

## Sujet A

---

**Exercice 1 – Rectangle**

---

**Q 1.1** Donner le code de la classe `Rectangle` qui possède :

- deux attributs largeur et longueur du type double.
- un constructeur à deux paramètres initialisant les deux attributs.
- un constructeur sans paramètre qui initialise les attributs avec des valeurs aléatoirement entre  $[10, 30[$ . Attention à la réutilisation de constructeurs existants.
- un constructeur de copie qui prend en paramètre un autre `Rectangle` et copie ses valeurs.

```
1 public class Rectangle {
2     private double largeur, longueur;
3
4     public Rectangle(double largeur, double longueur) {
5         this.largeur = largeur;
6         this.longueur = longueur;
7     }
8
9     public Rectangle(Rectangle r){
10        largeur = r.largeur;
11        longueur = r.longueur;
12    }
13
14    public Rectangle() {
15        this(Math.random()*20+10, Math.random()*20+10);
16    }
17
18
19 }
```

**Q 1.2** Ajouter les méthodes suivantes :

- L'ascenseur et modificateur pour la variable d'instance largeur.
- `toString` qui renvoie une chaîne de caractère avec les valeurs de largeur et longueur.
- `area` qui calcule l'aire du `Rectangle`.
- `egalite` qui prend en argument un autre `Rectangle` et renvoie `true` si ils sont égaux ou `false` si ils sont différent.

```
1     public double getLargeur(){
2         return largeur;
3     }
4
5     public void setLargeur(double largeur){
6         this.largeur = largeur;
7     }
8
9     public String toString(){
10        return "Rectangle:␣largeur=␣" + largeur + "␣longueur=␣" + longueur;
11    }
12    public double area(){
13        return (largeur*longueur);
14    }
15    public boolean egalite(Rectangle r){
16        return (largeur == r.largeur && longueur==r.longueur);
17    }
18
19 }
```

---

**Exercice 2 – TestRectangle**

---

**Q 2.1** Écrire une classe `TestRectangle` contenant une méthode `main` dans laquelle on crée trois `Rectangle`s, à chaque fois utiliser un de trois constructeurs créé précédemment.

**Q 2.2** Calculer et afficher l'aire du `Rectangle` créé à la question précédente.

**Q 2.3** Afficher les caractéristiques des trois `Rectangle`s.

```
1 public class TestRectangle() {  
2     public static void main(String [] args) {  
3         Rectangle r1 = new Rectangle();  
4         Rectangle r2 = new Rectangle(10, 28);  
5         Rectangle r3 = new Rectangle(r2);  
6  
7         System.out.println(r2.area());  
8         System.out.println(r1.toString());  
9         System.out.println(r2);  
10        System.out.println(r3);  
11    }  
12  
13  
14  
15 }
```

---

**Exercice 3 – Modification de la classe `Rectangle`**

---

Nous voudrions insérer un nouveau rectangle dans un rectangle existant, comme dans la figure 1. Les conditions à respecter sont :

- 1) le rectangle inséré doit être plus petit que le rectangle qui le héberge.
- 2) un rectangle ne peut pas être inséré dans plusieurs rectangles, et chaque rectangle contient au max un rectangle à son intérieur.

**Q 3.1** Créer le(s) variable(s) d'instance(s) nécessaire(s) pour pouvoir insérer un rectangle.

**Q 3.2** Créer la méthode `insérer` pour insérer un rectangle dans un autre rectangle. La méthode affichera "L'insertion est pas possible" si une de conditions n'est pas satisfaite.

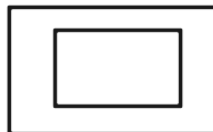


FIGURE 1 – Rectangle inséré

```
1     private Rectangle rectRef;  
2  
3     public void inserer(Rectangle in) {  
4         if (rectRef != null || r1.area() >= area() || in.rectRef != null) {
```

```
5         System.out.println("L'insertion est pas possible")
6     }
7     else{
8         rectRef = in;
9         in.rectRef = this;
10    }
11 }
12 }
```