Univesité Kofi Annan
Faculté de MIAGE : Licence 3
TP N° 5 : Héritage

Exercice 1 : animaux et zoo

Le directeur d'un zoo vous demande de gérer l'ensemble de son parc animalier. Un zoo gère des animaux. Un animal est caractérisé par :
- le fait qu'il soit vivant ou non,
- par son âge,
- son âge maxi (au-delà duquel il meurt) et,
- son nom.

Le système doit être capable de :
- créer un Animal : le seul paramètre donné est le nom de l'animal. Age est alors mis a 0. On suppose qu'à la création, un animal est vivant,
- afficher les caractéristiques de l'animal ;
- vieillir un animal. Par défaut un animal vieillit de 1 an; s'il atteint son âge limite, il meurt.
- faire mourir un animal : un animal ne peut mourir que s'il est vivant. Lorsqu'il meurt, il faut en avertir le zoo afin qu'il enlève l'animal de sa ménagerie;
- chaque type d'animal pousse un cri particulier. Tant que l'on ne connaît pas plus précisément de quel type d'animal il s'agit, il est impossible de préciser son cri.
- il doit être impossible de créer directement un animal. Par contre, on doit pouvoir créer un type d'animal en le faisant appartenir à une famille.
- un animal est rattaché à un zoo. Il doit pouvoir changer de zoo.

Les animaux peuvent être répartis en trois familles : les chiens, les oiseaux et les singes. Chaque famille peut avoir des caractéristiques et des comportements particuliers.

Les chiens :
- les chiens sont caractérisés par leur race;
- âge maximum d'un chien est 30 ans;
- afficher les caractéristiques d'un chien équivaut à afficher le fait qu'il soit un chien, ses caractéristiques en tant qu'Animal et à préciser sa race;
- lorsqu'un chien vieillit, il vieillit de 7 ans;
- le cri d'un chien est "Ouah-Ouah"

Les oiseaux :
- âge maximum d'un oiseau est 20 ans;
- afficher les caractéristiques d'un oiseau équivaut à afficher le fait qu'il soit un oiseau et ses caractéristiques en tant qu'Animal;
- lorsqu'un oiseau vieillit, il vieillit de 10 ans;
- le cri d'un chien est "Cui-Cui"

Les singes :
- un singe a un prénom,
- âge maximum d'un singe est 30 ans;
- afficher les caractéristiques d'un singe équivaut à afficher le fait qu'il soit un singe, ses caractéristiques en tant qu'Animal et à préciser son prénom;
- un singe vieillit normalement;
- un singe parle.

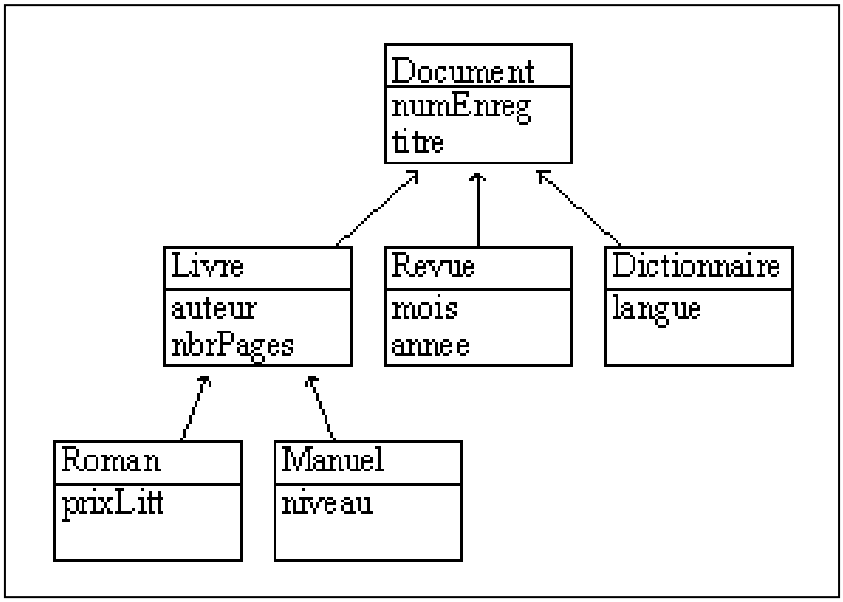
* Les attributs décrivant les animaux doivent être accessibles par tous les types d'animaux. Par contre, ils ne doivent pas être vus par les objets ne dérivant pas de Animal.
* Les méthodes sont accessibles par tout le monde.
Un zoo est constitué d'un ensemble d'animaux et est dirigé par un directeur.
Il doit être possible de :
- créer un zoo en précisant son directeur
- ajouter un animal
- fêter l'anniversaire du zoo c'est-à-dire faire vieillir chacun des animaux;
- faire une tournée de nourriture : nourrir chacun des animaux. Quand un animal est nourri, il crie;
- faire mourir un animal : il faut l'enlever de la liste des animaux du zoo.

Questions :

1 Faire le diagramme de classe correspondant :
1.1 définir les classes
1.2 définir les propriétés des classes
1.3 définir les méthodes
1.4 définir la visibilité des propriétés et des méthodes
2 définir les associations / composition
3 définir les héritages
4 définir les autres méthodes
5 Ecrire les programmes java correspondant

Exercice 2 : bibliothèque

Pour la gestion d'une bibliothèque, on nous demande d'écrire une application traitant des *documents* de nature diverse : des *livres*, des *revues*, des *dictionnaires*, etc. Les livres, à leur tour, peuvent être des *romans* ou des *manuels* comme on le voit sur le diagramme ci-dessous :



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 (un entier). 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 objets en question ici (*livres*, *revues*, *dictionnaires*, *romans*, etc.) doivent pouvoir être manipulées en tant que *documents*.

A. 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,
- si vous avez déclaré **`private`** les variables d'instance (c'est conseillé, sauf indication contraire) 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 différentes

B. Une bibliothèque sera représentée par un *ArrayList de documents*. Définissez une classe **`Bibliotheque`**, avec un tel ArrayList pour variable d'instance et les méthodes :

- **`Bibliotheque()`** - constructeur qui crée une bibliothèque en instanciant un objet ArrayList dont la référence sera stockée dans listDocs.
- **`void afficherDocuments()`** - affiche *tous* les ouvrages de la bibliothèque,
- **`Document document(int i)`** - renvoie le i^{ème} document,
- **`ajouter(Document doc)`** - ajoute le document indiqué dans listDocs,
- **`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`**)

C. Définissez, avec un effort minimal, une classe **`Livrotheque`** dont les instances ont les mêmes fonctionnalités que les **`Bibliotheques`** mais sont *entièrement constituées de livres*. Comment optimiser dans la classe **`Livrotheque`** la méthode **`afficherAuteurs`** ?

Exercice 3 : **classes abstraites**

Dans un établissement d'enseignement, on trouve trois catégories de personnes : des administratifs (au sens large, incluant des techniciens) représentés par la catégorie secrétaire, des enseignants et des étudiants. Chaque personne est caractérisée par son nom et prénom, son adresse (rue et ville) qui sont des attributs privés et communs à toutes les personnes. En plus des variables d'instances que nous venons de citer, la classe Personne contient une variable de classe (nbPersonnes) qui comptabilise le nombre de personnes dans l'établissement, une méthode abstraite (ecrirePersonne()) qui pourrait écrire les caractéristiques d'une personne, une méthode toString() qui fournit une chaîne de caractères correspondant aux caractéristiques (aatributs) d'une personne, une méthode de classe (nbPersonne()) qui écrit le nombre total de personne et le nombre de personnes par catégorie et une méthode modifierPersonne(String rue, String ville) qui permet de modifier l'adresse d'une personne et appelle ecrirePersonne() pour vérifier que la modification à bien été faite.

La classe Secretaire possède toutes les caractéristiques d'une personne en plus d'un numéro de bureau. De même, un Enseignant est une Personne enseignant une spécialité (mathématique, englais, informatique…). Un Etidiant est une Personne préparant un diplôme (diplomeEnCours). Les méthodes pour Enseignant et Etudiant sont similaire à celles de Secretaire. Une privée static dans chaque compte le nombre de personnes créés dans chaque catégorie. Une méthode statique du même nom que la variable fournit la valeur de cette variable static (nbSecretaire, nbEnseignant, nbEtudiant).

On vous demande de donner le schéma UML de cette hiérarchie de classes et d'écrire un programme Java permettant de les manipuler.