PARIS DESCARTES IUT – Département Informatique

Bases de la programmation orientée objet

– Projet –

Le Tic-Tac-Toe et ses variantes

1 Présentation

Le but du projet est de commencer par une application très simple puis de l'enrichir étape par étape. À chaque nouvelle étape, vous aurez à réorganiser le code issu de l'étape précédente pour pouvoir y introduire les nouvelles fonctionnalités demandées. Toutefois, vous prendrez soin à préserver au sein de la nouvelle version, les fonctionnalités de la version antérieure.

1.1 Une première application

Le premier programme que vous devez réalisez doit permettre à deux joueurs de s'affronter au Tic-Tac-Toe. Le Tic-Tac-Toe, aussi appelé Morpion et OxO en Belgique, est un jeu de réflexion se pratiquant à deux joueurs au tour par tour dont le but est de créer le premier un alignement. Nous commençons par le jeu dans sa forme la plus simple.

Le jeu se joue sur une grille de 3×3. Deux joueurs s'affrontent. Ils doivent remplir chacun à leur tour une case de la grille avec le symbole qui leur est attribué : `O` ou `X`. Le gagnant est celui qui arrive le premier à aligner trois symboles identiques, horizontalement, verticalement ou en diagonale. La partie est nulle si toutes les cases sont occupées et qu'aucun joueur n'a réalisé un alignement. Il est coutume de laisser le joueur jouant `X` effectuer le premier coup de la partie.

Pour ce premier programme (et les suivants), l'interface avec l'utilisateur vous est imposée. Initialement, ainsi qu'après chaque coup joué, votre programme doit afficher la grille de jeu. Une grille vide doit être affichée comme suit (sans aucun espace superflu) :

```
   1   2   3
1 [ ] [ ] [ ]
2 [ ] [ ] [ ]
3 [ ] [ ] [ ]
```

Lorsqu'une case contient un symbole `X` ou `O` (o majuscule), vous devez afficher la lettre correspondante entre les crochets.

Votre programme doit inviter le joueur courant à saisir son coup en affichant le message "`Au joueur X` (resp. `O`) `de jouer :  `". Le joueur saisit le numéro de ligne puis le numéro de colonne en

les séparant par un tiret. La saisie de n'importe laquelle des chaînes suivantes devrait être acceptée par votre programme : "`1-2`", "`01-0002`", "␣`1`␣`-`␣␣`2`␣".

Si le coup n'est pas valide (la chaîne saisie ne respecte pas le format ci-dessus, les numéros de ligne ou de colonne sont trop petits ou trop grands ou encore si ces numéros désignent une case déjà occupée), votre programme doit afficher le message "`Coup invalide :  `" et attendre la saisie d'un nouveau coup.

C'est à votre programme de détecter que la fin de partie est atteinte. Selon la situation, votre programme doit afficher le message "`Le joueur X` (resp. `O`) `remporte la partie`" ou le message "`Partie nulle`".

Une fois la fin de partie détectée et le message de fin affiché, votre programme doit se terminer. Seuls les affichages et saisies indiqués ci-dessus doivent être réalisés.

1.2 Une deuxième application

Votre programme doit être enrichi de manière à ce que les joueurs puissent choisir s'ils veulent jouer au Tic-Tac-Toe ou au Morpion. Le choix du jeu doit être réalisé au lancement de l'application via des paramètres (le tableau de chaînes de caractère souvent nommé `args` qui est reçu en paramètre par toute méthode statique `main`) dont vous préciserez le format.

Dans le jeu de Morpion, les grilles ont une taille quelconque. Les règles du jeu sont modifiées comme suit.

- La partie ne se termine plus au premier alignement mais continue en alternant les coups des deux joueurs jusqu'à ce que toutes les cases soient occupées.
- Un joueur ne peut poser un symbole que sur une case étant adjacente (horizontalement, verticalement ou en diagonale) à une case déjà occupée. Au premier coup, le placement est libre.
- Un même symbole ne peut compter que pour un alignement. Dès qu'un alignement est formé, les symboles qui le composent ne peuvent plus concourir à la réalisation d'un autre alignement. Ces symboles sont dits être fermés. Les symboles non encore fermés sont dits être ouverts.
- En fin de partie, le joueur ayant fait le plus d'alignements gagne. La partie est nulle en cas d'égalité.

Pour faciliter la visualisation du jeu, les symboles encore ouverts doivent être représentés par une majuscule et ceux déjà fermés par une minuscule. Par exemple, le coup "`5-1`" joué par `X` à partir de la grille représentée à gauche ci-dessous conduit à la grille représentée à droite. Les trois `X` formant un nouvel alignement, ils sont à présent fermés et sont donc représentés en minuscule.

```
   1   2   3   4   5   6
1 [ ] [ ] [ ] [ ] [ ] [ ]
2 [ ] [O] [ ] [O] [ ] [ ]
3 [ ] [ ] [X] [ ] [ ] [ ]
4 [ ] [X] [ ] [ ] [ ] [ ]
5 [ ] [ ] [ ] [ ] [ ] [ ]
```

```
   1   2   3   4   5   6
1 [ ] [ ] [ ] [ ] [ ] [ ]
2 [ ] [O] [ ] [O] [ ] [ ]
3 [ ] [ ] [x] [ ] [ ] [ ]
4 [ ] [x] [ ] [ ] [ ] [ ]
5 [x] [ ] [ ] [ ] [ ] [ ]
```

1.3 Une troisième application

Vous devez étendre votre programme de manière à ce qu'il soit possible de jouer à une partie d'un des jeux présentés ci-dessous. Comme dans l'application précédente, les arguments du programme doivent permettre à l'utilisateur de choisir le jeu auquel il veut jouer (`Tic-Tac-Toe`, `Morpion` ou l'un des jeux ci-dessous).

- Le nombre de symboles à aligner est fixé en début de partie (≥ 3 et ≤ 5).
- Ce n'est plus des alignements qu'il faut faire mais des formes particulières (une croix par exemple).
- La grille est initialement remplie aléatoirement d'autant de `X` que de `O` (plus un `X` ou un `O` choisi lui aussi aléatoirement si le nombre de cases est impair). Un coup ne consiste plus à déposer un symbole mais à permuter un `X` avec un `O`. Les symboles permutés peuvent être ouverts ou fermés. Un point est remporté si cette permutation conduit à réaliser une forme particulière de symboles ouverts (un alignement de 3 symboles, par exemple). Le joueur qui remporte le point ne dépend pas de qui joue le coup mais du symbole composant la forme. Si c'est un `X` (resp. `O`), le point est remporté par le joueur `X` (resp. `O`). Ainsi, une même permutation peut conduire à augmenter le score des deux joueurs.

2 Qui, quoi et quand?

Votre projet doit être fait en binôme. Les groupes de 3 ne sont pas acceptés. Évitez de faire votre projet tout seul (soit vous êtes très fort et des personnes ont besoin de votre aide, soit vous avez des difficultés et il faut vous faire aider).

Vous devez porter une attention particulière à la rédaction de votre dossier. Sa qualité est déterminante pour l'évaluation de votre travail. Votre dossier doit être un unique document `pdf` dont la composition est la suivante :

- Une page de garde indiquant le nom des membres du binôme, l'objet du dossier.
- Une table des matières de l'ensemble du dossier.
- Une brève introduction du projet.
- Le diagramme UML des classes formant vos applications (les attributs et méthodes des classes, interfaces et énumérations ne doivent pas être indiqués, par contre, leurs inter-dépendances et l'organisation des paquetages doivent être précisées de manière exhaustive).
- Le code Java des tests unitaires de vos classes.
- Le code Java complet de votre application.
- Un bilan du projet (les difficultés rencontrées, ce qui est réussi, ce qui peut être amélioré).

Nous vous rappelons que le critère principal de notation est la structuration de votre application. Votre rapport doit mettre en avant la qualité de celle-ci en démontrant que les points d'extension qu'elle propose permettent l'intégration aisée de nouvelles fonctionnalités. Des exemples d'extension sont des arguments convaincants.

Vous devez rendre votre rapport complet imprimé le **vendredi 10 janvier 2020**. De plus, vous devez déposer dans le puits (`BPO`) une archive portant votre nom et contenant l'ensemble de vos fichiers sources (et uniquement cela). Seules les archives au format `zip` seront acceptées.