12 Kétdimenziós tömbök

- kétdimenziós tömbök

12.1 Kétdimenziós tömbök

Az eddig használt egydimenziós tömbökön túl használhatunk többdimenziós tömböket is. Példaként nézzük meg, hogyan is képzelhetünk el egy 4x5 méretű kétdimenziós tömböt és hogyan hivatkozhatunk a tömb elemeire:

a[1,1] 0	a[1,2] 0	a[1,3] 0	a[1,4] 0	a[1,5] 0
a[2,1] 0	a[2,2] 0	a[2,3] 0	a[2,4] 0	a[2,5] 0
a[3,1] 0	a[3,2] 0	a[3,3] 0	a[3,4] 0	a[3,5] 0
a[4,1] 0	a[4,2] 0	a[4,3] 0	a[4,4] 0	a[4,5] 0

Láthotjuk, hogy az egyes elemekre az elem sorának és oszlopának megadásával hivatkozhatunk, például a harmadik sor harmadik eleme: a[3,3].

Az ábrán látható tömbben minden elem értéke 0. Mivel a többdimenziós tömbökkel egymásba ágyazott ciklusok segítségével dolgozunk (a külső ciklus adja meg a sorindexeket, a belső az oszlopindexeket), a tömb összes elemének 0-ra állítása is két ilyen egymásba ágyazott ciklus segítségével oldható meg:

```
program Pelda30a;
uses crt;
var a:array[1..4,1..5] of integer;
    i,j:integer;
begin
clrscr;
 for i:=1 to 4 do
  for j:=1 to 5 do a[i,j]:=0;
readln;
end.
```

A tömb kiírása a képernyőre is két egymásba ágyazott ciklus segítségével lesz megoldva. A belső ciklus fog kiírni egy sort, a külső ciklus pedig megadja a sorok számát. Az előző példa kiírással kiegészítve:

```
program Pelda30b;
uses crt;
var a:array[1..4,1..5] of integer;
    i,j:integer;
begin
clrscr;
 for i:=1 to 4 do
  for j:=1 to 5 do a[i,j]:=0;
 for i:=1 to 4 do
  begin
  for j:=1 to 5 do write(a[i,j]:2);
  writeln;
  end;
writeln;
end.
```

Készítsünk most egy programot, amely kigenerálja egy 10x10-es kétdimenziós tömbbe a 10-es szorzótáblát, majd kiírja a képernyőre. A tömbünkbe tehát a következő értékeket szeretnénk kigenerálni:

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	12	14	16	18	20
3	3	6	9	12	15	18	21	24	27	30
4	4	8	12	16	20	24	28	32	36	40
5	5	10	15	20	25	30	35	40	45	50
6	6	12	18	24	30	36	42	48	54	60
7	7	14	21	28	35	42	49	56	63	70
8	8	16	24	32	40	48	56	64	72	80
9	9	18	27	36	45	54	63	72	81	90
10	10	20	30	40	50	60	70	80	90	100

Ehhez a tömb mindegyik elemébe berakjuk az oszlopindexének és a sorindexének a szorzatát. Programunk így néz ki:

```
program Pelda31a;
uses crt;
var t:array[1..10,1..10] of integer;
    i,j:integer;
begin
 clrscr;
 { kigenerálás... }
 for i:=1 to 10 do
  for j:=1 to 10 do t[i,j]:=i*j;
 { kiírás... }
 for i:=1 to 10 do
  begin
  for j:=1 to 10 do write(t[i,j]:4);
  writeln;
  end;
 writeln;
end.
```

A fenti programban a tömb kigenerálását és kiírását elvégezhetjük ugyanabban a ciklusban is. Programunk ekkor így módosul:

```
program Pelda31b;
uses crt;
var t:array[1..10,1..10] of integer;
    i,j:integer;
begin
 clrscr;
 { kigenerálás és kiírás ugyanabban a ciklusban }
 for i:=1 to 10 do
  begin
  for j:=1 to 10 do begin
                    t[i,j]:=i*j;
                    write(t[i,j]:4);
                    end;
  writeln;
  end;
 readln;
end.
```