TD/TP 2 : Classes et encapsulation

1. Disque

Un disque est caractérisé par un **diametre**

a. Écrire une classe **Disque** qui permet de calculer et d'afficher les caractéristiques d'un disque et qui est définie par le diagramme de classe suivant :

Disque
- diametre : float
+ Disque () + Disque (float diametre) + Disque (Disque D) + getDiametre () : float + setDiametre (float diametre) : void + perimetre () : float + surface () : float + rayon () : float + afficher () : void

Cette classe possède :

- ✓ Une variable membre privé **diametre**
- ✓ Des constructeurs
- ✓ Une méthode modificatrice de diamètre
- ✓ Une méthode sélectrice pour retourner le diamètre
- ✓ Des méthodes pour calculer le périmètre, la surface et le rayon
- ✓ Une méthode pour afficher les **caractéristiques** (diamètre, rayon, périmètre et la surface) du disque

b. Écrire une classe de test qui permet de :

- ✓ Créer 2 objets d1 et d2 de type Disque de diamètres d1(2.5), d2(5.6).
- ✓ D'afficher les caractéristiques des 2 disques.

2. Classe Point3D :

Un point dans l'espace est définie par les coordonnées ***(x,y,z)***

a. Écrire une classe **Point3D** qui permet d'utiliser des points dans l'espace et qui est définie par le diagramme de classe suivant :

Point3D
- x : double - y : double - z : double
+ Point3D () + Point3D (double a, double b, double c) + Point3D (Point3D p) + getX () : double + getY () : double + getZ () : double + setPoint (double a, double b, double c) : void + deplacer (double dx, double dy, double dz) : void + distance () : double + distance (Point3D p) : double + egal (Point3D p) : boolean + afficher () : void

- ✓ La méthode distance permet de calculer la distance du point par rapport à l'origine ***O (0,0,0)***
- ✓ La méthode distance (**Point3D**) permet de calculer la distance du point par rapport à un autre point reçu comme paramètre.
- ✓ La méthode ***egal*** permet de retourner ***true*** si le point est égal au point reçu comme paramètre.

Formule :

La distance euclidienne entre deux points *P1(x1, y1, z1)* et *P2(x2, y2, z2)* est définie par l'expression $d(P1, P2) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2 + (z1 - z2)^2}$

b. Ecrire une classe de test qui permet de :
- ✓ Créer 2 objets p1 et p2 de type Point3D de coordonnées *p1(2,5,7), p2(5,9,16)*.
- ✓ D'afficher les caractéristiques des 2 points, ainsi que la distance qui les sépare.

3. Classe Sphere

a. Écrire une classe **Sphere** qui permet d'afficher les caractéristiques d'une sphère et qui est définie par le diagramme de classe suivant :

Sphere	
- centre : Point3D - rayon : double	
+ Sphere () + Sphere (double x, double y, double z, double r) + Sphere (Point3D p, double r) + getRayon () + getCentre () + setShere (Point3D p, double r) + surface () + volume () + afficher ()	 : double : Point3D : void : double : double : void

$$S = 4\pi R^2$$
$$V = \frac{4}{3}\pi R^3$$
Où R est le rayon de la sphère

b. Écrire une classe de test qui permet de :
- ✓ Créer un objet de type Sphere de rayon égale à 10 et de coordonnées pour le centre *(2,5,7)*
- ✓ D'afficher ses caractéristiques

4. Segment

On dispose de la classe Point3D pour manipuler les points dans l'espace :

a- Ecrire une classe Segment3D permettant de manipuler des segments dans l'espace. Un segment sera caractérisé par deux points : un point d'origine et un point d'extrémité.

Le diagramme de classe est le suivant :

Segment3D	
- origine : Point3D - extremite : Point3D	
+ Segment3D (double a1, double b1, double c1, double a2, double b2, double c2) + Segment3D (Point3D o, Point3D e) + longueur () + getOrigine () + getExtremite () + setSegment3D (Point3D o, Point3D e) + afficher ()	 : double : Point3D : Point3D : void : void

b- Écrire une classe de test qui permet de :
- ✓ Créer un segment s1 de caractéristiques : origine(2.5, 4, 3), extremite(13, 5.6, 2).
- ✓ D'afficher la longueur du segment.