Université Paris Diderot **Programmation Orientée Objet**
L3 **Année 2011-2012**

TP n° 5

Interface et classes astraites

Auto-évaluation : Vous devez au moins faire jusque l'exercice 5, en pensant, bien sûr à tester vos classes et méthodes au fur et à mesure.

Important : Dans ce TP, toutes les variables (d'objet ou de classe) seront **privées.**

Par ailleurs, nous fournissons ici des classes permettant de faire des dessins simples en tenant compte de vos connaissances actuelles. Comme vous verrez plus tard, ce n'est pas la bonne méthode pour faire ce genre de choses.

Préliminaires

Téléchargez sur Didel ou sur `http://igm.univ-mlv.fr/~fagnot/` le fichier `tp5.zip`. Compilez la classe `TestDessin` et testez-la.

Pour la suite, vous aurez besoin de regarder l'API de la classe `Polygon` (page `http://download.oracle.com/javase/6/docs/api/java/awt/Polygon.html`)

Exercice 1 On va définir une classe abstraite `Figure` dont voici le début :

```
import java.awt.Polygon;

public abstract class Figure{
    // coordonnées du centre approximatif de la figure
    private int posX;
    private int posY;

    public Figure(int x, int y){
        posX = x;
        posY = y;
    }
    ............
}
```

On définira aussi dans `Figure` les méthodes concrètes :

- `public int getPosX()` qui donnera la position horizontale du centre de la figure (abscisse) ;
- `public int getPosY()` qui donnera la position verticale du centre de la figure (ordonnée) ;

Ainsi que la méthode abstraite

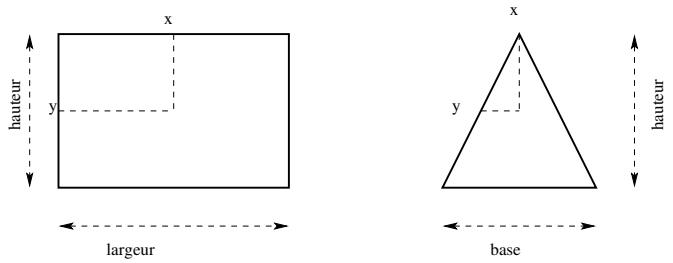
- `public abstract Polygon creePolygon();` qui créera un polygone correspondant à la figure.

Écrivez le code de `Figure`.

On définira aussi les classes concrètes suivantes : `Rectangle`, `Carre` et `Triangle`.

Donnez la hiérarchie de classes que vous choisiriez. Pour l'instant, n'écrivez pas de code pour ces classes.

Exercice 2 Écrivez le code de la classe `Rectangle`. On doit passer en paramètres du constructeur la position du centre, la largeur et la hauteur. Dans la figure ci-dessous, `x` et `y` représente la position du centre.

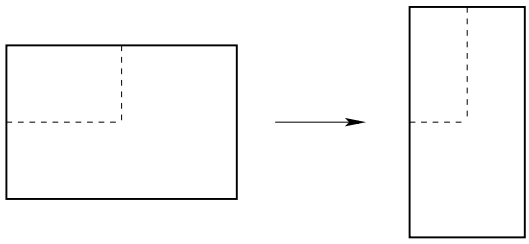


Exercice 3 Écrivez le code de la classe `Carre`. On doit passer en paramètres du constructeur la position du centre et la longueur d'un côté.

Exercice 4 Écrivez le code de la classe `Triangle`. On considère qu'un triangle est toujours isocèle et positionné comme sur le dessin ci-dessus. On doit passer en paramètres du constructeur la position du centre, la base et la hauteur (cf. dessin). Dans la figure ci-dessus, *x* et *y* représente la position du centre, *y* est à la moitié de la hauteur.

Exercice 5 On définit l'interface `Deformable` qui correspond aux figures qu'on peut déformer horizontalement et verticalement de manière à obtenir une nouvelle figure.

Cette interface contiendra la méthode `Figure deformer(double coeffH, double coeffV)` où `coeffH` est le coefficient de déformation horizontale, et où `coeffV` est celui de déformation verticale. Par exemple, dans le dessin ci-dessous, le rectangle de droite est la déformation du triangle de gauche `rec` par l'appel `rec.deformer(0.5, 1.5)`



On notera qu'un figure est déformable si la déformation donne une figure. Quelles figures implémenteront Deformable ?

Implémetez-là dans toutes les classes où c'est possible.

Exercice 6 Les méthodes des questions suivantes sont à ajouter (sauf indication contraire) dans `Figure`. Pour chacune d'entre elles, demandez-vous si elle être abstraite ou non, et dans quelles classes il convient de la définir ou la redéfinir. Vous pouvez faire ces questions dans l'ordre qui vous convient.

– Écrivez la méthode `double estDistantDe(Figure fig)` qui calculera la distance entre le centre de la figure sur laquelle est appelée la méthode et le centre de `fig` ;
– Écrivez la méthode `double surface()` qui calculera la surface de la figure.
Rappel surface d'un triangle $base \times hauteur/2$.
– En regardant la classe `Dessin`, trouvez comment les couleurs sont définies. changez vos classes de telles manière qu'une figure ait une couleur, puis surchargez la méthode `dessine()` de `Dessin` de façon à ce qu'elle prenne une figure en paramètre et la dessine dans la couleur voulue.
– Écrivez une méthode `déplacement(int x, int y)` qui déplace une figure (et la modifie donc) dans la direction indiquez par `x` et `y`.