

Prénom _____
 NOM _____

20

Interrogation de Java

(Date)

Remarques :

- ↪ vous travaillerez dans une répertoire `interro2`¹,
- ↪ vous créerez un package `gxxxxxx.interro2` pour l'occasion,
- ↪ vous pouvez utiliser **toutes les notes** que vous voulez ainsi que les programmes qui se trouvent dans votre répertoire,
- ↪ l'interro se fait sur linux1,
- ↪ vous disposez de 2 heures

L E PROBLÈME. Nous allons écrire un petit programme permettant à l'ordinateur de jouer ² à **Bataille**.

Nous simplifions le jeu afin que le programme se termine dans un temps raisonnable. La simplification concerne la récupération des cartes. Normalement lorsqu'un joueur gagne, il récupère ses cartes et celles de l'adversaire. Dans notre cas, les cartes seront écartées et nous compterons un point par *main ramassée*.

Afin de gérer tout ça, vous écrirez une classe `Carte` représentant une carte, une classe `JeuCarte` représentant un jeu de 52 cartes et une classe `Bataille` permettant de jouer.

La classe `Carte`

Cette classe comprend deux attributs, un `int` représentant la valeur de la carte (1->13) et un `String` représentant sa couleur (Pique, Coeur, Carreau ou Trèfle).

Vous écrirez les méthodes :

- ↪ `Carte(int, String)`, le constructeur³,
- ↪ `String toString()`, la représentation textuelle d'une carte,
- ↪ `int compareTo(Object)`, une méthode permettant de comparer 2 cartes ... nous éviterons les vraies batailles en donnant une valeur aux couleurs. Dans l'ordre des "forces" on aura : Pique, Coeur, Carreau et Trèfle⁴.

Remarque Définir une telle méthode `compareTo` revient à implémenter l'interface `Comparable`.

La classe `JeuCarte`

Un jeu de carte est un ensemble de 52 cartes⁵. Cette classe n'a qu'un seul attribut représentant cet ensemble de cartes. Le constructeur de la classe devra créer les 52 cartes et les stocker dans "l'ensemble".

¹Remarquez l'absence de majuscule au nom de répertoire

²Finalement pourquoi ne pourrait-il pas s'amuser aussi

³Vous pouvez déclarer 4 constantes de type `String` représentant Pique, ... histoire que le jeu de cartes soit homogène.

⁴Comme au Whist je crois

⁵Jusque là rien de bien neuf

Pour représenter cet ensemble, vous avez plusieurs possibilités (suivant votre connaissance de l'API du langage).

- ↪ Vous utilisez un tableau de cartes que vous gérez de bout en bout.
- ↪ Vous utilisez une *Collection* définie dans le langage. Voici un extrait de code ...

```
...
private List jeu ;
...
public JeuCarte () {
    jeu = new ArrayList();
    ...
    jeu.add( new Carte(1, Carte.PIQUE));
    ...
}
```

Cette classe implémentera également une méthode `distribuer(List l1, List l2)` répartissant les cartes entre les deux joueurs. Vous déciderez si ce choix de paramètres vous convient.

Vous n'êtes pas obligé de mélanger les cartes avant de les distribuer.

La classe **Bataille**

La classe `Bataille` est la classe mettant le tout en oeuvre. Elle contient une méthode `main` instanciant une `Bataille` et demandant de la jouer.

La classe possède les attributs suivant :

- ↪ deux attributs de type `List` (ou `Carte[]`) représentant la main (ou le pot) d'un joueur,
- ↪ deux attributs de type `int` représentant les points d'un joueur (le joueur remporte un point par main gagnée).
- ↪ un attribut de type `JeuCarte`

Remarque Si vous optez pour l'utilisation de l'API Java, plutôt que la gestion de vos tableaux de cartes à la main, pour stocker les cartes de chacun des joueurs, je vous conseille la classe `java.util.LinkedList` car elle offre les méthodes `removeFirst()` et `addLast()` qui seront bien utiles pour gérer le tas de cartes.