Travail pratique

But : Le but du présent travail est de vous faire faire des méthodes permettant de découper un problème en sous-problèmes. Votre classe devra comprendre des méthodes en plus de la méthode main qui appellera les autres méthodes (ou certaines d'entre elles) de votre classe.

Énoncé : Le programme que vous devrez concevoir permet à son utilisateur de parier un certain montant d'argent sur le résultat d'un jeu de hasard. Ce jeu consiste à faire piger deux ou trois cartes d'un jeu de cartes ordinaire par l'ordinateur et de parier sur ces cartes. Une "pige" est constituée des cartes pigées. Les paris qui peuvent être faits sont les suivants :

1. La pige contient au moins une figure (As, roi, dame ou valet).
2. Toutes les cartes de la pige sont strictement inférieures à 5. On considère que l'as a la valeur 1.
3. La somme de la valeur des cartes de la pige est paire.
4. Toutes les cartes de la pige ont la même couleur (aux cartes, il y a 4 couleurs possibles [et non deux] : coeur, carreau, trèfle ou pique).
5. Toutes les cartes de la pige ont la même valeur.

Au début du programme, celui-ci demande au joueur un nombre pour initialiser le procédé aléatoire pour pouvoir choisir au hasard des cartes. Puis, il lui demande le montant initial qu'il a en sa possession. Ce montant doit être supérieur ou égal à 4 $ (coût de l'achat de deux cartes). Ensuite, il lui demande s'il désire jouer une partie. Dans l'affirmative, il demande au joueur combien de cartes il veut piger (2 ou 3 cartes). Il faut s'assurer que le joueur a assez d'argent pour acheter le nombre de cartes désiré (par exemple, s'il a 5 $ en poche, il peut acheter au plus 2 cartes, il faut donc en tenir compte lorsqu'on demande au joueur le nombre de cartes qu'il désire). Chaque carte pigée coûtera 2 $ au joueur. Puis le programme lui demande sur quel résultat il veut parier (1, 2, 3, 4 ou 5). Ensuite, l'ordinateur pigera les cartes, et donnera les résultats. Voici un tableau donnant l'argent gagné si le résultat correspond au pari choisi. (Notons par nbCartes, le nombre de cartes pigées.)

Pari	Argent gagné, s'il y a lieu, en $
1	17 - 2 * nbCartes
2	4 * nbCartes
3	2 * nbCartes + 2
4	$3 * (nbCartes - 1)^2 + 2$
5	$2 * (nbCartes - 1)^3 + 2$

Le programme affiche premièrement les cartes qu'il a pigées, ensuite si le joueur a gagné ou perdu son pari et le montant qu'il a gagné s'il y a lieu. Enfin, il affiche le montant dont dispose le joueur une fois comptabilisé l'argent qu'il a gagné s'il y a lieu.

Si le joueur a moins de 4 $, le programme se termine. Sinon, le programme lui demande à nouveau s'il désire jouer une partie et ainsi de suite.

Exemple d'exécution :

À venir

Aide : La classe `PaquetDeCartes` vous sera fournie vous permettant de piger aléatoirement des cartes d'un jeu de 52 cartes. **Vous devez utiliser cette classe.** **Vous ne devez, en aucun cas, modifier cette classe**. Chacune des 52 cartes sera codée par un nombre entier de 0 à 51 de la façon suivante :

0, 1, 2, ... 11, 12 : Les coeurs ♥ : as, deux, trois, ..., dix, valet, dame, roi.
13, 14, 15, ..., 24, 25 : Les carreaux ♦ : as, deux, trois, ..., dix, valet, dame, roi.
26, 27, 28, ..., 37, 38 : Les trèfles ♣ : as, deux, trois, ..., dix, valet, dame, roi
39, 40, 41, ... 50, 51 : Les piques ♠ : as, deux, trois, ..., dix, valet, dame, roi

Exemples : la carte numéro 7 correspond au 8 de coeur, la carte 20 correspond au 8 de carreau, la carte 40 correspond au 2 de pique.

Voici quelques méthodes publiques (que vous pouvez donc utiliser dans votre Tp) disponibles dans cette classe :

La méthode `initialiserJeuDeCarte` doit être appelée (**une seule fois**) au début de votre programme (méthode `main`) avec une valeur choisie par l'utilisateur de votre programme. Ceci permettra de pouvoir générer les mêmes cartes à chaque exécution du programme. Ceci facilite la mise au point du programme. Voici une méthode que vous pouvez mettre dans votre programme pour ce faire :

```
public static void initialiserLeJeu () {
    int germe;
    System.out.println ( "Entrez un nombre entier pour initialiser le jeu : " );
    germe = Clavier.lireInt ();
    PaquetDeCartes.initialiserJeuDeCarte ( germe );
    PaquetDeCartes.brasser();
} // initialiserLeJeu
```

La méthode `piger` retourne un entier correspondant à la carte sur le dessus du paquet de cartes.

Je vous invite à aller voir le contenu de cette classe pour voir les méthodes disponibles.

Exigences logicielles, conseils et contraintes pour la deuxième partie

1. Les différentes saisies (lecture de oui/non, lecture du montant initial du joueur, lecture de la sorte de pari, lecture du nombre de cartes, etc) **doivent** se faire chacune dans une méthode.

2. Vous **devrez** avoir les méthodes suivantes et les appeler, soit de la méthode `main`, soit d'une autre méthode :

```
/**
 * Affiche la carte selon sa couleur et sa valeur
 * @param carte doit etre entre 0 et 51 inclusivement
 */
public static void afficherCarte ( int carte ) { ... }

/**
 * Détermine si les deux cartes ont la même valeur (ex.: deux rois, deux 9)
 * @param carte1 et carte2 doivent etre entre 0 et 51 inclusivement
 * @return true si les deux cartes ont la même valeur, false sinon
 */
public static boolean memeValeur ( int carte1, int carte2 ) { ... }
```

```
/**
 * Détermine si les deux cartes ont la même couleur.
 * Les 4 couleurs possibles sont : coeur, carreau, trèfle et pique.
 * @param carte1 et carte2 doivent etre entre 0 et 51 inclusivement
 * @return true si les deux cartes ont la même couleur, false sinon
 */
public static boolean memeCouleur ( int carte1, int carte2 ) { ... }

/**
 * Détermine si une carte est une figure : As, roi, dame ou valet.
 * @param carte doit etre entre 0 et 51 inclusivement
 * @return true si la carte est une figure, false sinon
 */
public static boolean estUneFigure ( int carte ) { ... }
```

Vous devrez aussi avoir d'autres méthodes.

3. Vous devez suivre les normes de programmation énoncés dans le document disponible sur le site : http://cyberzoide.developpez.com/java/javastyle/

4. **Sont interdits** :
   - les `break` ailleurs que dans un `switch`,
   - l'opérateur `?`,
   - un `return` ou un `break` dans un `for`, un `while`, un `do...while`, un `if`
   - un `return` dans un `switch`
   - l'énoncé `continue` est aussi interdit.
   - plusieurs `return` dans une même méthode.

3. Utiliser des variables réelles uniquement lorsque requis. Autrement dit, un compteur ne doit pas être `float` ou `double`.

6. Vous devez utiliser la classe `Clavier` fournie sur le site du cours pour la saisie des données. Vos programmes seront testés avec cette classe.

7. L'affichage des résultats doit se faire à la console.

Précisions supplémentaires :

1. Lorsque le programme demande à l'utilisateur s'il désire jouer, les seules réponses valables sont "oui" et "non", minuscules et majuscules acceptées (j'ai fait cette méthode en classe). Exemple :

```
Voulez-vous jouer une partie ?
n
*** vous devez repondre par oui ou non :
no
*** vous devez repondre par oui ou non :
oUi
```

2. Il ne doit y avoir aucune ligne blanche affichée à la console pour faciliter la validation.

3. Si l'utilisateur, après une partie, a 3 $ ou moins en poche, le programme doit se terminer puisqu'il ne peut acheter deux ou trois cartes. Exemple :

```
Vous disposez maintenant de 1 $
Vous n'avez plus assez d'argent, vous ne pouvez continuer.
Merci d'avoir joue avec moi !
Vous quittez avec 1 $ en poche.
```

4. Si l'utilisateur a 4 $ ou 5 $, il ne peut acheter 3 cartes, donc, le programme ne lui donne pas le choix du nombre de cartes à piger. Exemple :

```
Entrez le montant dont vous disposez :
4
Voulez-vous jouer une partie ?
oui
Je vais piger deux cartes.
```