Rapport de projet – S4

La pipopipette

groupe : Bovie Pierre Edouard
Credeville Louis-Maxime
Renard Vincent
Roche Julie

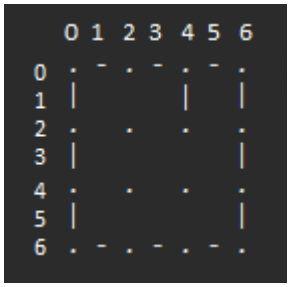
Mode d'emploi

Une fois l'application lancée dans le terminal grâce à "java -jar pipopipette.jar", la liste des commandes est rappelée, il faut alors saisir celle que l'on souhaite lancer. Selon la commande choisie, on peut soit obtenir directement un résultat, soit entrer en interraction avec le programme, pour jouer contre une IA par exemple. Lorsque la sous-application lancée est terminée, on peut de nouveau saisir une des commandes proposées.

Paramètres

- nblignes et nbcolonnes sont des entiers qui donnent la taille de la grille de jeu.
- type désigne si la grille est avec (C) ou sans (S) contours.
- strategie prend l'une des valeurs suivantes : simplet, prevoyant, idiot ou pondere, à laquelle il faut ajouter le type, le nombre de lignes, de colonnes.
- joueur vaut 1 si on souhaite jouer en premier, 2 si on souhaite que l'IA commence.

Coordonnées de la grille



Les batons sont pris en compte dans la grille selon les coordonnées ci-contre.
Par exemple, on peut voir dans cette grille un baton placé en (4,1)

Description du travail effectué

Notre groupe était composé de 4 personnes ayant des niveaux assez différents en java, ce qui a provoqué quelques difficultés de compréhension mutuelle par moments.
Nous avons utilisé Eclipse pour programmer, et GitHub pour se partager notre travail.

Lorsque nous avons eu le sujet, nous avons commencé à coder rapidement dans une première architecture jetable, notament des classes telles que Grille, Graphe et Poids.
Une fois cette architecture brouillon effectuée et remplie de morceaux de code jetable, nous avons réfléchi à une architecture finale. Cette étape de conception a été celle sur laquelle nous avons le plus bloqué. En effet, à cause de difficultés de compréhension des fonctions dynamiques, nous avons dû revoir plusieurs fois nos plans.
Lorsque l'architecture a été définie, nous nous sommes réparti les tâches de programmation, et notre programme a pris peu à peu forme. Les tests unitaires Junit étaient effectués dès qu'une nouvelle classe ou méthode était programmée, de manière à corriger les erreurs de manière rapide et efficace.

Nous avons enfin testé par binômes le main (classe Pipopipette), et ajouté la javadoc ainsi que certains commentaires, et après une relecture du projet complet par tous les membres du groupe visant à corriger les dernières erreurs qui auraient pû être oubliées, nous avons extrait et testé l'archive au format jar éxécutable.

problemes rencontrés

- Le paramètre 'type' a été codé et utilisé en tant que booléen lors de la phase de programmation, nous avons donc rectifié cela avec un switch afin d'avoir un appel des paramètres correct dans le terminal.
- Malgré une communication quasi permanente entre les membres du groupe, il est arrivé que nous ayons des 'merge conflicts' dans GitHub, il a donc fallu que certains morceaux de codes perdus soient de nouveau codé, et ce à plusieurs reprises.
- Ayant eu des soucis de compréhension concernant l'implémentation des classes Max et Moyenne, nous n'avons pas réussi à coder un apprentissage correcte, ou un calcul de stratégie exacte, qui les utilisait.

Répartition du travail

Travail effectué par			
Pierre Edouard Bovie	**Louis-Maxime Credeville**	**Vincent Renard**	**Julie Roche**
Interface utilisateur	Conception de l'architecture		Conception de l'architecture
Classe Pipopitette Classe Humain Classe Partie (+ certaines classes du code jetable)	Classe ToDot Classe Prevoyant Classe Idiot Classe Max Classe Moyenne Classe NombreCarre Classe Pondere (+ certaines classes du code jetable)		Tests unitaires Junit Gestion du groupe de projet (+ certaines classes du code jetable)
		Javadoc	Rédaction du rapport Commentaires / Javadoc

Experimentation

Lorsqu'on lance une simulation, via la commande '-simul', sur une grille de dimension 3x3 avec contours, on remarque certaines choses :
- Quelque soit le nombre de parties entre deux Prévoyants, le premier gagne toujours avec une moyenne de 9 carrés.
-Quelque soit le nombre de parties entre deux Idiots, le premier gagne toujours avec une moyenne de 6 carrés.
De même pour une grille de dimension 2x2 avec bords :
- Lors d'un nombre quelconque de parties entre deux Simplets, le premier gagne toujours avec 4 carrés.

On en conclut donc que lorsque deux joueurs ayant une même stratégie s'affrontent lors d'une partie, le premier à jouer gagne toujours, avec une moyenne de carrés constante.
Un joueur Humain a donc plus de chances de gagner s'il décide d'être le premier à jouer lors d'une partie contre une IA.

Architecture