

# Info 202 - TP n°3

## Gestion de Bibliothèque Universitaire

### Université Savoie Mont Blanc - L1 MISPI - PEIP

## Consignes

### Fichiers

Dans ce TP, nous allons créer plusieurs modules afin de simuler la gestion d'une bibliothèque universitaire.

Au début de chaque fichier, vous ajouterez les lignes suivantes :

```
1 # INFO 202 - TP 3
2 # NOM1 - NOM2
3 # Groupe
```

### Spécification des fonctions

Pour chaque fonction, vous **devez** écrire le type des entrées et sortie.

```
# Question 7
def maFonction(var1, var2,...):
    """ Entrees : Entier, Chaine,...
        Sortie : Tableau de caracteres """
```

Le nom des fonctions, des variables, ainsi que le type des entrées et sortie seront pris en compte dans la notation.

### Rendu du TP

Votre enseignant de TP vous donnera les consignes pour rendre le TP.

## 1 Le module Livre

Pour commencer, nous allons modifier le module livre.

Un livre sera enregistré sous forme de dictionnaire. Les informations nécessaires pour un livre donné sont : le titre, l'auteur, le nombre et si il est emprunté ou non.

Par exemple, le livre "Apprendre à programmer avec Python", sera stocké de la forme suivante. On a besoin des informations suivantes : le titre, le (ou les) auteurs, et d'un booléen permettant de savoir si le livre est emprunté ou non.

```
Livre_exemple = {
    "Titre" : "Apprendre a programmer avec Python",
    "Auteur" : "G. Swinnen",
    "categorie" : "Informatique",
    "identifiant" : 76698}
```

**Question 1** Dans le fichier `livre.py`, créez vos 3 livres préférés, avec ce même format

**Question 2** Écrire 3 fonctions `titre_livre`, `auteur_livre`, `categorie_livre`, qui prennent en entrée un livre et ressortent la donnée voulue

**Question 3** Écrire une fonction `afficher_livre`, qui affichera de manière agréable et lisible pour un utilisateur les informations concernant un livre donné.

```
>>> afficher_livre(Livre_exemple)
Titre du Livre : Apprendre a programmer avec Python,
Auteur : G. Swinnen,
Categorie : Informatique
```

**Question 4** Écrire une fonction `creer_livre`, qui prend en entrée un titre, un auteur et une catégorie, et renvoie un nouveau livre.  
Vous utiliserez la fonction `nouvel_identifiant` pour attribuer un identifiant au livre.

**Question 5** Écrire une fonction `saisir_livre`, qui permet de à l'utilisateur un titre, un auteur et un genre, et renvoie un nouveau livre.  
Vous vous servirez de la fonction `input`.

```
>>> saisir_livre()
Quel est le titre du livre ?
Apprendre a programmer avec Python,
Quel est l auteur de livre ?
G. Swinnen
Quelle est la categorie du livre ?
Informatique
```

## 2 Module bibliothèque

Nous appellerons bibliothèque la liste des livres gérées par la BU.

**Question 6** Créer un fichier `bibliotheque.py`, et y écrire les lignes ci-dessous.

Vérifier que votre variable `Bibliotheque` contient bien des livres

```
from import_livres import import_livres

Bibliotheque = import_livres("listelivres.csv")
```

**Question 7** Écrire une fonction `affiche_liste_livres` qui permet d'afficher la liste de tous les livres de la bibliothèque universitaire, en utilisant la fonction `affiche_livres`.

**Question 8 Ajout d'un livre.** Écrire une fonction d'ajout d'un livre dans la bibliothèque. Cette fonction prend en entrée un livre et ne renvoie rien.

**Question 9 Recherche par titre.** Écrire une fonction `recherche` qui permet de rechercher si un livre est présent dans la bibliothèque. Cette fonction renverra le livre si il est présent, un dictionnaire vide sinon.

**Question 10 Recherche par auteur, par catégorie** Ajouter les fonctionnalités de recherche par auteur, par catégorie . Ces fonctions retournent une liste de livres.

### 3 Module abonnés

**Question 11** Ajouter un module abonnés. Un abonné aura un nom, un prénom et un mail (son identifiant). Créer :

- 3 abonnés
- une variable globale contenant la liste des abonnés
- une fonction `identifiant`, qui prend en entrée un abonné et renvoie son adresse mail
- une fonction `creer_abonne` qui renvoie un dictionnaire du nouvel abonné
- une fonction `saisir_et_ajouter_abonne` qui demande des informations à l'utilisateur du logiciel sur le nouvel abonné

### 4 Module emprunts

Nous allons également gérer les emprunts, dans un nouveau module.

Un emprunt sera un dictionnaire, composé de 5 champs : "id\_livre", "id\_abonne", "date\_emprunt", "retour". Le "retour" sera un booléen, pour savoir si le livre a été retourné.

La liste des livres empruntés sera une variable globale `ListeEmpruntes`.

Au début du fichier, ajoutez la ligne suivante :

```
from datetime import date
```

Vous pouvez alors utiliser :

- `date.today()` pour avoir la date du jour
- `date(year, month, day)` pour créer une nouvelle date avec les paramètres voulus
- les opérateurs classiques de comparaison, `<`, `==`, `>`, `<=`, `=>`, pour comparer deux dates

**Question 12** Écrire une fonction qui affiche la liste des livres actuellement empruntés.

**Question 13** Ajouter les fonctions d'emprunt et de retour de livre.

**Question 14** Ajouter une fonction qui permet de connaître la liste des emprunts d'un utilisateur donné.

### 5 Interface bibliothécaire

En pratique, on va regrouper toutes les fonctionnalités développés ci-dessus dans un nouveau module `interface`, pour le bibliothécaire.

```
>>> gestionBU()
Que souhaitez-vous faire ?
1 - Emprunter un livre
2 - Retourner un livre
3 - Créer un nouveau livre
4 - Créer un nouvel abonné
5 - Rechercher un livre
6 - Afficher la liste des livres empruntés par abonné
7 - Quitter
```

**Question 15** Écrire cette interface de gestion de la BU. Le programme bouclera, exécutera l'action demandé, puis réaffichera le menu, jusqu'à ce que l'utilisateur souhaite quitter le programme.

*Il faudra certainement écrire de nouvelles sous-fonctions.*

## 5.1 Retards

Les utilisateurs ont un délai de 3 semaines pour rendre leurs livres.

**Question 16** Créer une fonction qui renvoie la liste des livres en retard. A vous de choisir la ou les structures de données adéquats.