16. TÖBBDIMENZIÓS LISTA – MÁTRIX, TÖMB

```
matrix=[['A','B','C'],['D','E','F'],['G','H','I']]
matrix=[['A','B','C'],
        ['D','E','F'],
        ['G','H','I']]
```

Ahogy az előző fejeztben is említettük a listánk egyik eleme lehet egy másik lista. Sőt egy lista minden eleme is lehet (belső) lista. Ilyenkor beszélünk többdimenziós listásól.
Tehát a „mátrix” lista első (0.) eleme: =['A','B','C']
és annak az első (0.) eleme: 'A'
Mindig a nagyobb (külső) listából indulunk a kisebb (belső) lista felé!
A többdimenziós listákat mindig téglalapként képzeljük el, írjuk fel! (második példa)

Az elemek téglalapszerű elrendezését kétdimenziós listának, vagy mátrixnak nevezzük.
Az első index a sort (sorindex), a második az oszlopot (oszlopindex) jelöli.
Azt nem szabad elfelejteni, hogy itt is az index értéke a nullától indul.
Tehát a matrix[1][2] ➔ F
Nem kell azonos sor és oszlopszámúnak lennie a tömböknek!
pl. n=5, m=6 (5*6)
Mondatszerű leírásban a tömbök megadása pl.: matrix(2,4)
Pythonban az előbb említett módon **pl.: matrix[2][4]**
Többdimenziós tömbök készítésénél fokozottan figyeljünk a négyzetes zárójelek „[]” és a vesszök”,” helyes használatára!

0,0	0,1	0,2
1,0	1,1,	1,2
2,0	2,1	2,2

A	B	C
D	E	F
G	H	I

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

(16a.py)

Készítsünk programot, melyben megadunk és kiíratunk egy 3*3-as tömböt! A feladatban egy „amőba” játék tábláját jelenítsük meg!
A táblában az üres helyet „_” karakter, a másik kettőt pedig a „X” és „O” karakterek jelenítsék meg!

```
16a.py
tabla=[['_','X','0'],
       ['X','_','0'],
       ['_','0','X']]
for i in range(0,3):
    for j in range(0,3):
        print(tabla[i][j],end=" ")
    print()
```

- Az első három sorban megadjuk a tábla elemeit. Sorokba tördelve megfelelő struktúrával! Az elemek megfelelő elválasztására figyeljünk!
- Majd indítunk egy külső ciklust, melyben a sorokat fogjuk váltani. (0,1,2 sorok)
- Aztán egy belső ciklust indítunk, amelyben a sorokban lévő elemeit íratjuk ki a minta szerint. A karaktereket szóközökkel választjuk el egymástól!
- Majd az utolsó sorban a külső ciklus sortörése miatt egy print() utasítást gépelünk be. Új sort kezdünk.
- Futtassuk és teszteljük a programunkat!

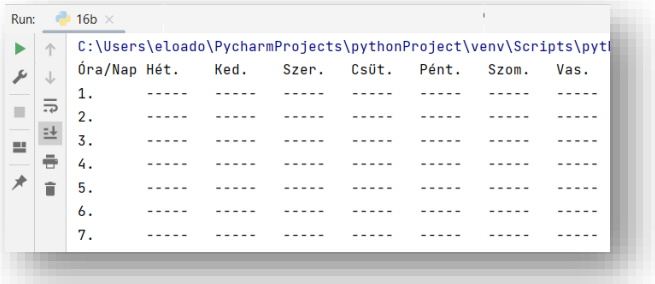
```
Run:  16a
E:\00_MM\12_Python_pr
_ X 0
X _ 0
_ 0 X

Process finished with
```

(16b.py)

Készítsünk programot, amely az előző példához hasonlóan egy fix tömböt kiírat a képernyőre!
Ebben a példában egy órarendet szeretnénk kiíratni a képernyőre, soronként soron belül tabulátorokkal elválasztva.

```
16b.py
orarend=[['Óra/Nap','Hét.','Ked.','Szer.','Csüt.','Pént.','Szom.','Vas.'],
         ['1.  ','-----','-----','-----','-----','-----','-----','-----'],
         ['2.  ','-----','-----','-----','-----','-----','-----','-----'],
         ['3.  ','-----','-----','-----','-----','-----','-----','-----'],
         ['4.  ','-----','-----','-----','-----','-----','-----','-----'],
         ['5.  ','-----','-----','-----','-----','-----','-----','-----'],
         ['6.  ','-----','-----','-----','-----','-----','-----','-----'],
         ['7.  ','-----','-----','-----','-----','-----','-----','-----']]
for i in range(0,8):
    for j in range(0,8):
        print(orarend[i][j],end="\t")
    print()
```

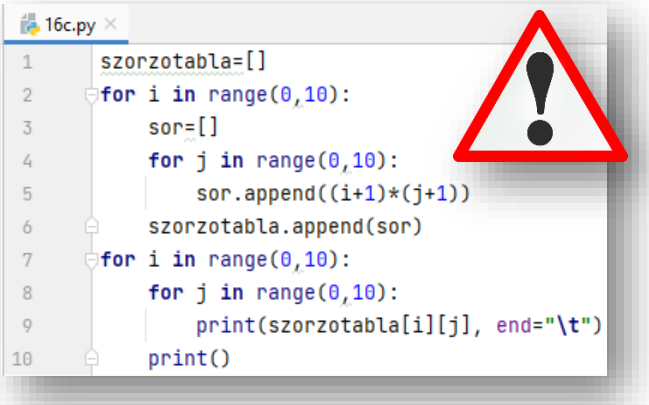
- A program elején megadjuk egy 8*8-as tömbben az első sorban a napokat, majd a sorszámokat és az órák helyét 5db „-„ karakterrel.
- Majd egymásba ágyazott ciklussal 0-7-ig kiíratjuk a tömb elemeit tabulátorokkal elválasztva, majd sort váltunk.

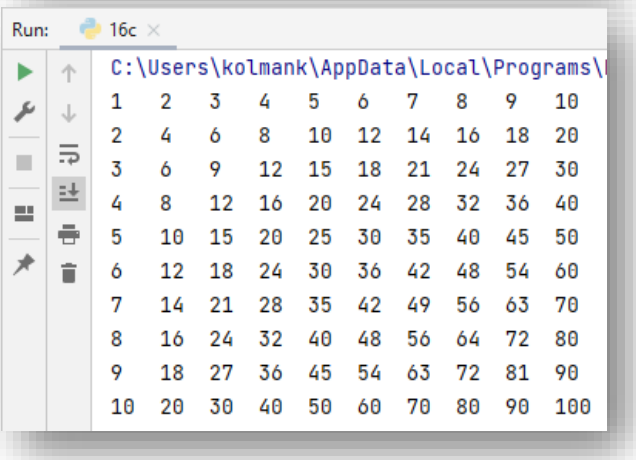


(16c.py)

Készítsünk programot, melyben egy kétdimenziós tömbben szorzótáblát készítünk, majd kiíratjuk a képernyőre!

- A program első sorában létrehozunk egy üres tömböt, szorzótábla néven.
- Majd (2-6. sor) feltöltjük tömbünket. Egymásbaágyazott külső for ciklussal indulunk. Az „i” változóval a sorokban lépkedünk, ezért minden sor elején kell egy „belső” tömböt létrehozni, melyben a „j” változós „belső” for ciklussal az „oszlopok” elemeit töltjük fel az ötödik sorban. Mivel a lista leme nullával indul, ezért hozzá kell adni egyet a változók aktuális értékéhez.
- Aztán minden feltöltött sor[] tömböt hozzáadunk a „szorzotabla” tömbhöz (6. sor).
- Végül tabulátorokkal elválasztva, egymásbaágyazott for ciklussal kiíratjuk a képernyőre a szorzótáblát.





(16d.py)

Készítsünk programot, melyben feltöltünk egy 5*12-es tömböt 1-99 közötti véletlen számokkal, majd kiíratjuk a tömb legkisebb és legnagyobb elmemét a minta alapján!

Tulajdonképpen az előző programot kell átírni és kibővíteni.