Sujet du projet – Première partie

Introduction

Le sujet du cours « Projet programmation 2 » consiste cette année en le développement en Scala d'un simulateur du jeu d'échecs permettant de faire s'affronter à la fois des joueurs humains et des intelligences artificielles (dans toutes les configurations d'affrontements possibles), ainsi que de faire progresser les utilisateurs, notamment en proposant la sauvegarde de parties et le fait de pouvoir rejouer des parties sauvegardées ou importées, les reprendre à un certain point, etc. Vous devrez utiliser avec soin les principes de la programmation orientée objet. Vous travaillerez en trinôme, sachant qu'il y aura éventuellement un ou deux binômes (qui seront évidemment évalués en conséquence).

1 Travail demandé

Pour cette première partie, il s'agira de développer un simulateur graphique basique du jeu d'échecs [1] permettant de faire s'affronter des joueurs humains ou pilotés par une intelligence artificielle (IA) primitive, qui jouera de manière aléatoire. Le programme devra satisfaire aux exigences suivantes.

- Celui-ci devra être muni d'une interface graphique permettant :
  - un jeu visuel et intuitif ;
  - de lancer une nouvelle partie (dans toutes les configurations possibles : humain contre humain, humain contre IA primitive, IA primitive contre IA primitive), arrêter une partie et quitter le programme.
- À tout instant d'une partie, le programme devra veiller au respect des règles « de base » du jeu, qui concernent :
  - le placement initial ;
  - le déroulement des tours de jeu ;
  - les déplacements et les prises des pièces (sans considérer les règles spéciales du roque, de la prise en passant et de la promotion) ;
  - la victoire par échec et mat (sans prendre en compte les cas particuliers du pat, des différentes situations menant à la partie nulle ou de la perte au temps).
- L'IA primitive se limitera simplement à faire des mouvements aléatoires valides.

Il faudra veiller à soigneusement tirer profit des concepts de la programmation orientée objet afin d'avoir un code facilement modulable et réutilisable en vue d'évolutions et d'améliorations futures. La notion de joueur, qui peut être humain, piloté par une intelligence artificielle donnée, tirer ses mouvements d'un fichier de sauvegarde ou encore à travers un canal de communication réseau, nous semble à ce sujet requérir une attention particulière de votre part.

2 Critères d'évaluation

2.1 Rapport et soutenance

Vous devrez rendre un rapport de 2 à 3 pages (en format PDF, généré par LaTeX) expliquant les choix d'implémentation que vous aurez rencontrés. Dans le cas où votre programme présenterait des défauts, il faudra les mentionner dans le rapport en précisant leurs raisons. Une soutenance d'une dizaine de minutes par groupe sera organisée à la fin de la première partie du projet durant laquelle vous nous ferez une démonstration de votre programme.

2.2 Fonctionnalités du code

Bien évidemment, votre projet sera évalué par ses fonctionnalités. S'il remplit tout ce qui est demandé, rajouter d'autres fonctionnalités pourra apporter un bonus. L'ergonomie de l'interface graphique sera également prise en compte dans l'évaluation de votre projet : il n'est pas nécessaire que celle-ci soit très évoluée ou ait une apparence particulièrement travaillée, mais elle devra être suffisament fonctionnelle et intuitive à l'utilisation.

1. `https://fr.wikipedia.org/wiki/%C3%89checs`

2.3 Organisation du code

Votre projet devra impérativement être organisé hiérarchiquement, en le séparant en fichiers, classes et méthodes. Vous tâcherez de séparer du mieux possible les différentes fonctionnalités en classes, typiquement les aspects « interface graphique » des aspects « règles du jeu ». Gardez sous le coude la règle de ne jamais avoir de fonction trop longue ou de fichier trop grand.

2.4 Qualité du code

L'utilisation adéquate de la programmation orientée objet et de Scala sera un critère important dans l'évaluation. Dupliquer du code dans des classes sous-entendrait une mauvaise compréhension de l'héritage. De même, préférez des directives fonctionnelles concises à des boucles `for` et `if` imbriquées, voir la Section 4.4 de l'introduction à Scala. La mise en forme, la présence de commentaires et la cohérence des noms de classes, méthodes et variables devront être suffisamment décentes pour une lecture agréable du code. Nous vous demandons également de documenter votre code, comme expliqué dans la Section 7 de l'introduction à Scala.

3 Dates importantes

— Le code et le rapport seront à rendre avant le mardi 21 février à 23h59.
— La soutenance pour la première partie aura lieu le vendredi 24 février.