UNIVERSITÉ DE FRANCHE-COMTÉ

Projet d'APOO

Analyse du jeu de blocage avec polyominos

Antoine CUINET
Gaspar QUENTIN

Table des matières

Introduction

Le projet est un projet de fin d'année de première année de licence informatique à l'université de Franche-Compté.

Ce rapport décrit et explique la conception de ce projet, qui est le jeu de blocage avec polyominos.

Le jeu est codé en Java et le joueur joue contre un ordinateur depuis le terminal.

Le but du jeu est relativement simple, chaque joueur dispose d'un ensemble de pièces, qui sont des polyominos, et qu'il place tour à tour sur un plateau, qui est sous forme de grille rectangulaire.
Il s'agit d'un jeu de blocage : le perdant est le premier joueur qui ne peut plus placer de pièce.
Le jeu se déroule en mode humain contre ordinateur : le joueur humain interagit via des affichages et des saisies sur la console.

Ce jeu à été coder par CUINET Antoine et QUENTIN Gaspard.

Les fonctionnalités

Les fonctionnalités du jeu sont multiples.

Toutes fonctionnalités demandées dans le sujet ont été implantés.

Le joueur peu donc choisir son pseudo, choisir les pièces qu'il souhaites poser (Dominos, Triominos ou Tetrominos, elles ont toutes étés implantés) s'il lui en reste (des vérifications ont étés mises en places sur le nombre de pièce possible de choisir ainsi que sur la saisie qui doit être correcte).

Un affichage de chaque étapes du processus de choix d'une pièce est présenté au joueur ainsi que l'affichage de la pièce finalement choisi sur la grille (si elle ne sort pas de la grille et qu'elle ne chevauche aucune autre pièce, des vérifications demande au joueur de recommencer sa saisie si celle-ci est incorrecte).

Ensuite, l'ordinateur peu également jouer, il choisi et place ses pièces sur la grille de la même façon que le joueur.

Un affichage de la grille est présenté au joueur une fois que l'ordinateur à jouer avec un affichage de la pièce sur la grille ainsi qu'un message indiquant le type de pièce poser et la position de celle-ci.

Enfin, des fonctions ont été mises en place pour détecter la fin d'une partie. A ce moment, la partie s'arrête automatiquement et un message est afficher spécifiant le nom du gagnant.

Les classe

Le jeu dispose de 10 classes.

3.1 La classe Main

La classe Main est comme son nom l'indique la classe principale de notre projet. Celle ci contient la méthode statique main qui s'occupe de créer les bons objets et d'appeler les bonnes méthodes afin de créer une partie. Cette classe contient également les méthodes permettant à l'utilisateur de jouer.

3.2 La classe Computer

La classe Computer s'occupe de faire jouer l'ordinateur contre le joueur. Les pièces placées par l'ordinateur sont toutes choisies de manière aléatoire et placées de manière aléatoire également.

3.3 La classe Piece

La classe Pièce est une classe abstraite servant de modèles pour les classes Domino, Triomino et Tetromino qui en hériteront. Cette classe contient plusieurs méthodes virtuelles pures telles que getPositions() (qui renvoie un tableau de Position représentant la pièce dans un espace orthonormé) ou encore la méthode toString() qui s'occupera d'afficher la pièce pour aider l'utilisateur. Ces dernières seront donc entièrement redéfinies dans les classes enfant.

3.4 La classe Grid

La classe Grid représente la grille de jeu. Elle se charge également de l'affichage du plateau de jeu. Elle est composée d'un tableau de 2 dimensions d'objets de type Case. Elle comporte également les méthodes permettant au joueur et à l'ordinateur de voir si il est possible de placer une pièce (méthode isPiecePlaceable()) et de la placer (méthode placePiece).

3.5 La classe Position

Cette classe est constituée de deux attributs : une positon en abscisse sur la grille et une position en ordonnée. Ces deux attributs sont des entiers. Elle possède également plusieurs méthodes utiles comme la méthode add() qui permet d'additionner deux positions entre elles.

3.6 La classe Case

Cette classe représente une case sur le plateau. Une case contient une pièce qui peut être nulle. Elle possède également une redéfinition de la méthode toString() permettant d'afficher un rond si une pièce se trouve sur cette case et qu'elle appartient au joueur, un croisillon si une pièce se trouve sur cette case mais que cette pièce est une pièce ennemie et rien sinon.

3.7 La classe Matrix

Une pièce étant constituée de plusieurs objets Position, pour trouver l'orientation voulue pour placer la pièce dans la grille, nous avons dû avoir recours à quelques petits calculs matriciels. Cette classe sert donc principalement à calculer des coordonnées rotationnées à l'aide de cette formule mathématique :

$$\begin{pmatrix} x' \\ y' \end{pmatrix} \quad \begin{pmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}$$

3.8 La classe Domino

La classe Domino, qui hérite de la classe Piece, se charge de créer des dominos avec le bon type en fonction du nombre d'instances déjà crées. Cette classe comporte une surchage de la métode toString() permettant d'afficher toutes les différentes dispositions des dominos et de la méthode getPositions() renvoyant toutes les positions représentant la pièce crée dans un repère orthonormé.

3.9 La classe Triomino

Cette classe fonctionne de façon identique à celle de la classe Domino.

3.10 La classe Tetromino

La classe Tetromino comporte la même logique d'implémentation que ces deux classes soeures, la classe Domino et la classe Triomino.

Initialisation de jeu

Au début du jeu, un affichage permettant de saisir de pseudo du jeueur est mis à l'écran.

Une fois cette saisie faite, une explication du jeu, de l'ordinateur (qui se nome Yumi), ainsi que des pièces disponible est présenté.

4.1 Dimensions du jeu

Le plateau de jeu est un rectangle de 12 cases par 10 cases, qui sont numérotés en lettres (de A à L) en abscisse et en chiffre (de 0 à 9) en ordonné.

4.2 Distribution des pièces

Au début du jeu, le joueur comme l'ordinateur ont à disposition un même nombre de pièce.

Il est distribué 3 Dominos, 6 Triomino posable en 2 positions avec 3 pièces disponibles par position, et 9 Tétrominos posable en 7 positions avec 1 pièce disponible par position sauf pour les tétrominos de formes I et T qui sont aux nombre de 2.

Ainsi, chaque joueur un nobre de 60 cases sur un plateau faisant en tout 120 cases.

Description des conditions

Plusieur types de conditions sont utilisé afin de faire fonctionner au mieux le jeu.

5.1 Placement des pièces

// ici à faire

5.2 La partie est finie

// ici à faire

Jeu de l'ordinateur

Par manque de temps pour réaliser ce projet, notre ordinateur choisi ses pièce et les places aléatoirement.

Il place tout de même les pièce de façon coherente. C'est à dire qu'il place, tout comme le joueur, un nombre pré-définie de pièces d'un certain type et d'une certaine forme.

Autrement dit, l'ordinateur possède au début de la partie les mêmes pièce que le joueur et des vérifications sont faites afin q'il ne pose que les pièce dont il a le droit de poser et dans le nombre qu'il possède.

Ajouts/libertés prises

Afin de rajouter de la créativité, l'affichage de la grille est réalisé de façon originale.

Le placement des pièces dans la grilles se fait en couleur, de façon à distinguer facilement celui que les a posées et afin de faire un affichage coloré (rouge pour l'ordinateur et bleu pour le joueur).

Le nom du jeueur ainsi que celui de l'ordinateur sont eux aussi en couleur (de même, rouge pour l'ordinateur et bleu pour le joueur).

Enfin, l'affichage des pièces en vert, qui est guidé tout au long du processus de choix d'une pièce, permet au joueur de facilement se rendre compte des pièces qu'il lui sont possible de poser.

Conclusion

Pour conclure ce rapport, nous avons produit un jeu amusant, autant dans la conception que dans son utilisation.

Certain problèmes ont été rencontrer comme celui de faire les tétrominons qui à été très long (autant dans la création des pièce que dans les affichages). De plus, celui de distinguer les formes possibles de chaques pièces, en plus des différentes pièces et de leurs orientations.

Ces problèmes ont bien évidement trouver leurs solutions et nous ont permis de nous surpasser dans la création de ce jeu.

Concevoir ce jeu nous à également permis de mettre en pratique les cours appris tout au long de l'année et spécifiquement la programmation orienté objet.