Estebe
Arthur

L1

Projet de programmation : Jeu d'Echec

Contenu

Introduction

Tout d'abord qu'est-ce qu'un jeu d'échec ?

Il faut savoir que le jeu d'échec est un jeu de rôle qui oppose deux joueurs. Chaque joueur possède des pièces d'une même couleur. Chacun en possède 16, dont la liste est la suivante :

- 8 pions
- 2 tours
- 2 fous
- 2 cavaliers
- 1 reine
- 1 roi

La partie se joue sur un plateau appelé Echiquier, c'est un plateau de 64 cases (8*8) avec une alternance de cases noires et de cases blanches.

Le but du jeu est de manger les pièces de son adversaire afin de le mettre en difficulté et facilité ainsi la prise du Roi, la pièce phare du jeu. En effet le principe fondamental est de capturer le roi du joueur adverse.

Pour cela il y a tout un stratagème à mettre en place et une technique de jeu à avoir. Il faut savoir prendre des risques pour réussir à gagner.

Présentation du projet

Ce projet consiste à reproduire un jeu d'échec sur ordinateur. Il faut donc le concevoir algorithmiquement puis le programmer. Les fonctionnalités qui sont mandées par le cahier des charges :

- Affichage du jeu
    - Affichage du plateau de jeu
    - Affichage d'un tableau de bord des scores
- Gestion de l'alternance des tours des joueurs
- Gestion du déplacement des pièces et des prises
    - Sélection de la pièce à jouer
    - Sélection de la position cible
    - Validation du coup s'il est possible
    - Mise à jour de l'état du jeu et de l'affichage du jeu
- Gestion des coups spéciaux (Roque, prise en passant, promotion, pat)
- Détection de la fin de partie (partie nulle, échec et mat, abandon, etc)

Pour réaliser ce projet, j'ai utilisé le logiciel codeblocks et le langage C/C++ pour un affichage en console.

Analyse fonctionnelle générale

Présentation des différents modules

Dans cette partie, je vais détailler dans les différents modules présents sans rentrer dans leur fonctionnement.

Insertion des noms des joueurs

Une partie d'échec se joue à deux, il m'a paru donc logique de pouvoir identifier le joueur à qui c'est le tour de jouer.
Dès le début du programme, les deux utilisateurs ont donc possibilité de rentrer leur pseudo qui sera le leur tout le long du jeu.

L'affichage de ce pseudo sera récurent lors de l'exécution du programme, il servira à identifier son tableau de score ainsi que si c'est ou non au joueur concerné de jouer.

Affichage d'un plateau de jeu

Comme dit plus haut, le jeu d'échec se joue sur un plateau de jeu, ainsi pour rendre le jeu le plus réaliste possible j'ai reproduit un plateau de jeu sous console. L'affichage console ne permet pas d'afficher des pièces réalistes, j'ai donc trouvé une méthode de substitution. J'affiche donc les initiales des lettres dans les cases concernées.

- Pion ➔ P
- Tour ➔ T
- Fou ➔ F
- Cavalier ➔ C
- Reine ➔ R
- Roi ➔ K (king)

Un joueur possède les minuscules et le second possède les majuscules.

Affichage d'un tableau des prises par joueurs

Il est impératif lors d'une partie, de pouvoir se situer par rapport à son adversaire d'un simple coup d'œil. Pour cela, un tableau affichant les prises qu'on a effectué me parait essentiel. A chaque fois qu'un joueur prend une pièce de son adversaire, je l'affiche donc dans son tableau des prises, tout comme un aventurier montrerait ses meilleures prises.
Ce tableau des prises permet aussi de détecter une fin de partie, en effet lorsqu'un des joueurs possède un tableau rempli, cela veut dire qu'il a mangé toutes les pièces de son adversaire. La partie est donc finie.

Affichage de l'état de la partie

Lors de la partie, j'ai rajouté la fonctionnalité suivante, l'état de la partie où trois états sont possibles :

- **En cours** (lors d'une partie normal, ce statut sera toujours affiché)
- **Abandon** (lorsqu'un joueur abandonne, le statut passe à abandon)
- **Terminée** (lorsqu'un joueur gagne la partie, le statut passe à terminée)

Cela permet à l'utilisateur de savoir ce qu'il en est au niveau de la partie. Le statut est bien évidemment actualisé en permanence.

Affichage du nombre de tour

Le nombre de tour permet au joueur de savoir combien de tour est passé depuis le début de la partie. Il permet aussi de vérifier la condition de fin de partie au bout du 50^{eme} tour.

Saisie de la case à jouer et de la case à déplacer

Ce module permet à l'utilisateur de saisir la pièce qu'il souhaite jouer, et la case dans laquelle il veut que sa pièce aille.
La réalisation de ce module qui bien qu'au premier abord parait simple, est en fait assez complexe. En effet un grand nombre de vérifications est nécessaire au bon fonctionnement de ce module. Par exemple un joueur ne peut pas sélectionner une case vide pour jouer, il ne peut pas non plus sélectionner une pièce qui ne lui appartient pas etc.

Choix d'abandon de partie

A tout moment, un joueur peut décider si oui ou non il veut abandonner la partie, pour ça, rien de plus simple, il lui suffit de répondre positivement à la question « Voulez-vous abandonner ? ».

Le statut de la partie sera alors : Abandon et l'autre joueur sera déclaré vainqueur.
La partie pourra alors recommencer à 0.

Promotion

Aux échecs, il y a moult coups spéciaux dont un qui s'appelle la promotion. Ce coup consiste à amener un de ses pions sur la ligne 1 ou la ligne 8 selon où se trouve son camp. Si le joueur arrive à y amener un de ses pions, il aura alors la possibilité de l'échanger contre une pièce de son choix entre :

- Un fou
- Une tour
- Une reine
- Un cavalier

Son pion lui disparaîtra. Sa nouvelle pièce aura bien entendu tous les avantages de la nouvelle classe choisie.

Couleur

Pour rendre le design plus attrayant et simplifier la compréhension du jeu, j'ai opté pour l'utilisation de conio. L'affichage se fait donc en couleur.

Fin de partie 50 coups

Cette option permet de compter le nombre de coups qui sont joués. Au bout du 50eme coup la partie s'arrête. En effet cela veut dire que la partie est bloquée.

Analyse fonctionnelle détaillée

Présentation des différents modules

Dans cette partie, je vais détailler dans les différents modules présents en présentant la partie technique.

Insertion des noms des joueurs

Pour ce module, j'ai utilisé deux tableaux de caractères de chacun d'une capacité de 15 caractères. Cela permet de rentrer une liste de 15 lettres. Pour la saisie, un simple **cin** suffit. Par la suite avec un **cout** je peux aisément afficher le tableau au moment souhaité.

Pour la saisie, il faut quand même la sécurisée puisque le **cin** ne prends pas en compte les espace, ni les caractères spéciaux. Il faut aussi vérifier que le nom de l'utilisateur ne dépasse pas 15 caractères sinon cela va poser des problèmes de segmentation à l'exécution du programme. Pour cette vérification, une boucle do...While est assez appropriée car cela demandera à l'utilisateur de recommencer tant que les conditions ne sont pas respectées.

Ces deux tableaux ne seront plus jamais modifier par la suite, ils resteront identique tout au long de la partie.

Affichage d'un plateau de jeu

L'affichage du tableau de jeu est assez simple à réaliser. J'ai tout d'abord crée un tableau de caractères à deux dimension d'une taille de 8*8 cases. En effet le jeu d'échec utilise un plateau de 64 cases. Je l'ai ensuite rempli du code ascii 'espace' pour avoir une impression d'une case vide lors de l'affichage. Pour les pièces, j'ai donc rempli le tableau avec les lettres P-p-F-f-T-t-C-c-R-r-K-k

Pour l'affichage du tableau, une double for, une pour les lignes, une autre pour les colonnes, permet d'afficher le tableau avec une forme d'un carré.

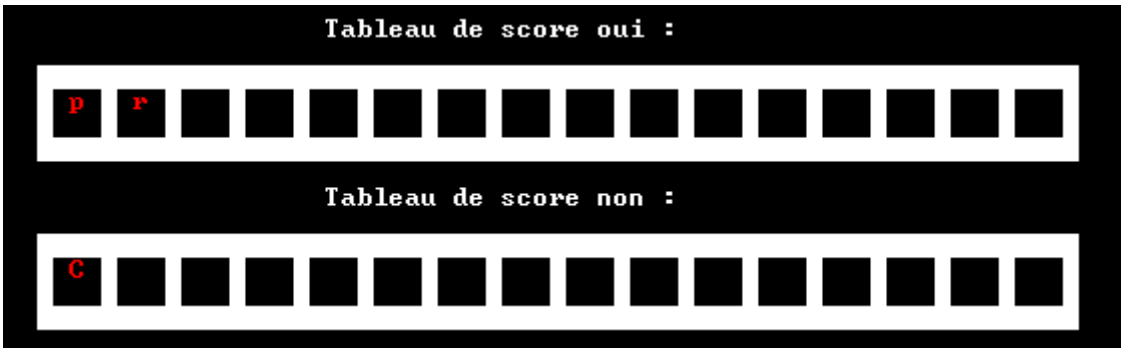
J'ai utilisé le caractère de code ascii 219 ce qui permet de créer un cadre autour de chaque case et de réaliser les bordures du tableau. Le résultat est celui sur l'image ci-contre.

Pour simplifier la lecture, j'ai utilisé la fonction textcolor() ainsi que des chiffres mit en bas et sur la gauche du tableau.

Affichage d'un tableau des prises par joueurs

Pour ce module, j'ai créé deux tableaux de caractères de chacun une taille maximum de 16 caractère. Au départ, ils sont remplis entièrement avec le caractère 'espace '. Toujours pour donner cette impression de tableau vide. Au fur et à mesure de la partie, les cases vont évidemment se remplir avec les pièces que le joueur aura capturées. Pour ce faire, à chaque tour je stocke dans une variable tampon la valeur de la case du tableau là où le joueur veut y amener sa pièce, et si cette valeur est différente de la valeur 'espace ' cela veut dire que la case contenait forcément une pièce du joueur adverse.

Voici l'exemple des tableaux des prises lors d'une partie. Ici le joueur 1 s'appelle « oui » et le joueur 2 s'appelle « non ». Ils auront le même nom tout au long de l'exécution du programme. On peut voir qu'ici oui a mangé deux pièces appartenant à non et non lui, n'en a mangé pour le moment qu'une seule.



Un compteur s'incrémente de +1 à chaque fois qu'une case du tableau est remplie, une fois que le compteur atteint 15, cela veut dire que toutes les pièces sauf le Roi ont été capturées. C'est un signe de fin de partie. Les tableaux permettent donc de vérifier si la partie peut continuer ou non.

Si la partie est abandonnée, le joueur perdant voit son tableau se remplir de toutes ses pièces.

Affichage de l'état de la partie

Pour ce module, la technique est très simple est très rapide, une variable est attribuée au statut de la partie. Selon sa valeur, le programme affiche un des 3 états. L'utilisateur peut changer la valeur de cette variable à chaque début de tour, pour savoir s'il veut abandonner ou non. S'il dit oui, la variable prendra alors la valeur d'abandon, et ça sera affiché sur le programme.

Avec l'utilisation de conio, cela permet de rendre un peu plus attrayant le programme et un peu plus esthétique.

Affichage du nombre de tour

Le nombre de tour est géré par un simple compteur, en effet à chaque fin de tour, il gagne 1 ce qui permet au tour suivant d'afficher le bon numéro du tour.

Saisie de la case à jouer et de la case à déplacer

Cette partie est assez complexe au niveau des boucles de vérifications. En effet il faut vérifier plein de conditions pour que le joueur puisse sélectionner une pièce à jouer.
Un joueur ne doit pas pouvoir jouer une pièce qui ne lui appartient pas. Pour cela, l'utilité d'utiliser des minuscules pour l'un et des majuscules pour l'autre prend tout son sens. En effet le code ascii est propre pour chaque caractère. Le joueur qui possède les majuscules aura ses codes ascii qui vont de 65 à 90. Il suffit alors de vérifier que la case qu'il a sélectionnée soit comprise entre ce code ascii pour qu'il puisse jouer.

L'autre vérification, c'est pour que le joueur ne puisse pas sélectionner une case vide. Il suffit alors de vérifier que le code ascii de la valeur de la case ne soit pas égal à 'espace'.

Maintenant il faut aussi vérifier que la case de destination choisie par le joueur soit valable. En effet un joueur ne peut pas jouer dans une case ou il possède déjà une pièce. Son déplacement doit aussi correspondre aux spécificités de la pièce déplacée. Chaque pièce à son déplacement propre. Pour cela, j'ai donc créé une fonction par pièce qui sera appelé si le joueur voudra bouger cette pièce.

Par exemple pour un pion, il faut vérifier que la ligne de départ soit différente de celle d'arrivée, mais seulement de +1 ou de +2 si le pion est dans sa position initiale de début de partie. Pour la colonne, pour un pion, elle doit toujours être la même sauf si le pion mange un autre pion en diagonale.

Si toutes les vérifications sont validées, une variable prend alors la valeur 1, la fonction de vérification, prend elle aussi la valeur de 1 et dans le main, si la fonction renvoi la valeur 1, alors le déplacement est validé et le joueur peut jouer.

Choix d'abandon de partie

Ce module est assez simple du point de vue de la technique. En effet une simple saisie en début de tour permet de demander à l'utilisateur s'il souhaite abandonner la partie. Si la valeur de la variable concerné vaut un abandon, au prochain tour du programme, la boucle d'abandon sera activée et le statut de l'état du jeu sera alors : Abandon.

Promotion

Pour gérer ce coup, il suffit de vérifier si le coup joué est un pion, et si la ligne d'arrivée correspond à la ligne 8 ou à la ligne 1 selon à qui c'est le tour de jouer.

Le joueur doit alors choisir entre les quatre choix qui s'offrent à lui. Selon son choix la valeur de la case concernée de son tableau sera alors remplacée par le caractère correspondant à son choix.

```
Bravo oui tu merites une promotion !
Choisi : 1- Reine  2- Fou  3- Cavalier  4- Tour :
```

Conclusion

Lors de la réalisation de ce projet, j'ai tout d'abord appris à maîtriser les techniques de jeux du jeu d'échec. De plus, j'ai aussi perfectionné ma technique de programmation et j'ai pu avoir une réflexion sur comment améliorer la complexité de mon programme.

Au départ, se dire que l'on doit concevoir un jeu d'échec peut être un peu déroutant, mais au fur et à mesure de mon avancée, je me suis rendu compte que ce n'était pas si dur que ça en avait l'air, bien que je ne sois pas allé jusqu'au bout du projet.

En effet, ce projet a quand même été assez conséquent à traiter en cette période de l'année ou nous avons tous nos partiels.

J'ai bien aimé travailler sur un jeu d'échec car c'est un projet concret, il y a quelque chose au bout. Il ne s'agit pas juste d'une entité calculatoire pour tester des fonctions mathématiques.

Il y a un rendu réel et surtout interactif.

Le point clé de ce projet est à mon sens, les vérifications de validation du coup s'il est possible ou non. En effet la complexité de ce programme réside en la simplification des vérifications, qui sont par ailleurs toujours perfectibles pour nous.

Je reste donc dans l'attente d'un nouveau projet tout aussi intéressant.