Université Abdelmalek Essaadi

Faculté des Sciences et Techniques d'Alhoceima

Rapport de Travaux Pratiques

LST : Ingénierie de données et développement logiciel(IDDL)

TP 06: POO en JAVA

Réalisé par : BAHLAOUANE Salaheddine

Encadré par : Pr. FAHIM Mohamed

Année Universitaire : 2023/2024

Table des matières

I- Introduction

Introduction

Dans ce TP, nous explorons le concept d'héritage en programmation orientée objet, ainsi que le polymorphisme. L'héritage est un pilier de la POO, permettant la création de nouvelles classes basées sur des classes préexistantes, ce qui favorise la réutilisation du code et la structuration hiérarchique des objets.

II- Exercices

1. Travail à faire :

On définit un classe mère “Personne” qui regroupa toutes les propriétés communes entre ces classes filles “Etudiant” et “Enseignant”. Après on va créer une classe nommée “Principale” pour tester.

2. La classe Personne :

a. Le programme :

```java
package InformationScolaire;

public class Personne {
    private String nom;
    private String prenom;
    private String adresse;

    public Personne(String nom, String prenom, String adresse) {
        this.nom = nom;
        this.prenom = prenom;
        this.adresse = adresse;
    }

    public String getNom() {
        return nom;
    }
    public void setNom(String nom) {
        this.nom = nom;
    }

    public String getPrenom() {
        return prenom;
    }
    public void setPrenom(String prenom) {
        this.prenom = prenom;
    }

    public String getAdresse() {
        return adresse;
    }
    public void setAdresse(String adresse) {
        this.adresse = adresse;
    }

    @Override
    public String toString() {
        return "nom : " + nom + "\nprenom : " + prenom + "\nadresse : " + adresse;
    }
}
```

b. Description :

La classe "Personne" représente une entité personne avec des attributs tels que le nom, le prénom et l'adresse.

3. La classe Etudiant :

a. Le programme :

```java
package InformationScolaire;
//L'heritage
public class Eleve extends Personne{
    private String classe;

    public Eleve(String nom, String prenom, String adresse, String classe) {
        super(nom, prenom, adresse);//appel au constructeur personne
        this.classe = classe;
    }

    public String getClasse() {
        return classe;
    }
    public void setClasse(String classe) {
        this.classe = classe;
    }

    @Override
    public String toString() {
        return "Eleve : "+"\nnom : "+getNom()+
                "\nprenom : "+getPrenom()+
                "\nadresse : "+getAdresse()+
                "\nclasse : " +classe;
    }
}
```

b. Description :

La classe "Eleve" qui étend la classe "Personne". La classe "Eleve" ajoute un nouvel attribut spécifique aux élèves, nommé "classe". Ce champ représente la classe à laquelle l'élève est affilié.

Le constructeur de la classe "Eleve" utilise le mot-clé "super" pour appeler le constructeur de la classe mère "Personne" afin d'initialiser les attributs hérités (nom, prénom, adresse) de l'élève. De plus, il initialise l'attribut spécifique "classe" de l'élève.

4. La classe Enseignant :

a. Le programme :

```java
package InformationScolaire;
import java.time.LocalDate;
//L'heritage
public class Enseignant extends Personne{
    private int echelon;
    private double nbHeures;
    private LocalDate dateNaissance;
    //tableaux a variable constante
    private static final double[] tauxSalaire = {40,42,43,44,46,50,52,54,56,60,65};
    //Constructeur Enseignant
    public Enseignant(String nom, String prenom, String adresse, int echelon,double nbHeures,LocalDate dateNaissance) {
        super(nom, prenom, adresse);
        this.echelon = echelon;
        this.nbHeures = nbHeures;
        this.dateNaissance = dateNaissance;
    }
    public int getEchelon() {
        return echelon;
    }
    public void setEchelon(int echelon) {
        this.echelon = echelon;
    }

    public double getNbHeures() {
        return nbHeures;
    }
    public void setNbHeures(double nbHeures) {
        this.nbHeures = nbHeures;
    }

    public LocalDate getDateNaissance() {
        return dateNaissance;
    }
    public void setDateNaissance(LocalDate dateNaissance) {
        this.dateNaissance = dateNaissance;
    }

    @Override
    public String toString() {
        return "Enseignant : \n"+super.toString()+
                "\nechelon : " +echelon+
                "\nnbHeures : "+nbHeures+
                "\nDate-Naissance : "+dateNaissance;
    }
    //methode de verification de l'échelon
    public boolean echelonValide(){
        return this.echelon > 0 && this.echelon <= tauxSalaire.length;
    }
    //methode pour savoir le prix par heure d'un echelon
    //en prenant en compte la gestion des erreurs
    public double getPrixHeure(int echelon) throws Exception{
        if (echelonValide()) {
            return tauxSalaire[echelon-1];
        }
        else {
            throw new Exception("Echelon invalide");
        }
    }
    //methode de calcul de salaire
    //en prenant en compte la gestion des erreurs
    public double calculSalaire() throws Exception{
        if (echelonValide()) {
            System.out.print("Le salaire est : ");
            return getPrixHeure(this.echelon)*getNbHeures();
        }
        else {
            throw new Exception("Echelon invalide");
        }
    }
    //methode pour calculer l'age d'un enseignant
    public int calculAge(){
        System.out.print("L'age est : ");
        return LocalDate.now().getYear() - this.dateNaissance.getYear();
    }
}
```

b. Description :

La classe "Enseignant", qui étend la classe "Personne", ajoute plusieurs attributs spécifiques à un enseignant tels que "echelon" (le niveau de l'échelon), "nbHeures" (nombre d'heures travaillées), "dateNaissance" (date de naissance de l'enseignant) ainsi qu'un tableau statique "tauxSalaire" contenant les taux de salaire correspondant à chaque échelon.

Le constructeur prend en paramètres le nom, le prénom, l'adresse, l'échelon, le nombre d'heures et la date de naissance de l'enseignant. Il utilise le mot-clé "super" pour appeler le constructeur de la classe parente `Personne` et initialise les attributs hérités.

5. La classe Principal :

a. Le programme :

```java
package InformationScolaire;
import java.time.LocalDate;

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

        Personne personne = new Personne("Simo","Live","Al hoceima");
        Personne eleve = new Eleve("Bahlaouane", "Salaheddine","Casablanca", "IDDL");

        Enseignant enseignant1 =new Enseignant("Fahim", "Mohamed","Hoceima",11,80,
                                LocalDate.of(1980,5,23));
        Enseignant enseignant2 = new Enseignant("Alaouite","ALi","Nador",12,125,
                                LocalDate.of(1990,12,30));

        System.out.println(personne);
        System.out.println("########################################");
        System.out.println(eleve);
        System.out.println("########################################");
        System.out.println(enseignant1);
        System.out.println(enseignant1.calculAge());
        try {
            System.out.println(enseignant1.calculSalaire());
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
        System.out.println("########################################");
        System.out.println(enseignant2);
        System.out.println(enseignant2.calculAge());
        try {
            System.out.println(enseignant2.calculSalaire());
        } catch (Exception e) {
            System.out.println(e.getMessage());
        }
    }
}
```

b. L’execution:

```
nom : Simo
prenom : Live
adresse : Al hoceima

Eleve :
nom : Bahlaouane
prenom : Salaheddine
adresse : Casablanca
classe : IDDL
Enseignant :
nom : Fahim
prenom : Mohamed
adresse : Hoceima
echelon : 11
nbHeures : 80.0
Date-Naissance : 1980-05-23
Le salaire est : 5200.0
L'age est : 43

Enseignant :
nom : Alaouite
prenom : ALi
adresse : Nador
echelon : 12
nbHeures : 125.0
Date-Naissance : 1990-12-30
Echelon invalide
L'age est : 33

Process finished with exit code 0
```