Python bevezető

Kabódi László

Python tulajdonságai

- dinamikusan típusos
- script nyelv: interpretált
- memóriakezeléssel nem kell foglalkozni - garbage collection
- többféle paradigmát is támogat (pl: objektum-orientált, funkcionális, imperatív)
- legtöbb nyelvtől eltérően a blokkokat a behúzás mértéke jelöli
- komment: #

Típusok

- nem kell előre deklarálni
- a típust az értékadásból kitalálja
- tömbök
  - `lista=[1, 2, 3, 4]`
  - `print(lista[1:3]) # [2, 3]`
  - `print(lista[1:]) # [2, 3, 4]`
  - `print(lista[:3]) # [1, 2, 3]`
  - `print(lista[:]) # [1, 2, 3, 4]`
  - `print(lista[-1]) # [4]`

Függvények

- `def function(args)`
- változó referenciát ad át érték szerint
- függvény hívásnál lehet sorrendben, vagy névvel hivatkozni a paraméterekre
-
```
def func(arg1, arg2, arg3, arg4)
    #itt van a függvény

param1=1
param2=2
papar4=4
func(param1, param2, arg4=param4)
```

Vezérlési szerkezetek - for

- ```
  for i in range(0, 11, 2):
      print(i)
  ```

- ```
  array = [1, 2, 3, 4]
  for e in array:
      print(e)
  ```

Vezérlési szerkezetek - if

```
if szam < 4:
    print("A szám kisebb, mint 4")
else:
    print("A szám legalább 4")
```

Vezérlési szerkezetek - while

```
i=1
while i < 11:
    print(str(i) + ". iteráció")
    i += 1
```

Futási időmérés

- `timeit.timeit(stmt, setup, timer, number)`

- stmt lehet python kifejezés, vagy függvény
- a függvény csak paraméter nélküli lehet

Input/output műveletek

- `f = open("filename", 'r')`
- a végén lehet w is, akkor írható lesz
- `f.read()`
- `f.readline()`
-
```
for line in f:
    #a sorokkal lehet varázsolni
```

numpy

- `import numpy as np`
- `A = np.array([[1, 2, 3], [4, 5, 6]], dtype = np.float64`
- np.add, np.multiply, np.dot