Université Paris 7 — Java
Licence Informatique — Année 2005-2006

TD n°6

Héritage

Exercice 1 On modélise une application devant servir à l'inventaire d'une bibliothèque. Elle devra traiter des documents de nature diverse : des livres, des dictionnaires, et autres types de documents qu'on ne connaît pas encore précisément mais qu'il faudra certainement ajouter un jour (articles, bandes dessinées...). Tous les documents possèdent un *numéro d'enregistrement* et un *titre*. À chaque livre est associé, en plus, un *auteur* et un *nombre de pages*, les dictionnaires ont, eux, pour attributs supplémentaires une *langue* et un *nombre de tomes*. On veut manipuler tous les articles de la bibliothèque au travers de la même représentation : celle d'un document.

1. Définissez les classes `Document`, `Livre` et `Dictionnaire`. Définnissez pour chacune un constructeur permettant d'initialiser toutes ses variables d'instances.
2. Définissez une classe `Bibliothèque` réduite à une méthode main permettant de tester les classes précédentes (ainsi que les suivantes).
3. Définissez la classe `ListeDeDocuments` permettant de créer une liste vide de documents, puis y adjoindre une fonction permettant d'ajouter un document.
4. Dans la classe `ListeDeDocuments` définissez une méthode `tousLesAuteurs()` qui affiche la liste des numéros des documents de la liste avec, pour chacun, l'éventuel auteur.
5. Redéfinnisez la méthode `toString()` dans la classe `Document` ainsi que dans les classes `Livre` et `Dictionnaire` et qui renvoie une chaîne de caractères décrivant un document, un livre ou un dictionnaire. Ajoutez alors dans la classe `ListeDeDocuments` une méthode `tousLesDocuments()` qui affiche consécutivement la description de tous les documents.
6. Proposez quelques lignes de codes à ajouter à la classe `Bibliotheque` afin de tester la classe `ListeDeDocuments`.

Exercice 2 Écrire trois classes `Figure`, `Carre`, et `Rectangle`, telles que :

1. `Figure` a des attributs abscisse et ordonnée, ainsi qu'une couleur (encodée par un entier).
2. `Carre` et `Rectangle` héritent de `Figure`, mais ont en plus une ou deux longueur pour les côtés.
3. `Figure` a un attribut privé `Vector` référençant **toutes les instances** de sa classe et de ses sous classes.
4. `Figure` a une méthode statique `getInstances()` renvoyant ce vecteur.
5. `Carre` et `Rectangle` redéfinissent cette méthode `getInstances()` de manière à ne récupérer que les instances qui correspondent à leur type.