

# Devoir Maison POO

Abou KOUTA, M1 Bioinformatique

May 2020

## 1 EXERCICE 1

La classe **Personne** est représentée par un nom, un prénom, un numéro de insee et une date de naissance. Le nom et le prénom seront représentés sous forme de chaîne de caractères, le numéro de insee par un entier et la date de naissance par une date. Il existe un constructeur, celui par défaut qui permet d'initialiser la classe **Personne**.

Pour que la classe **Patient** hérite de la classe **Personne**, il faudra retourner tous les attributs de la classe **Personne**.

La classe **Patient** (classe fille) hérite de la classe **Personne** donc elle contient tous les attributs et les méthodes de la classe **Personne** (classe mère). Dans la classe **Patient** j'ai ajouté un attribut correspondant à l'état du patient qui est un entier. Afin d'utiliser la classe **Personne** comme classe mère de la classe **Patient** nous devons mettre la classe *affiche* en virtual et mettre les attributs en protected afin de pouvoir accéder à ces attributs dans la classe **Patient**. Pour compiler le fichier main.cpp faites:

```
$ g++ main.cpp -o main
```

## 2 EXERCICE 2

Le compilateur communique 2 erreurs.

La première est qu'on essaye d'instancier un objet de la classe **Forme**, alors que cette classe est abstraite. On ne peut pas instancier de classe abstraite. La correction consiste donc à supprimer la ligne qui instancie l'objet, d'autant plus qu'on ne s'en sert pas par la suite.

La deuxième erreur est qu'on essaye d'assigner un pointeur d'un objet de la classe **Cercle** à un pointeur d'un objet de la classe **Sphere**. La correction consiste à caster le pointeur de **Cercle** en pointeur de **Sphere** explicitement dans le code. Voici le résultat obtenu lors de l'exécution du programme:

```
>>> ./main
Affiche Cercle x=4.2 y=5.3r=5
>>> █
```

Pour compiler le fichier main.cpp faites:

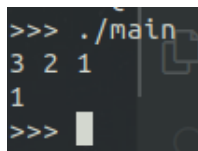
```
$ g++ main.cpp -o main
```

### 3 EXERCICE 3

Dans cet exercice, on nous demande de développer une liste ordonnée à l'aide de templates dans la classe **ListeOrd**.

Dans cette classe, j'ai mis un constructeur, celui par défaut, il initie le **ListOrd** avec une liste vide, un destructeur qui détruit tous les éléments récursivement. Le **ListOrd** peut insérer data, via la méthode **InsertInPlace**, en supprimant le premier élément via **popFirst**, regarder si il y a un élément dans la liste via **Is-empty**, affiche les éléments de la liste dans triés dans l'ordre avec **display**(cf listord.hpp).

Voici le résultat obtenu lors de l'exécution du programme:



```
>>> ./main
3 2 1
1
>>>
```

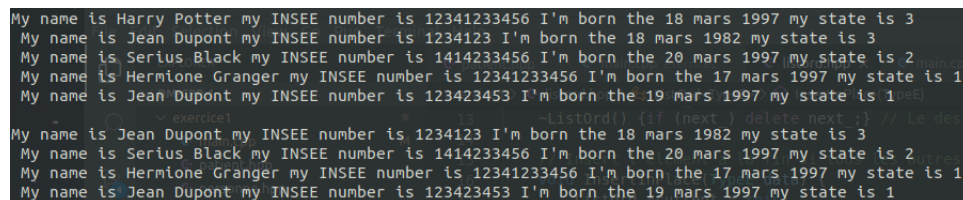
Pour compiler le fichier main.cpp faites:

```
$ g++ main.cpp -o main
```

### 4 EXERCICE 4

Cet exercice concerne les surcharges de l'opérateur, il utilise les classes **Personne** et **Patient** de l'exercice1 et charge aussi la liste des templates **ListOrd** de l'exercice 3. Dans la classe **Patient**, j'ai ajouté des opérateurs de comparaison d'infériorités et d'égalités **bool operator<** et **bool operator==**. Ces fonctions vont utiliser des références de constantes puis vont comparer et renvoyer un booléen indiquant si les deux objets sont identiques ou non. Puis j'ai défini une fonction amie de la classe **Patient** avec **friend**, elle permet d'avoir accès à toutes les données et fonctions de la classe.

Voici le résultat obtenu lors de l'exécution du programme:



```
My name is Harry Potter my INSEE number is 12341233456 I'm born the 18 mars 1997 my state is 3
My name is Jean Dupont my INSEE number is 1234123 I'm born the 18 mars 1982 my state is 3
My name is Sirius Black my INSEE number is 1414233456 I'm born the 20 mars 1997 my state is 2
My name is Hermione Granger my INSEE number is 12341233456 I'm born the 17 mars 1997 my state is 1
My name is Jean Dupont my INSEE number is 123423453 I'm born the 19 mars 1997 my state is 1
```

Pour compiler le fichier main.cpp faites:

```
$ g++ main.cpp -o main
```