Projet

Le but du projet est d'écrire un programme en language C en binome.

Groupe :

L1-Y, Groupe 2

Membres du groupe :

- Valentin GUILLON, 20002588
- Lylian DENISET, 21001107

Jeu du All Queens Chess

- **règles**

- 2 joueurs
- Sur un plateau de 5*5
- Chaque joueur a 6 Reines
- Elles sont placées sur les lignes du bas et du haut, en alternant j1-j2, + 2 reines qui s'opposent sur la ligne du milieu, séparées par 3 cases
- Le but du jeu est d'aligner 4 Reines pour gagner

- **Déroulement du jeu**

Chaque joueur déplace une de leur Reine à tour de rôles.
Une Reine peut se déplacer de 1 à 4 cases dans toutes les directions (horizontale, verticale, diagonale).
Une Reine ne peut pas passer par dessus une autre, prendre sa place, ou la capturer.

Annoncer le gagnant à la fin de la partie.

Spécifications informelles:

- Bibliothèques :
  - **stdio.h** -> printf() scanf()
  - **stdlib.h** -> FILE et fonction de fichier

- Structures :
  - **Position** -> x et y sur le plateau

- Variables :
  - **chessboard** -> liste les pièces sur le plateau
  - **fichier** -> stock l'adresse d'un fichier
    - contenu formaté du fichier : (ex avec les positions de base des reines. Le "r" indique le joueur qui joue)
      - 21212
      - 00000
      - 10002
      - 00000
      - 12121
      - r

  - **winner** -> gagnant
  - **player** -> joueur dont c'est le tour
  - **pos_origin** -> coordonnées de la reine selectionnée
  - **pos_dest** -> coordonnées de la destination de la reine selectionné

- Constantes :
  - **taille** -> taille de la grille (n*n)

- Fonctions :
  - **afficher_chessboard()** -> Affiche le plateau dans le terminal
  - **lire_chessboard()** -> lit un fichier pour remplir le plateau
  - **ecrire_chessboard()** -> écrit l'état actuel du plateau dans un fichier
  - **choose_queen()** -> selection de la position d'une reine (choix d'une reine du joueur)
  - **place_queen()** -> selection de la destination d'une reine (choix d'une position libre)
  - **sans_conflit()** -> vérifie si la reine peut être déplacé vers la destination
  - **winning()** -> vérifie si 4 reines sont alignés et retourne le gagnant

- Fonctions (détails) :
    - void **afficher_chessboard(**int *chessboard, Position chosen, int player**)** :
        - plusieurs print formaté qui affiche ligne par ligne le contenu du tableau à 2 dimensions

    - int **lire_chessboard(**FILE *fichier, int *chessboard**)** :
        - initialise **fichier** au début du fichier
        - pour chaque char du fichier, l'ajouter dans chessboard[i][j]
        - retourne 1 ou 2 en fonction du dernier char du fichier (soit 'r' soit 'n')

    - void **ecrire_chessboard(**FILE *fichier, int *chessboard, int player**)** :
        - initialise **fichier** au début du fichier
        - pour chaque élément du tableau, l'ajouter dans le fichier (en respectant le format)
        - ajouter 'r' ou 'n' en fontion de **player**

    - Position **choose_queen(**int *chessboard, int player**)** :
        - tant que la piece n'appartient pas au joueur
            - demande une position (l'input est un char suivit d'un int (ex: B4))
            - si l'input est "z0" (zéro) ou "z1" (ou en majuscule)
                - return -> -10 ou -11
        - convertir l'input en coordonnées (B4 devient 13, E1 devient 40)
        - return -> les coordonnées

    - Position **place_queen(**int *chessboard, Position origin, int player**)** :
        - tant que la position choisit n'est pas libre :
            - demande une position (l'input est un char suivit d'un int (ex: B4))
            - si l'input est "z0" (zéro) ou "z1" (ou en majuscule)
                - return -> -10 ou -11
        - convertir l'input en coordonnées (B4 devient 13, E1 devient 40)
        - return -> les coordonnées

    - int **sans_conflit(**Position origin, Position dest, int *chessboard**)** :
        - verifier si le déplacement est horizontal, verticale ou diagonale :
            - char dir = 'h' ou 'v' ou 'd'
            - si aucune dir n'est trouvé, return -> 0
        - verifier s'il y'a des reines sur le trajet (en fonction de dir) :
            - si une reine est croisée -> return 0
        - return 1

- int **winning(**int *chessboard**)** :
    - déclaration de 2 int : player et count
    - pour chaque reine croisée, player en prends la valeur (sauf pour zéro)

    - si lors d'un croissement de reine, elle est la même que player --> count ++
    - si c'est une reine adverse ----------> count = 1
    - else ----------> count = 0

  - count = 0
  - vérifie s'il y'a un alignement en horizontal :
    - si count == 4 --> return player
  - count = 0
  - vérifie s'il y'a un alignement en vertical :
    - si count == 4 --> return player
  - count = 0
  - vérifie s'il y'a un alignement en vertical :
    - si count == 4 --> return player

  - return 0

Spécification formelles : Language C

- Variables :
  - int **chessboard[TAILLE][TAILLE]** -> chaque cellule contient un entier entre 0 et 2 (0 = pas de reines, 1 = j1, 2 = j2)
  - FILE ***fichier** -> contient l'adresse du fichier (en lecture/écriture)
- Structures :
  - **Position** {int x; int y}
- Constantes :
  - int **TAILLE** = 5
- Fonction principale :
  - initialiser **chessboard[][]** (explicitement)
  - player = 1
  - winner = 0
  - demander si on veux charger la partie sauvegarder
    - si oui -> **lire_chessboard()**
    - fermer le fichier
  - tant qu'il n'y a pas de gagnant:
    - choose_again: (goto)
    - pos_origin = take_queen()
    - pos_dest = place_queen()
    - if sans_conflit() = 1 -> goto choose_again;
    - else ----------> chessboard[pos_origin] = 0, chessboard[pos_dest] = player
    - winner = winning()
    - changer le joueur
  - afficher winner