9. hét: fájlkezelés

Czirkos Zoltán · 2023.11.07.
Szövegfájlok beolvasása. Formátumhibák kezelése.

Felkészülés a laborra:

- A fájlkezelésről szóló előadásrészlet áttekintése.
- A kivételkezelésről szóló előadás feleleventése.

Tartalom

1. Harmadik kis ZH
2. Szavak beolvasása és szűrése
3. Metróvonalak
4. ZH eredmények
5. Szám vagy nem szám?
6. Hibás fájl
7. Dolgozatok
8. ZH eredmények oszlopdiagramon

1. Harmadik kis ZH

A hétfői alkalmakon lesz a harmadik kis ZH az informatikusoknak.

2. Szavak beolvasása és szűrése

Adott a következő fájl: szavak_50.txt. Ez ötven darab magyar szót tartalmaz. Mentsd le ezt a *fájlt* a programod mellé, mert ezzel kell dolgoznia!

Írj programot, amelyik az alábbi feladatokat végzi el:

- Beolvassa a fájl szavait egy listába. Ügyelj arra, hogy ne legyen sorvége jel a szavak végén! Legyen ez a **betoltes()** nevű függvény, amelynek paramétere a fájlnév, visszatérési értéke a szavak listája! Ellenőrizd a szavak számát, és magukat a szavakat is a beolvasás után!
- Rendezd sorba a listát a beépített rendező függvénnyel, **sorted()** vagy **.sort()**.
- Írd ki egy másik fájlba csak a **k** betűvel kezdődő szavakat! Legyen ennek a neve **szavak_kbetus.txt**, és legyen ugyanolyan a formátuma, soronként egy darab szó! Ellenőrizd a keletkező fájlt, pl. jegyzettömbbel! A fájlba írást végezze a **mentes()** nevű függvény, amelynek paraméterben lehet megadni a fájlnevet és a mentendő listát!

Ne felejtsd el sehol bezárni a fájlt! Használd a **.close()** függvényt, vagy **with** blokkot, ahogy az előadáson szerepelt.

Karakterkódolások

Előfordulhat (tipikusan Windowson), hogy a fájlt a Python nem olyan karakterkódolással olvassa be, ahogy kellene. Ilyenkor az ékezetes betűk helyett furcsa karakterek jelennek meg, pl. „árvíztűrő tükörfúrógép" helyett „árvÍztűrő" vagy „ÁárvÍztLarL". Ha ilyet látsz, az **open()** függvénynek add meg paraméterként a karakterkódolást is. Ez a többi feladatra is érvényes:

```
open("szavak_50.txt", "rt", encoding="utf-8")
```

A témáról többet a karakterkódolások oldalon olvashatsz. Ez nem vizsgaanyag, de érdemes tudni róla, mert gyakran okoz galibát.

Azt is észreveheted, hogy a sorba rendezés is hibás: az ékezetes karakterrel kezdődő szavak a lista végére kerülnek. Ennek az az oka, hogy pl. a 'á' betű kódja az nagyobb, mint a 'z' betűjé.

```
print(ord('á'),ord('z'))
225 122
```

Ezen úgy lehet segíteni, hogy egyrészt megadjuk azt a nyelvet, amin a sorbarendezés történik, ezt az ún. locale segítségével tehetjük meg, másrészt explicite kérjük a sorted függvényt, hogy a nyelvfüggő összehasonlító függvényt használja. Részletek a referenciában, többnyelvű program írásakor érdemes használni.

```
import locale

szavak=['alma','áfonya','banán','szőlő']
locale.setlocale(locale.LC_COLLATE, 'hu_HU') # így csak a sorrend vagy LC_ALL, akkor mindent magyar nyelv szerint.
print(*sorted(szavak))
print(*sorted(szavak, key= locale.strxfrm))

alma banán szőlő áfonya
áfonya alma banán szőlő
```

3. Metróvonalak

Adott két fájl: m1.txt és m2.txt. Ezek a budapesti 1-es és 2-es metró megállóinak neveit tartalmazzák.

- Írj függvényt, amely paraméterként a fájl nevét kapja, és visszaad egy listát, amelyik a megállók neveit tartalmazza!
- Írj függvényt, amelyik paraméterként két fájl nevét kapja, és meghatározza, hogy át lehet-e szállni az egyik metróvonalról a másikra! Ha igen, térjen vissza a megálló nevével, ha nem, akkor **None** értékkel!

Teszteld a programod további adatokkal! A másik két metróvonal: m3.txt és m4.txt. Az M1–M4 között nincs átszállási lehetőség, a többinél van.

4. ZH eredmények

A következő fájl egy régebbi évfolyam első kis ZH-n elért eredményeit tárolja: kzh_pontszam.txt. A fájlban a pontszámok összeszőve vannak, minden sorban egy szám.

- Olvasd be a fájlban tárolt számokat egy listába! Ügyelj arra, hogy ezeket **int**-té kell alakítanod. Legyen ez a **beolvas()** függvény, amelynek paramétere a fájlnév, értéke a pontszámok sorozata!
- A legkisebb pontszám 0, a legnagyobb 10. Készítsd statisztikát: hány 0 pontos, hány 1 pontos, ... ZH lett! Tedd ezt a **stat()** nevű függvénybe, amely paraméterként az összes pontszámok kapja, visszatérési értékként pedig a statisztikát állítja elő! (Pontszámmal indexelhető lista, amely létszámokat tartalmaz.)
- Írd ki a statisztikát, pontszámok szerinti eloszlást a **stat_kiir()** függvényben, amely paraméterként a **[pontszam]-áltszám listát kapja**! Írd ki azt is, hogy hány sikeres ZH lett, ahol a pontszám legalább 4! Hány %-a ez az évfolyamnak?
- Ügyelj itt is a fájl bezárására! Hívd meg a főprogramból a függvényeket, hogy megjelenjen az eredmény!

Ez kell legyen az eredmény:

```
 1 db  0 pontos
 3 db  1 pontos
10 db  2 pontos
11 db  3 pontos
 9 db  4 pontos
12 db  5 pontos
14 db  6 pontos
41 db  7 pontos
11 db  8 pontos
18 db  9 pontos
29 db 10 pontos
Átment: 134 fő, 84.28%
```

▶ Megoldás

5. Szám vagy nem szám?

Tedd félre kicsit a pontszámos programot, később még lesz vele feladat!

Írj egy függvényt, amelyik paraméterként egy sztringet vesz át, és visszatérési értékében megmondja, hogy szám van benne vagy nem szám! Pl. **szam_e("123")** értéke **True**, **szam_e("almafa")** értéke viszont **False** kell legyen. Emlékezz vissza: ezt legegyszerűbben úgy lehet megoldani, hogy számmá konvertálod a kapott sztringet, és elkapod a kivételt, ha kell.

▶ Megoldás

6. Hibás fájl

A kzh_pontszam_hibas.txt fájl ugyanazokat az adatokat tartalmazza, mint az előző – azzal a különbséggel, hogy ebben hibás sorok is vannak. Némelyik üres, máshol beficzen egy-egy szó vagy egy valós szám, ami nem való a helyes adatok közé.

A feladatod úgy módosítani az előző feladat beolvasó függvényt, hogy kihagyja ezeket a hibás sorokat, és csak a helyes adatokat adja vissza a listában!

▶ Tipp

▶ Megoldás

7. Dolgozatok

A zheredmeny.txt fájl, ahogy a tartalmán is látszik, kitalált neveket, NEPTUN kódokat és dolgozatok eredményeit tárolja:

```
DHSF7F:Fül Elek:24
ZFO9PY:Szőke Barna:17
LGNUKG:Metall Ica:3
XMXO5D:Füle Imre:12
...
```

Minden sor egy dolgozat adatait tárolja: NEPTUN kód, név és pontszám; mindezek kettősponttal elválasztva.

- Definiálj osztályt, amelyik egy nevet, NEPTUN kódot és pontszámot tárol!
- Írj függvényt, amelyik paraméterként kapott nevű fájlból beolvassa ezeket az adatokat egy listába! Ügyelj itt is, hogy **int**-ként tárold a pontszámot.

Hozz létre egy második listát, amelyik ugyanazokat a dolgozat objektumokat tartalmazza, mint az előbbi! Nagyon figyelj arra, hogy ez mit jelent: nem kell beolvasni még egy fájlt, nem kell létrehozni új dolgozat objektumokat, hanem csak egy új listába be kell tenni ugyanazon dolgozatok referenciáit!

A következőképp tudod ellenőrizni, hogy ezt jól csináltad-e:

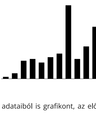
```
lista1 = ...
lista2 = ...

print(lista1 is lista2)                  # False: mert két külön listáról van szó
print(lista1[0] is lista2[0])            # True, ugyanazok az objektumok vannak benne
lista1[0].pontszam = 27                  # Reklámáll , változott a pontszáma
print(lista2[0].pontszam == 27)          # True, mert ugyanaz a dolgozat objektum
```

Rendezd ezek után a két listát, egyiket név szerint, másikat pontszám szerint! Írd ki az eredményt a képernyőre, ellenőrizd az adatokat!

8. ZH eredmények oszlopdiagramon

Dolgozz tovább a kis ZH pontszámos feladattal! Oldd meg teknőgrafikával, hogy egy oszlopdiagramot kapj az eredményekről! Az x tengelyen a pontszámok vannak, az y tengely pedig a létszámot mutatja, hogy hányan érték el annyi pontot. Ehhez hasonló kell legyen az eredmény:



▶ Megoldás

Készíts a „dolgozatok" című feladat adataiból is grafikont, az előzőhöz hasonlóan!