Laboratoire 5 – Héritage

ITI 1521. Introduction à l'informatique II

Semaine 1-5 Mars 2021

Dû en ligne après une semaine de votre Lab

/10

- ## Objectifs

  - Appliquer et approfondir le concept d'héritage pour l'implémentation de classes
  - Utiliser et maitriser le polymorphisme (méthode pouvant avoir un comportement différent selon les situations).
  - Utiliser la documentation JavaDoc

- ## Introduction

  Vous allez dans ce Lab réaliser des applications simples afin d'approfondir le concept d'héritage.

I. Première partie – Héritage simple

On va dans cette partie, modélisez des animaux et des mamifères avec des classes.

Question 1 : (4 POINTS : 1 POINT pour chaque classe)

Implémenter les classes suivantes :

- une nommée `Animal` qui a :
  - un attribut nommé `name` de type `String`
  - deux constructeurs, un sans paramètre (qui fait rien) et un qui ne prend en paramètre que le nom de l'animal.
  - une méthode `getType` qui devra fournir une description de l'instance.
- une nommée `Mammal` qui hérite de la classe `Animal` et qui a :
  - un attribut nommé `nbMonthPregnacy` de type `int`
  - deux constructeurs, un sans paramètre (qui fait rien) et un qui ne prend en paramètre que le nom de l'animal.
  - une méthode `getType` qui devra fournir une description de l'instance et affichera "I am a Mammal ".
- une nommée `Dog` qui hérite de la classe `Mammal` et qui a :
  - deux constructeurs,
    - un sans paramètre qui doit modifier l'attribut `nbMonthPregnacy` à 3 et qui afficher "I am a dog and just born after 3 months of pregnacy. I d'ont have a a name yet ".

        - et un qui ne prend en paramètre que le nom de l'animal.
    - une méthode `getType` qui devra fournir une description de l'instance et affichera " " I am a dog ".
- une nommée `Cat` qui hérite de la classe `Mammal` et qui a :
    - deux constructeurs,
        - un sans paramètre qui doit modifier l'attribut `nbMonthPregnacy` à 2 et qui afficher "I am a cat and just born after 2 months of pregnacy. I d'ont have a name yet ".
        - et un qui ne prend en paramètre que le nom de l'animal.

    - une méthode `getType` qui devra fournir une description de l'instance et affichera " " I am a cat ".

Arrangez-vous pour que les classes filles puissent manipuler directement les nouvelles variables.

Voici ci-dessous une classe de test et l'affichage que vos classes devront fournir. La méthode `getType` de vos classes devra fournir une description de l'instance, en incluant les descriptions de toutes les classes mères. Pour cela, il vous est imposé d'utiliser le mot-clé `super`.

```
/******** classe TestAnimal ******/
public class TestAnimal {
  public static void main(String[] args) {
    Animal[] animals = new Animal[6];
    animals[0] = new Animal("Bof");
    animals[1] = new Animal();
    animals[2] = new Dog("Lassie");
    animals[3] = new Cat("Puffy");
    animals[4] = new Dog();
    animals[5] = new Cat();
    for (int i = 0; i < animals.length; i++) {
      System.out.println(animals[i].getType());
    }
  }
}
```

```
/******** Exemple de sortie ******/
I am a dog and just born after 3 months of pregnacy. I don't have a name yet.
I am a cat and just born after 2 months of pregnacy. I don't have a name yet.
I am an animal and my name is Bof.
I am an animal.
I am an animal and my name is Lassie. I am a Mammal. I am a dog
I am an animal and my name is Puffy. I am a Mammal. I am a cat.
I am an animal. I am a Mammal. I am a dog
I am an animal. I am a Mammal. I am a cat.
```

- **Deuxième partie –Polymorphisme**

Dans cette partie, vous allez programmer le calcul des salaires hebdomadaires des employés d'une compagnie.

Cette compagnie comporte plusieurs types d'employés :

- Des employés contractuels qui sont payés suivant le nombre d'heures qu'ils ont travaillé dans la semaine. Ils sont payés à un certain tarif horaire et leurs heures supplémentaires (au-delà de 40 heures) sont payées 30 % de plus que les heures normales.
- D'autres employés contractuels sont payés de la même façon, mais leurs heures supplémentaires sont payées 50 % de plus que les heures normales.
- Les employés permanents sont payés avec une somme fixe à laquelle on ajoute 1% du chiffre d'affaires qu'ils ont fait dans la semaine.

Question 2 : (6 POINTS : 1.5 POINTS pour chaque classe)

Modélisez cette situation à l'aide des classes suivantes:

- une classe `Employee` dont hériteront les autres classes. Vous donnerez un nom à chacun des employés. On ne pourra modifier le nom d'un employé. La classe `Employee` contient :
  - un attribut nommé `name` de type `String` pour le nom de l'employé.
  - un constructeur qui ne prend en paramètre que le nom de l'employé.
  - une accesseur (méthode) `getName` qui devra renvoyer le nom d'un employé
  - une méthode `getSalary` qui devra faire le calcul du salaire et le renvoyer.
  - une méthode `toString` qui doit afficher le nom et le salaire de chaque employé.
- une classe `Contract` qui contient :
  - 4 attributs de type double nommés :
    - `nbHours` pour le nombre d'heures travaillé par semaine.
    - `hourRate` pour le tarif horaire.
    - `percentHourSup` pour le pourcentage de leurs heures supplémentaires
    - `dueHour` (static) pour le nombre d'heures dû par semaine (40).
  - 2 constructeurs : un qui ne prend en paramètre que le nom de l'employé et l'autre qui prend en paramètres le nom et toutes les informations pour le calcul du salaire.
    - un modificateur (méthode) `setSalaryInfo` pour entrer ou modifier les informations brutes nécessaires au calcul des salaires (nombre d'heures de travail, tarif horaire, pourcentage des heures supplémentaires).

- 3 accesseurs (méthodes) :
  - `getNbHours`
  - `getHourRate`
  - `getPercentHourSup`
- une méthode `getSalary` (accesseur) qui devra faire le calcul du salaire et le renvoyer.
- une méthode `toString` qui doit afficher le nombre d'heures travaillés par semaine, le tarif horaire et le pourcentage pour les heures supplémentaires.

- une classe `FullTime` qui contient :
  - 3 attributs de type double:
    - `pay` : pour la somme fixe payée aux employés
    - `turnover` : pour le chiffre d'affaires du mois
    - `percentTurnover` : pourcentage du chiffre d'affaires pour le calcul du salaire
  - 2 constructeurs : un qui ne prend en paramètre que le nom de l'employé et l'autre qui prend en paramètres le nom et toutes les informations pour le calcul du salaire.
  - 3 modificateurs (méthodes) :
    - `setSalaryInfo` pour entrer ou modifier les informations brutes nécessaires au calcul des salaires (paye fixe, chiffre d'affaire).
    - `setTurnover`
    - `setPay`
  - une méthode (accesseur) `getSalary` qui devra faire le calcul du salaire et le renvoyer.
  - une méthode `toString` qui doit afficher le nombre d'heures travaillés par semaine, le tarif horaire et le pourcentage pour les heures supplémentaires.

- Une classe `Test` pour tester vos classes, comportant une méthode `main()` qui entrera les informations sur des employés des différents types. Les employés seront enregistrés dans un tableau `employees`.

Au moins un des employés sera créé avec le constructeur qui n'a que le nom en paramètre, et vous entrerez ensuite les informations pour son salaire avec la méthode `setSalaryInfo`. La méthode `main` affichera le salaire hebdomadaire de chacun des employés dans une boucle "`for`" qui parcourra le tableau des employés. On utilise le polymorphisme avec l'accesseur pour le salaire.

/****** Exemple de sortie ******/**

Alex earns 63000.0 $
Dave earns 7125.0 $
Louise earns 4000.0 $
Demi has an unspecified pay

Créer et soumettre un fichier zip comme d'habitude (Q1,Q2)

III. 3ème partie – Documentation : JavaDoc (Démonstration seulement)

Votre démonstrateur (TA) va maintenant faire une brève présentation sur JavaDoc :

- Expliquer JavaDoc

- Représenter les commentaires à l'aide de JavaDoc

- Produire des pages Web à partir de spécifications JavaDoc.

La documentation des programmes est une autre activité importante. JavaDoc est un ensemble de règles pour les commentaires de vos programmes et un ensemble d'outils afin de produire des pages Web. En ITI1521, nous vous demandons de commenter tous vos programmes (variables, méthodes, etc.) à l'aide de JavaDoc.

- Expliquer comment produire des fichiers HTML automatiquement, soit à l'aide de votre environnement de programmation favori (DrJava, Eclipse, Netbeans, etc.), ou encore à l'aide de la commande javadoc dans un shell, les paramètres sont -d doc suivi d'une liste de fichiers sources à traiter :
> javadoc -d doc code.java
Lorsqu'il y a plusieurs fichiers à traiter simultanément, utilisez * à la place des noms de fichiers.
> javadoc -d doc *