Szöveges állományok kezelése

Készítette: Vastag Atila

2017

Kezdjük egy file megnyitásával és tartalmának a képernyőre írásával, és e file tartalma legyen a következő:

```
# -*- coding: ISO-8859-2 -*-
from typing import *
from io import open

file: TextIOWrapper = open ("./data/ip.txt", encoding='utf-
8,mode=„r")

oneLine: str = file.readline()
allLines: List[str] = file.readlines()

file.close()

print(allLines)
```

```
project
|
-- data
     |
     --ip.txt
|
--main.py
```

```
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4
10.0.0.5
10.0.0.6
```

Egész file beolvasása egyetlen karakterláncba az alábbiak szerint történik.

Először is az **open()** függvény segítségével megnyitjuk a beolvasni kívánt file-t. Az **open()** függvény egy előkészítő függvény, amely arra készíti fel a rendszert, hogy az adott merevelemezen (vagy más adattárolón) fellelhető file-lal dolgozni tudjunk. Legegyszerűbb esetben ez a file nevének a megadásával történik.

```
file: TextIOWrapper = open ("./data/ip.txt", encoding='utf-8')
```

A fenti kódsor hatására a **file** változón keresztül férhetünk hozzá a file tartalmához.

Meghívható függvények a **file**-on:

- **read(n)** → függvény a megadott **n** számú karaktert olvassa ki az aktuális filefolyamból

- **tell()** → függvényt alkalmazva megkaphatjuk az aktuális folyampoziciót (épp hanyadik karekter kiolvasásánál tart).

- **seek()** → függvény segítségével állíthatjuk be egy megnyitott filefolyam aktuális pozicióját.

- **readline()** → függvény segítségével a megnyitott filefolyam egy sorát olvashatjuk ki mint string

- **readlines()** → függvény segítségével a megnyitott filefolyam több (összes) sorát olvashatjuk ki mint string lista

A nyitott fájlok erőforrásokat fogyasztanak, és a fájl módjától függően más programok esetleg nem tudják elérni azokat. Fontos a fájlokat azonnal bezárni, amint végeztél a feldolgozásukkal.

```
file.close()
```

Ha már minden olvasás és írás befejeződött, akkor a **close()** utasítással zárhatjuk be a folyamot, azaz ez a parancs szakítja meg a kommunikációt az adattárolóval.

Hogy véletlenül se felejtsük el bezárni a file-t pythonban van egy kényelmes/hasznos konstrukció a **with** . Ez meghívja az adott objektum __enter__() es __exit__() tagfüggvényeit. A with block után a változó megmarad, de a file már zárva van.

A **with** előnyei:

- biztos nem felejtjük el bezárni a filet
- ha valami (i/o) hiba történik, a fájl akkor is bezáródik
- nyertünk egy sor kódot
- szinte biztos, hogy nem keverjük össze a különböző fájlokat,amibe írni akarunk file1,file2, …
- kizárólag ebben a blokkban tudunk a fájlba írni
- rövid ideig van nyitva egy fájl, kevesebb az esélye, hogy több helyről van írva olvasva egyszerre
- érthető a kódban, hogy mi hol történik

```
# -*- coding: ISO-8859-2 -*-

from typing import *
from io import open

with open ("./data/ip.txt", encoding='utf-8,mode="r")) as file:
    oneLine: str = file.readline()
    allLines: List[str] = file.readlines()

print(oneLine)
print(allLines)
```

Most úgy ahogy az a nagykönyvben is meg van írva

```
# -*- coding: ISO-8859-2 -*-
from typing import *
from io import open

oneLine: str = None
allLines: List[str] = None
try:
    with open ("./data/ip.txt", encoding='utf-8, mode="r") as file:
        oneLine = file.readline()
        allLines = file.readlines()
except FileNotFoundError as ex:
    print(f"{ex.filename} nem található!")
else:
    print(oneLine)
    print(allLines)
```

Most úgy ahogy az a nagykönyvben is meg van írva adatok olvasása sorról sorra nagy méretű szöveges állományból

```
from typing import *
from io import open
filename:str = “data/ip.txt”
oneLine:str
allLines:List[str]=[]

try:
        here: str = os.path.dirname(os.path.abspath(__file__))
        path: str = os.path.join(here, fileName)

    with open(path,encoding='utf-8', mode="r") as file:
        for line in file:
            oneLine = line.strip()
            allLines.append(oneLine)
except  FileNotFoundError as ex:
    print(f"{ex.filename} nem található!")

for line in allLines:
    print(f"{line}")
```

Most úgy ahogy az a nagykönyvben is meg van írva adatok olvasása sorról sorra nagy méretű szöveges állományból, ha egy objektum adatai több sorban vannak

```
from typing import *
from io import open

class Line:
    def __init__(self, line1: str, line2: str, line3: str, line4: str) -> None:
        super().__init__()
        self.row:List[str] = [line1, line2, line3, line4]

    def __str__(self) -> str:
        return f"{line1} {line2} {line3} {line4}"

oneLine:str
line: Line
lines: List[Line] = []

try:
    with open("./data.txt",encoding='utf-8', mode="r") as file:
        while(oneLine := file.readline().strip()):
            line1:str = oneLine
            line2:str = file.readline().strip()
            line3:str = file.readline().strip()
            line4:str = file.readline().strip()

            line=Line(line1, line2, line3, line4)
            lines.append(line)
except  FileNotFoundError as ex:
    print(f"{ex.filename} nem található!")

for item in lines:
    print(f"{item.row}")
```

File megnyitási módok:

Character	Meaning
`'r'`	open for reading (default)
`'w'`	open for writing, truncating the file first
`'x'`	open for exclusive creation, failing if the file already exists
`'a'`	open for writing, appending to the end of the file if it exists
`'b'`	binary mode
`'t'`	text mode (default)
`'+'`	open a disk file for updating (reading and writing)

JSON beolvasása

```
{
  "firstName": "John",
  "lastName": "Smith",
  "isAlive": true,
  "age": 27,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021-3100"
  },
  "phoneNumbers": [
    {
      "type": "home",
      "number": "212 555-1234"
    },
    {
      "type": "office",
      "number": "646 555-4567"
    }
  ]
}
```

```
class Address:
    def __init(self, streetAddress: str, city: str, state: str,
postalCode: str) -> None:
        self.streetAddress = streetAddress
        self.city = city
        self.state = state
        self.postalCode = postalCode


class Phone:
    def __init__(self, type: str, number: str) -> None:
        self.type = type
        self.number = number


class Person:
    def __init__(self, firstName:str, lastName:str, isAlive:bool,
age: int, address: Address, phoneNumbers:List[Phone]) -> None:
        self.firsName = firstName
        self.lastName = lastName
        self.isAlive = isAlive
        self.age = age
        self.address = address
        self.phoneNumbers = phoneNumbers
```

```python
from typing import List
import json

from person import Person

# read file
with open('data.json', 'r') as myfile:
    data=myfile.read()

# parse file
persosns: List[Person] = json.loads(data)

print(persosns)
```

Írás szövegfájlokba

A fájlokba nagyjából ugyanúgy írhatsz, ahogy olvasod azokat. Először megnyitsz egy fájlt, és kapsz egy adatfolyam-objektumot, aztán metódusokat használsz az adatfolyam-objektumon, hogy adatokat írj a fájlba, majd bezárod a fájlt.

Egy fájl írásra való megnyitásához használd az **open()** függvényt, és add meg az írás módot. Két fájlmód használható az íráshoz:

Az „*írás*” mód felülírja a fájlt. Add át a **mode='w'** paramétert az **open()** függvénynek.
A „*hozzáfűzé*s” mód a fájl végéhez fog adatokat adni. Add át a **mode='a'** paramétert az **open()** függvénynek.

Mindkét mód automatikusan létrehozza a fájlt, ha még nem létezik, így nincs szükség semmi szöszmötölős „ha a fájl még nem létezik, akkor hozz létre egy új fájlt csak hogy először megnyithasd” függvényre. Csak nyisd meg a fájlt, és kezdd írni.

Mindig zárd be a fájlokat, amint végeztél az írásukkal, hogy felszabadítsd a fájlhivatkozást, és biztosítsd, hogy az adatok valójában kiírásra kerülnek a háttértárra. Ahogyan az adatok fájlból olvasásakor, itt is meghívhatod az adatfolyam-objektum **close()** metódusát, vagy használhatod a with utasítást, és rábízhatod a fájl bezárását a Pythonra.

Fogadni mernék, hogy kitalálod, melyik módszert javaslom.

```
# -*- coding: ISO-8859-2 -*-
from typing import *
from io import *


try:
    with open ("./data/ip.txt", encoding='utf-
8', mode="a") as file:
        file.write("123.123.123.123")
except FileNotFoundError as ex:
    print(f"{ex.filename} nem található!")
```

```
10.0.0.1
10.0.0.2
10.0.0.3
10.0.0.4
10.0.0.5
10.0.0.6
123.123.123.123
```

1 – Egy szöveges állományban, **eredmeny.txt**, az érettségizők pontjai vannak elmentve a következő módón, soronként és a sorokban tabulátorral elválasztva:

Virág 9,28
Jázmin 6,26

Feladatunk, hogy kikeressük a legtöbb és legkevesebb pontot elért érettségizőt és a **max.txt** illetve a **min.txt** állományokba írjuk bele őket.
Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

2 – A programunk feladata, hogy hőmérsékletet mérjen reggel délben és este, majd ezeket a hőmérsékleteket a **meresek.txt** állományokba mentse a hét minden napján (napi három mérés).
A hőmérsékleteket **Random** számmal adjuk meg 0 és 40 közt!
Keressük ki a hét végén, miután megtörtént az ösz mérés, hogy mekkora volt az átlag hőmérséklet reggel, délben és este és az **atlag.txt** állományba mentsük el.

Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.

3 – Egy **forrás.txt** állományban számoljuk meg a magán és mássalhangzókat, számokat és egyéb szimbólumokat, majd az eredményt írjuk az **eredmeny.txt** állományba.
Egyes logikai egészeket alkotó műveleteket függvényekkel oldjuk meg.
forrás.txt:
“Microsoft suspends the Intel Kaby Lake and Ryzen AMD processors for Windows 7 and 8. Unfortunately, the tech mogul has no plans in bringing the next-gen chips to the old operating systems in the future. In fact, just last week, Microsoft released the KB 4012982 error entitled "Your PC uses a processor that isn't supported on this version of Widows.”

http://www.universityherald.com/articles/69674/20170317/microsoft-blocks-windows-7-8-updates-ryzen-amd-intel-kaby.htm

4 – Egy szöveges dokumentumban e-mail címek találhatóak és a hozzájuk tartozó jelszó. Kérjük meg a felhasználót hogy adja meg az e-mail címét és a jelszót és ha ez megtalálható a szöveges dokumentumban léptessük be a rendszerbe és egy log.txt állományba írjuk be, hogy mikor és ki lépett be. Az e-mail cím bekérésénél ügyeljünk arra, hogy megfeleljen az e-mail cím formátumának.
A feladatot az OOP-al oldjuk megoldjuk meg.

5 – A **konyvek.txt** állományban az adatok a következő módón vannak tárolva:

- Vezetéknév (íróé),
- Keresztnév (íróé),
- SzületésiDátum,
- Cím,
- ISBN,
- Kiadó,
- KiadvasiÉv,
- ár,
- Téma,
- Oldalszám,
- Honorárium (amit a könyvért kapott az író)

a) Írjuk ki a képernyőre az össz adatot
b) Keressük ki az informatika témajú könyveket és mentsük el őket az **informatika.txt** állömányba
c) Az 1900.txt állományba mentsük el azokat a könyveket amelyek az 1900-as években íródtak
d) Rendezzük az adatokat a könyvek oldalainak száma szerint csökkenő sorrendbe és a **sorbarakott.txt** állományba mentsük el.
e) „kategoriak.txt” állományba mentse el a könyveket téma szerint. Például:

Thriller:
- könnyv1
- könnyv2

Krimi:
- könnyv1
- könnyv2

6 – A **roplabda.txt** állományban az adatok a következő módón vannak tárolva:

Név,
Magasság,
Poszt,
Nemzetiség,
Csapat,
Ország (ahol a csapat játszik)

a) Írjuk ki a képernyőre az össz adatot
b) Keressük ki az ütő játékosokat az **utok.txt** állömányba
c) A **csapattagok.txt** állományba mentsük a csapatokat és a hozzájuk tartozó játékosokat a következő formában:

Telekom Baku: Yelizaveta Mammadova,Yekaterina Gamova,

d) Rendezzük a játékosokat magasság szerint növekvő sorrendbe és a **magaslatok.txt** állományba mentsük el.
e) Mutassuk be a **nemzetisegek.txt** állományba, hogy mely nemzetiségek képviseltetik magukat a röplabdavilágban mint játékosok és milyen számban.
f) **atlagnalmagasabbak.txt** állományba keressük azon játékosok nevét és magasságát akik magasabbak mint az „adatbázisban” szereplő játékosok átlagos magasságánál.
g) állítsa növekvő sorrendbe a posztok szerint a játékosok ösz magasságát
h) egy szöveges állományba, „*alacsonyak.txt*” keresse ki a játékosok átlagmagasságától alacsonyabb játékosokat. Az állomány tartalmazza a játékosok nevét, magasságát és hogy mennyivel alacsonyabbak az átlagnál, 2 tizedes pontossággal.

7 – A **magyarvarosok.txt** állományban az adatok a következő módón vannak tárolva:

Nev (város neve),
Város típusa,
Megye név,
Járas,
Kistérség,
Népesség,
Terület

a) Írjuk ki a képernyőre az össz adatot
b) Keressük ki a megyeszékhely megyei jogú városokat és mentsük el a **megyejoguvarosok.txt** állományba
c) Az **nepesseg.txt** állományba mentsük el azokat a településeket és a hozzájuk tartozó adatokat, ahol a népesség 50.000 és 100.000 közt van
d) Keressük ki azokat a településeket, melyek területei meghaladják az 200-at és a **nagyteruletek.txt** állományba mentsük el.
e) Keressük ki Békés megye össz települését és a **bekes.txt** állományba mentsük el.
f) **megyeterületek.txt** állományba mentsük el a megye nevét és területének nagyságát.

8 – A **lotto.txt** állományban az adatok a következő módón vannak tárolva:

Név,
tippek

a) Írjuk ki a képernyőre az össz adatot
b) Random számok segítségével generáljuk le a napi 7 nyerő számot és írjuk őket egy szüveges állományba melynek az aktuális nap lesz a neve
c) Keressük ki, van(ak)-e 7 találatos szelvény(ek), ha igen írjuk ki a nyertesek nevét a **nyertesek-{mai dátum}.txt** állományba.
d) Keressük ki, hogy a befizetett játékosok hány találatot értek el, és mentsük el a **talalatok-{mai dátum}.txt** állományba a játékos nevét és a találatainak számát

9 – A **vezetok.txt** állományban az adatok a következő módón vannak tárolva:

Vezetéknév,Keresztnév,Anya vezetékneve, Anya keresztneve
Születés időpont,Születés helye,Megye,Ország
Utca,Házszám,Irányítószám,Város,Megye,Ország
Kategóriák

Az egyes egésszek tabulátorral vannak elválasztva, az egésszek adatai pedig vesszővel.

Írjunk programot, mely menü segítségével lehetővé teszi a következő adatok kírását:

a) Írjuk ki a képernyőre az össz adatot a vezetőkről
b) A felhasználó álltal megagadott megyére a **megye-vezetoi.txt** állományba elmenti a megadott megyében lakó vezetőket.
c) A felhasználó álltal megagadott kategóriával rendelkező vezetőket a {**kategoria nev**}-**kategoria.txt** állományba menti el.
d) A **fiatalok.txt** állóménybа kikeresi azokat a vezetőket akik 18 és 21 év közt vannak.
e) **kulfoldi.txt** állományba azokat a vezetőket keresi ki, akik nem Magyarországon születtek.

10 – Az **nb1.txt** állományban az adatok a következő módón vannak tárolva:

A labdarúgó mezére írt szám (szám)
A labdarúgó utóneve (szöveg); előfordul, hogy valaki felvett nevet használ, ilyenkor üres is lehet
A labdarúgó vezetékneve (szöveg)
A labdarúgó születési dátuma (dátum)
Értéke igaz, ha magyar állampolgár (is) a labdarúgó (logikai) [-1 igaz, 0 hamis]
Értéke igaz, ha külföldi állampolgár (is) a labdarúgó (logikai) [-1 igaz, 0 hamis]
A labdarúgó euró ezrekben kifejezett értéke (szám)
A klub neve (szöveg)
A poszt neve (szöveg), például kapus, bal oldali védő, bal szélső

Írjunk programot, mely menü segítségével lehetővé teszi a következő adatok kírását:

a) A kapusokon kívül mindenkit mezőnyjátékosnak tekintünk. Keresse ki a legidősebb mezőnyjátékos vezeték- és utónevét, valamint születési dátumát! (Feltételezheti, hogy csak egy ilyen játékos van.)

b) Határozza meg hány magyar, külföldi és kettős állampolgárságú játékos van!

c) Határozza meg játékosok összértékét csapatonként és írja ki a képernyőre! A csapatok neve és a játékosainak összértéke jelenjen meg!

d) Keresse ki, hogy mely csapatoknál mely posztokon van csupán egy szerződtetett játékos! Írja ki a csapat nevet és a posztot amire csak egy játékost szerződtettek!

e) Keressük ki azon játékosokat, akiknek az értékük nem haladja meg a játékosok értékének átlag értékét.

f) Írja ki azon játékosok nevét, születési dátumát és csapataik nevét, akik 18 és 21 év közt vannak és magyar állampolgárok. Ha nincs ilyen, akkor megfelelő üzenettel helyettesítse a kimenetet.

g) A „hazai.txt” illetve a „legios.txt” állományokba keresse ki a magyar, illetve a külföldi állampolgárságú játékosokat csapatonként. A szöveges állományoknak tartalmazniuk kell a csapat nevét majd alatta felsorolva a játékosok teljes nevét, poszt nevet és értéküket.

11 - Adva van az adatok.txt állományban a Magyar Női Röplabda Bajnogság csapatainak pontászámai a következő képpen:

Békéscsaba
1,2,1,3,3,3,3,3,3,3,1,2,2,1,3,3,1,3

ahol a csapat nevét tabulátorral elválasztva követik a az elért pontok mérközésenként (max 18 lejátszott mérközés).

a) Hány csapat vett részt a bajnokságban?
b) Ki nyerte a bajnogságot?
c) Döntetlen mérkőzéskor a csapat 2 pontot szerez. Mutassa be csapatonként ki hány döntetlen mérkőzést játszott le!
d) Ha egy mérkőzés 5 szetben dől el, akkor a vesztes csapat 1 pontot szerez. Mely csapatok játszottak 5 szettes mérkőzést és hányat?
e) Ki a bajnogság első három helyezete. Mutassa be mintának megfelelően:
   helyezés - csapat neve *pontszám*
e) Az elért pontok alapján, az utolsó három csapat kiesik az első osztályból. Kik ők?
f) Mutassa be csapatonként a győzelmi és verességi arányt csapatonként!
g) Mely csapatok győzelmi aránya van az átlag alatt?