06_PythonBev_Fajlok_Fgv

November 13, 2024

1 6. alkalom: Fájlkezelés és ismerkedés a függvényekkel

1.1 Fájlkezelés

- Fájlkezelés a dokumentációban
- A fájl valamilyen adathordozón tárolt, logikailag összefüggő adatok összessége.
- Egy fájl életciklusa a következő lépésekből áll:
  1. megnyitás
  2. olvasás, írás, pozícionálás, …
  3. bezárás

1.1.1 Fájl írása `.write()`

A megnyitás lehetséges módjai: - `"r"`: olvasásra (read) - Csak meglévő fált tud megnyitni - `"w"`: írásra (write) - Létrehozza az új üres fájt. (Ha létezett, ha nem) - `"a"`: hozzáfűzésre (append) - A meglévő fájlt megnyitja és a fájl végére áll.

[1]:
```
## Klasszikus szintaxissal
#1 Fájl megnyitása (írásra) - létre is hozza a fájlt!
f = open('valami.txt', 'w')
#2 Sztring fájlba írása
f.write('1 alma\n')
#3 Fájl bezárása.
f.close()
```

[2]:
```
## with utasítás használatával     - Vigyázat!! felül írtam az előzőt
with open('valami.txt', 'w') as f:
    f.write("2 körte\n")
```

[3]:
```
# Fűzzünk hozzá valamit az előzőhöz!
with open('valami.txt', 'a') as f:
    f.write("3 barack\n")
```

Hová került ez a fájl? - A Colab online felületén a notebook fájl mellé tartozó ideiglenes könyvtárba. Lásd ball oldalt a könyvtár ikonra kattintva. (Ha a kapcsolat megszakad és újracsatlakozunk akkor eltűnik!) - Ha Jupyter-notebookot használunk az asztali gépünkön, akkor a notebook fájlt tartalmazó könyvtárba. - Természesesen létrehozhatok új könyvtárat, majd abba is írhatok, de akkor a fájlnévben az elérési utat is meg kell adjam.

Példa: Hozzunk létre egy `data` nevű könyvtárat a munkakönyvtárunkban, majd írjunk egy txt fájlt bele.

[4]:
```
## könyvtárat parancssori utasítással a notebookból is létrehozhatok!
## Ez a "mágikus parancs" csak Linux rendszerben, vagy a Colab online felületén␣
 ↪működik
!mkdir data
```

[5]:
```
# fájlnévben legyen ott az elérési útvonal!
with open('./data/valami.txt', 'w') as f:
    f.write("2 körte\n")
    f.write("3 barack\n")
```

Példa: készítsünk egy `gyumolcs.txt` nevű fájlt, amely az alábbi gyümölcsök neveit tartalmazza sorszámmal ellátva. Az első sorban szerepeljen egy fejléc a megadott sztringgel.

[6]:
```
nevek = ["alma", "körte", "szilva"]
fejlec = "index\tname\n"
```

[7]:
```
# Megoldás with használatával
with open('gyumolcs.txt', 'w') as f:
    f.write(fejlec)
    for num, gy in enumerate(nevek):
        f.write(f"{num}\t{gy}\n")
```

1.1.2 Fájl olvasása `.read()` `.readline()` `.readlines()`

Győződjünk meg róla, hogy az előbbi `gyumolcs.txt` fájl ott van a munkakönyvtárunkban. Ha esetleg nincs, akkor futtassuk az alábbi cellát!

[ ]:
```
## Ha nincs meg, akkor töltsük le (Colabban / Linuxos gépen működik)
!wget https://raw.githubusercontent.com/zoldbirka/colab-test-pub/master/_files/
 ↪gyumolcs.txt
```

[ ]:
```
## innen is ránézhetünk a megfelelő parancssori utasítással a fájlok listájára
!ls -l
```

[10]:
```
## Klasszikus szintaxis - Fájl tartalmának beolvasása sztringbe
f = open('gyumolcs.txt',"r")
s = f.read()
f.close()
```

[11]:
```
# Nézzünk rá a sztringre
print(s)
```

```
index   name
0       alma
1       körte
```

```
2       szilva
```

[12]:
```
# with használatával - Fájl tartalmának beolvasása sztringbe
with open('gyumolcs.txt',"r") as f:
    s = f.read()

print(s)
```

```
index   name
0       alma
1       körte
2       szilva
```

Példa: Készítsünk egy fájlt, majd olvassuk be az első sorát! - Az `example_file.txt`-t a munkakönyvtárban a New / Text File menüponttal hozhatjuk létre. (Desktop alkalmazás) - A Colab felületén pedig a bal oldali menü könyvtár ikonja, majd jobb egérgomb felugró menü Új Fájl. *Ez csak a munkamenet alatt létezik!* - De persze bármilyen egyszerű szövegszerkesztővel is elkészíthetjük, majd a notebook mellé menthetjük

[13]:
```
# Fájl elkészítése valamilyen módon
with open('example_file.txt', 'w') as f:
    f.write("2 körte\n")
    f.write("3 barack\n")
```

[14]:
```
# Első sor beolvasása
with open('example_file.txt',"r") as file:
    s = file.readline()
print(s)
```

```
2 körte
```

[19]:
```
# Megjegyzés: A readline a sortörést is beteszi az eredménybe.
s
```

[19]:
```
'2 körte\n'
```

[15]:
```
# A sortörést pl. a strip eljárással vághatjuk le:
s.strip()
```

[15]:
```
'2 körte'
```

[16]:
```
# Fájl sorainak beolvasása sztringlistába.
with open('example_file.txt',"r") as file:
    sorok_lista = file.readlines()
print(sorok_lista)
```

```
['2 körte\n', '3 barack\n']
```

Példa: Olvassuk be a korábban megírt `gyumolcs.txt` fájlt tartalmát adatpárok listájaként!

[ ]:
```
## Ha nincs meg, akkor töltsük le (Colabban / Linuxos gépen működik)
!wget https://raw.githubusercontent.com/zoldbirka/colab-test-pub/master/_files/
 ↪gyumolcs.txt
```

[22]:
```
## Sztring darabolása egy határoló jelsorozat mentén (tokenizálás).
line = 'aa,bb ,ccc'
line.split(",")
```

[22]:
```
['aa', 'bb', 'ccc']
```

[34]:
```
# Alapértelmezés szerint a split fehér karakterek mentén darabol.
line = 'aa  bb\tccc\n'
line.split(",")
```

[34]:
```
['aa  bb\tccc\n']
```

[35]:
```
# Iterálás egy szövegfájl sorain.
with open("gyumolcs.txt") as f:
    for line in f:
        print(line)
```

```
0       alma

1       körte

2       szilva
```

[17]:
```
# Fájl első sorának átugrása, a további sorok tokenizálása.
data = []
with open('gyumolcs.txt',"r") as f:
    f.readline() # 1. sor átugrása
    for line in f:
        tok = line.strip().split()
        record = int(tok[0]), tok[1]
        data.append(record)

print(data)
```

```
[(0, 'alma'), (1, 'körte'), (2, 'szilva')]
```

[25]:
```
# Ezt a tokenizálást nem feltétlenül nekünk kell megcsinálni
# csv modul, vagy egyébb adat feldogozó lehetőségek!
import csv
```

```
with open('gyumolcs.txt',"r", newline = '') as f:
    csvread = csv.reader(f, delimiter='\t')
    for line in csvread:
        print(line)
```

```
['index', 'name']
['0', 'alma']
['1', 'körte']
['2', 'szilva']
```

1.2 Függvények: bevezetés, alapfogalmak

- Függvények a dokumentációban
- A függvény névvel ellátott alprogram, amely a program más részeiből meghívható.
- Függvények használatával a számítási feladatok kisebb egységekre oszthatók. A gyakran használt függvények kódját könyvtárakba rendezhetjük.
- A matematikában egy függvénynek nincsenek mellékhatásai. Egy Python nyelvű függvénynek lehetnek!
- Pythonban a függvények "teljes jogú állampolgárok":
  - Egy változónak értékül adhatunk egy függvényt.
  - Függvényeket lehet egymásba ágyazni.
  - Egy függvény kaphat paraméterként függvényt ill. adhat eredményül függvényt.
- Fontos különbséget tenni egy függvény *definíciója* és *meghívása* között:
  - A függvény definíciója megadja, hogy milyen bemenethez milyen kimenet rendelődjön (és milyen mellékhatások hajtódjanak végre). A függvény definíciója a programban általában csak egy helyen szerepel (ha több helyen szerepel, akkor az utolsó definíció lesz az érvényes.)
  - A függvény meghívása azt jelenti, hogy egy adott bemenethez kiszámítjuk a hozzárendelt értéket. Egy definiált függvényt a programban többször is meg lehet hívni.

[ ]:
```
# Példa: n-edik gyök függvény definiálása.
def gyok(x , n = 2):
    return x**(1/n)
```

[27]:
```
# Példa: n-edik gyök függvény definiálása.
# Dokumentációs sztringgel és adattípusok megadásával

def gyok(x:float , n:int = 2) -> float:
    """
    x -első változó
    n - második
    x n-edik gyökét számítja ki alapbeállítás n = 2
    """
```

```
    return x**(1/n)
```

Ha a függvény első utasítása egy sztring, akkor ez lesz a függvény dokumentációs sztringje.

[68]:
```
# Dokumentációs sztring (docstring) lekérdezése.
# 'dunder' doc

print(gyok.__doc__)
```

```
    x -első változó
    n - második
    x n-edik gyökét számítja ki
```

- Pythonban a függvényeknek *pozícionális* és *kulcsszó* paraméterei lehetnek.
  - Függvénydefiníciókor először a pozícionális majd a kulcsszó paramétereket kell felsorolni.
  - A pozícionális paramétereknek nincs alapértelmezett értékük, a kulcsszó paramétereknek van.
  - Mindegyik paramétertípusból lehet nulla darab is.
- Függvényhíváskor…
  - Az összes pozícionális paraméter értékét meg kell adni, olyan sorrendben, ahogy a definíciónál szerepeltek,
  - A kulcsszó paraméterek értékét nem kötelező megadni.

[28]:
```
# Gyök 4 kiszámítása.  - Kulcsszó paraméter nem feltétlen szükséges
print(gyok(4, 2))
print(gyok(4))
```

```
2.0
2.0
```

[29]:
```
# Köbgyök 4 kiszámítása. - A kulcsszó paramétert nem kell feltétlen nevesíteni
print(gyok(4, n = 3))
print(gyok(4, 3))
```

```
1.5874010519681994
1.5874010519681994
```

[74]:
```
# Változónak értékül adhatunk függvényt.
y = gyok
```

[75]:
```
# nézzük meg az új függvény változó típusát, dokumentációs sztringjét!
print(type(y))
print(y.__doc__)
```

```
<class 'function'>
```

```
    x -első változó
```

```
n - második
x n-edik gyökét számítja ki
```