

# 03\_PythonBev\_vezerlesi\_szerk

August 23, 2024



## 1 Vezérlési szerkezetek I.

- Pythonban a vezérlési szerkezetek belsejét behúzással kell jelölni.
- Ezt a behúzást a Tab billentyű lenyomásával érhetjük el
- Emiatt garantált, hogy a program megjelenése és logikai jelentése összhangban van.

### 1.1 for ciklus:

- [for ciklus a Python dokumentációban](#)
- Szintaxis:

```
for ELEM in SZEKVENCIA:  
    UTASÍTÁS
```

- Megjegyzések:
  - A szekvencia lehet egész számok folytonos sorozata, de lehet más is (pl. sztring, tuple, lista, halmaz, szótár, megnyitott fájl).
  - Akkor érdemes for ciklust alkalmazni, ha A) a szekvencia már rendelkezésre áll vagy B) a ciklus kezdetekor tudjuk az iterációk számát.

#### 1.1.1 1. példa

Írjuk ki az alábbi könyvcímeket egymás alá a képernyőre:

Jane Eyre, Shirley, Agnes Grey, Üvöltő szelek

```
[21]: ## Érdemes listána gyűjteni őket:  
konyv_lista = ['Jane Eyre', 'Shirley', 'Agnes Grey', 'Üvöltő szelek']
```

```
[22]: ## ha csak az eddigi tudásunkat alkalmazzuk  
print(konyv_lista[0])  
print(konyv_lista[1])  
print(konyv_lista[2])  
print(konyv_lista[3])
```

Jane Eyre  
Shirley

Agnes Grey  
Üvöltő szelek

Na de mi van, ha bővül a lista további könyvekkel?

- Akkor az előző megoldás már nem lesz jó.
- Ráadásul mennyi sort írtunk már az előzőnél is.

Pont erre való a for utasítás

```
[23]: # írjuk ki a neveket for ciklussal
for konyv in konyv_lista:
    print(konyv)
```

Jane Eyre  
Shirley  
Agnes Grey  
Üvöltő szelek

Új könyvek a láthatáron! Fűzzük őket hozzá az előző könyv listánkhoz!

```
[24]: uj_konyvek = ['Wildfell asszonya', 'Villette', 'Henry Hastings kapitány']
```

```
[25]: # új könyvek hozzáfűzése az eredeti listához
konyv_lista = konyv_lista + uj_konyvek
print(konyv_lista)
```

['Jane Eyre', 'Shirley', 'Agnes Grey', 'Üvöltő szelek', 'Wildfell asszonya',  
'Villette', 'Henry Hastings kapitány']

```
[26]: # írjuk ki ismét a neveket for ciklussal
for konyv in konyv_lista:
    print(konyv)
```

Jane Eyre  
Shirley  
Agnes Grey  
Üvöltő szelek  
Wildfell asszonya  
Villette  
Henry Hastings kapitány

```
[27]: # sorszámozzuk is a kiírt könyvcímeket
sorszam = 1
for konyv in konyv_lista:
    print(str(sorszam)+". könyv: " , konyv)
    sorszam += 1
```

1. könyv: Jane Eyre  
2. könyv: Shirley  
3. könyv: Agnes Grey

4. könyv: Üvöltő szelek
5. könyv: Wildfell asszonya
6. könyv: Villette
7. könyv: Henry Hastings kapitány

### 1.1.2 Megjegyzés:

a könyvlisták összefűzését is elvégezhetem for ciklussal

```
[ ]: ## az eredeti listák:  
konyv_lista = ['Jane Eyre', 'Shirley', 'Agnes Grey', 'Üvöltő szelek']  
uj_konyvek = ['Wildfell asszonya', 'Villette', 'Henry Hastings kapitány']
```

```
[ ]: # összefűzésés for ciklussal:  
for konyv in uj_konyvek:  
    konyv_lista.append(konyv)  
  
# képernyőre íratás:  
print(konyv_lista) # ez már a cikluson kívüli utasítás!
```

### 1.1.3 1. példa folytatása

Írjuk ki az alábbi könyvcímeket egymás alá a képernyőre sorszámozva. - a könyvcímek után tüntessük fel az egyes címek hosszát is zárójelben - mekkora a könyvcímek összhosszúsága? - mekkora a könyvcímek átlagos hossza?

```
[22]: konyv_lista = ['Jane Eyre',  
                    'Shirley',  
                    'Agnes Grey',  
                    'Üvöltő szelek',  
                    'Wildfell asszonya',  
                    'Villette',  
                    'Henry Hastings kapitány']
```

```
[44]: # Megoldás:  
sorszam = 1  
ossz_hossz = 0  
for konyv in konyv_lista:  
    hossz = len(konyv)  
    print(str(sorszam)+". könyv: " , konyv, f"\t({hossz} karakter)")  
    sorszam += 1  
    ossz_hossz += hossz  
atlag = ossz_hossz/len(konyv_lista)  
print()  
print("A könyvcímek összhossza: ", ossz_hossz, "karakter")  
print()  
print("A könyvcímek átlagos hossza: ", f'{atlag:.2f}' )
```

1. könyv: Jane Eyre (9 karakter)
2. könyv: Shirley (7 karakter)
3. könyv: Agnes Grey (10 karakter)
4. könyv: Üvöltő szelek (13 karakter)
5. könyv: Wildfell asszonya (17 karakter)
6. könyv: Villette (8 karakter)
7. könyv: Henry Hastings kapitány (23 karakter)

A könyvcímek összhossza: 87 karakter

A könyvcímek átlagos hossza: 12.43

#### 1.1.4 2. példa

- a) Írjuk ki a képernyőre az első 10 négyzetszámot!
- b) Gyűjtsük össze egy listába az első 10 négyzetszámot!

```
[47]: # Értéktartomány (range) létrehozása.
list(range(10))
```

```
[47]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9]
```

```
[49]: # 3 paraméter: kezdő érték (1), minél kisebb (11), mekkora lépésközzel (2)
list(range(1, 10 + 1, 2))
```

```
[49]: [1, 3, 5, 7, 9]
```

```
[52]: # kiírás:
n = 10
for elem in range(1, n + 1):
    print(elem**2)
```

```
1
4
9
16
25
36
49
64
81
100
```

```
[53]: # listába gyűjtés
n = 10
negyzetszamok = [] # üres lista
for elem in range(1, n + 1):
    negyzetszamok.append(elem**2)
```

```
print(negyzetszamok)
```

```
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
```

### 1.1.5 3. példa

- a) Írjuk ki a képernyőre  $n$  darab  $*$  karaktert!
- b) Készítsünk  $n$ -szer  $n$ -es háromszöget  $*$  karakterekből!

```
n = 4
*
**
***
****
```

```
[3]: # Példa: n darab * karakter kiírása.
n = 3
# for ciklussal
for elem in range(n):
    print("*",end=" ")

print()
# beépített lehetőséggel egyszerűbb
print("*"*n)
```

```
***
***
```

```
[58]: # Példa: n-szer n-es háromszög * karakterekből.
n = 4
for elem in range(n):
    print("*" * (elem + 1))
```

```
*
**
***
****
```

### 1.1.6 4. példa:

- a) Számoljuk meg a magánhangzókat egy angol kisbetűs szövegben!
- b) Készítsünk statisztikát a magánhangzókról! Azaz külön, külön legyen meg minden magánhangzóra a darabszáma.

**a) Stratégia:** Végigmegyünk a szöveg karakterein és ha találtunk egy magánhangzót azt följegezzük egy gyűjtő változóba.

```
[16]: # a) Magánhangzók megszámlálása (angol kisbetűs szövegben).
text = 'This is a short text for testing the algorithm.'
vowels = {'a', 'e', 'i', 'o', 'u'}

v_num = 0
for ch in text:
    if ch in vowels:
        v_num = v_num + 1
print("A magánhangzók száma: ", v_num)
```

A magánhangzók száma: 12

**b) Stratégia 1.:** Végigmegyünk az egyes magánhangzókra külön-külön.

Mindegyik magánhangzónál megvizsgáljuk, hogy hányszor szerepel a szövegben.

Az eredményt (magánhangzó → darabszám) például szótárban eltároljuk.

```
[17]: # b) Magánhangzók statisztikája szótárral
text = 'This is a short text for testing the algorithm.'
vowels = {'a', 'e', 'i', 'o', 'u'}

# teljesen üres szótár feltöltése
d = dict()
for v in vowels:
    for ch in text:
        if v == ch:
            if v in d:
                d[v] += 1
            else:
                d[v] = 1

print(d)
```

{'e': 3, 'a': 2, 'o': 3, 'i': 4}

**b) Stratégia 2.:** Készítünk egy szótárat aminek a magánhangzók a kulcsai és minden értéket kezdetben 0-ra állítunk.

A szövegen végighaladva minden betűnél megvizsgáljuk, hogy szerepel-e a szótárkulcsok között. Ha igen akkor eggyel növelem a megtalált magánhangzónál szereplő értéket.

```
[18]: # b) Magánhangzók statisztikája szótárral (angol kisbetűs szövegben).
text = 'This is a short text for testing the algorithm.'
vowels = {'a', 'e', 'i', 'o', 'u'}

d = dict()

# üres szótár inicializálása kulcsokkal
```

```

for ch in vowels:
    d[ch] = 0
print(d)

# adatok feltöltése
for ch in text:
    if ch in d:
        d[ch] += 1

print(d)

```

```

{'e': 0, 'a': 0, 'o': 0, 'i': 0, 'u': 0}
{'e': 3, 'a': 2, 'o': 3, 'i': 4, 'u': 0}

```

## 1.2 if (ha) utasítás

- [if utasítás a Python dokumentációban](#)
- Szintaxis:

```

if FELTÉTEL1:
    UTASÍTÁS1
elif FELTÉTEL2:
    UTASÍTÁS2
else:
    UTASÍTÁS3

```

- Megjegyzések:
  - Több elif ág is szerepelhet.
  - Az elif ágak és az else ág is elhagyható.
  - Ha az utasítás 1 soros, akkor írható az if-fel elif-fel ill. az else-zel azonos sorba.

### 1.2.1 1. példa: Kérsz sört?

```

[ ]: ## Megoldás
kor = 15
if kor >= 18:
    print(kor, "éves vagy. Kaphatsz sört!")
else:
    print(kor, "éves vagy. Nem kaphatsz sört!")

```

15 éves vagy. Nem kaphatsz sört!

### 1.2.2 2. példa: Mennyi 2\*4? Az eredmény függvényében adjunk visszajelzést: - “Ügyes vagy!” - “Gyakorold még a szorzótáblát!”

```

[20]: # Megoldás:
n = int(input("Mennyi 2*4? "))

if n == 8:

```

```

    print("Ügyes vagy!")
else:
    print("Gyakorold még a szorzótáblát!")

```

Gyakorold még a szorzótáblát!

### 1.3 for ciklus if utasításokkal kombinálva

#### 1.3.1 1. példa: Számlista elemeinek vizsgálata

Az alábbi számlistában vizsgáljuk meg az elemeket: - írjuk ki az adott elemet, majd utána - írjuk ki ha negatív - írjuk ki ha egyjegyű - írjuk ki hogy páros, vagy páratlan

```
[ ]: num_list = [0, 99, -7, 56, 23, 6, 1, -5, 9]
```

```

[15]: # Megoldás:
print("-" * 42 + "\nSzámlista elemeinek vizsgálata:\n" + "-" * 42)

for num in num_list:
    # negatív? -----
    s = ""
    if num < 0:
        s = s + "negatív"

    # egyjegyű? -----
    e = ""
    if 1 <= num < 10:
        e = "egyjegyű"
    # páros/páratlan? -----
    if num % 2 == 0:
        p = "páros"
    else:
        p = "páratlan"

    # eredmény kiírása -----
    print("A vizsgált szám:", num, s, e, p)

```

```

-----
Számlista elemeinek vizsgálata:
-----

```

```

A vizsgált szám: 0    páros
A vizsgált szám: 99   páratlan
A vizsgált szám: -7   negatív páratlan
A vizsgált szám: 56   páros
A vizsgált szám: 23   páratlan
A vizsgált szám: 6    egyjegyű páros
A vizsgált szám: 1    egyjegyű páratlan
A vizsgált szám: -5   negatív páratlan

```



A vizsgált szám: 9 egyjegyű páratlan

### 1.3.2 2. példa: összetett feltételek

- a) Az alábbi számlistából válogassa ki a 2-vel és 3-mal is osztható számokat!
- b) A 100-nál kisebb pozitív egészek közül mely számok oszthatók 7-tel úgy, hogy a négyzetük 3-mal osztva 1 maradékot ad?

```
[9]: num_list=[0, 99, -7, 56, 23, 6, 1, -5, 9]
```

```
[10]: # a) Megoldás
print("\n2-vel és 3-mal is osztható számok listája.")
for num in num_list:
    if (num % 2 == 0) and (num % 3 == 0):
        print(num, end=', ')
```

2-vel és 3-mal is osztható számok listája.  
0, 6,

```
[11]: # b) Megoldás
print("\nMely számok oszthatók 7-tel úgy, hogy a négyzetük 3-mal osztva 1_
↳maradékot ad?")
for num in range(100):
    if (num % 7 == 0) and (num**2 % 3 == 1):
        print(num, end=', ')
```

Melyik számok oszthatók 7-tel úgy, hogy a négyzetük 3-mal osztva 1 maradékot ad?  
7, 14, 28, 35, 49, 56, 70, 77, 91, 98,