Université Hassan II de Casablanca
ENSET de Mohammedia
Département Mathématiques et Informatique
Filière d'Ingénieurs : GLSID 1 & BDCC 1
Prof. M. K. MANSOURI

TP 5
PROGRAMMATION ORIENTEE OBJET EN C++
Héritage simple, protection de données, constructeurs, transmissions de données et surcharge des opérateurs

Problème : Formes géométriques

On définit une hiérarchie des classes représentant des formes géométriques qui comportent le cercle, le triangle, le rectangle et le carré.

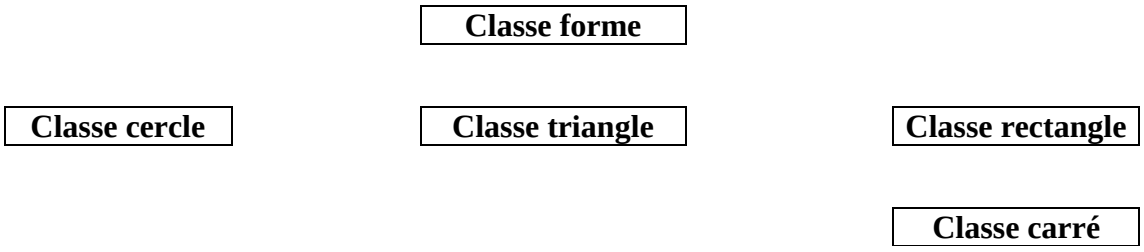
Pour chaque forme, on veut connaître son périmètre et sa surface. Il sera également possible de déplacer une forme sur un plan, on définira pour cela une classe coordonnée.

On rappelle qu'un :

- Cercle est caractérisé par un centre, point de coordonnées x et y un rayon r,
- Triangle est caractérise par trois points a, b et c,
- Rectangle est caractérisé par deux points celui du coin bas gauche et celui du coin haut droite,
- Carré est caractérisé par un point du coin bas gauche est un coté a.

En supposera que toutes les formes possèdent une couleur et on souhaite que chaque classe qu'on va implémenter durant ce TP possède une forme canonique.

La hiérarchie des classes est la suivante :



1ère partie :

1- Ecrire la classe coordonne (structure et corps) selon le modèle suivant :

```
Classe coordonne
Private :
   Int x, y;
Public :
   Coordonne(int a=0, int b=0)
   {
      x = a ;
      y = b ;
   }

déplace (int,int)
{
```

```
    x += a ;
    y += b ;

affiche()
{
    coût <<"x = "<< x << "Y = " << Y;
}
```

2- Ecrire le code déclaratif de la classe forme selon le modèle suivant :

Classe forme
Protected : Short couleur;
Public : Forme(short = 1) ; Forme(forme &) ; Void affiche() Forme operator = (forme &) ;

3- Implémenter les différentes fonctions membres pour cette classe,

2ème partie :

4- Ecrire une nouvelle classe cercle qui hérite publiquement de La classe forme selon le modèle suivant :

Classe cercle
Protected : Coordonne centre; Short rayon ;
Public : Cercle(int, int, short, short) ; Cercle(cercle &) ; Cercle operator = (cercle &) ; Void affiche() ; Deplace(int, int) ; Int surface ; Int périmètre ;

On remarque que le constructeur de la classe **cercle** reçoit 4 paramètres les coordonnés **de son** centre, son rayon **et sa** couleur.

5. Implémenter ses différentes fonctions membres, sachant que le constructeur usuel doit faire appel à la fois au constructeur de la classe **coordonne** et le constructeur de la classe **forme,** par contre le constructeur de recopie de la classe **cercle** doit réutiliser celui de la classe **forme.** La fonction **affiche** doit réutiliser celle de la classe **forme** et **coordonne.**

5- Dans la fonction principale main écrire le code suivant :

```
Main()
{
//création d'un cercle de coordonnées 10, 20
de rayon 5 et de couleur 12
Cercle cl (10,20,5, 12);
```

```
cl.deplate(5,4);
cl,affiche();
getch();
```

3ème partie :

6- Créer une nouvelle classe triangle qui hérite publiquement de la classe forme selon le modèle suivant :

Classe triangle
Protected : Coordonne a, b, c;
Public : Triangle(int,int,int,int,int,int,int,short) ; triangle(triangle &) ; triangle operator = (triangle &) ; Void affiche() ; Deplace(int, int) ; Int surface ; Int périmètre ;

On remarque que le constructeur de la classe **cercle** reçoit 7 paramètres, les coordonnés des trois coins qui caractérisent un triangle **ainsi** que sa couleur.

7- Implémenter ses différentes fonctions membres, sachant que le constructeur usuel doit faire appel à la fois au constructeur de la classe coordonne et le constructeur de la classe forme, par contre le constructeur de recopie de la classe triangle doit réutiliser celui de la classe forme. La fonction affiche doit réutiliser celle de la classe forme et coordonne.

Ajouter le code suivant dans la fonction principale:

```
Triangle T(10,20,30,40,50,50,11) ;
T.affiche();
T.deplace(5,4} ;
T.affiche() ;
```

4ème partie :

8- Créer une nouvelle classe rectangle qui hérite publiquement de la classe forme selon Je modèle suivant :

Classe rectangle
Protected : Coordonne a, b;
Public : Triangle(int,int,int,int,int,short) ; Rectangle(rectangle &) ; Rectangle operator = (rectangle &) ; Void affiche() ; Deplace(int, int) ; Int surface ; Int périmètre ;

On remarque que le constructeur de la classe **rectangle** reçoit 5 paramètres qui représentent les coordonnés du **coin** bas gauche et celui du coin haut droite qui caractérise un rectangle ainsi que sa couleur.

9- Implémenter ces différentes fonctions membres, sachant que le constructeur usuel doit faire appel à la fois au constructeur de La classe coordonne et le constructeur de la classe forme, par contre le constructeur de recopie de la classe triangle doit réutiliser celui de la classe forme. La fonction affiche doit réutiliser celle de la classe forme et coordonne.

Ajouter le code suivant dans la fonction principale et tester votre programme :

```
rectangle R(l0,20,30,40,50,60) ;
T.affiche() ;
T.deplace(5,4) ;
T.affiche();
```

5ème partie :

10- Créer une nouvelle classe carre qui hérite publiquement de la classe Rectangle selon le modèle suivant :

Classe carre
Short cote;
Public : Carre(int,int,int,short) ; carre(carre &) ; carre operator = (carre &) ; Void affiche() ; Deplace(int, int) ; Int surface ; Int périmètre ;

On remarque que le constructeur de la classe **carre** reçoit 4 paramètres qui représentent tes coordonnés du **coin** bas gauche , son côté ainsi que sa couleur.

11- Implémenter ses différentes fonctions membres, sachant que le constructeur usuel doit faire appel à la fois au constructeur de la classe coordonne et le constructeur de la classe forme, par contre le constructeur de recopie de la classe triangle doit réutiliser celui de la classe forme. La fonction affiche doit réutiliser celle de la classe forme et coordonne et qui doit afficher les coordonnées du coin bas gauche du carré ainsi la mesure de son côté et sa couleur.

Ajouter le code suivant dans la fonction principale et tester votre programme :

```
Carre c(l0,20,5,10) ;
c.affiche() ;
c.deplace(5,4) ;
c.afiche() ;
```