Exercice 62

Énoncé

Réaliser une classe `point` permettant de manipuler un point d'un plan. On prévoira :

- un constructeur recevant en arguments les coordonnées (`float`) d'un point ;
- une fonction membre `deplace` effectuant une translation définie par ses deux arguments (`float`) ;
- une fonction membre `affiche` se contentant d'afficher les coordonnées cartésiennes du point.

Les coordonnées du point seront des membres donnée privés.

On écrira séparément :

- un fichier source constituant la `déclaration` de la classe ;
- un fichier source correspondant à sa `définition`.

Écrire, par ailleurs, un petit programme d'essai (`main`) déclarant un point, l'affichant, le déplaçant et l'affichant à nouveau.

Exercice 63

Énoncé

Réaliser une classe `point`, analogue à la précédente, mais ne comportant pas de fonction `affiche`. Pour respecter le principe d'encapsulation des données, prévoir deux fonctions membre publiques (nommées `abscisse` et `ordonnee`) fournissant en retour l'abscisse et l'ordonnée d'un point. Adapter le petit programme d'essai précédent pour qu'il fonctionne avec cette nouvelle classe.

Exercice 64

Énoncé

Ajouter à la classe précédente (comportant un constructeur et trois fonctions membre `deplace`, `abscisse` et `ordonnee`) de nouvelles fonctions membre :

- `homothetie` qui effectue une homothétie dont le rapport est fourni en argumentþ;
- `rotation` qui effectue une rotation dont l'angle est fourni en argumentþ;
- `rho` et `theta` qui fournissent en retour les **coordonnées polaires** du point.

Exercice 65

Énoncé

Modifier la classe `point` précédente, de manière que les données (privées) soient maintenant les coordonnées polaires d'un point, et non plus ses coordonnées cartésiennes. On évitera de modifier la déclaration des membres publics, de sorte que l'interface de la classe (ce qui est visible pour l'utilisateur) ne change pas.

Exercice 66

Énoncé

Soit la classe `point` créée dans l'exercice 62, dont la déclaration était la suivante :

```
class point
{  float x, y ;
  public :
   point (float, float) ;
   void deplace (float, float) ;
   void affiche () ;
}
```

Adapter cette classe, de manière que la fonction membre `affiche` fournisse, en plus des coordonnées du point, le nombre d'objets de type `point`.