Séance 3: Jouer à Pierre-Feuille-Ciseaux

Université Paris Cité

Objectifs:

- Un programme qui interagit avec l'utilisateur
- Création d'un petit jeu en plusieurs manches

L'objectif de ce TP est de programmer un jeu de Pierre-Feuille-Ciseaux dont les deux joueurs sont l'ordinateur et l'utilisateur. Une partie se jouera en une succession de manches et le programme maintiendra une fiche de score afin de déterminer le vainqueur à la fin.

Lors de ce TP, il faudra donc faire interagir l'utilisateur de votre programme avec l'ordinateur en lui demandant son coup. Il faudra interpréter sa réponse et réagir en conséquent. Il faudra également tenir à jour une fiche de score.

Pour cette séance, les programmes de chaque exercice sont à réaliser dans un même fichier que vous enregistrerez dans le bon sous-répertoire.

Exercice 1 (Jeu en une manche, ⋆)
Dans ce premier exercice, nous créerons un jeu de Pierre-Feuille-Ciseaux en une manche.

Rappel : *le jeu de Pierre-Feuille-Ciseaux est un jeu de mains où les deux joueurs choisissent simultanément chacun un élément parmi pierre, feuille et ciseaux. Le vainqueur est décidé selon les principes suivants : la pierre casse les ciseaux, les ciseaux coupent la feuille, la feuille enveloppe la pierre.*

1. *Écrire une fonction* `tirage` *sans paramètre qui renvoie dans une chaîne de caractères le choix – aléatoire – de l'ordinateur.*

   Contrat:
   Un appel de `tirage()` *renvoie une chaîne de caractères contenant aléatoirement* `"Pierre"`, `"Feuille"` *ou* `"Ciseaux"`.

   Note : *La fonction* `randRange(a, b)` *suivante renvoie aléatoirement un entier* `n` *compris entre les paramètres* `a` *et* `b` *avec* `a <= n < b`. *Si vous l'utilisez, pensez à ajouter la directive* `import java.util.Random;` *au tout début de votre programme.*

```
static Random rand = new Random();

public static int randRange(int a, int b) {
    return (rand.nextInt(b-a) + a);
}
```

2. *Écrire une fonction* `coupJoueur` *sans paramètre qui utilise la fonction* `System.console().readLine()` *pour demander le choix du joueur et le renvoie dans une chaine.*

   Contrat:
   Un appel de `coupJoueur()` *demande à l'utilisateur de choisir entre Pierre, Feuille et Ciseaux et renvoie une chaîne de caractères correspondant à ce choix.*

3. *Écrire une fonction* `uneManche` *qui fait jouer l'ordinateur contre l'utilisateur à Pierre-Feuille-Ciseaux et qui renvoie une chaîne de caractères indiquant le vainqueur.*

   Contrat:
   Un appel de `uneManche()` *fait jouer une manche à l'utilisateur et l'ordinateur et renvoie le résultat sous la forme d'une chaîne de caractères :*
   — `"J"` *si le joueur remporte la manche*
   — `"O"` *si l'ordinateur remporte a manche*
   — `"E"` *si il y a égalité*

□

Exercice 2 (Jeu en plusieurs manches 🔍, ⋆)
Dans cet exercice, nous créerons le jeu de Pierre-Feuille-Ciseaux en plusieurs manches.

Écrire une procédure `chifoumi(n)` *qui fait jouer une partie de Pierre-Feuille-Ciseaux en* `n` *manches entre l'utilisateur et l'ordinateur. La fonction doit afficher le vainqueur, son nombre de victoires et un résumé de la partie sous forme d'une chaîne de caractères.*

Contrat:
Un appel de `chifoumi(5)` *fait jouer* `5` *manches de Pierre-Feuille-Ciseaux à l'ordinateur et l'utilisateur, et il affiche à la fin le nom du vainqueur ainsi que le résultat de chaque manche sur une seule ligne :* `JOEJJ` *par exemple si le joueur utilisateur a gagné la première, la quatrième et la cinquième manche et que la troisième a été une égalité.*

□

Exercice 3 (Pour aller plus loin 🔍, ⋆— ⋆⋆⋆)
Dans cet exercice, nous ajoutons quelques fonctionnalités à notre jeu. Chaque question est indépendante. Pour ne pas perdre de code, il est conseillé de travailler sur une copie `chifoumi2(n)` *du programme de l'exercice précédent.*

1. *Modifier votre programme pour que le joueur choisisse le nombre de manches avant le début de la partie.*
2. *Modifier votre programme pour qu'une partie ne se joue plus en* `n` *manches mais qu'une partie soit terminée dès qu'un des joueurs a au moins* `n` *victoires.*
3. *Modifier votre programme pour qu'il vous fasse jouer à Pierre-Feuille-Ciseaux-Lézard-Spock.*
   Les premières règles s'appliquent toujours, mais il faut ajouter que le lézard mange le papier, empoisonne Spock, est écrasé par la pierre et est décapité par les ciseaux. Spock vaporise la pierre, casse les ciseaux, et il est discrédité par le papier.
4. *Proposez une (ou plusieurs) fonction* `coupOrdi` *qui choisit le coup de l'ordinateur selon une autre stratégie que le tirage aléatoire uniforme. Cette fonction peut utiliser, ou non, un ou plusieurs paramètres, selon vos envies. Testez vos stratégies face à la stratégie du tirage aléatoire uniforme.*

□