FACULTÉ DES SCIENCES DE TUNIS
ANNÉE UNIVERSITAIRE 2024-2025
FILIÈRE : DEUXIÈME ANNÉE DU CYCLE PRÉPARATOIRE INTÉGRÉ - MI2

TP 6 : La Programmation Orientée Objet en PYTHON

Exercice 1 Classe Polynôme

1. Écrire une classe *Polynome* définissant un polynôme par la liste de ses coefficients. Par exemple, le polynôme $P = 5 + 12X - 9X^3 + 7X^5$ est représenté par la liste [5,12,0,-9,0,7]. Pour cela, écrire :
   (a) une méthode constructeur définissant l'attribut *coefs* représentant la liste des coefficients et un attribut *degre* représentant le degré du polynôme. On supposera que pour un polynôme non nul, la liste de ses coefficients se termine par le coefficient dominant du polynôme. Le polynôme nul est représenté par la liste [0] et son degré est égale à -1.
   (b) une méthode de représentation du polynôme. Par exemple, pour le cas du polynôme P, on affichera $5 * X * *0 + 12 * X * *1 + 0 * X * *2 + (-9) * X * *3 + 0 * X * *4 + 7 * X * *5$.
   (c) une méthode *coeff* permettant d'obtenir le coefficient du degré i du polynôme. Remplacer le nom de la méthode par __getitem__.
   NB : Si i>$degre$ du polynôme, la fonction retourne 0.
   (d) une méthode *eval* permettant d'évaluer la valeur du polynôme en un point. Remplacer le nom de la méthode par __call__.
   (e) une méthode *trace* qui accepte comme paramètres $self$, $xmin$ et $xmax$ tel que l'appel P.trace(x0,x1) permet de tracer la courbe de P sur l'intervalle [x0,x1] en utilisant 1000 points. On utilisera la méthode *eval* et on n'oubliera pas d'importer les modules nécessaires.
   (f) une méthode *addition* pour effectuer la somme de deux polynômes. Remplacer le nom de la méthode par __add__.
2. Écrire une sous-classe *Monome* de la classe *Polynome* en gardant les mêmes attributs que la classe Polynome, à savoir *coefs* pour la liste des coefficients et *degre*. Pour cela, écrire :
   (a) un constructeur acceptant comme paramètres self, le coefficient et le degré du monôme. Par exemple l'instruction suivante $M = Monome(2, 3)$ permettra de créer le monôme $M = 2X^3$.
   (b) Redéfinir la méthode de représentation pour l'affichage d'un monôme.
   (c) Redéfinir la méthode *eval* pour l'évaluation d'un monôme.

Exercice 2 Classe Polygone

1. Définir une classe Point avec :
   — un constructeur __init__(self, x, y, nom) permettant de définir deux attributs x et y représentant les coordonnées et un attribut nom correspondant au nom du point.
   — la méthode de transformation en chaîne de caractères __repr__(self).
   — la méthode dessiner(self) qui permet d'afficher un point dans un repère en utilisant le module matplotlib.pyplot

     NB : On utilisera la fonction annotate('etiquette',(x,y)) du sous-module pyplot de matplotlib pour annoter le point avec la chaîne 'etiquette' au coordonnées (x,y).
   — la méthode milieu(self, B) retournant le milieu du segment reliant le point self au point B.
2. Définir une classe Polygone comportant :
   — la méthode __init__(self, sommets) qui permet de construire un polygone à partir de la liste de ses sommets. L'attribut $sommets$ est la liste contenant les sommets du polygone (liste de points).
   — la méthode aire(self) qui calcule la surface du polygone.

     Indication : L'aire S d'un polygone ayant les sommets (x_0,y_0), (x_1 ,y_1),... (x_{n-1},y_{n-1}) est défini comme suit :

     $$S = \frac{1}{2}|(x_0+x_1)(y_0-y_1)+(x_1+x_2)(y_1-y_2)+...+(x_{n-2}+x_{n-1})(y_{n-2}-y_{n-1})+(x_{n-1}+x_0)(y_{n-1}-y_0)|$$

   — la méthode __repr__(self) permettant d'afficher l'expression textuelle du polygone. Par exemple : "polygone [A(1.0,1.0),B(5.0,1.0),C(3.0,2.0)]"
   — la méthode dessiner(self) qui permet de dessiner un polygone dans un repère en utilisant le module matplotlib.pyplot (les sommets du polygone doivent être annotés).
3. Définir une classe Triangle, sous-classe de Polygone, avec un constructeur __init__(self,a,b,c) ayant les trois sommets indiqués.
4. Définir une classe Rectangle, sous-classe de Polygone, munie d'un constructeur __init__(self, xMin, xMax, yMin, yMax) qui permet de construire un rectangle à partir des 4 points de coordonnées (xMin,yMin), (xMax,yMin), (xMax,yMax) et (xMin,yMax).
5. Définir une classe PolygoneRegulier, sous-classe de Polygone, disposant d'un constructeur __init__(self, c, n) permettant de construire un polygone régulier de centre c et constitué de n sommets.

   Indication : Un polygone régulier centré autour du point (0,0) est construit à partir de la liste des n points de coordonnées : $(\cos(\frac{2k\pi}{n}), \sin(\frac{2k\pi}{n}))$ pour k de 0 à n-1.
6. Écrire un programme de test construisant et affichant des polygones de diverses sortes.