Projet d'année | F-106

Le jeu de Dames

Malik FASSI FIHRI
Matricule : 000364845

Année Académique 2012-2013

Table des matières

Pour implémenter un jeu de Dames il est nécessaire de diviser le travail en fonction.
Il faut donc analyser toutes les actions que l'on doit être capable de faire dans ce jeu.

1. Déplacements

Dans la partie 1 nous retrouvons les fonctions pour créer le plateau, et l'imprimer correctement. Nous avons aussi pris en charge le déplacement dans le plateau.
On se limitait a bouger le pion d'une seule case puisqu'il n'y avait pas encore de dames.
Il nous suffisait alors de déplacer le pion d'une case vers la gauche ou la droite en fonction du mouvement, sans tenir compte des captures.
Dans la partie 2, en prenant compte des captures et de la dame, le déplacement de pièce est traduit par ce speudo-code

1.1. Pseudo-code

```
movePiece(board,rowIndex,colIndex,direction,length=1 )

    directionligne=-player
    directionColonne=player
    si longueur de direction == 2 faire :
        directionligne=player
    si direction == « L » faire :
        directionColonne =-player
    longueurJusqueCapture=countFree(board,rowIndex,colIndex,direction,player=playe
r)
    si longueurJusqueCapture < length faire :
        length=longueurJusqueCapture+2
         #+2 car il faut rajouter la case de départ et celle d'arrivée

    ancienneCase devient vide
    nouvelleCase=(rowIndex+directionLigne*length,colIndex-directionColonne*length)
    insérer le pion dans la nouvelle case

    retourner tuple de destination et la capture si tel est le cas
```

« countFree » est une fonction qui calcule le nombre de case vide jusqu'à la prochaine capture. Le but de faire cela est que l'on va déplacer du bon nombre de case puisque le paramètre « length » vaut 1 par default pour les pions.

Voici le pseudo-code destiné à représenter countFree.

1.2. Pseudo-code

countFree(board,rowIndex,colIndex,direction,player=None,length=0)

```
s'il n'y pas de joueur faire
        joueur=board[rowIndex][colIndex]
        si joueur = dame :
                diviser par deux
directionLigne=-player
directionColonne=player
si longueur de direction == 2 faire :
        # ==2 car la direction est vers l'arrière si il y a 2 éléments dans la direction
        directionLigne=player
si direction == gauche faire :
        directionColonne =-player
ligneSuivante=rowIndex+directionLigne
colonneSuivante=colIndex+directionColonne
caseSuivante=(ligneSuivante,colonneSuivante)
si caseSuivante dans plateau, faire :
        si caseSuivante est vide  faire
                length+=1
                length=countFree(board,destI,destJ,direction,
                                                        player=player,length=length)
retourner length
```

2. « CANNOT_JUMP_OUTSIDE » et « CANNOT_GO_OUTSIDE »

Dans la partie 2 nous avons aussi du gérer les erreurs.
Pour cela nous trouvons autant de fonction qu'il y a d'erreur à gérer. Ces fonctions renvoient un code d'erreur comme prévu dans l'énoncé.
Pour vérifier si un coup et permis il suffit alors de vérifier tour a tour si ces fonctions ne renvoient pas d'erreur.

Cependant, pour « CANNOT_JUMP_OUTSIDE » et « CANNOT_GO_OUTSIDE » j'ai regroupé leurs fonctions en une seule et même fonction. Et ce, parce que les paramètres « destI » et « destJ » sont les cases d'arrivée du pion. Dans le cas ou un mouvement est sensé retourner « CANNOT_JUMP_OUTSIDE », en vérifiant si la case d'arrivée du mouvement est en dehors du plateau, on reçoit directement l'erreur « CANNOT_GO_OUTSIDE » pour laquelle il suffit justement de ne vérifier que cela. C'est pour cela qu'il faut vérifier deux choses en plus :

- que le mouvement mène a une case remplie qui se trouve sur les rebords du plateau. Voici comment faire :

2.1. Pseudo-code

*si rowIndex+(directionLigne*length) – 9==1 ou*
*colIndex+(directionColonne*length)-9==1 faire…*

- qu'il y a bien eu capture grâce a la fonction findCapture:

2.2. Pseudo-code

```
capture =None
resultat=capture

pour i = 1 jusque i=length-1 faire

    ligneSuivante=rowIndex+(directionLigne*length)-directionLigne*i
    colSuivante=colIndex+(directionColonne*length)-directionColonne*i
    caseSuivante=(ligneSuivante ,colSuivante)
    si caseSuivante n'est pas vide faire
        capture=caseSuivante
        resultat=capture
        arrêter la boucle

retourner resultat
```

3. Déroulement du jeu

Pendant le jeu le joueur se verra demander les coordonnées du déplacement, la longueur du déplacement et la direction du déplacement. Dans un premier lieu il faut demander vérifier que le mouvement est autorisé. Tant que le mouvement proposé par l'utilisateur est interdit, on lui redemande un nouveau mouvement.
Si le coup est autorise il faut maintenant procéder au déplacement du pion en appelant « movePiece » détaillée plus haut.
La variable « hasPlayed » peut maintenant être assignée à « True » puisque le joueur vient de déplacer un pion.
Il faut ensuite demander au joueur si il a terminé son tour. Ainsi, s'il veut il peut faire une rafle.
Enfin, si il a terminé son tour, on peut appeler la fonction « becomeKing » qui transforme les pions adéquats en Dames. Elle reçoit comme paramètre le plateau « board », et les indices de la case actuelle, « rowIndex » et « colIndex ». il faut donc l'appeler deux fois dans une boucle « for » ou elle recevra comme indices une fois comme indice (0,i) puis (9,i).

Voici le fonctionnement de « becomeKing » :

3.1. Pseudo-code

```
pion=caseActuelle
si pion est noir et ligneActuelle==9 faire
        caseActuelle-=1
sinon :
        si pion est blanc et ligneActuelle ==0 faire
                caseActuelle+=1
```

4. Bug

Je tiens à préciser que je n'ai malheureusement pas pu venir a bout du projet. Je suis resté bloqué sur une erreur sans pouvoir m'en dénouer. L'erreur se trouvant dans « checkMove », ou plus précisément lorsque l'on veut vérifier que la case choisie n'est pas déjà prise, je n'ai pas pu vérifier ma fonction « checkEndOfGame » avec une fonction « checkMove » défectueuse. Cela induit donc que je n'ai pas pu vérifier les codes erreurs suivants, à savoir : TOO_LONG_JUMP, NO_FREE_WAY, OPPONENT_PIECE, MUST_CAPTURE.