TP-2 JAVA OBJET

Programmation orienté objet : définition de classes, création d'objets, héritage, polymorphisme et classes abstraites

Calcul des impôts locaux

Dans le cadre de l'informatisation d'une mairie, on veut automatiser le calcul des impôts locaux. On distingue deux catégories d'habitation : les habitations à usage professionnel et les maisons individuelles, l'impôt se calculant différemment selon le type d'habitation. Pour cela, on d´définit les classes ***HabitationProfessionnelle*** et ***HabitationIndividuelle*** et les caractéristiques communes à ces deux
classes sont regroupées dans la classe Habitation. On a donc un schéma de classes Les classes où
HabitationProfessionnelle et ***HabitationIndividuelle*** héritent de la classe Habitation.
L'objet de cet exercice est d'implémenter ce schéma d'héritage et de mettre en œuvre le m´mécanisme de liaison dynamique.

Définition de la classe Habitation

Objectif : D´définir une classe avec un constructeur et créer une instance de cette classe

La classe Habitation comprend les attributs :

Propriétaire du type chaine de caractères et qui correspond au nom du propriétaire,

adresse du type chaine de caractères et qui correspond à l'adresse de l'habitation,

surface du type double et qui correspond à la surface de l'habitation et qui permet de calculer le montant de l'impôt.

les m´ethodes :

double Impot() qui permet de calculer le montant de l'impôt que doit payer le propriétaire de l'habitation à raison de 2F par m2.

void Affiche() qui permet d'afficher les trois attributs de la classe Habitation.

et un constructeur à trois paramètres permettant d'initialiser une instance de la classe Habitation :
Habitation(String P, String A, double S);

Question 1 D´efinissez la classe Habitation

Question 2 Tester votre programme avec le code donn´e en annexe A.1.

Définition des classes HabitationIndividuelle et HabitationProfessionnelle

Objectif : Utiliser l'héritage pour définir de nouvelles classes, redéfinir des méthodes dans les classes héritières.

Le calcul de l'impôt d'une maison individuelle est différent de celui d'une habitation, il se calcule en fonction de la surface habitable, du nombre de pièces et de la présence ou non d'une piscine. On compte 100 €/pièce et 500 € supplémentaire en cas de présence d'une piscine.

Question 3 Définir la classe `HabitationIndividuelle` qui hérite de la classe Habitation en utilisant l'en-tête suivant :

```
public class HabitationIndividuelle extends Habitation {
    ...
}
```

Ajouter les attributs `NbPieces` de type entier et `Piscine` de type booléen. Redéfinir les méthodes `Impot` et `Affiche`. La mméthode `Affiche` doit afficher, les attributs `proprietaire`, `adresse` et `surface` de la classe `Habitation`, et les attributs `NbPieces` et `Piscine` propres à la classe `HabitationIndividuelle`. La première ligne de la méthode `Affiche`, commencera par l'instruction `super.Affiche()`, permmettant d'appeler la méthode `Affiche` de la classe mère `Habitation`.

Question 4 Tester votre programme avec le code donné en annexe A.2.

Le calcul de l'impôt d'une habitation à usage professionnel est également différent de celui d'une habitation. Il se calcule en fonction de la surface occupée par le bâtiment et du nombre d'employés travaillant dans l'entreprise. On compte 1000 € supplémentaire par tranche de 10 employés.

Question 5 Définir la classe HabitationProfessionnelle qui hérite de la classe Habitation en utilisant l'en-tête suivant :

```
public class HabitationProfessionnelle extends Habitation {
    ...
};
```

Ajouter l'attribut `NbEmploye`s de type entier. Redéfinir les méthodes `Impot` et `Affiche`. La méthode `Affiche` doit afficher, en plus des attributs `proprietaire`, `adresse` et `surface`, l'attribut `NbEmploye`s.

Question 6 Tester votre programme àl'aide du code donné en annexe A.3.

Gestion des habitations d'une commune

Objectif : Mettre en oeuvre le mécanisme de liaison dynamique.

On désire à présent calculer l'impôt local des habitations (individuelles ou professionnelles) d'une commune. Pour cela, on utilise une collection d'objets représenter par un tableau oùchaque él'ment désigne une habitation individuelle ou professionnelle.

Question 7 Tester votre programme à l'aide du code donné en annexe A.4

Annexe A : Programmes de test

A.1 Test de la classe Habitation

```
// Classe TestHabitation permettant de tester la classe
Habitation public class TestHabitation{
    public static void main (String[]
        args){ double I;

        // creation d'un objet de type Habitation
        Habitation H = new Habitation("Jean", "METZ",
        120);
        // calcul de
        l'impôt I =
        H.Impot();
        // affichage des attributs de la classe
        Habitation H.Affiche();
    }
}
```

A.2 Test de la classe HabitationIndividuelle

```
// Classe TestHabitationIndividuelle pour tester la classe
// HabitationIndividuelle
public class TestHabitationIndividuelle{
    public static void main (String []
    args){
        double I;

        // creation d'un objet de type HabitationIndividuelle
        HabitationIndividuelle HI = HabitationIndividuelle new("Paul", "METZ", 120, 5, False);
        // calcul de
        l'impôt I =
        HI.Impot();
        // affichage des attributs de la classe
        HabitationIndividuelle HI.Affiche();
    }
}
```

A.3 Test de la classe HabitationProfessionnelle

```
// Classe TestHabitationProfessionnelle permettant de tester la classe
// HabitationProfessionnelle
public class TestHabitationProfessionnelle {
```

```
    public static void main (String [] args){
        double I;

        // creation d'un objet de type HabitationProfessionnelle
        HabitationProfessionnelle HP = new HabitationProfessionnelle("ImportExport", "METZ",
        2500, 130)
        // calcul de
        l'impôt I =
        HP.Impot();
        // affichage des attributs de la classe
        HabitationProfessionnelle HP.Affiche();
    }
}
```

A.4 Test de la classe Collection

```
//  Définition de la classe
 TestCollection public class
 TestCollection{
    public static void main (String [] args){
        Habitation [] TableauHabitation;

        // creation d'un tableau contenant 5 habitations
        TableauHabitation = new Habitation [5];
        // Initialisation des éléments du tableau
        TableauBatiment[0] = new HabitationProfessionnelle("ImportExport", "METZ", 2500, 130);
        TableauBatiment[1] = new HabitationProfessionnelle("Export", "METZ", 250, 10);
        TableauBatiment[2] = new HabitationIndividuelle("Paul", "METZ", 100, 5, false);
        TableauBatiment[3] = new HabitationProfessionnelle("Import", "METZ", 1200, 90);
        TableauBatiment[4] = new HabitationIndividuelle("Jean", "METZ", 130, 6, true);
        // affichage des attributs de chaque élément du
        tableau for (int i = 0; i < 5; i++)
            TableauHabitation[i].Affiche();
       // calcul et affichage de
        l'impôt for (int i = 0; i < 5;
        i++)
            TableauHabitation[i].Impot();
    }
}
```