



Szkriptyelvek

Szathmáry László
Debreceni Egyetem
Informatikai Kar

9. Gyakorlat

- modulok (folyt.)
- haladó rendezés
- kivételkezelés

(utolsó módosítás: 2025. aug. 14.)

2025-2026, 1. félév



Modulok (folyt.)

Láttuk, hogy ha egy projekten belül ugyanazt a függvényt több szkriptben is fel akarjuk használni, akkor ezeket a közös függvényeket érdemes egy modulban összegyűjteni (lásd `ex1.py` és `ex2.py` szkriptek, illetve `pygyak.py` modul).

Mi a helyzet akkor, ha **egy modult több projektben** is fel szeretnénk használni?

Lehetséges megoldás:

Tegyük a modult egy kitüntetett könyvtárba (pl. `HOME/lib`), majd ezt a könyvtárat tegyük be a `PYTHONPATH` környezeti változóba. Mivel a `sys.path`-ba az inicializáció során bekerülnek a `PYTHONPATH`-ban megadott könyvtárak, így a modulunkat bármelyik szkriptbe be tudjuk importálni.

Lépések

1. Hozzuk létre a `HOME` könyvtárunkban a `lib` alkönyvtárat, majd tegyük be ide a `pygyak.py` modult.
 2. A `~/.bashrc` fájlhoz adjuk hozzá a következőt:

```
PYTHONPATH=$PYTHONPATH:$HOME/lib  
export PYTHONPATH
```

3. Nyissunk egy új terminált (vagy `source ~/.bashrc`), majd teszteljük, hogy működik-e:

```
>>> import pygak  
>>> pygak.hello()
```

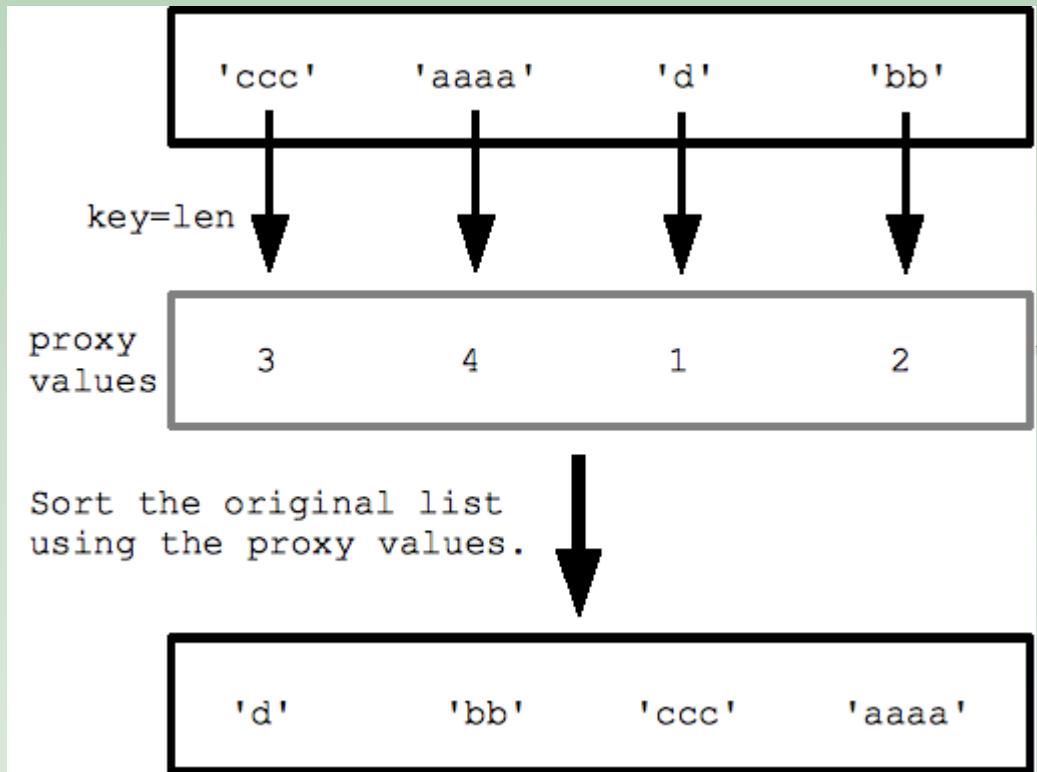
ha nincs hibaüzenet: OK

Haladó rendezés

Összetettebb rendezések esetén a `sorted()` függvénynek megadható egy „`key`” paraméter, melynek értékül egy függvényt lehet megadni. Ez a függvény a rendezés előtt az elemeket áttranszformálja. A függvény vesz egy értéket s egy új értéket ad vissza. A rendezés a függvény által visszaadott érték alapján történik.

```
1 >>> words = ['ccc', 'aaaa', 'd', 'bb']
2 >>> sorted(words)
3 ['aaaa', 'bb', 'ccc', 'd']
4 >>>
5 >>> sorted(words, key=len)
6 ['d', 'bb', 'ccc', 'aaaa']
```

A „`key`” értéke most a beépített „`len`” függvény. A szavakat a szavak hossza szerint szeretnénk rendezni. A rendezés során a `len` meghívódik a lista minden egyes elemére, s a rendezés a `len` által visszaadott érték szerint történik.



a „len” által visszaadott értékekből készít egy „árnyéklistát” s ezt rendezi:
 $[1, 2, 3, 4]$

majd visszahelyettesít:
 $\begin{aligned} 1 &\rightarrow 'd' \\ 2 &\rightarrow 'bb' \\ 3 &\rightarrow 'ccc' \\ 4 &\rightarrow 'aaaa' \end{aligned}$

```

1 >>> words = ['Cc', 'BB', 'aa', 'zz']
2 >>> sorted(words)
3 ['BB', 'Cc', 'aa', 'zz']
4 >>>
5 >>> sorted(words, key=str.lower) ←
6 ['aa', 'BB', 'Cc', 'zz']

```

rendezzük úgy, hogy a kis- és nagybetűket *nem* különböztetjük meg

```

1 >>> words = ['xc', 'zb', 'yd', 'wa']
2 >>> sorted(words)
3 ['wa', 'xc', 'yd', 'zb']
4 >>>
5 >>> def my_func(s): ←
6 ...     return s[-1]
7 ...
8 >>> sorted(words, key=my_func)
9 ['wa', 'zb', 'xc', 'yd']

```

rendezzük az elemeket
az utolsó karakter szerint

saját függvény is
megadható

Feladat

Oldjunk meg néhány haladó rendezéssel kapcsolatos feladatot.

Link: <https://arato.inf.unideb.hu/szathmary.laszlo/pmwiki/index.php?n=Py3.20121006e>

Kivételek

```
1 #!/usr/bin/env python3
2
3 import sys
4
5
6 def cat(fname):
7     f = open(fname, 'r')
8     text = f.read()
9     print('---', fname)
10    print(text)
11    f.close()
12
13 #####
14
15 if __name__ == "__main__":
16     args = sys.argv[1:]
17     for arg in args:
18         cat(arg)
```

a Unix cat parancsához
hasonló kiíratás



Feladat:

Nem létező file-ok esetén
hibaüzenet, majd a szkript
folytassa a feldolgozást a köv.
argumentummal.

Lásd még: <https://arato.inf.unideb.hu/szathmary.laszlo/pmwiki/index.php?n=Py3.20121120a>

```

6 def cat(fname):
7     try:
8         f = open(fname, 'r')
9         text = f.read()
10        print('---', fname)
11        print(text)
12        f.close()
13    except IOError:
14        print('--- I/O error:', fname)
  
```

A kivételt futásidőben elkapjuk s lekezeljük.

Ha nincs ilyen file, akkor hibaüzenet, s a program fut tovább.

```

6 def cat(fname):
7     try:
8         f = open(fname, 'r')
9         text = f.read()
10        print('---', fname)
11        print(text)
12        f.close()
13    except IOError as e: ←
14        print('--- I/O error:', e)
  
```

„e” a kivételobjektum

kiíratjuk a kivételobjektumot

Többféle kivételt is el lehet kapni egy except ágban, ekkor a kivételek típusait egy **tuple**-be kell rakni.

```
except (KeyboardInterrupt, EOFError):
```

Feladat

Szkript bővítése kivételkezelőkkel.

Link: <https://arato.inf.unideb.hu/szathmary.laszlo/pmwiki/index.php?n=Py3.20121120b>



házi feladat



Feladatok

1. [\[20130920a\]](#) kis- és nagybetűk felcserélve
2. [\[20120904b\]](#) lista egy részének a megfordítása
3. [\[20130326a\]](#) utolsó N sor, **B** változat