ECOLE DES HAUTES ETUDES D'INGENIERIE D'OUJDA

ANNEE DE FORMATION : 2019– 2020

GENIE INFORMATIQUE: 3EME ANNEE

Unité de formation : Langage C++ TP12 Formateur : A.Serghini

Exercice :

I-On considère une classe vecteur3d permettant de manipuler des vecteurs de trois composantes. On prévoit que sa déclaration se présente comme suit:

```
Class vecteur3d
  { char*Nom   // Nom d'un vecteur
    float x, y, z ;    // les trois composantes (cartésiennes)
   };
```

1- On souhaite pouvoir déclarer un vecteur, soit en fournissant explicitement ses quatre composantes, soit en fournissant aucune, auquel cas le vecteur créé possédera quatre composantes nulles.
Ecrire le ou les constructeur(s) correspondant(s):

- En utilisant des constructeurs surchargés
- En utilisant un seul constructeur.
- En utilisant un seul constructeur en ligne.

2- Introduire une fonction membre nommée sym_vec permettant d'obtenir la symétrie d'un vecteur.

- En utilisant un retour par valeur,
- En utilisant un retour par référence,
- En utilisant un retour par adresse.

N.B Le nom du vecteur symétrique est obtenu en concaténant le nom du vecteur avec ''sym''

3- Introduire une fonction membre nommée coïncide permettant de savoir si deux vecteurs sont égaux:

- En utilisant une transmission par valeur,
- En utilisant une transmission par adresse,
- En utilisant une transmission par référence.
- En utilisant une fonction constante et une transmission par référence constante.

4- Définir le destructeur associé.
5- Implémenter une méthode d'affichage : void affiche() const
6- Définir le constructeur par recopie associé à cette classe.
7- Surcharger les opérateurs =, == et >>.
8- Ajouter une fonction membre de la classe **Vecteur3d** qui retourne le vecteur produit d'un vecteur et d'un réel (donne une signification à **P2 = P1 * h; le nom reste inchangeable**)
9- Ajouter une fonction AMIE de la classe **Vecteur3d** qui retourne le vecteur produit d'un réel et d'un vecteur (donne une signification à **P2 = h * P1;**)
10- Pourquoi faut-il déclarer cette dernière fonction AMIE de la classe **Vecteur3d**.

II- On appellera *dimension* d'une **pile de Vecteur3D** le nombre maximal de vecteurs qu'elle peut contenir et *taille* d'une pile le nombre de vecteurs qu'elle contient réellement. Le tableau représentant la pile est donc indexé de 0 (bas de la pile) à *taille-1* (haut de la pile). Le vecteur que l'on peut dépiler est donc dans la case d'indice *taille-1*.

Ecrire le fichier Pile.cpp correspondant au fichier Pile.h suivant :

```
#ifndef Pile H
#define Pile H
class Pile
{ public :
        Pile(int t = 10); //Constructeur qui construit une pile de dimension t (10 par defaut)
        ~Pile(); //Destructeur
        void empile(Vecteur3D); //empile n en haut de la pile
        void depile(); //depile le sommet de la pile
        bool vide() const; //teste si la pile est vide
        bool pleine() const; //teste si la pile est pleine
        int donnetaille() const; //renvoie la taille de la pile
        Pile(const &Pile) ;//constructeur par recopie
      Pile& operator=(const & Pile)  // surcharge de l'operateur d'affectation
private :
        int dim;
        int taille;
        Vecteur3D *adr;
};
#endif
```