a folytonos növekedést 1-től 10-ig ciklus segítségével fogjuk megoldani. A ciklusmagban mindig ki fogjuk íratni az i változó értékét és hozzá a szám négyzetét is, tehát az i*i értékét. Programunk így nézni ki:

```
program Pelda12a;
uses crt;
var i:integer;
begin
 clrscr;
 for i:=1 to 10 do writeln(i,' negyzete = ',i*i);
 readln;
end.
```

Ez a program a következőt fogja kiírni a képernyőre:

```
1 negyzete = 1
2 negyzete = 4
3 negyzete = 9
4 negyzete = 16
5 negyzete = 25
6 negyzete = 36
7 negyzete = 49
8 negyzete = 64
9 negyzete = 81
10 negyzete = 100
_
```

Itt fontos megjegyeznünk, hogy egy szám négyzetének kiszámítására létezik a pascalban matematikai függvény is, mégpedig az **sqr()**. A programunkban tehát az i*i helyett írhattunk volna sqr(i) kifejezést. A későbbiekben már ezt a függvényt fogjuk használni egy szám négyzetének a kiszámítására.

Hasonlóan a négyzetgyök kiszámítására is létezik egy függvény, ez pedig az **sqrt()**.

3.2 A szám ill. szöveg kiírása előre megadott hosszúságú helyre

A kiírásnál megfigyelhettük, hogy a program az 1, 2, ... 9 számokat egymás alá írta, de a 10 számnál (mivel ez kétjegyű), a szöveg további része egy hellyel arrébb tolódott. Szöveg kiírásánál van arra is lehetőségünk, hogy egy bizonyos szöveget, számot előre megadott hosszúságú helyre írjunk ki. Tehát például az 1, 2, ... 9 számokat is két helyre írjuk ki, még ha ezek csak egyet foglalnak is el. Ezt a **writeln** parancsban adhatjuk meg a kiírandó szám mögé írva a **:2**-t, ahol a 2 azt jelenti, hogy az előtte levő számot 2 helyre szeretnénk kiírni. Programunk ez után a módosítás után így néz ki:

```
program Pelda12b;
uses crt;
var i:integer;
begin
 clrscr;
 for i:=1 to 10 do writeln(i:2,' negyzete = ',sqr(i));
 readln;
end.
```

És ez fog megjelenni a képernyőn, miután lefuttattuk:

```
 1 negyzete = 1
 2 negyzete = 4
 3 negyzete = 9
 4 negyzete = 16
 5 negyzete = 25
 6 negyzete = 36
 7 negyzete = 49
 8 negyzete = 64
 9 negyzete = 81
10 negyzete = 100
_
```

Láthattuk, hogy az egyjegyű számokat is most már két helyre írja ki a program, mégpedig úgy, hogy a szám elé rak egy szóközt. Hasonlóan megoldhatjuk, hogy az 1, 4, 9, 16, ... számoknál is az egyesek az egyesek alatt, a tízesek a tízesek alatt, stb. legyenek. Mivel itt a legnagyobb szám háromjegyű, ezért itt minden számot három helyre íratunk ki:

```
program Pelda12c;
uses crt;
var i:integer;
begin
 clrscr;
 for i:=1 to 10 do writeln(i:2,' negyzete = ',sqr(i):3);
 readln;
end.
```

Ez a program már így fogja kiírni a számokat a képernyőre:

```
 1 negyzete =   1
 2 negyzete =   4
 3 negyzete =   9
 4 negyzete =  16
 5 negyzete =  25
 6 negyzete =  36
 7 negyzete =  49
 8 negyzete =  64
 9 negyzete =  81
10 negyzete = 100
_
```

A számhoz hasonlóan egy szöveges változónál vagy egy szövegnél is megadhatjuk, hogy azt milyen hosszú helyre akarjuk kiíratni, például így:

```
writeln('Hova kerul ez?':20);
```

Ilyenkor a szám kiírásához hasonlóan a szöveg elé megfelelő számú szóközt tesz ki a program. Néha ügyelnünk kell arra, hogy a szöveges képernyő felbontása 80 x 25, tehát egy sorban 80 karakter fér el. Ha ennél többet írunk ki, akkor a maradékot (ami nem fér ki) már a következő sor elejére fog kerülni.

3.3 A BEGIN ... END; kulcsszavak használata

Gyakran szükségünk lehet arra, hogy egy cikluson belül több utasítást is elvégezzen a programunk. Például az előző programunk ne csak kiírja a ciklusban azt, hogy melyik számnak mennyi a négyzete, hanem mindegyik sor után egy új sorba rajzoljon (írjon ki) egy vonalat is egyenlőség jelekből. Ekkor már a ciklusunkon belül két **writeln** parancsot kéne használnunk (egyet a ciklusváltozó négyzetének a kiírására, egyet pedig a vonal kiírására). Ez csak úgy oldható meg, ha a ciklus után a parancsokat **begin** ... **end;** kulcsszavak közé rakjuk. Ezt használva természetesen nem csak kettő, de akár több parancsot is összekapcsolhatunk a cikluson belül. Ilyen esetben a ciklusunk így néz ki:

```
for ciklusváltozó := kifejezés1 to kifejezés2 do  begin
                                                     első parancs ;
                                                     második parancs ;
                                                     ...
                                                     utolsó parancs ;
                                                  end ;
```

Így valójában a ciklusváltozó minden egyes értékére végrehajtódik a cikluson belüli első, második, ... utolsó parancsot. Ilyen **begin** ... **end;** parancsot máshol is fogunk még használni, ahol a cikluson (feltételen) belül egy parancs helyett többet szeretnénk elvégezni (összekapcsolni).

Próbáljuk meg most átírni a négyzetszámok nevű programunkat úgy, hogy minden kiírás után írjon ki új sorba egy egyenlőség jelekből álló vonalat is. Ezt a vonalat egy újabb **writeln** paranccsal fogjuk kiíratni, melyet az eddigi **writeln** parancsal együtt a **begin** ... **end;** kulcsszavak közé rakunk. Programunk tehát így néz ki:

```
program Pelda12d;
uses crt;
var i:integer;
begin
 clrscr;
 for i:=1 to 10 do begin
                   writeln(i:2,' negyzete = ',sqr(i):3);
                   writeln('==================');
                   end;
 readln;
end.
```

És ha lefuttatjuk, ezt írja ki a képernyőre:

```
 1 negyzete =   1
 2 negyzete =   4
 3 negyzete =   9
 4 negyzete =  16
 5 negyzete =  25
 6 negyzete =  36
 7 negyzete =  49
 8 negyzete =  64
 9 negyzete =  81
10 negyzete = 100
_
```

Feladat: Készítsünk programot, amely az előző feladathoz hasonlóan kiírja 1-től 10-ig mindegyik egész szám négyzetét egymás alá (az egyenlőségekből álló vonalakat most ne írja ki). Ez után az egész kiírás után rakjon ki egy mínusz jelekből álló vonalat és ez alá a vonal alá írja ki a program hogy mennyi a kiírt négyzetszámok összege, tehát hogy mennyi az 1+4+9+16+25+36+...+100 összeg.

Megoldás: A feladat megoldásához bevezetünk egy s változót, melynek típusa **integer**. Ennek a változónak a program elején 0 értéket adunk, majd a ciklusban mindig miután kiírtuk a négyzetszámot, hozzáadjuk ehhez a változóhoz is az éppen kiírt négyzetszámot. Így a ciklus lefutása után az s változóban a keresett összeg lesz. A ciklus lefutása után már csak kiírunk egy mínusz jelekből álló vonalat, majd kiírjuk hogy mennyi ennek az s változónak az értéke. Programunk így néz ki:

```
program Pelda13;
uses crt;
var i,s:integer;
begin
 clrscr;
 s:=0;
 for i:=1 to 10 do begin
                   writeln(i:2,' negyzete = ',sqr(i):3);
                   s:=s+sqr(i);
                   end;
 writeln('-----------------');
 writeln('Ezek osszege: ',s);
 readln;
end.
```

Miután lefuttattuk a programot, a következő jelent meg a képernyőn:

```
 1 negyzete =   1
 2 negyzete =   4
 3 negyzete =   9
 4 negyzete =  16
 5 negyzete =  25
 6 negyzete =  36
 7 negyzete =  49
 8 negyzete =  64
 9 negyzete =  81
10 negyzete = 100
-----------------
Ezek osszege: 385
_
```

A fenti programban érdekes lehet még az s:=s+sqr(i); sor. Ez azt csinálja, hogy az **s** változó értékéhez (melyet a program legelején beállítottunk 0-ra) hozzáadja az **i** változó értékének a négyzetét, majd az eredményt beteszi az **s** változóba. Tehát valójában az **s** változó értékéhez hozzáadja az **i** négyzetét. Mivel a ciklusban az **i** 1-től 10-ig megy, ezért az **s**-hez sorban hozzáadja az 1 négyzetét, majd a 2 négyzetét, 3 négyzetét, stb. egészen a 10 négyzetéig.