POO & langage Java

Sujet 4 - Construction et utilisation d'une classe

L'objectif de cet exercice consiste à réaliser puis à utiliser, en pseudo langage orienté objet, les classes `Point` et `Rectangle` dont la description est fournie ci-après. Afin de bien comprendre la substantifique moelle de la programmation par objets, il vous est demandé d'appliquer à la lettre les principes d'encapsulation et d'abstraction de données.

— Un point est défini par ses coordonnées réelles `X` et `Y`.

— Un rectangle est défini un point (son coin inférieur gauche) et par sa longueur (axe des X) et sa largeur (axe des Y). Outre sa longueur et sa largeur, un rectangle doit également être capable de calculer sa surface, son périmètre et de renvoyer un point représentant son centre. Un rectangle doit en plus disposer d'une méthode lui permettant de s'afficher sur un objet de la classe `Ecran` disposant d'une méthode `tracerSegment(Point,Point)`.

⋆ ⋆ ⋆

Travail demandé

1. Écrire la classe `Point`.
2. Écrire la classe `Rectangle`.
3. Écrire un programme qui crée un rectangle, lui demande de s'afficher, puis dessine[1] ses deux diagonales. Pour cette dernière opération, seules les informations retournées par les méthodes de la classe `Rectangle` peuvent être utilisées. Tracer ensuite l'exécution de ce programme sur un exemple pour voir tous les envois de messages.
4. Supposons que l'on ne veuille plus représenter un rectangle par le triplet (coin inférieur gauche, longueur, largeur) mais par deux points diamétralement opposés. Refaire les questions 2 et 3 avec ces nouvelles contraintes.
5. Écrire maintenant la spécification puis l'implémentation de la classe `Carre` qui, comme son nom l'indique, représente un carré, c'est-à-dire un rectangle dont la longueur et la largeur sont toujours égales. Dans la mesure de vos possibilités, essayez de répondre à cette question de manière "intelligente"... `;-)`

   NB : Afin de pouvoir distinguer visuellement les carrés des rectangles, vous ferez en sorte que la méthode `afficher` de la classe `Carre` trace à la fois le contour du carré mais également ses deux diagonales.
6. La dernière question consiste à écrire un nouveau programme de test qui va stocker dans un tableau plusieurs rectangles et carrés (i.e. plusieurs instances des classes `Rectangle` et `Carre`), puis va les afficher sur un écran (via une simple boucle).

Informations pratiques Afin de pouvoir mener à bien cette "entreprise", vous prendrez soin de recopier chez vous les trois fichiers `Ecran.class`, `Ecran$1.class` et `SimplePanel.class` que vous trouverez sur le web de l'UPPA à l'url suivante : http://munier.perso.univ-pau.fr/temp/M2207/. Il s'agit, entre autres, de l'implémentation de la classe `Ecran` qui dispose d'un constructeur sans argument et d'une seule et unique méthode dont la signature est `void tracerSegment(Point p1, Point p2)`. Pour que celle-ci fonctionne correctement, la classe `Point` que vous écrirez devra comporter au moins les deux méthodes `double getX()` et `double getY()`. Inutile de vous rappeler que les majuscules et minuscules sont **extrêmement importantes**...

1. C'est le programme qui dessine les deux diagonales, pas le rectangle lui-même!

Classe `Point`

```java
package m2207.tp04;

//M2207 - Consolidation des bases de la programmation
//TP04 - POO & Java: classes Point & Rectangle
//Manuel Munier - IUT des Pays de l'Adour - 03/2014

public class Point {

   // Attributs

   private double X,Y;

   // Constructeurs

   public Point() {
      this(0.0,0.0);
   }

   public Point(double a, double b) {
      X=a;
      Y=b;
   }

   public Point(Point pt) {
      this(pt.getX(), pt.getY());
   }

   // Methodes

   public double getX() {
      return X;
   }

   public void setX(double a) {
      X=a;
   }

   public double getY() {
      return Y;
   }

   public void setY(double b) {
      Y=b;
   }


   // Redefinition de la methode toString pour debuggage

   public String toString() {
      return "("+X+","+Y+")";
   }


   // Programme principal (test embarque dans la classe)

   public static void main(String[] args) {
      Point p1 = new Point(10,6);
      System.out.println("p1 = "+p1);
```

```
      Point p2 = new Point(p1);
      System.out.println("p2 = "+p2);
      p2.setX(5);
      p2.setY(12);
      System.out.println("p2 = "+p2);
   }

}
```

Classe Rectangle

```java
package m2207.tp04;

//M2207 - Consolidation des bases de la programmation
//TP04 - POO & Java: classes Point & Rectangle
//Manuel Munier - IUT des Pays de l'Adour - 03/2014

public class Rectangle {

   // Attributs

   private Point coin;
   private double longueur, largeur;

   // Constructeurs

   public Rectangle(Point pt, double a, double b) {
      coin = new Point(pt);
      longueur = a;
      largeur = b;
   }

   public Rectangle() {
      this(new Point(), 0.0, 0.0);
   }


   // Methodes d'acces

   public Point getCoin() {
      return new Point(coin);
   }

   public void setCoin(Point pt) {
      coin = new Point(pt);
   }

   public double longueur() {
      return longueur;
   }

   public void longueur(double a) {
      longueur = a;
   }

   public double largeur() {
      return largeur;
   }

   public void largeur(double b) {
      largeur = b;
   }


   // Methodes "+ evoluees"

   public double surface() {
      return this.longueur() * this.largeur();
   }
```

```
   public double perimetre() {
      return 2*(this.longueur() + this.largeur());
   }

   public Point centre() {
      return new Point(coin.getX()+this.longueur()/2, coin.getY()+this.largeur()/2);
   }

   public void afficher(Ecran e) {
      Point p1 = coin;
      Point p2 = new Point(p1.getX(), p1.getY()+this.largeur());
      Point p3 = new Point(p1.getX()+this.longueur(), p2.getY());
      Point p4 = new Point(p3.getX(), p1.getY());

      e.tracerSegment(p1, p2);
      e.tracerSegment(p2, p3);
      e.tracerSegment(p3, p4);
      e.tracerSegment(p4, p1);
   }

   // Programme principal (test embarque dans la classe)

   public static void main(String[] args) {
      Rectangle r = new Rectangle(new Point(50,20), 60, 40);
      Ecran ecr = new Ecran();
      r.afficher(ecr);
   }

}
```