TP d’Algorithmique

N°10

**Table des matières :**

[**Fonction afficher\_bibliothèque : 1**](#_7d61urfndks5)

[**Fonction afficher\_menu : 2**](#_xp87odg9dqkl)

[**Fonction ajouter\_livre : 4**](#_qc9mjhdrnkw3)

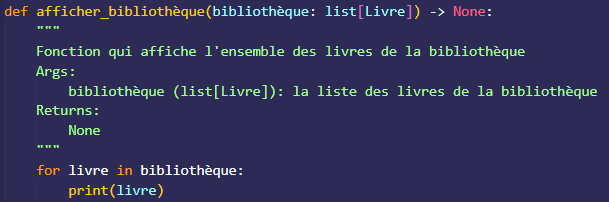
[**Fonction rechercher\_livre : 5**](#_lirwh5lpe8i1)

[**Programme principal : 6**](#_pma0krlso3lb)

[**Type Livre : 8**](#_1rjys1m3ptea)

[**Jeux d’essais : 9**](#_x6eew1yixh72)

# Fonction afficher\_bibliothèque :

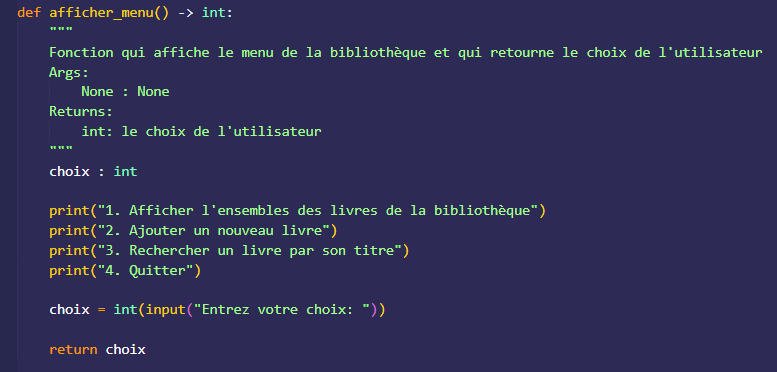


La fonction **afficher\_bibliothèque** est conçue pour afficher tous les livres présents dans une bibliothèque, représentée sous forme de liste d'objets de type **Livre**.

**Détails du code :**

1. **Docstring de la fonction :**
   * Explique :
     + Que la fonction prend en paramètre une liste de livres (**list[Livre]**).
     + Que la fonction affiche les livres et ne retourne rien (**None**).
2. **Paramètre :**
   * **bibliothèque** : une liste d'objets de type **Livre**, où chaque livre contient des informations telles que le titre, l'auteur, l'année de publication, etc.
3. **Traitement :**
   * **Boucle for :**
     + Parcourt chaque livre dans la liste **bibliothèque**.
     + Utilise la méthode **\_\_str\_\_** de la classe **Livre** (déjà définie) pour afficher les informations du livre.
   * **Condition if :**
     + Vérifie si la liste **bibliothèque** est vide en utilisant **len(bibliothèque) == 0**.
     + Si c'est le cas, affiche : **"La bibliothèque est vide"**.
4. **Type de retour :**
   * La fonction n'a pas de valeur de retour explicite. Elle affiche directement les informations.

# Fonction afficher\_menu :



Cette fonction **afficher\_menu** est utilisée pour afficher un menu d'options liées à une bibliothèque, puis récupérer et retourner le choix de l'utilisateur.

**Détails du code :**

1. **Docstring de la fonction :**
   * Elle explique :
     1. Que la fonction affiche un menu.
     2. Qu'elle retourne un entier correspondant au choix de l'utilisateur.
2. **Affichage du menu :**
   * Quatre options sont proposées :
     1. **Afficher l'ensemble des livres de la bibliothèque.**
     2. **Ajouter un nouveau livre.**
     3. **Rechercher un livre par son titre.**
     4. **Quitter.**
3. **Saisie de l'utilisateur :**
   * La fonction demande à l'utilisateur d'entrer un choix en utilisant **input**.
   * La saisie est convertie en entier avec **int()**.
   * Ce choix est ensuite renvoyé par la fonction.
4. **Type de retour :**
   * Le type de retour est un entier (**int**) correspondant au numéro de l'option choisie.

# Fonction ajouter\_livre :

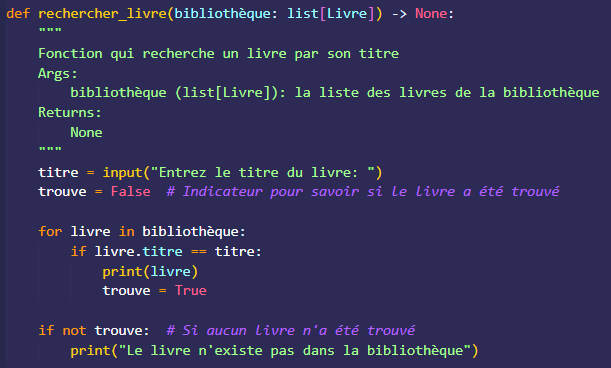


La fonction **ajouter\_livre** permet d'ajouter un nouveau livre à une bibliothèque représentée par une liste d'objets de type **Livre**.

**Détails du code :**

1. **Docstring de la fonction :**
   * Explique que :
     + La fonction prend en entrée une liste de livres (**bibliothèque**).
     + Elle ne retourne rien (**None**) mais modifie la liste en y ajoutant un nouveau livre.
2. **Paramètre :**
   * **bibliothèque** : une liste où chaque élément est un objet **Livre**.
3. **Traitement :**
   * **Demande d'informations à l'utilisateur :**
     + La fonction interagit avec l'utilisateur pour obtenir les détails du livre :
       - **Titre** : récupéré comme une chaîne de caractères.
       - **Auteur** : récupéré comme une chaîne de caractères.
       - **Année de publication** : récupérée comme un entier.
       - **Nombre de pages** : récupéré comme un entier.
   * **Création d'un objet Livre :**
     + Utilise les informations saisies pour créer un nouvel objet de type **Livre**.
   * **Ajout à la bibliothèque :**
     + L'objet **Livre** est ajouté à la liste **bibliothèque** grâce à la méthode **append**.
4. **Type de retour :**
   * La fonction ne retourne pas de valeur, elle modifie directement la liste passée en argument.

# Fonction rechercher\_livre :

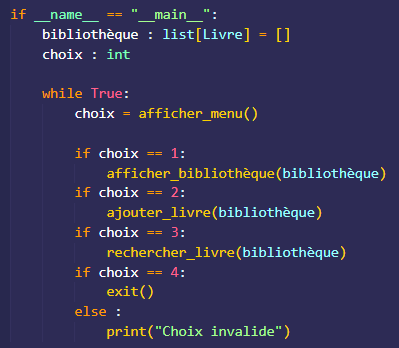


La fonction **rechercher\_livre** permet de chercher un livre dans une bibliothèque (représentée par une liste d'objets de type **Livre**) en se basant sur le **titre** fourni par l'utilisateur. Elle affiche les informations du livre trouvé ou informe que le livre n'existe pas si aucun résultat ne correspond

### 

1. **Demande du titre :**
   * La fonction invite l'utilisateur à entrer le titre du livre qu'il cherche.
2. **Parcours de la bibliothèque :**
   * La fonction utilise une boucle **for** pour examiner chaque livre dans la liste.
3. **Vérification :**
   * Si un livre à un titre qui correspond exactement à celui saisi (**livre.titre == titre**), la fonction affiche les informations du livre en utilisant la méthode **\_\_str\_\_** de l'objet **Livre**.
   * Une variable **trouve** est utilisée pour indiquer qu'un livre a été trouvé.
4. **Message d'erreur (si nécessaire) :**
   * Une fois la boucle terminée, si **trouve** est toujours **False** (aucun livre trouvé), la fonction affiche le message : **"Le livre n'existe pas dans la bibliothèque"**.
5. **Type de retour :**
   * La fonction ne retourne rien (**None**), elle affiche uniquement des messages.

# Programme principal :



Ce code est le **point d'entrée principal** du programme, où toutes les fonctions précédemment définies sont utilisées pour créer une application simple de gestion d'une bibliothèque. Voici une explication étape par étape :

### 

**1. Initialisation :**

* **bibliothèque : list[Livre] = []**
  + Une liste vide est créée pour stocker les livres. Elle représente la bibliothèque.
* **choix : int**
  + Une variable est déclarée pour enregistrer le choix de l'utilisateur.

**2. Boucle principale :**

* Une boucle **while True** est utilisée pour que le programme continue à s'exécuter tant que l'utilisateur ne choisit pas de quitter (option 4).

**3. Affichage du menu :**

* La fonction **afficher\_menu** est appelée pour afficher les options disponibles et récupérer le choix de l'utilisateur.

**4. Traitement des choix :**

* En fonction de la valeur de **choix**, différentes actions sont exécutées :
  + **choix == 1** : Appelle la fonction **afficher\_bibliothèque** pour afficher tous les livres de la bibliothèque.
  + **choix == 2** : Appelle la fonction **ajouter\_livre** pour ajouter un nouveau livre à la bibliothèque.
  + **choix == 3** : Appelle la fonction **rechercher\_livre** pour rechercher un livre par son titre.
  + **choix == 4** : Utilise la fonction **exit()** pour quitter le programme.
  + **Autre choix (par exemple, < 1 ou > 4)** : Afficher le message **"Choix invalide"**.

# Type Livre :

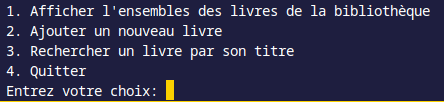


Ce code définit une classe Python appelée **Livre**, qui permet de représenter un livre avec ses principales caractéristiques.

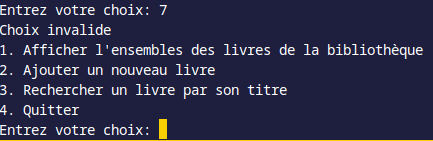
**Détails du code :**

1. **Docstring de la classe :**
   * Une description de la classe et de ses attributs :
     + titre : le titre du livre (chaîne de caractères).
     + auteur : le nom de l'auteur (chaîne de caractères).
     + annee : l'année où le livre a été publié (entier).
     + nbPages : le nombre de pages du livre (entier).
2. **Constructeur (\_\_init\_\_) :**
   * Une méthode spéciale utilisée pour créer un objet de type **Livre**.
   * Elle prend quatre paramètres :
     + **titre, auteur, annee, et nbpages.**
   * Ces paramètres sont utilisés pour initialiser les attributs de l'objet.
3. **Méthode spéciale \_\_str\_\_ :**
   * Permet de définir comment afficher un objet **Livre** sous forme de texte.
   * Retourne une chaîne de caractères formatée contenant les informations du livre (titre, auteur, année, nombre de pages).

# Jeux d’essais :



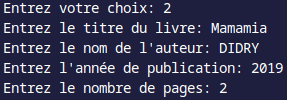
Au lancement du programme , on affiche le menu afin d’indiquer à l’utilisateur les choix qui lui sont possibles.



Si l’utilisateur ne rentre pas un choix valide, cela le ramène sur le menu en lui indiquant qu’il a fait un mauvais choix.



Si l’utilisateur veut afficher la bibliothèque mais qu'elle est vide, cela lui est indiqué.



Quand il veut entrer un nouveau livre, le programme lui demande la saisie de tous les paramètres de la classe Livre.

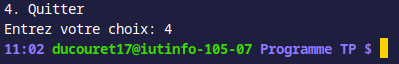


Si il ré-affiche la bibliothèque après y avoir inséré des nouveaux livres, le programme les lui affichera dans l'ordre de saisie.





Enfin la recherche de livre permet à l’utilisateur de rentrer le titre d’un livre ce qui va lui renvoyer soit toutes les informations du livre s'il est présent, sinon un message d’erreur.



Pour finir, si l’utilisateur souhaite quitter le programme, il le peut en sélectionnant le choix 4.

Classe Auteur :

**Objectif :**

La classe **Auteur** représente un auteur littéraire ou une personne qui a écrit des ouvrages, avec des attributs décrivant des informations personnelles.

**Attributs de la classe :**

* **nom** : (*str*) Le nom de l'auteur.
* **prenom** : (*str*) Le prénom de l'auteur.
* **nationalité** : (*str*) La nationalité de l'auteur (exemple : "Française").
* **dateNaissance** : (*int*) L'année de naissance de l'auteur.
* **annee\_deces** : (*int*) L'année de décès de l'auteur. Si l'auteur est vivant, cette valeur est -1

Fonction ajouter\_auteur :

**Objectif :**

Cette fonction permet d'ajouter un nouvel auteur à une bibliothèque, qui est une liste d'objets Auteur.

**Arguments :**

* **bibliothèque\_auteur** : Une liste contenant des instances de la classe Auteur.

**Fonctionnement :**

1. La fonction demande à l'utilisateur d'entrer les informations nécessaires sur un auteur :
   * **Nom**, **prénom**, **nationalité**, **date de naissance**.
   * Une question est posée pour déterminer si l'auteur est décédé :
     + Si oui, on demande l'année de décès.
     + Sinon, la valeur -1 est utilisée pour représenter un auteur encore vivant.
2. Une instance de la classe Auteur est créée à partir des informations collectées.
3. Vérification d'unicité :
   * Si l'auteur n'est pas déjà dans la liste bibliothèque, il est ajouté.
   * Sinon, un message indique que l'auteur est déjà présent.

Modification fonction ajouter\_livre :

**Résumé du fonctionnement**

1. La fonction parcourt tous les auteurs dans une liste et demande à l'utilisateur de confirmer si l'un d'eux est celui recherché.
2. Si aucun auteur ne correspond, l'utilisateur est invité à en ajouter un nouveau.
3. L'auteur trouvé ou ajouté est ensuite stocké dans la variable **auteur**.