Chapitre 1

TD TP POO-COO

Objectifs

1. Lecture et compréhension d'un programme Objet (Bataille)
2. Manipulation d'objets (déclaration, appel de méthodes ...)
3. Programmer des méthodes dans des classes existantes.

Récupérez sur Moodle les classes mises à votre disposition (et importez-les dans un projet Intellij ou Eclipse) : Carte.java, FabriqueDeJeuxde32Cartes.java, Joueur.java, Partie.java, Main.java.

1. (fondamental) Exécutez ce programme pour vérifier qu'il fonctionne
2. (fondamental) Lisez attentivement le code Java de toutes les classes.
3. (fondamental) Suivez mentalement la suite des appels de méthodes réalisés lorsque le programme est exécuté (y compris les appels à des constructeurs).
4. (fondamental)
   Dans le main d'une classe Essai :
   — Créez deux cartes (par exemple valet de carreau et 5 de trèfle) et affichez-les (en lisant l'aide mémoire sur `System.out.println` et `toString`

> **Aide mémoire :**
> **println et toString**
> — La méthode prédéfinie `System.out.println`, lorsqu'elle est appelée avec un objet en paramètre se charge d'appeler la méthode `toString` de l'objet. Autrement dit, `System.out.println(o)` est équivalent à `System.out.println(o.toString())`.
> Pour afficher un objet, il suffit donc de définir la méthode `toString` dans la classe de l'objet puis de faire un simple `System.out.println(o)`.
> — La méthode `toString` définie dans la classe `ArrayList` appelle en boucle le `toString` sur chacun des éléments de la liste. Pour afficher une liste d'objet `l`, il suffit donc de définir la méthode `toString` dans la classe des objets puis de faire un simple `System.out.println(l)`.

   — Créez un jeu de cartes trié et affichez-le.
   — Créez un jeu de cartes battu et affichez-le.
   — Comparez les deux première cartes du jeu de cartes battu et en utilisant des if...else affichez selon le cas :
   "la première est plus grande que la deuxième" ;
   "la deuxième est plus grande que la première" ;
   "les deux première cartes sont de même valeur".
   — Créez un joueur en utilisant le premier constructeur de la classe Joueur (celui qui ne prend en paramètre que le nom du joueur), affichez le nom de ce joueur (uniquement le nom), puis affichez la représentation du joueur donnée par la méthode toString.

— Faites-lui ramasser un carte puis affichez le joueur, à nouveau faites-lui ramasser une deuxième carte puis affichez le joueur.
— Faites-lui poser une carte (que vous affichez), puis affichez le joueur.
— Avec un if..else affichez selon le cas "Le joueur a au moins une carte" ou bien "Le joueur n'a aucune carte"
— Faites-lui poser encore une carte (que vous affichez), puis affichez le joueur.
— Avec un if..else affichez selon le cas "Le joueur a au moins une carte" ou bien "Le joueur n'a aucune carte"
— Créez deux Joueurs avec un jeu de cartes distribué aléatoirement entre eux (vous devez pour cela utiliser la méthode creerEtDistribuerJeu32 de la classe FabriqueDeJeuxDe32Cartes ainsi que le deuxième constructeur de la classe Joueur), affichez ces joueurs.

5. (fondamental)
— Développez dans la classe Carte la méthode suivante :

```
/*
* Retourne :
* un entier positif (de valeur absolue quelconque) si la valeur de la carte est strictement superieure
*       a la valeur de la carte c recue en parametre
* un entier negatif (de valeur absolue quelconque) si la valeur de la carte est strictement inferieure
*       a la valeur de la carte c recue en parametre
* zero si les deux cartes sont de meme valeur.
*/

public int compareTo (Carte c) {}
```

— Utilisez cette méthode dans le main de votre classe Essai (vous pouvez utiliser compareTo pour comparer deux cartes et afficher la plus grande ou bien si elles sont égales).

6. (consolidation)
— Développez dans la classe Joueur une méthode qui retourne le nombre de cartes que possède le joueur.
— Utilisez cette méthode dans le main de votre classe Essai.
— Modifiez la classe Partie pour que à chaque tour de jeu le programme affiche le nombre de cartes qu'il reste à chaque joueur.

7. (fondamental)
— Développez dans la classe Joueur la méthode :

```
/*
* Methode qui retourne la plus forte carte que possede le joueur.
*/
public Carte tricher1() {
```

— Utilisez cette méthode dans le main de votre classe Essai.
— Modifiez le programme pour que une fois tous les trois tours le joueur 1 utilise la méthode tricher1

8. (avancé) Implantez dans ce programme les vraies règles de la bataille.

9. (avancé)
— Développez dans la classe Joueur la méthode :

```
/*
* Methode qui retourne la plus petite carte strictement superieure a la carte recue en parametre.
* Si aucune carte n'est plus forte, c'est la plus faible carte qui est retournee
*/

public Carte tricher2(Carte carteAdverse) {}
```

— Utilisez cette méthode dans le main de votre classe Essai.
— Modifiez le programme pour que le joueur 2 utilise la méthode tricher2 à chaque tour.