Rádió

# Bevezetés:

# **Tartalom:**

## Főbb modulok:

* Universal\_tools.py
* Show\_management.py
* Show\_time\_management.py
* Playlist.py
* Render.py
* Management.py

## universal\_tools.py

-Ebben a modulban összefoglaltam a főbb, funkciókat, amelyeket, szinte az összes egyéb modulban használunk.

**Funkciók:**

* Ellenörző:
  + check\_file()
  + check\_if\_length\_right()
* Kereső:
  + show\_header()
  + return\_column()
  + search\_for()
* Fájl szerkesztő:
  + write\_full\_csv()
  + add\_row()
  + edit\_row()
  + delete\_row()

## show\_management.py

-Ebben a modulban vannak a műsorvezetőket, és műsoraikat összerakó metódusok

**Klasszok és metódusai:**

* ShowManager()
  + add\_show\_manager()
* Show()
  + add\_show

## show\_time\_management.py

-Ebben a modulban, a műsoridőket tudjuk összerakni

**Klasszok és metódusai:**

* ShowTime()
  + add\_show\_time()

**Funkciók:**

* check\_time\_format()
* check\_time

**Egyéb fontosabb változók:**

* days\_of\_week

## playlist.py

-Ez a modul felelős azért, hogy a zenékből lejátszási listákat lehessen készíteni.

**Klasszok és metódusai:**

* Playlist()
  + random\_music()

## render.py

-ez a modul felelős azért, hogy a megfelelő grafika legyen kiírva konzolba

**Funkciók:**

* render\_table()
* render\_choices()

## management.py

-ez a modul foglalja egybe az összes többi modult, és alkotja eggyé.

**Funkciók:**

* deleting()
* editing()
* listing()
* backing()

**Egyéb fontosabb változók**

* add\_functions
* functions

# **universal\_tools.py**

## check\_file(file,tag)

parametéterek:

* file – a fájl neve, amit ellenőrizni szeretnénk, ha nem létezik akkor készít egyet specifikus fejléccel.
* tags – ha fájlt készít, milyen címkék legyenek a fejlécen.

-Nem ad vissza semmilyen értéket, csak ha a fájl nem létezik, akkor egy specifikus fejléccel(amit megadtunk a tags parameternél) készít egyet.

## check\_if\_length\_right(string,max\_length)

paraméterek:

* string – az a string típusú adat, aminek megszeretnék nézni a hosszát.
* max\_length – mennyi lehet a maximum hossza a stringnek
* A funkció vagy True-t vagy False-t add vissza értékként, attól függően, hogy len(string) <= int(max\_length).

## show\_header(file)

paraméterek:

* file – fájl, aminek ki fogja írni, a fejlécét.

## return\_column(file,column)

paraméterek:

* file – fájl, amelynek vissza fogja adni az egyik oszlopát.
* column – ez egy string, ami lehetőleg az egyik címke a fejlécek között.
* A funkció visszaad egy listát, amiben megtalálható a specifikus oszlop elemeit. Ha a címkét nem találja, vagy az oszlop üres, üres listát ad vissza.

## search\_for(file,search\_tag,specify)

paraméterek:

* file - az a fájl, amelyet végig keres
* search\_tag – mit keressen
* specify – alapból egy üres string, ilyenkor az egész fájlt átkeresi, ha valamit beírsz akkor abban a specifikus oszlopban fog keresni.
* Vissza adja azokat a sorokat, amelyekben a search\_tag, megtalálható. Lista formában adja vissza.

## write\_full\_csv(file,content)

paraméterek:

* file – a neve az új fájlnak, amelyet létrehoz.
* content – mit írjon a sorba. Figyelem: a content egy lista típus kér be, és a sorban a listában lévő elemeket, mindig új sorba fogja írni.

Példa:

.>data = [[’name’,’id’,interest’],[’Bob Ross’,’1’,’Painting’],[’Isaac Asimov’,’2’,’writing’]

.> write\_full\_csv(file,data)

.>’name’,’id’,’interest’

.’Bob Ross’,’1’,’Painting’

.’Isaac Asimov’,’2’,’writing’

## add\_row(file,content)

paraméterek:

* file – fájl, amelybe írni fogja az új sort.
* content – ez egy lista, és a listában lévő elemeket vesszővel elválasztva fogja beírni.

-Mindig az utolsó sorba ír

## edit\_row(file,id,tag,content)

paraméterek:

* file – fájl, amelyben azt a specifikus sort fogja átírni
* id – a sornak az idja (a programban szükséges hogy a soroknak idjük legyen), amit át fog majd írni
* tag – megadja, hogy melyik címke alatt írja át.
* content – mire írja át

-Van pár dolog, amit a fájl éppsége és átláthatósága érdekében, amit nem lehet át írni:

1. A 0.ID, ami a fejléc nem lehet átírni,
2. Az ID-kat sem lehet átírni, így minden ID egy adott fájlban egyedi

## delete\_row(file,id)

paraméterek:

* file – megadja, hogy melyik fájlban töröljük ki az egyik sort.
* id – megadja, hogy melyik idjűt sort törölje ki
* Nem lehet kitörölni a fejléce, azaz a 0 ID-jű elemet.

# **show\_management.py**

* Két fő klasszból áll. ShowManager() és Show(), mindkettőnek van egy-egy metódusa. Ez a modul foglalkozik azzal, hogy hozzá lehessen adni a műsorvezetőket (1.a és 1.b feladat) a megfelelő címkékkel

## Class - ShowManager()

* az objektum készítésekor meg kell adni a fájl nevét, amiben fogunk foglalkozni az adatokkal, ha a fájl nem létezik készít egyet a megfelelő fejléccel (name,conent,picture). Az id-t kiszámlolja a meglévő id-k alapján

Metódusai:

### add\_show\_manager()

* Bekéri a megfelelő adatokat a sor elkészítéséhez, és le ellenőrzi, hogy azok megfelelőek-e (pl. len(name) < 50)

## Class – Show()

* az objektum készítésekor meg kell adni a fájl nevét, amiben fogunk foglalkozni az adatokkal, ha a fájl nem létezik készít egyet a megfelelő fejléccel (title,content,manager\_id).

Metódusai:

### add\_show()

* Bekéri a megfelelő adatokat a sor elkészítéséhez, és le ellenőrzi, hogy azok megfelelőek-e (pl. len(name) < 50)

# **show\_time\_management.py**

* Egy klasszból áll, ShowTime(), ez a modul foglalkozik azzal, hogy hozzá lehessen adni műsoridőket( 1. c feladat)

## Class – ShowTime()

* az objektum készítésekor meg kell adni a fájl nevét, amiben fogunk foglalkozni az adatokkal, ha a fájl nem létezik készít egyet a megfelelő fejléccel (day,start,end,show).

Metódusai:

### add\_show\_time()

* Bekéri a megfelelő adatokat a sor elkészítéséhez, és le ellenőrzi, hogy azok megfelelőek-e (pl. egy nap benne a van a hétben(1-7))

## Egyéb funkciók:

### Check\_time\_format(data)

paraméterek:

* data – string, amelynek megmodja, hogy megfelelő e a formája
* True-t vagy False-t ad vissza, az alapján hogy a string megfelelő formájú- e - óra:perc,

Az óra maximum értéke 23 és a perc maximum értéke 59

### Check\_time(data)

paraméterek:

* data – string, amelynek helyes formája van(óra:perc)
* Vissza ad egy értéket ami az óra és a perc átváltása másodperccé, integer formát ad vissza

## Adat halmazok:

### Days\_of\_week

* Ez egy segéd szótár a add\_show\_time() modulhoz, a számokat amit megadsz át alakítja a hét megfelelő napjává (1 – hétfő, 2 – kedd,..)

# playlist.py

## Class – Playlist()

* az objektum elkészítésekor meg kell adni, hogy melyik fájból, szedje ki a zeneszámokat és később dolgozhasson vele. (2. feladat)

Metódusai:

### random\_music()

* kimutatja a fájlban talált zenék címkéit, mint választható lehetőség, utána kiválaszthatjuk, hogy mennyi zenét választjon ki véletlen szerűen(maximum annyi, amennyi abból a címkéből van a fájlban), és rakja bele egy új fájlba, aminek a neve [címke]-[zenék száma]-[év-hónap-nap-óra-perc] jelölést követi.

# **render.py**

* ez a modul foglalkozik azzal, hogy a parancssorba kiírt szöveg, rendszerezett legyen.

## Funkciói:

### Render\_table(file)

paraméterek:

* file – azt a .csv fájlt kell megadni, amelynek tartalmát szeretnék kiiratni.
* A fejléc alapján az adatokat a .csv fájlból, a címkéket soronkét elválasztva írja ki, egy sorral elválasztja a különböző .csv beli sorokat is.

Példa:

Csvfile.csv

Id,name,interest

1, Bob Ross, painting

2, Isaac Asimov, writing

A render table lefuttatása után

.> render\_table(csvfile)

+-----------------------------

| id: 1

| name: Bob Ross

| interest: painting

+-----------------------------

| id: 2

| name: Isaac Asimov

| interest: writing

+-----------------------------

### render\_choice(choices)

paraméterek:

* Choices – lehetőségek, amiket ki akarunk iratni. Lista formátumban kell megadni
* Kiírás előtt kiszámolja a doboz hosszát, így bármennyi lehetőséget írsz be, a körülötte lévő doboz alkalmazkodni fog hozzá.

Példa:

.> choices\_list = [’add’,’edit’,’delete’,’list’,’exit’]

.> render\_choices(choices)

+------------------------------------------------+

| add | edit | delete | list | exit |

+------------------------------------------------+

# **management.py**

* Ez a modul foglalkozik, hogy az összes modult összekovácsolja. Itt alakítjuk ki a megfelelő objektumokat.

## Funkciók:

### Deleting(file)

paraméterek:

* File – megadja, hogy melyik file-al kell dolgozni.
* Ez a modul összerakja a felhasználó számára az egyik sor törlése funkciót, ki listázza a lehetőségeket és kiválaszthatjuk az id alapján, hogy melyik sort akarjuk kitörölni.

### Editing(file)

paraméterek:

* File – megadja, hogy melyik file-al kell dolgozni.
  + Ez a modul összerakja a felhasználó számára az egyik sor módosítésa funkciót, ki listázza a lehetőségeket és kiválaszthatjuk az id alapján, hogy melyik sort akarjuk módosítani, a tag alapját pedig, hogy melyik címkét akarjuk módosítani, végül a content-tel megadjuk, hogy mire akajuk átírni a kívánt címkét.

FIGYELEM: a fejlécet és a azonosítókat nem lehet átírni

### Listing(file)

paraméterek:

* File – megadja, hogy melyik file-al kell dolgozni.
* Ez a modul összerakja a felhasználó számára a listázás funkciót, ki listázza a fájlban található sorokat a render\_tables() funkció segítségével.

### Back()

* Ezzel a funkcóval lehetőségünk van visszaugorni, egy menüpontot.

### User\_input(choices,object)

paraméterek:

* choices – a választot döntésünk (pl.delete, edit, list..).
* object – hozzá van rendelve, az az objektum, aminek a fálját megnézzük az object.file modul segítségével, ezek mellet a különböző add funkciók végett, itt rendeljük hozzá a megfelelőt add funkciót.
* Ennek a modul segítségével, a felhasználó parancsát át tudjuk alakítani a megfelelő program funkcióvá.

## Adat Halmazok:

### Add\_functions

* Ez egy szótár, ami az add funkciónál vesz rész, megnézi az objektum típusát, és hozzá rendeli a megfelelő add funkciót.

(pl. "<class'show\_managment.ShowManager'>":show.add\_show())

### Functions

* Ez egy szótár, ami az összes többi funkciónál vesz részt, ami nem add. Megnézi a bejövő kulcsot és hozzárendeli a megfelelő funkciót.