1 Description générale

Pour le projet du second semestre, vous allez développer un jeu appelé *LineUp3*. Les deux joueurs de ce jeu disposent chacun d'une couleur et de trois pions de cette couleur, pions qu'il faut réussir à aligner sur le plateau.

Le plateau de jeu est représenté par un graphe régulier de forme polygonale dont le nombre de côté est fixé. La figure 1 illustre un plateau à 3 côtés (à gauche) ainsi qu'un plateau à 4 cotés (à droite). Un plateau de jeu est composé de positions (représentées par des cercles) et par des arcs représentant les mouvements possibles. On peut imaginer des plateaux dont le nombre de côtés est quelconque, mais seuls les plateaux à 3 ou 4 côtés sont pris en compte.

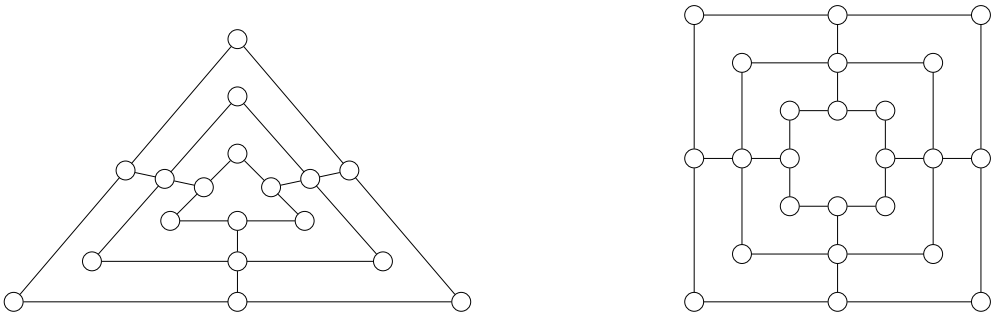

FIGURE 1 – Plateau à 3 côtés à gauche, plateau à 4 côtés à droite.

Déroulement d'une partie Une partie de *LineUp3* se déroule en deux phases: une phrase de déploiement et une phase de confrontation. En début de partie, le plateau est vierge. Dans la première phase, la seule action utilisable par les joueurs est la pose d'un pion. Ils vont tour à tour poser un de leur pions sur le plateau. Une fois tous les pions en jeu, la seconde phase est enclenchée. Les joueurs disposent d'autres actions qu'ils peuvent utiliser à tour de rôle afin d'aligner leurs trois pions. Parmi ces actions, certaines sont utilisables indéfiniment (comme le déplacement d'un pion), alors que d'autres sont limitées à un nombre d'utilisation donné (comme la pause d'un piège). Certaines actions sont permanentes (comme le déplacement) alors que d'autres ont un effet à durée limitée (comme le blocage d'un arc).

Conditions de victoire Un joueur gagne dès que ses trois pions sont alignés. Afin de préciser ce qu'est un alignement dans un tel graphe, nous allons définir une règle de numérotation des positions, **à respecter impérativement**. Une position est définie comme un couple (x, y) où

- x représente la couche à laquelle le pion appartient (la couche intérieure ayant le numéro le plus faible);
- y représente le numéro d'ordre du sommet dans sa couche, en sachant qu'ils sont numérotés dans le sens des aiguilles d'une montre à partir du sommet "*en haut à gauche*".

Les figures 2 et 3 illustrent respectivement un plateau à 3 et à 4 côtés dont les positions ont été entièrement numérotées.

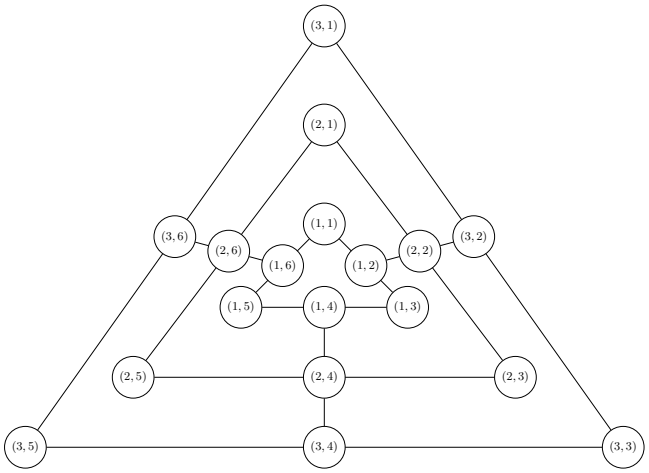

FIGURE 2 – Numérotation des sommets d'un plateau à 3 côtés.

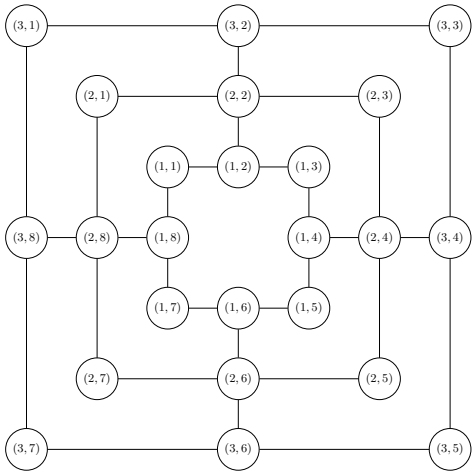

FIGURE 3 – Numérotation des sommets d'un plateau à 4 côtés.

D'après la figure 2 représentant un plateau de taille 3, les triplets de positions suivants sont des situations gagnantes: `{(3,5),(3,6),(3,1)}` ou `{(1,6),(2,6),(3,6)}`. En revanche, les triplets de positions suivants ne sont pas alignés: `{(2,2),(2,1),(2,6)}` ou `{(1,1),(2,1),(3,1)}`. Les triplets représentant une situation gagnante sur un plateau à 3 côtés ne sont pas forcément gagnants sur un plateau à 4 côtés. En effet, comme l'illustre la figure 3, les triplets des positions suivantes correspondent à des situations gagnantes: `{(2,7),(2,8),(2,1)}` ou `{(1,2),(2,2),(3,2)}` mais pas `{(3,8),(3,1),(3,2)}`.

2 Spécifications

Votre application doit être extensible sans nécessiter de lourdes modifications du code déjà écrit. Que vous ayez le temps ou non de développer les aspects optionnels suggérés, l'organisation du code de votre projet doit permettre leur intégration. Cela doit être visible sur le schéma UML que vous devez produire et garder à jour systématiquement.

— Les deux joueurs sont humains mais il doit être possible d'intégrer un joueur artificiel;
— La phase de déploiement doit pouvoir être réalisée par les joueurs ou générée aléatoirement;
— On doit pouvoir choisir la taille du plateau que l'on veut;
— Il doit être possible de spécifier un fichier de configuration permettant de démarrer une partie pré-configurée;
— La première version textuelle du jeu doit pouvoir être étendue par l'utilisation d'une interface graphique;
— Une partie doit pouvoir être interrompue et reprise plus tard;
— Les actions possibles sont:
  - la pose d'un pion (utilisable autant de fois qu'il y a de pions à poser),
  - le déplacement d'un pion (utilisable indéfiniment dès que la seconde phase de jeu démarre),
  - le blocage d'un arc (utilisable un nombre de fois donné, action à durée limitée),
  - la pose/génération d'un piège (utilisable une seule fois),
  - ...
— La liste des actions fournies ici n'est pas exhaustive;
— Certaines actions sont configurables via le fichier de configuration;

3 Objectifs, conditions de réalisation et organisation du travail

Modalité d'évaluation Ce projet tutoré est évalué sous deux aspects différents: l'aspect conception et programmation orientée objet d'un côté, et l'aspect gestion de projet de l'autre. Le projet est décomposé en deux jalons, où chacun des deux aspects est évalué.

Suivi Le suivi du projet sera donc assuré par un enseignant de POO/COO ainsi que par un enseignant en gestion de projet. 3 heures par semaine dans l'emploi du temps sont dédiés à votre projet tuteuré. Une partie de ce temps est non-encadré. Une semaine sur deux, 1h30 est encadré par un enseignant. À vous de vous organiser en conséquence.

Lors des séances de suivi du projet, veillez à vous munir d'un diagramme UML à jour en tout temps. Ce schéma sera la seule base de discussion recevable. L'objectif de l'encadrant n'est pas tellement de discuter "code", mais "conception" ce qui n'est possible qu'à partir d'un schéma UML convenable. De même, il vous sera demandé de présenter des outils de gestion de projet à jour. Le défaut de présentation d'un diagramme UML à jour ou d'outils de gestion de projet à jour sera pénalisé.

Constitution des groupes Le projet est relativement conséquent et doit être réalisé par équipe de 4 ou 5. La constitution de ces groupes relève de votre responsabilité (sous la contrainte du groupe TD: pas d'équipe inter-groupe TD).

D'après le programme du DUT, ce projet demande au moins 30 heures de travail individuel par étudiant, soit 3 heures par semaine en moyenne. C'est à chacun d'entre vous de s'assurer qu'il fait sa partie du travail.

En cas de conflits dans le groupe, en parler à l'enseignant de TP au plus vite ; il/elle pourra jouer le rôle de médiateur pour vous aider à résoudre le problème. Aucun changement de groupe ne sera possible une fois les projets démarrés.

Jalons d'évaluation Avant le rendu de chaque jalon, il est demandé de préparer une présentation succincte (5 minutes maximum) de votre travail. Cette présentation doit mettre en valeur votre travail en général mais aussi mettre en exergue les spécificités de votre projet. Elle sera suivie de quelques questions qui nous serviront à mieux évaluer votre travail. Cette séance (de 20 minutes maximum) se fera donc en présence de l'équipe du projet au complet et des deux encadrants.

La note de projet tiendra compte de :

— la régularité du travail : un travail régulier témoigne d'une bonne organisation et une bonne coordination et sera récompensé. Vous aurez une meilleure note si vous travaillez sur le projet toutes les semaines (par rapport à tout faire la semaine qui précède la date de rendu);
— l'implication équitable de tous les membres du groupe : réussir à impliquer chacun quel que soit son niveau est une compétence très importante, et sera récompensée;
— mettez dans votre rendu tous les documents produits (rapport, maquettes, schémas), pas seulement le code;
— si une contribution n'est pas visible, la faire figurer dans le rapport. Toute contribution sera prise en compte. Cependant, un étudiant ne peut pas se consacrer uniquement à des tâches qui ne nécessitent pas de programmer. Chacun doit contribuer en écrivant du code original. Il n'est pas demandé que tous les membres d'un groupe écrivent la même quantité de code, mais il ne doit pas y avoir de grands déséquilibres. Si vous avez contribué avec quelque chose qui n'est pas visible (par ex. recherche de modèles, capture de la vidéo de présentation, préparation de l'archive à rendre, etc.), assurez vous que c'est mentionné dans le rapport.

4 Calendrier et livrables

4.1 Démarrage du projet

Toutes les notions nécessaires à la réalisation du projet n'ont pas encore été abordées. Il est tout de même possible de commencer l'implémentation de certaines parties. Le premier aspect auquel vous devez vous intéresser est l'environnement. Il vous faut coder l'environnement sous forme d'une matrice d'adjacence. Pour rappel, l'élément sur la i-ème ligne et j-ème colonne décrit la possibilité de se déplacer du sommet i au sommet j, dans les deux sens. Une fois l'environnement codé, il faut implémenter les méthodes nécessaires pour déterminer si une position est occupée, si un mouvement donné est possible, si trois positions données constituent un alignement... Pour chaque taille d'environnement, il vous faut constituer une liste de triplets "gagnants".

4.2 Jalon 1: le jeu simplifié en mode texte (semaine du 10 mai)

La première version du jeu est restreinte aux mécanismes fondamentaux du jeu. Cette version repose sur un mode texte uniquement. La phase de déploiement du jeu se résume à une affectation aléatoire des pions de chacun des joueurs. L'ensemble des actions disponibles est restreint : les joueurs peuvent seulement déplacer un pion ou bloquer un arc. Les paramètres éventuellement nécessaires seront spécifiés directement à la méthode `main`. Une représentation ASCII de l'environnement est fournie afin d'aider les étudiants.

À rendre Vous devez rendre une archive dont la structure est la suivante:

— un diagramme UML représentant votre application dans le répertoire `doc`;
— un fichier `readme.md` qui donne un premier aperçu du projet à la racine du projet;
— une archive `jar` exécutable qui permet de lancer votre programme à la racine du projet;
— un répertoire `data` qui contient les fichiers de données dont vous pourriez avoir besoin;
— un répertoire `src` qui contient votre code source
— un répertoire `doc` qui contient la documentation;
— un répertoire `test` qui contient les tests.

4.3 Jalon 2 (semaine du 7 juin)

Les caractéristiques obligatoires à ajouter sont:

— la possibilité de suspendre une partie pour la poursuivre plus tard en l'état;
— la phase de déploiement jouée par les joueurs (au lieu de l'affectation aléatoire);
— l'introduction des autres actions comme la pose d'un nouveau pion (pour la phase de déploiement) ou la pause d'un piège téléporteur aléatoire;
— l'utilisation d'une IHM;
— l'utilisation d'un fichier de configuration.

Vous devez inclure au moins un aspect optionnel comme le développement d'un joueur artificiel paramétrable (qui peut suivre différentes stratégies), la génération d'un plateau de jeu polygonal régulier à n côtés (et les conditions de victoire associées)… Cette liste est loin d'être exhaustive et chaque groupe peut proposer ses propres extensions (à condition qu'elles soient pertinentes d'un point de vue POO/COO).

4.4 Présentation orale de votre projet

À chaque rendu (Jalon 1 et jalon 2) est associée une séance de 3h pendant laquelle vous devrez présenter oralement votre projet. Cette séance est en présence du groupe complet et des deux enseignants (gestion de projet et programmation).

— 5 min de présentation pendant lesquelles vous nous présenterez les fonctionnalités de votre projet. N'hésitez pas à présentez des fonctionnalités partielles, que nous pourrons approfondir pendant les questions.
— 5 min de questions informatiques où nous entrerons un peu plus en détail sur la conception globale du projet, donc pensez à avoir un ou des diagrammes UML à jour ! Et nous parlerons aussi des algorithmes mis en oeuvre dans votre implémentation.
— 5 min de questions plus orientées gestion de projet.

5 Attendus en termes de gestion de projets

5.1 Le module

— Travailler en équipe.
— Communiquer entre les membres de l'équipe.
— Gérer les conflits éventuels.
— Trouver des solutions devant les problèmes rencontrés.
— Se répartir les rôles.
— Gérer un projet sur la durée.
— Finaliser le travail pour les jalons.
— Un coordinateur de projet sera choisit au sein de l'équipe.

5.2 Le module

— Pour chaque réunion, faire le compte rendu et le déposer sur moodle ou l'envoyer à l'enseignant.
— Définir les tâches et les regrouper en lots.

Groupe				Charge (en heures)				
Lots et tâches	Qui	Date		Total (jour)	Totale (heure)	Période 1	Période 2	Pé
		Début	Fin	0	0	0	0	
Lot 1				0,0	0	0	0	
Tâche1					0			
Tâche2					0			
Tâche3					0			
Tâche4					0			
Tâche5					0			
Tâche6					0			
Tâche7					0			
Tâche8					0			
Réunion					0			
CR de réunion					0			
Lot 2				0,0	0			
Tâche1					0			
Tâche2					0			
Tâche3					0			
Tâche4					0			
Tâche5					0			
Tâche6					0			
Tâche7					0			
Tâche8					0			

— Cumuler et enregistrer le temps passé par chacun, sur chaque tâche (support libre).
— Mettre à jour la TODOLIST.

	A	B	C	D	E	F	G	H
1	Date de mise à jour							
2	Travail à faire	Qui	Livrable le	Livré le	Charge	avancement	Validé par	Remarque
3								
4								
5								
6								
7								
8								
9								
10								
11								
12								
13								
14								

— Lors du TD «suivi de projet», l'enseignant mettra à jour le tableau général suivi de projet et le déposera sur moodle.

6 Exemple de schéma UML