TP-2 POO en C++ (2ème Séance)

Ce TP aborde les points suivants :
- Définition simple d'une classe, Encapsulation, Allocation dynamique,
- Utilisation basique de la librairie d'entrées/sorties
- Constructeurs et destructeur, Listes d'initialisations, Surcharge de fonctions

Chapitre 3,4 du Polycopie : Les fonctions C++, Classes et Objets

Exercice 1

On souhaite réaliser une classe vecteur 3d permettant de manipuler des vecteurs à trois composantes. On prévoit que sa déclaration se présente ainsi :

```
class Vecteur3d {... double _x,  _y, _z ; ... } ;
```

On souhaite pouvoir déclarer un vecteur soit en fournissant explicitement ses trois composantes, soit en fournissant aucune, auquel cas le vecteur créé possédera trois composantes nulles. Il peut être aussi constant ou non constant ! Définir constructeur, destructeur, fonction d'affichage pour cette classe. Pensez à afficher dans le constructeur et le destructeur (cout<< "Je suis dans le … "<<endl ;)

1) Définir le ou les constructeurs correspondants en utilisant :

1.a Deux fonctions (constructeurs) membre sur-définies,

1.b Une seule fonction membre qui remplace les deux fonctions de 1.a.

1.c Une seule fonction « en ligne » qui remplace 1.b

1.e Une seule fonction avec la liste d'initialisation (corps de la fonction vide ! ! !) qui remplace 1.c.

2) Définir le constructeur de copie pour cette classe.

Exercice 2

Recopier le premier exemple du cours (Pile d'entier) et le préparer. Compléter le programme en ajoutant un fichier principal qui contient le «main». Corriger les erreurs. **Ajouter** une donnée et une fonction permettant de retourner le nombre d'objets piles créées. Exécuter et Valider.

Remarque. Il faut créer un fichier d'en-tête ***(pile.h)***, un fichier de détail ***(pile.cpp)*** et un fichier principal contient le main ***(testpile.cpp)***

Exercice 3

Réaliser une classe «***Point***» permettant de manipuler un point du plan. On prévoira : Un constructeur avec argument les coordonnées en «***Double***» d'un point, valeur par défaut «**0,0**»; Une méthode «***translate()***» effectuant une translation définie par ses deux arguments (double) ; Deux méthodes «***abscisse()*** et ***ordonnée()***» fournissant abscisse et ordonnée d'un objet point, Deux méthodes membres «***rho()*** et ***teta()***» fournissant les coordonnées polaires du point ; Une méthode membre «***rotation()***» qui effectue une rotation dont l'angle est donné en argument. On écrira séparément : (un fichier d'en-tête « ***point.h*** » constituant la déclaration de la classe ; un fichier source «***point.cpp***» constituant la définition des méthodes de cette classe;)

Un programme d'essai main «***mainpoint.cpp***» déclarant un point, affichant ses coordonnées, le déplaçant ou le faisant subir une rotation et l'affichant à nouveau.