À faire individuellement ou par petits groupes de deux ou trois.

Utilisation des *dataclass* – Librairie musicale

Dans cet exercice, nous allons modéliser une librairie musicale. Celle-ci reposera principalement sur deux *dataclasses*: **`Song`** et **`Album`**. Par la suite, cette librairie pourra également utiliser une troisième classe **`Artist`**, que nous n'allons pas implémenter dans cette série d'exercices.

Voici comment ces deux classes seront utilisées :

- **`Song`**:
  - **`title`**: contient le titre de la chanson;
  - **`duration`**: contient la durée de la chanson (en secondes).
- **`Album`**:
  - **`title`**: contient le titre de l'album;
  - **`artist`**: contient le nom de l'artiste;
  - **`year_released`**: contient l'année de sortie de l'album;
  - **`tracks`**: contient une liste de **`Song`**.

Dans la suite de cette exercice, nous allons construire et compléter ces deux classes pour pouvoir afficher des informations sur un album. Voici un exemple d'affichage que l'on souhaiterait obtenir :

```
Titre de l'album: Wish You Were Here
Artiste: Pink Floyd
Année de sortie: 1975
1. Shine On You Crazy Diamond (Parts I-V) (13:40)
2. Welcome to the Machine (7:38)
3. Have a Cigar (5:08)
4. Wish You Were Here (5:34)
5. Shine On You Crazy Diamond (Parts VI-IX) (12:31)
```

Classe `Song`

Voici le début la déclaration de la classe **`Song`**. Cette classe déclare une méthode **`pretty_print()`** qui doit afficher le titre de la chanson ainsi que sa durée en utilisant le format comme indiqué dans l'exemple ci-dessus: **`titre (mm:ss)`**.

```python
from dataclasses import dataclass

@dataclass
class Song:

    title: str
    duration: int

    def pretty_print(self):
        ... #TODO
```

a. Observez bien l'exemple d'affichage donné plus haut. Notez que la chanson `Have a Cigar` a une durée de `5:08` et non `5:8`. À partir de la durée totale de la chanson indiquée en secondes, comment pouvez-vous obtenir cet affichage ?

b. Compléter la définition de la méthode **`pretty_print()`** pour obtenir l'affichage souhaité, peu importe le nombre de minutes et de secondes.

Pour vous aider à tester votre méthode, voici un morceau de code vous permettant de lire le contenu d'un fichier CSV, qui créera autant d'objets de la classe **`Song`**, et qui fera à chaque fois appel à la méthode **`pretty_print()`**.

```python
# À ajouter en haut du fichier
from typing import List


# À ajouter après la déclaration de la classe
songs: List[Song] = []
# Cette ligne nous permet d'avoir une variable `f` qui stocke le contenu du fichier songs.csv
with open('songs.csv', 'r') as f:
    # lines: List[str], où chaque élément de la liste sera une ligne du fichier songs.csv
    lines = f.readlines()
    for line in lines:
        # La méthode split() permet de découper une chaîne de caractère en fonction de la valeur passée en paramètre.
        # Ici, on profite du format CSV pour séparer la ligne en `titre` et `duration`
        title, duration = line.split(',')
        # On ajoute un nouvel objet Song à la liste `songs`
        # Attention à la conversion en `int` de la variable `duration`, qui est de type str
        songs.append(Song(title=title, duration=int(duration)))
for song in songs:
    song.pretty_print()
```

Classe `Album`

La classe **`Album`** doit donc contenir les champs **`title`**, **`artist`**, **`year_released`**, et **`songs`**.

a. Déclarer la classe **`Album`** ainsi que ses champs en précisant également leurs types.

b. Dans la classe **`Album`**, définissez une méthode **`pretty_print()`** qui permette d'obtenir l'affichage proposé en début d'énoncé.
Aide: Vous pouvez utiliser **`print(..., end="")`** pour éviter un retour à la ligne à la fin de l'affichage d'un message en console.

c. Reprenez le programme utilisé pour tester l'affichage d'une chanson et complétez-le pour créer un objet de la classe **`Album`**, puis affichez son contenu en appelant la méthode **`pretty_print()`**.

Bonus

Vous pouvez compléter votre programme en complexifiant la modélisation d'une librairie musicale. En particulier, vous pouvez :

a. Calculer la durée totale d'un album et l'afficher en bas du message de description construit par la méthode **`pretty_print()`** ;

b. Compléter le fichier CSV avec d'autres albums (attention à la séparation entre un album et un autre) ;

c. Rajouter un **`genre`** aux chansons ou aux albums ;

d. Lister les chansons ou albums par genre ;

e. Rajouter une classe **`Artist`** ;

f. …