Programmation Orientée Objet

TP2 : Héritage et polymorphisme 2020

Exercice 1

a) Créer une classes nommée *Personne* :
   - Créer la classe *Personne* représentant une personne qui est caractérisée dans le cadre de notre TP par un nom, un prénom et un âge.
   - Doter la classe *Personne* d'un constructeur initialisant tous ses champs.
   - Écrire la méthode afficher de signature *void afficher()* qui affiche le nom, le prénom et l'âge d'une personne.

   Créer la classe TpH ne comportant pas de champs et comportant uniquement la méthode main.
   - Au sein de la méthode main, créer une instance de type *Personne* dont les valeurs des attributs seront données en ligne de commandes

b) Tester la classe *Personne* en exécutant le code de la classe TpH.

c) Nous désirons avoir la possibilité de connaître le nombre de Personnes créées. Modifier le programme et ré-exécuter la méthode main afin d'afficher cette valeur.

d) Créer une classe nommée *Enseignant* héritant de la classe *Personne* et ayant deux champs : *nbHeures* de type entier et *module* de type chaine de caractères.
   - Créer un constructeur pour la classe *Enseignant* appelant celui de la classe mère et initialisant les champs propres à la classe *Enseignant*.

e) Créer de même une classe *Etudiant* héritant de *personne* et ayant comme champs : *matricule* de type entier, *notes* de type tableau d'entier indiquant les notes de l'étudiant dans les différents modules qui sont au nombre de 8, et *moyenne* de type réel.
   - Créer un constructeur pour la classe *Etudiant* appelant celui de la classe mère et initialisant les champs propres à la classe *Etudiant*.
   - Doter la classe *Etudiant* de la méthode *float claculMoyenne(int [] notes)* qui calcule la moyenne de l'étudiant en supposant que tous les modules ont un coefficient de 1.
   - Redéfinir la méthode *afficher* de *personne* dans les deux classes *Enseignant* et *Etudiant* afin d'afficher les champs qui leurs sont propres.

f) Modifier la méthode main dans TpH en créant un tableau de 5 éléments de type *Personne* contenant deux éléments de type *Enseignant* et 3 de type *Etudiant*. Les éléments du tableau seront initialisés à travers les constructeurs.
   - Afficher pour chaque élément du tableau les informations le concernant en précisant s'il s'agit d'un enseignant ou d'un étudiant en utilisant l'opérateur *instanceOf*
   - Surcharger la méthode *afficher* de la classe personne en créant une méthode de signature *afficher(boolean reduit)*. Si reduit est à vrai, l'affichage se limite au nom, et au prénom de la personne.

g) Modifier la méthode main en ajoutant des appels aux différentes méthodes *afficher* et exécuter afin de tester les nouvelles modifications.

Exercice 2 *Héritage de classe et constructeur*

a) Un site internet est spécialisé dan la vente de livres pour enfant. Ces livres sont soit des bandes dessinées, soit des albums à colorier. Un livre est défini par son titre, son auteur, son prix et son nombre de pages. Les bandes dessinées sont soit en couleur soit en noir et blanc alors l'utilisateur a la possibilité de colorier une page d'un album présenté. Proposer et implémenter et tester sous Eclipse une solution à ce problème.

b) Le site web veut donner la possibilité aux utilisateurs de revendre un livre et de s'échanger deux bandes dessinées si elles ont un prix équivalent. Modifier le programme précédent pour prendre en compte ces fonctions supplémentaires.

c) Enfin, le site web veut étendre son offre d'un site culturels à des films (DVD) qui sont définis eux-aussi par un titre, un auteur et un prix mais avec en plus une information sur la durée du film. Comment modifier la hiérarchie de classe pour intégrer ces modifications ? Programmer-le.

Exercice 3 *Héritage de classe et polymorphisme*

Une entreprise possède plusieurs types de collaborateurs :

— Les employés qui sont payés en fonction du 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 35 heures) sont payées 25 de plus.

— Les managers qui sont payés de la même façon que les employés mais dont les heures supplémentaires sont payées 50

— Les commerciaux qui sont payés une somme fixe, équivalente à 35 heures au tarif horaire des employés, à laquelle on ajoute 1Le travail demandé est de modéliser cette situation à l'aide de classes JAVA.

a) Écrivez les trois classes **Employe**, **Manager** et **Commercial**, modélisant respectivement les employés, les managers et les commerciaux, sachant que :
   — Chaque collaborateur a un nom qui ne doit pas pouvoir être modifié.
   — Chaque classe doit disposer de deux constructeurs. L ?un prend seulement en paramètre le nom du collaborateur, l ?autre prend le nom et toutes les informations nécessaires au calcul du salaire de la semaine.
   — Dans chaque classe, le calcul du salaire se fait via une méthode dont la signature est double calculSalaire().

b) Écrivez une classe **Gestion** comportant uniquement la méthode **main**. Dans cette méthode, vous créerez plusieurs collaborateurs de plusieurs types et vous les enregistrerez tous dans un même tableau. Vous veillerez à utiliser les différents types de constructeur. La méthode **main** affichera le salaire hebdomadaire de tout le personnel dans une boucle parcourant le tableau des collaborateurs. Pour un collaborateur donné, l'affichage aura la forme :

```
Salaire de Gates : 2008 euros.
```