



Szkriptyelvek

Szathmáry László
Debreceni Egyetem
Informatikai Kar

2. Gyakorlat

- sztringek (folyt.)
- a lista adattípus
- for ciklus

(utolsó módosítás: 2026. jan. 16.)

2025-2026, 2. félév



Standard adattípusok

Standard adattípusok Pythonban:

- szám
 - sztring
 - lista
 - tuple
 - szótár (dictionary)
 - halmaz (set)
- { közös nevük: szekvencia }

Sztringek formázása #1:

```

4 def hello(name, color, obj):
5     print('{0}, {1} az {2}!'.format(name.capitalize(), color, obj))
6     # vagy
7     print('{}, {} az {}!'.format(name.capitalize(), color, obj))
8     # vagy
9     print('{n}, {c} az {o}!'.format(n=name.capitalize(), c=color, o=obj))
10
11
12 def main():
13     hello('geza', 'kek', 'eg')
14     print('-' * 30)
15     hello('peti', 'piros', 'auto')
16
17 if __name__ == "__main__":
18     main()

```

(lásd még: H függelék)

Gyakori hiba:

„konstans”



```

4 PI = 3.14159
5
6 # print('PI értéke: ' + PI)      # nem jó
7 print('PI értéke: ' + str(PI))  # jó
8 print('PI értéke:', PI)        # jobb

```

Sztringek formázása #2:

```
4 def hello(name, color, obj):  
5     print(f"{name}, {color} az {obj}!")  
6     # tetszőleges kifejezés is megadható:  
7     print(f"1 + 1 = {1+1}")      # 1 + 1 = 2  
8  
9 def main():  
10    hello("geza", "kek", "eg")
```

B	a	t	m	a	n
0	1	2	3	4	5

```

4  >>> a = 'Batman'
5  >>> a
6  'Batman'
7  >>> len(a)
8  6
9  >>> a[0]
10 'B'
11 >>> a[1:4] ←
12 'atm'
13 >>> a[0:4]
14 'Batm'
15 >>> a[0:3]
16 'Bat'
17 >>> a[3:6]
18 'man'
19 >>> a[3:]
20 'man'
21 >>> a[:3]
22 'Bat'
23 >>> a[:]
24 'Batman'
25 >>>

```

slice (szelet)

B	a	t	m	a	n
0	1	2	3	4	5
-6	-5	-4	-3	-2	-1

```

1  >>> a
2  'Batman'
3  >>> a[-1]
4  'n'
5  >>> a[-2] ←
6  'a'
7  >>> a[-6]
8  'B'
9  >>> a[-3:]
10 'man'
11 >>> a[:-3]
12 'Bat'
13 >>>

```

Negatív indexelés
(jobbról balra).

Megjegyzés:

$s[:n] + s[n:] == s$

(ahol n lehet pozitív vagy negatív érték is)

HF: string1.py kiegészítése.
Ha kész vagyunk vele, folytassuk a string2.py fájllal.

```
>>> s = "python programming"  
>>> s[::-2] ←  
'pto rgamm'  
>>> s[::-1]  
'python programming'  
>>> s[::-1] ←  
'gnimmargorp nohtyp'  
>>> s[:6]  
'python'  
>>> s[:6:2]  
'pto'
```

lépésköz

sztring megfordítása

```
>>> multi = """elso sor
... masodik sor"""
>>> multi
'elso sor\nmasodik sor'
>>> print(multi)
elso sor
masodik sor
>>>
>>> s = "hi\nthere"
>>> print(s)
hi
there
>>> len(s)
8
>>>
>>> s = r"hi\nthere"
>>> print(s)
hi\nthere
>>> len(s)
9
>>>
```

többsoros sztring

normál sztring

raw sztring
(főleg reguláris
kifejezéseknel használatos)

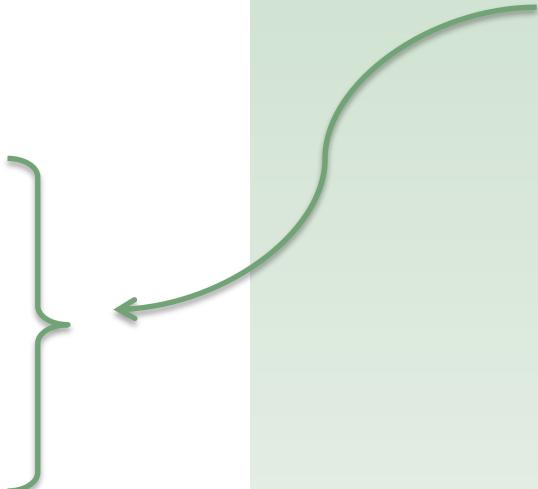
Gyakori hiba a Pythonnal ismerkedőknél

```

>>> a = 5
>>> print(++a)
5
>>> print(--a)
5
>>> print(a++)
File "<input>", line 1
    print(a++)
          ^
SyntaxError: invalid syntax
>>> print(a--)
File "<input>", line 1
    print(a--)
          ^
SyntaxError: invalid syntax
>>> --5
5
>>> a
5
>>> a += 1
>>> a
6
>>> a = 5
>>> a -= 1
>>> a
4
>>>
  
```

A + és - unáris operátorok,
 vagyis $a++$ jelentése: $+ (+5)$, aminek értéke 5.
 $a--$ jelentése: $- (-5)$, ami szintén 5...

Inkrementálásra / dekrementálásra
 $a+=$ és $-=$ operátorokat használjuk.



Listák

```
4  >>> [1, 2, 3]
5  [1, 2, 3]
6  >>> a = [1, 2, 3]
7  >>> a
8  [1, 2, 3]
9  >>> li = [] ←
10 >>> a = [1, 2, 'ab', 3.14]
11 >>> a
12 [1, 2, 'ab', 3.14]
23 >>> a = [1, 2, 3]
24 >>> a
25 [1, 2, 3]
26 >>> len(a)
27 3
28 >>> [1, 2] + [5, 6]
29 [1, 2, 5, 6]
```

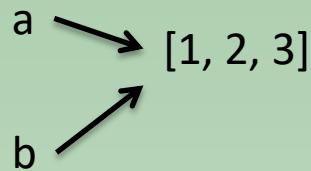
üres lista



a sztringeknél látott műveletek
nagyrésze itt is működik

```

1  >>> a = [1, 2, 3]
2  >>> b = a
3  >>> a
4  [1, 2, 3]
5  >>> b
6  [1, 2, 3]
7  >>> a[0] = 10
8  >>> a
9  [10, 2, 3]
10 >>> b
11 [10, 2, 3]
12 >>
13 >>> a
14 [10, 2, 3]
15 >>> b = a[:]
16 >>> b
17 [10, 2, 3]
18 >>> a[0] = 20
19 >>> a
20 [20, 2, 3]
21 >>> b
22 [10, 2, 3]
23 >>
24 >>> a == b
25 False
26 >>> [1, 2] == [1, 2]
27 True
28 >>> a
29 [20, 2, 3]
30 >>> a[1:]
31 [2, 3]
  
```

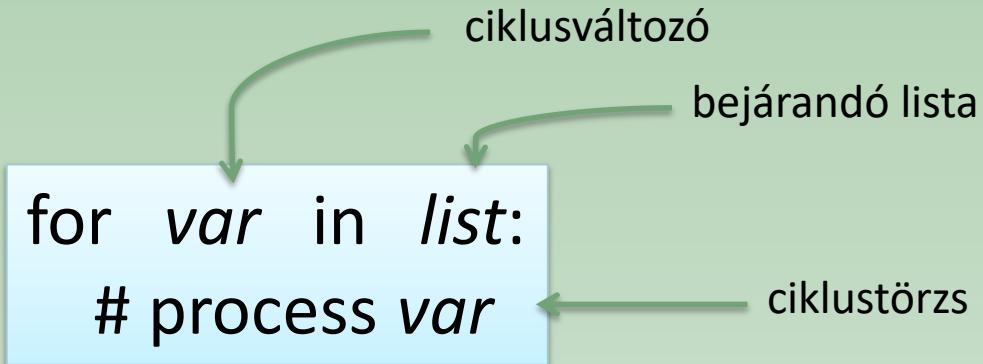


a-ról teljes másolat készül

két tömböt is össze lehet hasonlítani

slices (szeletek): ugyanúgy működik, mint a sztringeknél

foreach ciklus Pythonban



```
>>> li = [1, 2, 3]
>>> for e in li:
...     print(e)
...
1
2
3
```

- sztringekre is működik
- a listánkat soha ne hívjuk „list”-nek, ui. van egy ilyen nevű beépített függvény
- az előző pont a sztringekre is igaz: ott az „str” változónevet kell kerülni

gyakori minta

```
res = [] # üres lista
for e in lista:
    res.append(e)
# res feldolgozása
```

```
>>> li = [1, 2, 3, 4, 5, 6, 7, 8]
>>> paros = []
>>> for szam in li:
...     if szam % 2 == 0:
...         paros.append(szam)
...
...
...
>>> paros
[2, 4, 6, 8]
```

stílus: az operátorok előtt és után hagyunk ki egy szóközt

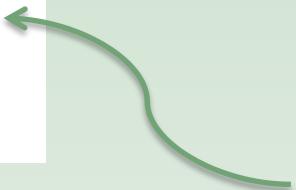


adott érték benne van-e egy listában

value in list

- True
- False

```
1 >>> li = [1, 2, 3]
2 >>> 2 in li
3 True
4 >>> 15 in li
5 False
6 >>>
7 >>> s = 'Python, C, C++, Java'
8 >>> '++' in s
9 True
```



sztringekre is működik



házi feladat



Feladatok

1. [\[20120815b\]](#) sztringek #1
2. [\[20120815c\]](#) sztringek #2
3. [\[20130218a\]](#) csodálatos elme
4. [\[20120815e\]](#) palindróm (triviális és rekurzív módszerrel)
5. [\[20120815j\]](#) egész szám megfordítása
6. [\[20120818j\]](#) számjegyek száma
7. [\[20120815a\]](#) két szám összege
8. [\[20141005a\]](#) valami_1 or valami_2 or ... valami_N
9. [\[20141005b\]](#) haladó sztringformázás