Compte-rendu de Projet

Circonstance
Faire un programme (en C) pour le cours de Programmation Impérative (S2 2022) en binome

Jeu
All Queens Chess

Nom des participants et Groupe de TP

- Valentin GUILLON (20002588) : valentin.guillon93430@gmail.com
- Lylian DENISET (21001107) : lyltis974@gmail.com
    - Groupe L1-Y

Guide d'utilisation du programme
Fichiers nécessaire :

- **projet_2022.c**
    - contenant l'entièreté du programme
- **sauvegarde/chessboard_save.txt**
    - contenant les informations d'une partie sauvegardées (peut être créé vide)
- (bonus) **sauvegarde/chessboard_test.txt**
    - contenant les informations d'une configuration test, dans laquelle Joueur N, peut gagner en un déplacement dans n'importe quel sens (horizontal, vertical ou diagonal)

Compilation :

- à l'emplacement du fichier "projet_2022.c", taper la commande suivante :
    - **gcc -Wall -Wextra -Wfatal-errors projet_2022.c -o projet_2022**
- puis, pour exécuter le programme :
    - **./projet_2022**

Utilisation :

- Choisir s'il on veut charger la partie sauvegardée (l'option caché '7', charge une partie test)
    - ~$ y ou n
- La partie est lancée. Le joueur indiqué choisie une reine en écrivant une position :
    - ~$ A1 ou a1 ou E5 ou e5
- Choisir la destination, écrivant une position :
    - ~$ A1 ou a1 ou E5 ou e5

- En cas de déplacement non autorisé, le joueur est invité à choisir entre le choix d'une autre reine ou d'un autre position
    - ~$ 1 ou 2

- L'abandon ou la sauvegarde de la partir, est permis au moment de ma selection de la reine, en donnant un position spéciale
    - ~$ Z0 ou z0 pour abandonner
    - ~$ Z1 ou z1 pour sauvergarder

La programme s'arrête en cas de victoire, d'abandon, ou de sauvegarde

Etat d'avancement

1. **Fait** : Création du Github, et des fichiers principaux
2. **Fait** : Lecture du cahier des charges, et discutions plus profondes des spécificités du programme
3. **Fait** : Réécriture du CdC/spécifs
4. **Fait** : Écriture du l'architecture de base du programme (déclaration des fonctions, écritures partielle du main())
5. **Fait** : Répartitions du programme
6. **Fait** : Complétion des fonctions
7. **Fait** : Assemblage du programme
8. **Fait** : Écriture et complétion du rapport
9. **non-fait** : Préparation de la soutenance

Répartition du travail

- Création du Github et des fichiers principaux : **Valentin**
- Analyse du cdc originel et rédaction du notre : **Valentin** et **Lylian**
- Fonctions :
    - afficher_chessboard() : **Valentin**
    - lire_chessboard() : **Valentin**
    - ecrire_chessboard() : **Valentin**
    - take_queen() : **Lylian**
    - place_queen() : **Lylian**
    - sans_conflit() : **Valentin**
    - winning() : **Valentin**
- Assemblage du programme : **Valentin**

Le programme et ses fonctions

- afficher_chessboard() : **Valentin**
    - affiche jolie le plateau dans le terminale
        - Cette fonction reçoit beaucoup d'arguments afin d'offrir la meilleure expérience utilisateur
        - Étant donné la nature de la fonction (principalement composée de printf() et de condition), il n'est pas vraiment utile de l'expliquer

- lire_chessboard() : **Valentin**

    - remplis le tableau 2D de reines en fonction du contenu d'un fichier formaté
        - Parcours un fichier caractère par caractère
        - Chaque caractère est converti en entier et ajouté à une cellule correspondante du tableau 2D

- ecrire_chessboard() : **Valentin**
    - écris dans un fichier, de manière formaté, l'état actuel du tableau 2D de reines
        - parcours un tableau 2D
        - Pour chaque élément, il est écrit dans un fichier

- take_queen() : **Lylian**
    - le joueur choisie une de ses reines
        - Pour une question de jouabilité, le joueur donne des coordonnées tel que "A3" ou "D2"
        - Chaque élément de la coordonnée est converti en entier lisible dans un tableau (pour A3, A deviens 0, et 3 deviens 2). Une lettre étant représenté par un code ASCII, il est facile de faire une conversion vers un entier
        - Vérifier dans le tableau, que ses coordonnées corresponde bien à une reine du bon joueur

- place_queen() : **Lylian**
    - le joueur choisis un emplacement vide
        - Pareil que pour take_queen()
        - Cependant, pour la vérification, il faut que les coordonnées correspondent à un espace vide (donc == 0)

- sans_conflit() : **Valentin**
    - vérifie si le déplacement demandé est possible
        - Déterminer le sens d'alignement entre deux position (horizontal, vertical ou diagonal)
            - horizontal, les deux y sont égaux
            - vertical, les deux x sont égaux
            - diagonal, en soustrayant les coordonnées d'un position avec l'autre, on doit obtenir une nouvelle position dont l'absolu de x et de y sont égales
        - Créer un vecteur permetant de connaitre la direction du déplacement
        - En fonction du sens, parcourir la distance entre les deux positions. Il ne devrait y avoir aucune reine sur le chemin, si le déplacement est possible

- winning() : **Valentin**
    - vérifie dans le tableau si 4 reines sont alignés
        - verification horizontale et verticale
            - parcours du tableau avec un compteur, qui augmente si des reines de même couleur se suivent (remis à zéro si une autre couleur ou une case vide est observée, ou en cas de changement de ligne/colonne)

- vérification diagonale (vers bas-droite, puis vers haut-droite)
    - initialisation de la lecture du tableau à partir d'une extrémité que permet un alignement d'au moins 4 cases en diagonale (donc pour A1, B2, et A2 (vers bas-droite) et A5, B5, et A4 (vers haut-droite))
    - en utilisant le même principe avec le compteur