Université Paris Diderot **Langages à objets avancés**
M1 **Année 2019-2020**

TP n° 7 : surcharge d'opérateurs et héritage de classes - suite

Exercice 1

1. Définir une classe `Vecteur` qui définie un vecteur de $\mathbb{R}^3$ avec ses constructeurs.
2. Surcharger les opérateurs `==` et `!=`, afin de pouvoir tester si deux vecteurs sont égaux ou différents.
3. Surcharger l'opérateur `+` afin de pouvoir faire la somme de deux vecteurs et l'opérateur `*` qui permet de réaliser le produit scalaire de deux vecteurs.
4. Surcharger l'opérateur `[]` afin de pouvoir accéder à une coordonnées du vecteur. Tester en particulier :

```
Vecteur v;
v[0] = 9;
```

5. Faites en sorte que l'on puisse afficher un vecteur `v` en invoquant l'instruction `cout << v << endl`.

Exercice 2 Créez une classe A qui contient un champ de type B, et une classe B qui contient un champ de type A. Vous utiliserez d'abord des pointeurs, des références et des variables "normales". Dans quels cas devez vous faire une declaration anticipée ? Quels sont les `include` nécessaires ?

Exercice 3 (const et pointeurs)

On considère la fonction

```
void f(int * x, int  *y){
    *x=0;
    x++;
    y++;
    *y=0;
}
```

1. Modifiez votre fonction pour que le premier argument soit de type pointeur constant vers un entier, et que le second soit de type pointeur vers un entier constant. Quelles instructions deviennent interdites ? Discutez, en particulier, le cas de `y`. On note `f2` ce qui reste de `f` lorsqu'on a supprimé les instructions interdites.
2. Si `t` est un tableau de deux entiers, lorsqu'on appelle `f2(t,t)` qu'est ce qui est réellement constant ?

Exercice 4 En 2112, un voyage sur Alpha du Centaure a permis de découvrir la richesse de la société Centaurienne. On y a découvert qu'à la différence des êtres humains les Centauriens peuvent être possiblement de trois sexes différents : Truc, Bidule et Machin. D'autre part, les Centauriens ont un nom qui ne peut être modifié. Il ont aussi un âge.

- Définissez la hiérarchie de classes correspondante. Réfléchissez à l'ensemble des attributs et opérations que l'on peut associer à un Centaurien. Pensez à déclarer les opérations permettant d'afficher le nom et l'âge.
- Utilisez le modifieur `const` à chaque fois que c'est utile.
- Créez l'ensemble des constructeurs adéquats dans les différentes classes de la hiérarchie. Peut-on construire un centaurien ?
- On souhaite écrire une méthode `string getSexe()` dans la classe `Centaurien`. Comment la déclarez vous ?
- L'âge des Centauriens évolue bizarrement : un Centaurien ne vieillit que lorsqu'on intéragit avec lui (vous lui demandez son âge et hop il prend un petit coup de vieux, vous lui demandez son nom et hop il prend un autre coup de vieux, etc)...
  Implémentez le mécanisme de vieillissement des Centauriens sans enlever les modifieurs `const` !