TP N°03

Héritage en Java

Exercice 1 :

Soit le programme suivant :

```
// Forme.java
class Forme {
    double x, y ;

    public Forme() {
        x = 0 ; y = 0 ;
    }

    public Forme(double x, double y) {
        this.x = x ; this.y = y ;
    }

    public String toString() {
        return "Position : (" + x + "," + y + ")" ;
    }
}
// Cercle.java

 class Cercle extends Forme {
    final static double PI = 3.141592564 ;

    private double rayon ;

    public Cercle() {
        rayon = 0 ;
    }

    public Cercle(double x, double y, double r) {
        super(x,y) ;
        rayon = r ;
    }

    public String toString() {
        return super.toString() + " Rayon : " + rayon ;
    }
}
public class MainForme {
    public static void main(String[] args) {
        Cercle c1,c2 ;
        c1 = new Cercle(1,1,3) ;
        c2 = new Cercle() ;
        System.out.println(c1.toString() + "\n" + c2.toString()) ;
    }
}
```

1. De quelles variables d'instance de **Forme** hérite la classe Cercle ?
2. La variable ***rayon*** étant déclarée ***private***, on ne peut pas la modifier de l'extérieur de la classe. Ajoutez une méthode ***setRayon*** pour pouvoir le fixer et ***getRayon*** pour le lire.
3. Ajoutez une variable ***surface*** à la classe **Cercle**. Modifiez les méthodes ***setRayon*** et ***toString*** en conséquence. Ajoutez les méthodes d'accès à ce champ.
4. Écrivez une méthode c1.***estGrand***(**Cercle** c2) renvoyant ``vrai'' si le cercle c1 est plus grand que c2.
5. Ecrivez une méthode ayant un comportement similaire, mais prenant les deux cercles en argument.
6. Étendez la classe **Cercle** avec une classe **Cylindre** ajoutant une variable ***h*** pour la ***hauteur*** et une méthode ***volume***. Ajoutez les constructeurs adéquats et surchargez ***toString***.
7. Étendez la classe **Cercle** avec une classe **CercleColorié**. La classe **Cylindre** hérite-t'elle de l'attribut de ***coloration*** ?
8. Ecrivez une classe abstraite **CollectionForme** ayant comme méthodes : ***ajout*** et ***suppression*** d'un objet de type **Forme**, la suppression prend en argument les infos sur la **Forme**. les deux méthodes sont abstraites.
9. Ecrivez une classe **CollectionFormeTab,** qui est une version concrète (implémentant ***ajout*** et ***suppression***) de la classe **CollectionForme**, dans laquelle un tableau est utilisé pour stocker les formes créées.
10. Testez la classe **CollectionFormeTab** dans la classe principale **MainForme.**