Université Paris Cité

Compléments en Programmation Orientée Objet
TD et TP n° 6 : Classes scellées, énumérations

1 Fin des TP précédents

Faire/finir les exercices des TP 4 et 5 dont le thème n'a pas été abordé lors du TP noté.
Demandez à votre enseignant de TP si vous avez un doute.

2 Cartes

Exercice 1 : Jeu de cartes – Modélisation

Le jeu de cartes classique se compose de 54 cartes : 52 cartes dans 4 familles correspondant à 4 enseignes différentes (pique, cœur, carreau, trèfle), chacune composée de 13 valeurs (as, deux, trois, ... neuf, dix, valet, dame, roi), ainsi que 2 jokers.

1. Pour le jeu à 52 "cartes normales" (c'est-à-dire non-joker), proposez une modélisation du type `CarteNormale` en utilisant les enregistrements (**`record`**) et les énumérations (**`enum`**), permettant de représenter toutes les cartes de ce jeu de cartes.

   Pour une carte donnée, il faut être capable de récupérer :
   - son enseigne (pique/cœur/carreau/trèfle, appelée souvent "couleur", mais pour éviter les confusions, nous éviterons ce terme)
   - sa valeur
   - sa couleur (rouge ou noir)
   - l'information si cette carte une figure (valet, dame, roi) ou pas

   Déterminez bien les propriété primaires (qu'on représente par des attributs) et les propriétés dérivées (déduites à partir des attributs). Toutes les propriétés sont bien sûr lisibles grâce à des accesseurs (*getters*), et les attributs sont privés.
2. Le type `CarteNormale` est-il récursivement scellé ? (c.-à-d. il est impossible de créer de sous type direct ou indirect depuis un autre fichier)

   Le cas échéant, apportez les modifications nécessaires pour que cela soit bien le cas.
3. Ce type est-il récursivement immuable ? (c.-à-d. il est impossible de créer des instances modifiables, y compris en créant un nouveau sous-type)

   Le cas échéant, apportez les modifications nécessaires pour que cela soit bien le cas.
4. Proposez une modélisation pour le type `Joker`, sachant qu'il y a exactement deux Jokers dans un jeu. Mêmes questions concernant l'extensibilité et la mutabilité.
5. Créez un type `Carte` qui serait la somme du type `Joker` et du type `CarteNormale` défini auparavant.

   Quel mot-clé faut-il utiliser pour s'assurer que `Carte` n'ait pas d'autre sous-type que ces deux-là ?

   Quel autre mot-clé ajouter pour permettre de définir `CarteNormale` et `Joker` dans des fichiers sources différents de `Carte.java` ?

   Réfléchissez aussi vous devez définir `Carte` comme classe ou interface.
6. `Carte` est-il aussi récursivement scellé et immuable ? (Si ce n'est pas le cas, corrigez ce problème)
7. Question subsidiaire : réécrivez ce programme seulement avec les possibilités qui existaient dans Java 11.

8. Programmez des méthodes statiques permettant de générer : un jeu de 52 cartes standard (retourné sous la forme d'une collection), un jeu de 32, un jeu de 54 (avec jokers),
9. Même question pour générer une collection contenant plusieurs exemplaires d'un jeu de carte (par exemple, pour jouer au Rami, il faut deux jeux de 54).

   Que signifie alors `a.equals(b)` pour 2 cartes dans cette collection ? Comment faire pour savoir si `a` et `b` sont deux cartes distinctes (possiblement avec le même marquage) ?

Exercice 2 : Jeu de cartes – 8 américain (pour ceux qui ont vraiment fini le reste)

En utilisant les cartes programmées à l'exercice 1, programmez le jeu du 8 américain (cf. Wikipédia [1]), ancêtre du célèbre jeu Uno. Programmez juste la version présentée comme "version minimale". Une version simple à 2 joueurs conviendra (humain contre humain, prévoyez un code simple pour désigner les cartes en ligne de commande).
Réfléchissez s'il faut préférer des méthodes définies et redéfinies dans les types de cartes, ou bien des méthodes extérieures utilisant des `switch` (et du *pattern matching*, si vous utilisez Java 21). Les deux techniques ont leurs avantages et inconvénients et ne se prêtent pas forcément à tous les contextes.

1. `https://fr.wikipedia.org/wiki/8_am%C3%A9ricain`