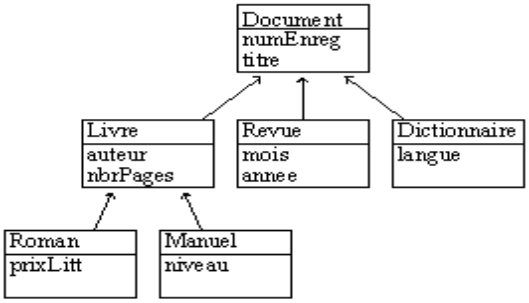
P.O.O en JAVA

Série 3 : Délégation, Héritage, polymorphisme

Exercice 1 : Gestion d'une bibliothèque

L'objectif est d'écrire une application traitant des documents de nature diverse : des livres, qui peuvent être des romans ou des manuels, des revues, des dictionnaires, etc…

Tous les documents ont un numéro d'enregistrement (un entier) et un titre (une chaîne de caractères). Les livres ont, en plus, un auteur (une chaîne) et un nombre de pages (un entier). Les romans ont éventuellement un prix littéraire (un entier conventionnel, parmi : GONCOURT, MEDICIS, INTERALLIE, etc.), tandis que les manuels ont un niveau scolaire (une chaîne). Les revues ont un mois et une année (des entiers) et les dictionnaires ont une langue (une chaîne de caractères convenue, comme "anglais", "allemand", "espagnol", etc.). Tous les divers objets en question ici (livres, revues, dictionnaires, romans, etc.) doivent pouvoir être manipulées en tant que documents.



1- Définissez les classes Document, Livre, Roman, Manuel, Revue et Dictionnaire, entre lesquelles existeront les liens d'héritage que la description précédente suggère. Dans chacune de ces classes définissez.
   - le constructeur qui prend autant arguments qu'il y a de variables d'instance et qui se limite à initialiser ces dernières avec les valeurs des arguments, une méthode public String toString() produisant une description sous forme de chaîne de caractères des objets,
   - Définissez également des « accesseurs » publics get... permettant de consulter les valeurs de ces variables.
   - Écrivez une classe exécutable TestDocuments qui crée et affiche plusieurs documents de types différents.

2. Une bibliothèque sera représentée par un tableau de Documents. Définissez une classe Bibliotheque, avec un tel tableau pour variable d'instance et les méthodes :
   - Bibliotheque(int capacité) - constructeur qui crée une bibliothèque ayant la capacité (nombre maximum de documents) indiquée,
   - void afficherDocuments() - affiche tous les ouvrages de la bibliothèque,
   - Document document(int i) - renvoie le ième document,
   - boolean ajouter(Document doc) - ajoute le document indiqué et renvoie true (false en cas d'échec),
   - boolean supprimer(Document doc) - supprime le document indiqué et renvoie true (false en cas d'échec)
   - void afficherAuteurs() - affiche la liste des auteurs de tous les ouvrages qui ont un auteur (au besoin, utilisez l'opérateur instanceof).

3. Définissez, une classe Livrotheque dont les instances ont les mêmes fonctionnalités que les Bibliotheques mais sont entièrement constituées de livres.

Cette classe hérite des attributs et des méthodes de la classe Bibliotheque. Elle possède un constructeur qui prend comme argument la capacite et les méthodes suivantes :
   - public boolean ajouter(Document doc)
   - public Livre livre(int i)
   - public void afficherAuteurs

Comment optimiser dans la classe Livrotheque la méthode afficherAuteurs ?

Exercice 2 : Employés d'une entreprise

L'objectif est de définir un ensemble de classes permettant de programmer le calcul des salaires hebdomadaires des employés d'une entreprise.

Cette entreprise comporte plusieurs types d'employés :

- Des employés 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 39 heures) sont payées 30 % de plus que les heures normales.
- D'autres employés, payés de la même façon, mais leurs heures supplémentaires sont payées 50 % de plus que les heures normales.
- Les commerciaux sont payés avec une somme fixe à laquelle on ajoute 1 % du chiffre d'affaires qu'ils ont fait dans la semaine.

Modélisez cette situation à l'aide de classes.

- Vous donnerez un nom à chacun des employés. On ne pourra modifier le nom d'un employé.
- Vous commencerez par écrire une classe Employe dont hériteront les autres classes.
- Pour ne pas avoir trop de modificateurs vous simplifierez en ne donnant qu'un seul modificateur setInfosSalaire pour entrer ou modifier les informations brutes nécessaires au calcul des salaires (nombre d'heures de travail, chiffre d'affaire,...). N'essayez pas de faire du polymorphisme avec cette méthode.
- Les classes comporteront au moins 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.
- Le calcul des salaires se fera dans la méthode getSalaire() qui sera utilisée pour faire du polymorphisme.

- Une classe Paie comportera une unique méthode main() qui entrera les informations sur des employés des différents types (au moins 3 commerciaux). Les employés seront enregistrés dans un tableau employes.

- 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 setInfosSalaire. Pour au moins un autre employé, vous utiliserez le constructeur pour entrer les informations sur le salaire.

- La méthode main affichera le salaire hebdomadaire de chacun des employés dans une boucle "for" qui parcourra le tableau des employés. Vous utiliserez le polymorphisme avec un accesseur pour le salaire. L'affichage aura exactement la forme : « Ahmed gagne 25000 dh".

- Vérifiez les calculs des salaires !