CHAPUS 17/05/2023

TP n°3 Java

Louka 3A IE

Exercice 1 :

1./

```java
public abstract class FormeGeometrique {

    protected double epaisseurTrait;

    public FormeGeometrique(double epaisseurTrait){
        this.epaisseurTrait = epaisseurTrait;
    }

    public abstract double calculPerimetre();
    public abstract double calculSuperficie();
    public abstract String toString();

}
```

Figure [1] classe FormeGeometrique

On déclare des méthodes abstraites, ainsi que l'épaisseur du trait en protected pour pouvoir y avoir accès dans les classes filles.

2./

On met l'attribut protected pour que les classes filles aient accès aux variables de la classe mère.

3./

Pour le constructeur, on initialise simplement l'épaisseur du trait puisque c'est le seul paramètre qui est commun aux deux classes

4./

```java
public class Cercle extends FormeGeometrique {

    protected double rayon;

    public Cercle (double epaisseurTrait ,double rayon){
        super(epaisseurTrait);
        this.rayon = rayon;
    }

    public double getRayon () {
        return this.rayon;
    }
```

Forme [2] classe Cercle

```java
public class Rectangle extends FormeGeometrique {

    protected double longueur;
    protected double largeur;

    public Rectangle (double epaisseurTrait ,double longueur ,double largeur ){
        super(epaisseurTrait);
        this.longueur = longueur;
        this.largeur = largeur;
    }

    public double getLongueur () {
        return this.longueur;
    }
    public double getLargeur () {
        return this.largeur;
    }
```

Forme [3] classe Rectangle

Là aussi on déclare simplement les classes avec les getters bon getters. Les deux classes héritent de la classe FormeGeometrique c'est pourquoi on appelle la fonction super() dans le constructeur.

5./

```java
public double calculPerimetre(){
    double S=0.0;
    S = 2 * 3.14 * this.rayon;
    return S;
}

public double calculSuperficie(){
    double S=0;
    S = 3.14 * this.rayon * this.rayon;
    return S;
}
```

Figure [4] méthodes de la classe Cercle

```java
public double calculPerimetre(){
    double S=0;
    S = 2*this.largeur + 2*this.longueur;
    return S;
}

public double calculSuperficie(){
    double S=0;
    S = this.largeur * this.longueur;
    return S;
}
```

Figure [5] méthodes de la classes Rectangle

A chaque fois on calcule le périmètre et la superficie des formes géométriques avec la bonne formule pour le rectangle et pour le cercle.

7./

```
Main.java:4: error: FormeGeometrique is abstract; cannot be instantiated
                FormeGeometrique forme1 = new FormeGeometrique(1.0);
                                          ^
Main.java:5: error: FormeGeometrique is abstract; cannot be instantiated
                FormeGeometrique forme2 = new FormeGeometrique(1.0);
                                          ^
```

Figure [6] erreur FormeGeometrique

Lorsqu'on essaye de déclarer des objets de type FormeGeometrique on obtient une erreur qui est logique car c'est une classe abstraite donc on ne peut pas créer d'objet.

8./

```
Cercle forme1 = new Cercle(1.0,5);
Rectangle forme2 = new Rectangle(1.0,5,9);

System.out.println("Périmètre du cercle : " + forme1.calculPerimetre() + "\nSuperficie du cercle : " + forme1.calculSuperficie());
System.out.println("Périmètre du rectangle : " + forme2.calculPerimetre() + "\nSuperficie du rectangle : " + forme2.calculSuperficie());
```

Figure [7] classe main

```
Périmètre du cercle : 31.400000000000002
Superficie du cercle : 78.5
Périmètre du rectangle : 28.0
Superficie du rectangle : 45.0
```

Figure [8] Résultat après exécution du main

On voir bien qu'après exécution du main on obtient bien les bonnes valeurs pour les périmètre et pour les superficies.

9./

La compilation se passe bien mais normalement il faudrait redéfinir la fonction toString dans chaque classe. C'est donc ce qu'on va faire par la suite.

```
public String toString(){
    String info;
    info = "Périmètre du cercle : " + this.calculPerimetre() + "\nSuperficie du cercle : " + this.calculSuperficie();
    return info;
}
```

Figure [9] méthode de la classe Cercle

```
public String toString(){
    String info;
    info = "\nPérimètre du rectangle : " + this.calculPerimetre() + "\nSuperficie du rectangle : " + this.calculSuperficie();
    return info;
}
```

Figure [10] méthode de la classe Rectangle

On met juste bien en forme le texte avec les bonnes informations et le bon texte à renvoyer. Donc ensuite dans le main pourra simplement appeler la méthode toString et on aura toutes les information dans le bon formalisme.

Exercice 2 :

1./

```
class TableauFormeGeometrique {

    protected int nbFormeGeometrique;
    protected FormeGeometrique tabFormeGeometrique[];
```

Figure [11] classe TableauFormeGeometrique

On pourrait mettre les variables en private cela reviendrait à la même chose.

2./

```
public TableauFormeGeometrique(int nbFormeGeometrique){
    this.nbFormeGeometrique = nbFormeGeometrique;
    this.tabFormeGeometrique = new FormeGeometrique[nbFormeGeometrique];
}
```

Figure [12] Constructeur de la classe TableauFormeGeometrique

On initialise un tableau vide de longueur nbFormeGeometrique.

3./

```
public void ajouterForme(FormeGeometrique forme){
    for(int k=0; k<nbFormeGeometrique;k++){
        if(this.tabFormeGeometrique[k] == null){
            this.tabFormeGeometrique[k]=forme;
            break;
        }
    }
}
```

Figure [13] méthode pour ajouter une forme géométrique

Pour ajouter une forme on parcourt tout le tableau et si l'élément est vide alors on met la forme à cet emplacement et on arrête la boucle pour ne pas remplir le tableau avec le même élément.

4./

```
TableauFormeGeometrique tab1 = new TableauFormeGeometrique(5);
tab1.ajouterForme(forme1);
tab1.ajouterForme(forme2);

System.out.println(tab1.toString());
```

Figure [14] Main

```java
public String toString(){
    String info="";
    for(int k=0 ; k<nbFormeGeometrique ; k++){
        if(this.tabFormeGeometrique[k] != null){
            info = info + this.tabFormeGeometrique[k].toString();
        }
    }
    return info;
}
```

Figure [15] Méthode de la classe TableauFormeGeometrique

Pour la fonction toString on appelle simplement les fonctions toString sur chaque éléments du tableau, cela nous permet de réutiliser le code précédent.