Programmation Orientée Objet

TP2 : Héritage et Polymorphisme (Partie 2) 2022-2023

Exercice 4 :

Dans cet exercice, nous allons définir une classe Personne qui permettra de gérer un individu et nous allons ensuite spécialiser ces classes par le mécanisme d'héritage en deux sous-classes qui chacune hériteront de la classe Personne: la classe Etudiant et la classe Enseignant.

- La classe **personne** possède le nom, le prénom et l'âge comme attributs.
- La classe **Etudiant** possède un tableau qui stock les 5 notes de chaque étudiant.
- Créer la méthode **RemplirNotes**() qui remplit les 5 notes dans le tableau à la ligne de commande en utilisant la classe Scanner.
- Créer la méthode **CalculMoyenne()** qui calcul la moyenne de chaque étudiant.
- La classe **Enseignant** possède un tableau pour remplir les modules affectés pour chaque enseignant (max 4 modules).
- Ajouter la méthode **AffecterModules()** qui affecte les modules de chaque enseignant dans le tableau.
- Ajouter
- Redéfinir la méthode **toString()** pour l'affichage dans les 3 classes, et adapter la méthode pour ne faire appel qu'à l'affichage des objets dans le main tout en ayant les résultats de traitement.
- Essayer les opérations de Upcasting et de DownCasting suivantes :

```
public static void main(String[] args) {
    personne person= new personne("Mohamed","Soualmi",50);
    Etudiant etud= new Etudiant ("Akram","Mounoub",19);
    Enseignant ens= new Enseignant ("Islam","Ali",30);

    person =  ens;
    ens = person;
    etud=(Etudiant) person;

}
```

Qu'est ce vous remarquez ?

Exercice 5 :

Nous allons dans cet exercice créer une hiérarchie entre 3 classes, on commencera à créer la classe mère qui s'appelle **« Animal »**. Cette classe possède le **nom** comme un attribut et en ajoutera aussi l'attribut **végétarien** de type de booléen.

- Par la suite créer les classes filles « **Chat** » et « **Eléphant** ». Sachant que la classe Chat possède un attribut nommé **couleur** pour désigner les couleurs du chat.
- Utiliser la méthode toString() pour l'affichage dans les 3 classes.

Maintenant, on revient à la classe Animal pour ajouter la méthode ***Diagnostique(double poids)***.
Cette méthode teste le poids d'un chat ou d'un éléphant pour signaler son obésité dans ces deux cas :

- Si le poids du chat est supérieur à 8kg on affiche que le chat est obèse, et même chose pour l'éléphant si son poids est supérieur à 1000kg.

Après il faut redéfinir la même méthode ***Diagnostique(int age)***, pour tester le degré de vieillissement d'un chat et d'un éléphant comme suite :

- Si l'âge du chat est supérieur à 10 ans on affiche que le chat est vieux, et pour un éléphant l'âge de vieillissement doit être supérieur à 30 ans.

- La méthode diagnostique doit être capable de faire la différence entre un chat et un éléphant en utilisant ***instanceof***.
- Essayer de faire appel à la méthode diagnostique avec une valeur de type entier, qu'est-ce que vous remarquer ? Essayer ensuite avec une valeur double.
- On veut maintenant créer un objet de type Animal et faire appel à la méthode diagnostique. Comment on fait ? Proposez une solution à base de polymorphisme ou de casting d'objets pour résoudre cela.