Algorithmes et Pensée Computationnelle

Programmation orientée objet : Héritage et Polymorphisme

Le code présenté dans les énoncés se trouve sur Moodle, dans le dossier **Ressources**.

1 Rappel : Surcharge des opérateurs

Dans cette section, vous manipulerez des fractions sous forme d'objets. Vous ferez des opérations de base sur ce nouveau type d'objets.
Question 1: (🕒 *5 minutes*) Dans un projet que vous aurez au préalable préparé, créez un fichier appelé **surcharge.py**.À l'intérieur de ce fichier, créer une classe **Fraction** qui aura comme attributs un numérateur et un dénominateur.

Question 2: (🕒 *5 minutes*) Définir un constructeur à votre classe. Assignez des valeurs par défaut à vos attributs.

> **Conseil**
>
> Les valeurs par défaut seront assignées à votre objet au cas où il est instancié sans valeurs. Ainsi en faisant **f = Fraction()**, on obtiendra un objet **Fraction** ayant pour valeurs un numérateur et un dénominateur à 1 soit $\frac{1}{1}$.

Question 3: (🕒 *5 minutes*)

2 Notions d'héritage

Language : Java

Le but de cette partie est de pratiquer et d'assimilier les notions liées à l'héritage. Pour cela nous allons nous inspirer de l'exemple présenter dans le cours.
Nous allons créer une classe **Livre()** qui contiendra deux sous-classe, **Livre_Audio()** et **Livre_Illustre()**. Les sous-classes cont hériter des attributs et méthode de la classe mère.
Question 4: (🕒 *10 minutes*) Création de classe et sous-classes

Créez la classe mère **Livre()** avec les caractéristiques suivantes :
- une variable privée **titre**
- une variable privée **auteur**
- une variable privée **annee**
- une variable privée **note** (initilisée à **–1**)
- le constructeur public prenant en argumant les trois premières variables ci-dessus
- une méthode **printInfo()** qui affiche le titre, l'auteur, l'année et la note d'un ouvrage
- une méthode **setNote()** qui permet de définir la variable **note**

Créez les classes filles avec les caractéristiques suivantes :

class Livre_Audio extends Livre
- une variable supplémentaire **narrateur**

class Livre_Illustre extends Livre
- une variable supplémentaire **dessinateur**

```
1  public class Livre {
2
3  }
4
5  public class Livre_Audio extends Livre {
6
7  }
8
9  public class Livre_Illustre extends Livre {
10
11 }
```

Conseil

En java, lors de la déclaration d'une classe, le mot clef **extends** permet d'indiquer qu'il s'agit d'une sous-classe de la classe indiquée.
Le mot clef **super** permet à la sous classe d'hérité d'éléments de la classe mère. **super** peut être utilisé dans le constructeur de la sous-classe selon l'example suivant : **super(variable_mère_1, variable_mère_2, variable_mère_3, etc.);**. Ainsi, il n'est pas nécessaire de redefinir toutes les variables d'une sous-classe !
L'instruction **super** doit toujours être la première instruction dans le constructeur d'une sous-classe.

Solution

```
1  public class Livre {
2
3     private String titre;
4     private String auteur;
5     private int annee;
6     private int note = -1;
7
8     public Livre(String titre, String auteur, int annee){
9        System.out.println("Création d'un livre");
10       this.titre = titre;
11       this.auteur = auteur;
12       this.annee = annee;
13    }
14
15    public void printInfo() {
16       System.out.println(" propos du livre");
17       System.out.println("--------------------");
18       System.out.println("Titre : \""+titre+"\"");
19       System.out.println("Auteur : "+auteur);
20       System.out.println("Année : "+annee);
21       System.out.println("Note : "+note);
22    }
23
24    public void setNote(int note) {
25       this.note = note;
26    }
27 }
28
29 public class Livre_Audio extends Livre {
30    private String narrateur;
31
32    public Livre_Audio(String titre, String auteur, int annee, String narrateur){
33       super(titre, auteur, annee);
34       System.out.println("Création d'un livre audio");
35       this.narrateur = narrateur;
36    }
37 }
38
39 public class Livre_Illustre extends Livre {
40
41    private String dessinateur;
42
43    public Livre_Illustre(String titre, String auteur, int annee, String dessinateur) {
44       super(titre, auteur, annee);
45       System.out.println("Création d'un livre illustré");
46       this.dessinateur = dessinateur;
47    }
48 }
```

Question 5: (*5 minutes*) Méthode et héritage

Maintenant que vous avez créer la classe et les sous classes correspondantes, vous pouvez créer un objet **Livre** à l'aide du constructeur de la sous-classe **Livre_Audio**. Si vous manquez d'inspiration vous pouvez indi-

quer les valeurs suivantes : titre : ”Hamlet”, auteur : ”Shakespeare”, année : ”1609” et le narrateur ”William.

Une fois le livre créer, attribuez lui une note à l'aide de la méthode définie précédemment.

Finalement, utilisé la méthode **printInfo()** pour afficher les informations du livre.

La méthode étant définie dans la classe mère, elle n'a pas connaissance de la variable **narrateur** définie dans la sous-classe. Redefinissez la méthode dans la sous-classe pour y inclure l'information sur le narrateur.

Conseil

Attention, on vous demande de créer un objet **Livre** et non pas **Livre_Audio**.
Le mot clef **super** peut être utilisé dans la redefinition d'une méthode selon l'example suivant : **super.nom_de_la_methode();**. Cette instruction permet d'inclure tout ce qui est défini dans la ”méthode mère” et vous pouvez la complétez selon les caractéristiques de votre sous-classe.
L'instruction **super** doit toujours être la première instruction dans le redéfinition d'une méthode dans une sous-classe.

Solution

```

public class Livre_Audio extends Livre {
    private String narrateur;

    public Livre_Audio(String titre, String auteur, int annee, String narrateur){
        super(titre, auteur, annee);
        System.out.println(”Création d'un livre audio”);
        this.narrateur = narrateur;
    }

    // redéfinition de la fonction printInfo() dans la sous-classe Book
    public void printInfo() {
        super.printInfo(); //permet de reprendre les éléments de la fonction mère
        System.out.println(”Narrateur: ”+ narrateur); //On ajoute l'attribut supplémentaire propre à la sous-classe
    }
}
```

```

class Main {
    public static void main(String[] args) {

        Livre monLivre = new Livre_Audio(”Hamlet”, ”Shakespeare”, 1609, ”William”);
        monLivre.setNote(5);
        monLivre.printInfo();

    }
}
```

3 Polymorphisme

4 Héritage en Python