Université de Versailles – Saint-Quentin-en-Yvelines
IATIC 3 ISTY – Projet 2020-2021
Encadrante : Laurence Pilard
laurence.pilard@uvsq.fr

Jeu de dames

Ce projet DOIT être réalisé en binôme.

Attention, il existe un grand nombre de règles différentes pour le jeu des dames. Vous trouverez ci-dessous LA règle qu'il vous est demandé d'implémenter. Tout ce qui est décrit doit être suivi précisément, si une règle n'est pas implémentée comme demandé, un malus sera appliqué à la notation de votre projet.

1 Règle du jeu

But du jeu : Capturer ou immobiliser les pièces de son adversaire.

Le jeu se joue sur un damier noir et blanc composé de 100 cases (10 sur 10). Chaque joueur a 20 pions. Chaque joueur place ses pions sur les cases noires. Avant le début d'une partie, il y a deux lignes au milieu du damier qui sont vides et séparent les deux camps. Ce sont les blancs qui commencent.

Le déplacement d'un pion : Les pions se déplacent sur les diagonales, du joueur vers le joueur adverse. Donc les pions se déplacent toujours vers l'avant. Ils ne se déplacent que d'une case à la fois sauf lorsqu'il y a une prise (*cf.* plus bas).

La prise par un pion :

- Une prise peut s'effectuer seulement vers l'avant. Lorsqu'une case voisine sur une des deux diagonales vers l'avant est occupée par un pion du joueur adverse, et qu'il y a une case libre juste derrière, ce pion doit être sauté. Il est ainsi pris et supprimé du jeu. Un pion ne peut pas prendre en arrière.
- *Prise multiple :* Le pion déplacé peut capturer plusieurs pions dans un même tour, à partir du moment où il peut être posé sur le damier entre deux prises.
- La prise des pièces de l'adversaire n'est jamais *obligatoire*. Un joueur a donc le choix de prendre ou de ne pas prendre. Cette flexibilité s'applique également au sein des prises multiples. Ainsi, si un joueur peut prendre trois pions en un tour, par une prise multiple, il peut en réalité prendre 0, 1, 2 ou 3 pions adverses.

La promotion : La promotion a lieu lorsqu'un pion atteint la dernière rangée du joueur adverse : celui-ci devient alors automatiquement une dame. On devra aisément distinguer les dames des pions dans l'affichage du jeu.

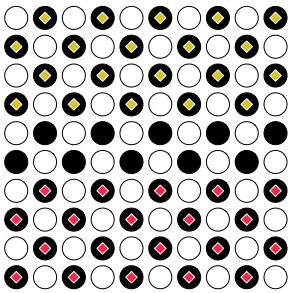
Déplacements et prises de la dame : La dame se déplace d'une seule case par coup en suivant les diagonales, mais a le privilège de pouvoir se déplacer, et prendre, en avant comme en arrière.

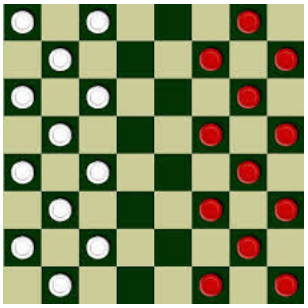
Fin de la partie : Le joueur a perdu la partie lorsqu'il ne lui reste plus aucune pièce en jeu, ou bien, si c'est à lui de jouer et que toutes ses pièces sont bloquées, c'est-à-dire qu'il est dans l'impossibilité de prendre ou de se déplacer.

2 L'implémentation

- Vous devez implémenter les règles du jeu présentées ci-dessus. **Aucun changement de règle n'est accepté**.
- **ATTENTION : une seule variables globale est autorisée** : celle qui est de type tableau à 2 dimensions et qui contient le plateau de jeu.

Interfaces graphiques : Il vous est demandé d'implémenter deux interfaces graphiques. Ces interfaces sont présentées dans la figure ci-dessous. Vous devez absolument conserver la forme des cases (rond versus carré), la forme des pièces (losange versus rond) et l'orientation du plateau (j1 joue au fond versus à gauche). En revanche, vous êtes libres de choisir la taille et les couleurs des éléments. ATTENTION, l'image de droite contient un plateau 8×8, mais vous devez faire un plateau 10×10.





Modèle de programmation MVC : Dans la mesure où votre jeu doit fonctionner sous deux interfaces graphiques différentes, vous devez utiliser la méthode MVC de programmation. Dans cette technique, votre code est divisé en trois parties :

- Le *Modèle*, qui contient toutes les fonctions relatives à la partie logique du jeu (`fin_partie`, `est_coup_valide`, `appliquer_coup`,...).
- La *Vue*, qui contient toutes les fonction relatives à l'affichage du jeu (`affiche_plateau`, `affiche_coup`, `affiche_coup_invalide`,...).
- Le *Contrôleur*, qui contient toutes les fonctions faisant le lien entre le modèle et la vue. (`main`,...)

Par exemple, si on s'intéresse à la fonctionnalité qui teste si le coup joué par un joueur est valide. Dans le *modèle*, on trouvera la fonction qui teste si le coup est valide et qui renvoie un booléen. Dans la *vue*, on trouvera la fonction qui affiche un message d'erreur dans l'interface graphique du jeu, indiquant que le coup n'est pas valide. Enfin, dans le *contrôleur*, on trouvera la fonction qui effectue le chose suivante : si la fonction du modèle qui teste la validité du coup renvoie *faux* (*i.e.* coup non valide), alors appeler la fonction de la partie *vue* qui affiche le message d'erreur.

Il ne vous est pas demandé de faire un fichier par partie. En revanche, vous devez rendre un fichier `.c` dans lequel vous avez trié vos fonctions. Ainsi, vous devez avoir un code source qui contient une première partie intitulée *La vue* (ce titre est indiqué en commentaire bien sûre) et qui regroupe toutes les fonctions relatives à la vue. De même, vous devez avoir une partie pour le *modèle* et une partie pour le *contrôleur*.

Mode de jeu : Il vous est demandé d'implémenter le mode de jeu «joueur-humain contre joueur-humain». En Bonus, vous pouvez implémenter le mode de jeu «joueur-humain contre intelligence-artificielle». Mais ceci ne sera qu'un bonus : vous devez avoir implémenté toutes les autres fonctionnalités avant d'implémenter le bonus.

Début et fin de parte : À chaque nouvelle partie, le programme propose au joueur de choisir son interface graphique. Il propose également de choisir son mode de jeu (mode deux joueurs humains ou joueur contre IA si cette option bonus a été implémentée). Une fois la partie terminée, le jeu doit afficher qui a gagné, puis proposer une nouvelle partie aux joueurs. Pour cette nouvelle partie, le programme devra à nouveau laisser le choix au joueur de l'interface graphique.

Gestion d'un tour : A cause de la prise multiple, le tour d'un joueur n'est pas nécessairement constitué d'un seul coup. En effet, un tour peut, par exemple, être composé de trois coups, si le joueur effectue trois prise d'affilées. De même, un tour peut être composé de 0 coup, si le joueur n'a plus de pion/dame ou s'il ne peut effectuer aucun déplacement/prise avec ses pions/dames.

Dans le cas où le joueur peut jouer, votre programme doit animer les coups joués au fur et à mesure. Ainsi, dans l'exemple précédent, si un joueur effectue trois coups lors de son tour, cela veut dire que le joueur va cliquer 4 fois, pour indiquer (1) le pion à déplacer, (2) la case d'arrivée après le premier coup, (3) la case d'arrivée après le deuxième coup et enfin (4) la case d'arrivée après le dernier coup. Après CHAQUE clic, votre programme doit mettre en évidence la case cliquée.

Votre programme doit détecter tout seul si un joueur est bloqué (aucun déplacement ou prise possible) et dans ce cas, arrêter le jeu en affichant qui a gagné.

3 Ce qu'il faut rendre

Date importante. Le programme devra être rendu le **lundi 12 octobre à 8H** au plus tard.

Comment rendre le projet ? Le projet devra être envoyé par mail à l'adresse suivante : `laurence.pilard@uvsq.fr`

Que faut-il rendre ? Il est demandé de rendre **le code source** de votre programme (votre fichier .c). Si vous rendez plusieurs fichiers .c et .h, alors il est impératif de décrire précisément dans le mail que vous m'envoyez, comment compiler et exécuter votre programme.

Sous quelle forme doit se présenter votre programme ? Votre programme devra compiler **sans erreur ni warning** dans les salles machines de l'ISTY. Il est *impératif* de mettre, en début de chaque fichier, un commentaire rappelant vos nom et prénom.

Triche : Si deux projets sont similaires dans leur réalisation, la note minimale entre ces deux projets sera prise, puis divisée par deux, puis donnée comme note aux deux projets. Généralisable à n projets similaires.

4 Notation

1. Le code : Une attention particulière sera portée sur votre code (environ 50% de la note) :

   respect du modèle MVC, usage pertinent des fonctions, noms de variables et de fonctions explicites, code bien indenté, bonne lisibilité du code en général.

2. L'exécution : 50% environ de la note. Concerne principalement le respect des règles, la jouabilité, l'ergonomie du jeu et l'absence de bug.