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)
- a 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]`
  - `print(lista[1:-1]) # [2, 3]`
  - `lista.append(5) # [1, 2, 3, 4, 5]`
  - `lista.extend([7, 8, 9]) # [1, 2, 3, 4, 5, 7, 8, 9]`
  - `lista.insert(5, 6) # [1, 2, 3, 4, 5, 6, 7, 8, 9]`

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
param4=4
func(param1, param2, arg4=param4)
```

Külső könyvtárak

- `from library import function`
- `import library as lib`

Véletlen

- random könyvtár
- `x = random()`
- $0 \leq x < 1$ lebegőpontos szám
- `y = uniform(a, b)`
- $a \leq y \leq b$ lebegőpontos

Formázott kimenet

- nagyjából C-s printf szintaxis
- ```
  print("Név: %s, ár: %.2f,\
  darab: %5d" %("alma", 199.99, 5))
  ```

- python szintaxis
- ```
  print("Név: {nev:s}, ár: {ar:.2f},\
  darab: {darab:5d}".format(nev="alma",\
  ar=199.99, darab=5))
  ```

- stringek összefűzése
- ```
  nev = "alma"
  ar = 199.99
  darab = 5
  print("Név: "+nev+", ár: "+str(ar)+\
  ", darab: "+str(darab))
  ```
  ```

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

```
szam = rnd.uniform(1, 10)
if szam < 4:
    print("A szám kisebb, mint 4! (%.2f)" %szam)
else:
    print("A szám nem kisebb, mint 4! (%.2f)" %szam)
```

Vezérlési szerkezetek - while

```
szam = 0
i = 0
while szam < 10:
    i = i+1
    szam=rnd.uniform(1,12)
print("A(z) %d. szám lett nagyobb, mint 10" %(i))
```

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, a-val hozzáfűz, r+-al írható és olvasható
- `f.write()`
  - file-ba ír
- `f.read()`
  - egész file-t olvassa
- `f.read(n)`
  - n karaktert olvas
- `f.readline()`
  - egy sort olvas
-
```
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, np.matmul, np.transpose, ...