Créer des objets en Python – Exercices

1 Compétences ciblées

Savoir :

1. écrire et lire un fichier
2. créer un module
3. créer et utiliser une classe

2 Créez des modules

1. Écrire un module de calcul des racines du trinôme réel : $ax^2 + bx + c$. Le module définit une fonction `trinome` avec trois paramètres du trinôme, a, b, et c. La fonction doit retourner une tuple dont le premier élément est le nombre de racines du trinôme $(0, 1, \text{ou } 2)$, et les autres éléments sont les racines éventuelles.

   Testez votre fonction avec trois jeux de valeurs suivantes : $1, -3, 2, 1$; $1, -2, 1$ et $1, 1, 1$.
2. Écrivez un programme principal utilisant le module précédent. Les trois paramètres seront ainsi saisis dans une `flotbox` du module `easyguiB` et les résultats seront affichés dans une `msgbox`.

3 Manipulez un fichier

1. Créez un fichier `NbLigne` qui a pour paramètre un *nom de fichier existant* (.txt) et qui renvoie le nombre de lignes de ce fichier.
2. Créez un fichier texte nommé `La_mesure_de_lhomme.txt` et écrivez la chaîne de caractères suivante dedans :

   "La mesure de l'homme.

   Ce n est pas celui qui critique qui est important, ni celui qui montre du doigt comment l homme fort trébuche ou comment l homme d action aurait pu faire mieux.
   L hommage est dû à celui qui se bat dans l arène, dont le visage est couvert de poussière et de sueur, qui va de l avant vaillamment, qui commet des erreurs et

en commettra encore, car il n y a pas d efforts humains sans erreurs et imperfections. C est à lui ou à elle qu appartient l hommage, à celui u à celle dont l enthousiasme et la dévotion sont grands, à celui ou à celle qui se consume pour une cause importante, à celui ou à celle qui, au mieux, connaîtra le triomphe du succès, et au pis, s il échoue, saura qu il a échoué alors qu il risquait courageusement."

3. Écrivez le contenu du fichier `Demi.txt` (créer et éditer au préalable) dans le fichier `La_mesure_de_lhomme.txt`.
4. Affichiez le contenu du fichier `La_mesure_de_lhomme.txt`.

4 Créez et utilisez une classe

1. Définissez la class `MaClasse` possédant les attributs suivants :
   — données : deux attributs de classes : $x = 23$ et $y = x + 5$.
   — méthode : une méthode `affiche` contenant un attribut d'instance $z = 42$ et les affichages de y et de z.
   Dans le programme principal, instanciez un objet de la classe `MaClasse` et invoquez la méthode `affiche`.
2. Définir une classe `Vecteur2D` avec un constructeur fournissant les coordonnées par défaut d'un vecteur du plan (par exemple : $x = 0$ et $y = 0$). Dans le programme principal, instanciez un `Vecteur2D` sans paramètre, un `Vecteur2D` avec ses deux paramètres, et affichez-les.
3. Enrichissez la classe `Vecteur2D` précédent en lui ajoutant une méthode d'affichage et une méthode de surchage d'addition de deux vecteurs du plan. Dans le programme principal, instanciez deux `Vecteur2D`, affichez-les et affichez leur somme.
4. Définissez une classe `Rectangle` avec un constructeur donnant les valeurs (`longueur` et `largeur`) par défaut et un attribut `nom = "rectangle"`, une méthode d'affichage et une méthode `surface` renvoyant la surface d'une instance.
   — Définissez une classe `Carre` héritant de `Rectangle` et qui surcharge l'attribut d'instance : `nom = "carré"`. Dans le programme principal, instanciez un `Rectangle` et un `Carre` et affichez-les.
   — Définissez une classe `Point` avec un constructeur fournissant les coordonnées par défaut d'un point du plan (par exemple : `x=0.0 et y = 0.0`).
5. Définissez une classe `Segment` dont le constructeur possède quatre paramètres : deux pour l'origine et deux pour l'extrémité. Ce constructeur définit deux attributs : `orig` et `extrem`, instances de la classe `Point`. De cette manière, vous concevez une classe *composite* : la classe `Segment` est composée de deux instances de la classe `Point`.
   (a) Ajouter une méthode d'affichage.
   (b) Enfin écrivez un auto-test qui affiche une instance de `Segment` initialisée par les valeurs 1, 2, 3 et 4.