M2103 – projet de programmation

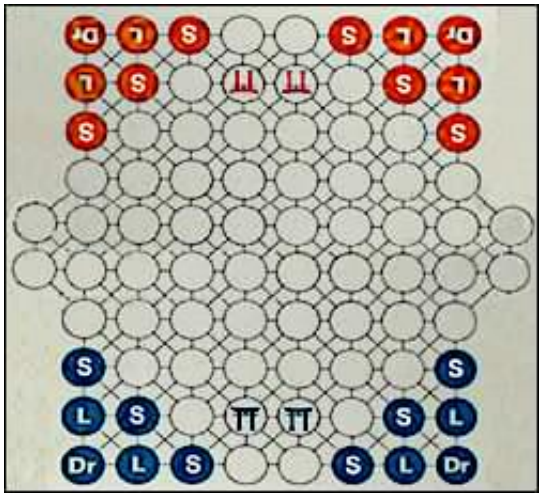
23 mars 2018

1 Description du projet

Pour le projet de programmation du module M2103, nous allons reprendre le jeu Shing Shang que vous avez programmé pendant le premier semestre.

Comme vous le savez déjà, le Shing Shang est un jeu de stratégie qui se joue sur un damier de 84 cellules avec 2×12 pièces (Bushis).

Préparation : Chaque joueur possède une armée de 12 Bushis d'une même couleur (rouge ou noir). Cette armée est composée de 3 groupes de Bushis : 2 Dragons, 4 Lions et 6 Singes. La différence entre les groupes est indiquée par sa taille (les Singes étant les plus petits Bushis, les Lions sont les Bushis moyen et les Dragons correspondent aux plus grands Bushis). Chaque joueur place son armée dans les coins du damier (voir le schéma ci-dessous [1]).



Déroulement du jeu : Les joueurs jouent à tour de rôle. Lors de son tour de jeu, un joueur doit déplacer l'un de ses Bushis de deux façons :

- Glisser vers une case suivante (vide) horizontalement, verticalement ou en diagonale, aussi bien en avant qu'en arrière.
- Sauter par dessus un Bushi si celui-ci est de **la même taille ou plus petit** que le sauteur. Pour pouvoir sauter, le Bushi doit se trouver sur

1. © `http://jeuxstrategieter.free.fr/Shing_shang_complet.php`

une case contiguë à une case occupée par un Bushi rouge ou noire. Il peut sauter la pièce, verticalement, horizontalement ou en diagonale, à condition que la case suivante soit vide. On peut enchaîner plusieurs sauts par dessus des Bushis **distincts** au cours d'un même tour.[2] Cet enchaînement de sauts s'appelle un **Shing Shang**.

Les règles suivantes complètent celles indiquées au-dessus :

- Les singes peuvent se déplacer d'une ou deux cases dans n'importe quelle direction, horizontalement, verticalement ou en diagonale mais sans changer de direction au cours du tour.
- Les Lions peuvent se déplacer d'une case dans n'importe quelle direction, horizontalement, verticalement ou en diagonale.
- Les dragons ne peuvent se déplacer qu'en sautant.

- Si au cours d'un **Shing Shang** on saute par dessus une ou plusieurs figurines adverses, celle-ci sont retiré du plateau. **Note :** On ne retire pas de figurine adverse pour un **saut simple**.
- Après avoir effectué un Shing Shang, le joueur a le droit à un tour supplémentaire avec un **autre** Bushi.
- Il est interdit de **terminer** un mouvement (saut ou déplacement simple) sur l'une des 4 cases portail du plateau de jeu, exception faite d'amener l'un de ses dragons sur l'un des portails de son adversaire, ce qui termine la partie.

Fin du jeu : Un joueur gagne la partie lorsqu'il parvient à amener l'un de ses dragons sur l'un des portails (i.e. deux cases spéciales de la deuxième et avant-dernière ligne) de son adversaire ou qu'il capture les deux dragons de son adversaire.

Votre programme devra permettre de visualiser une séquence de jeu en affichant à chaque tour : le plateau de jeu, les déplacements et/ou sauts courant du joueur.

Voici quelques indications des points à réaliser :

- placement des pièces
- gestion du début de partie (déterminer qui commence soit au choix de l'utilisateur, soit aléatoirement)
- saisie d'un coup (donc choix d'une pièce puis séquence de déplacements)
- validation d'un coup (est-il valide ? sinon retour en saisie)
- déplacer les pièces, retirer les pièces mangées et retour en saisie, ou début du tour de l'adversaire si pas de Shing Shang
- vérification de fin de partie

2 Notes de mise en œuvre

Dans un premier temps, définissez un diagramme de classes contenant les classes principales du jeu, donc les informations sur les Bushis, les joueurs, le damier et le gestionnaire de jeu (contenant les informations globales concernant

2. Sauter par dessus un Bushi et le resauter pour arriver à la case de départ n'est pas un Shing Shang !

l'état du jeu courant). Pour chaque classe, définissez les données correspondantes et les fonctions associées.

Dans un deuxième temps, définissez la boucle de contrôle globale du jeu qui assure les différentes phases du jeu. Il est conseillé d'avoir en permanence un programme qui compile et qui définit une version partielle du jeu. Seule devra être réalisée une version textuelle de ce jeu. Le programme devra permettre de visualiser la partie.

Aller plus loin au cas où le programme de base **fonctionne parfaitement** : Toute initiative personnelle (et justifiée) sera prise en compte et valorisée. Il existe de nombreux moyens d'étendre le projet. Une possibilité appréciable est de pouvoir interrompre une partie et la sauvegarder pour la reprendre ultérieurement. Le menu devra donc comporter les options "nouvelle partie", "sauvegarder la partie", "charger une partie en cours". Une seconde possibilité est de développer une version graphique de votre programme. Il est toutefois préférable dans un premier temps de définir une version non graphique afin de bien séparer les fonctions associées au jeu et l'affichage graphique du jeu. Si vous êtes très motivé, vous pouvez essayer d'implémenter un algorithme qui permet à l'ordinateur de jouer intelligemment un (ou deux) des joueurs.

3 Travail à rendre

Le projet est à réaliser en binômes (ou, avec l'accord de votre chargé de TP, en monôme).

Le diagramme de classes doit être rendu dans le contexte du module M2104 et va être noté. Après avoir rendu votre diagramme, vous recevrez un diagramme de classes préparé pour vous aider à la conception du projet.

Le programme à rendre est un projet sous forme d'une archive **zip** ou sous forme d'un **jar** (contenant les fichiers sources) + rapport à déposer sur eCampus.

Le nom de l'archive doit avoir la forme suivante : Nom1Nom2.grTP.zip/.jar ou Nom1.grTP.zip où Nom1 et Nom2 sont les noms de famille des membres des binômes et grTP est le nom du groupe de TP auquel ils appartiennent (1.1, 1.2, etc.).

Ce qu'il faut rendre :

- Le code source complet de votre application en Java qui compile sous Linux. **ATTENTION :** si votre programme fait appel à d'autres librairies externes, il est impératif de les inclure pour les besoins de test.
- La JavaDoc de vos classes, accessible en-dehors du code, c.-à-d. les fichiers html générés par le IDE.
- Un court rapport d'une longueur comprise entre 3 et 10 pages présentant :
  - les fonctionnalités implémentées (très brièvement),
  - l'organisation du programme : explications de votre démarche,
  - l'organisation et la répartition des tâches au sein du binôme durant la durée du projet (brièvement),
  - un mode d'emploi avec quelques illustrations (p. ex. capture d'écrans, scénario d'exécution...), destinées à montrer l'opérationnalité de votre application,
  - un bilan qualitatif du travail : difficultés rencontrées, l'apport (ou non) du projet en termes techniques, points qui vous ont paru intéressants.

  Le code source ne doit pas faire partie du rapport.

4 Évaluation du projet et calendrier

Le diagramme de classes est à rendre le vendredi 30 mars avant 23h59 heure de Paris.

Le projet est à rendre le dimanche 20 mai 2018 avant 23h59 (tout retard conduira à des pénalités).

Les soutenances auront lieu pendant les séances de TP de la semaine du 28 mai.

L'évaluation sera réalisée en fonction des critères ci-dessous :

— qualité technique du code : découpage en fonctions (préférez les fonctions courtes avec un rôle précis), modularité, un style de codage clair et cohérent.

— gestion des erreurs lors de saisies (les entrées erronées de l'utilisateur ne doivent pas interrompre le jeu).

— utilisation du paradigme orienté objet (utilisation des objets, attributs, méthodes d'objet, exceptions).

— lisibilité du code : présentation du programme (indentation), usage de variables et de fonctions ayant des noms explicites, commentaires des parties clés de votre code.

— documentation : JavaDoc, organisation du programme et son mode d'emploi, bilan.

— présentation orale : démonstration du programme et questions sur le travail réalisé.

— qualité de la démo : prévoir plusieurs scénarios permettant d'illustrer le bon fonctionnement des différents déplacements des Bushis ainsi que le cas de victoire.