

**Exercice 1 :**

1. Définir une classe **Livre** avec les attributs suivants : **titre**, **auteur (Nom complet)**, **prix**.
2. Définir les accesseurs. Un *accesseur* est une méthode qui renvoie la valeur d'un attribut. Par exemple, la méthode **get\_titre** renvoie le titre du livre.
3. Définir un constructeur (**\_\_init\_\_**) permettant d'initialiser les attributs de la méthode par des valeurs saisies par l'utilisateur.
4. Définir la méthode **afficher** créant une chaîne de caractère à partir des informations du livre en cours.
5. Écrire un programme qui :
  - instancie la classe **Livre** avec les informations d'un livre quelconque,
  - affiche le titre,
  - affiche les informations du livre.

**Exercice 2 :**

1. Définir une classe **Rectangle** avec les attributs **longueur** et **largeur**.
2. Définir les accesseurs.
3. Définir les *mutateurs*. Un mutateur est une méthode qui modifie la valeur d'un attribut. Par exemple, la méthode **set\_longueur** modifie la valeur de l'attribut **longueur**.
4. Ajouter les méthodes suivantes :
  - **perimetre** : renvoie le périmètre du rectangle (nombre flottant).
  - **aire** renvoie l'aire du rectangle (nombre flottant).
  - **est\_carre** : renvoie **True** si le rectangle est un carré, **False** sinon.
5. Écrire un programme qui :
  - crée un rectangle de dimensions 5.3 et 2.8,
  - affiche le périmètre et l'aire de ce rectangle,
  - vérifie si c'est un carré,
  - modifie la largeur.

**Exercice 3 :** En mathématiques un nombre complexe est défini par :

- sa partie réelle  $a$ ,
- sa partie imaginaire  $b$

et tel que :

$$z = a + b \times i \text{ où } i^2 = -1$$

$a$  et  $b$  sont des nombres réels.

Par exemple, le nombre complexe  $z = 2.5 + 3.1i$  a pour partie réelle 2.5 et pour partie imaginaire 3.1. L'addition de deux complexes  $z_1$  et  $z_2$  s'effectue tel que :

$$z_1 + z_2 = (a_1 + a_2) + (b_1 + b_2) \times i$$

1. Écrire une classe **Complexe** permettant de définir un nombre complexe.
2. Écrire la méthode **afficher** qui renvoie une chaîne de caractère de la forme  $a + b * i$ .
3. Écrire la méthode **addition** qui ajoute (sans le modifier) au nombre en cours, un nombre complexe passé en argument et retourne le nombre complexe obtenu sous forme d'un tuple **réel**, **imaginaire**.

- Écrire la méthode **soustraction** qui soustrait (sans le modifier) au nombre en cours un nombre complexe passé en argument et retourne le nombre complexe obtenu sous forme d'un tuple **réel**, **imaginaire**.
- Écrire un programme permettant de tester la classe **Complexe**.

**Exercice 4 :** On définit une classe **Date** pour représenter une date avec trois nombres entiers pour attributs : **jour**, **mois**, **annee**.

- Écrire son constructeur.
- Écrire la méthode **afficher** qui renvoie une chaîne de la forme *8 mai 1945*.
- Écrire la méthode **est\_avant** qui prend une **Date** pour paramètre et renvoie **True** si la date en cours est plus ancienne que celle passée en paramètre.
- Écrire un programme qui teste cette classe.

**Exercice 5 :** Le *loto* est un jeu de hasard. Chaque tirage est composé de 6 entiers distincts compris entre 1 et 49 et d'un entier complémentaire (également distinct) compris entre 1 et 49.

- Définir une classe **Loto**. Cette classe possède un attribut **numeros** de type **list** et un attribut **complementaire**.
- Écrire la méthode **afficher** qui renverra une chaîne de caractères des numéros du loto de la forme : *1 - 2 - 3 - 4 - 5 - 6 / 7*.
- Écrire la méthode **est\_present** qui vérifiera si le numéro donné en argument est présente dans les 6 numéros du loto.
- Écrire la méthode **est\_gagnant** qui possède deux paramètres : une liste d'entiers et un entier. Elle renverra **True** si le tirage correspond exactement à la proposition.
- Écrire une *fonction* **creer\_tirage** qui renvoie un objet **Loto** avec des entiers tirés au hasard.

### Information

La méthode **seed** de la bibliothèque **random** initialise le générateur de nombres aléatoires. En phase de test le code 1 permet de fixer les entiers tirés au sort.

```
1 from random import seed
2 seed(1)
```

Code 1 – initialise le générateur

- Placer le code 1 en début de programme puis tester la fonction **creer\_tirage**. Observer les tirages obtenus.
- Tester alors la méthode **est\_gagnant**.