TP de Programmation avancée en Python : Série 1

Exercice 1 : Classe rectangle.

1. Écrire une **classe Rectangle** en langage Python, permettant de construire un rectangle dotée d'attributs **longueur** et **largeur.**
2. Créer une méthode **Perimetre()** permettant de calculer le périmètre du rectangle et une méthode **Surface()** permettant de calculer la surface du rectangle

Exercice 2 : Classe Compte bancaire

1. Créer une classe Python nommée **CompteBancaire** qui représente un compte bancaire, ayant pour attributs : numeroCompte (type numérique) , nom (nom du propriétaire du compte du type chaine), solde.
2. Créer un constructeur ayant comme paramètres : numeroCompte, nom, solde.
3. Créer une méthode **Versement()** qui gère les versements.
4. Créer une méthode **Retrait()** qui gère les retraits.
5. Créer une méthode **Agios()** permettant d'appliquer les agios à un pourcentage de 5 % du solde
6. Créer une méthode **afficher()** permettant d'afficher les détails sur le compte
7. Donner le code complet de la classe **CompteBancaire**.

Exercice 3 : Cercle

1. Définir une classe Cercle permettant de créer un cercle **C(O,r)** de centre **O(a,b)** et de rayon r à l'aide du constructeur :

```
def __init__(self , a , b ,
    self.a = a
    self.b = b
    self.r = r
```

2. Définir une méthode **Surface()** de la classe qui permet de calculer la surface du cercle
3. Définir une méthode **Perimetre()** de la classe qui permet de calculer le périmètre du cercle
4. Définir une méthode **testAppartenance()** de la classe qui permet de tester si un point **A(x,y)** appartient ou non au cercle **C(O,r)**.

Exercice 4 : Classe Calcul arithmétique

1. Créer une classe Calcul ayant un constructeur par défaut (sans paramètres) permettant d'effectuer différents calculs sur les nombres entiers.
2. Créer au sein de la classe Calcul une méthode nommée **Factorielle()** qui permet de calculer le factorielle d'un entier. Tester la méthode en faisant une instanciation sur la classe.
3. Créer au sein de la classe Calcul une méthode nommée **Somme()** permettant de calculer la somme des n premiers entiers: 1 + 2 + 3 + .. + n. Tester la méthode.
4. Créer au sein de la classe Calcul une méthode nommée **testPrim()** permettant de tester la primalité d'un entier donné. Tester la méthode.

5. Créer au sein de la classe Calcul une méthode nommée **testPrims()** permettant de tester si deux nombres sont premier entre eux.
6. Créer une méthode **tableMult()** qui crée et affiche la table de multiplication d'un entier donné. Créer ensuite une méthode **allTablesMult()** permettant d'afficher toutes les tables de multiplications des entiers 1, 2, 3, ..., 9.
7. Créer une méthode **listDiv()** qui récupère tous les diviseurs d'un entier donné sur une liste Ldiv. Créer une autre méthode **listDivPrim()** qui récupère tous les diviseurs premiers d'un entier donné

Exercice 5 : Classe myString

Coder une classe **myString** permettant de doter les chaines de caractères de la méthode **append()** faisant la même opération que celle des listes. Exemple si on crée des chaines via l'instanciation **s1 = myString("Hello")** et **s2 = " World !"**, et on lui applique la méthode :

```
1 | print(s1.append(" world !")) # affiche  'Hello world !'
```

Exercice 6 : Classe Book

1. Définir une classe Book avec les attributs suivants : Title, Author (Nom complet), Price.
2. Définir un constructeur ayant comme attributs: Title, Author, Price.
3. Définir la méthode **View()** pour afficher les informations d'une instance object Book.
4. Ecrire un programme pour tester la classe Book.

Exercice 7 : Classe Geometry

1. Ecrire une classe Python nommée **Geometry** avec un **constructeur** par défaut **sans paramètres**.
2. Ajouter une **méthode** nommée **distance()** à la classe geometry qui permet de calculer la distance entre deux points **A = (a1, a2), B = (b1, b2)** (avec la convention: un point est est identifié à ses coordonnées **M = (xM , yM))**
3. Ajouter une **méthode** nommée **middle()** à la classe geometry qui permet de déterminer le milieu d'un bipoint (A , B).
4. Ajouter une **méthode** nommée **trianglePerimeter()** à la classe geometry qui permet de calculer le périmètre d'un triangle ABC.
5. Ajouter une **méthode** nommée **triangleIsocel()** qui renvoie True si le triangle est isocel et False sinon.

Exercice 8 : Classe Student

Créez une classe Python nommée "Student" avec les caractéristiques suivantes :

1. Un attribut "nom" : de type chaine de caractère string.
2. Des attributs "note_semestre1" et "note_semestre2" : de type réel (float).
3. Une méthode "**moyenne()**" : qui calcule la moyenne de l'étudiant.
4. Une méthode "**result()**" : qui renvoie 'Admis' ou 'Non admis' en fonction de la moyenne.
5. Une méthode "**afficher()**" : qui affiche le nom, la moyenne et le résultat de l'étudiant.
6. Créez deux instances de la classe "Student" : pour représenter un étudiant admis et un étudiant non admis.