***Projet FreeCell Réalisé par : Mohammed Benaissa***

Le projet FreeCell est structuré en plusieurs fichiers pour une meilleure organisation et modularité du code. Voici un rapport complet pour les fichiers **freecell.h**, **impl.c**, et **main.c** :

1. ***freecell.h :***

**Structures de Données :**

**Card** : Contient un numéro et un type pour représenter une carte.

**Element** : Représente un élément dans la pile, liant une carte à l'élément suivant.

**StackCards** : Une pile de cartes avec une tête et un nom. (Pour faciliter l’accès à chaque pile à l’aide de son nom et la manipuler au niveau de la tête )

**Zone** : Représente l'une des trois zones du jeu avec un tableau de piles.

**Fonctions :**

**Fonctions de Design** : setConsoleColor, resetConsoleColor : Gèrent les couleurs de la console.

**Fonctions de Carte (Card)** : createCard, displayCard, isSameColor : Manipulent les cartes.

**Fonctions de Pile (StackCards)** : createElement, createStack, Empiler, Depiler, displayStack : Opérations sur les piles.

**Fonctions de Zone (Zone)** : createZone, createZone1, createZone2, createZone3, displayZone, isZoneEmpty, initZone1 : Opérations sur les zones.

**Fonctions de Mouvement** : move, wantedStack, executeMove, isValideMove : Gèrent les déplacements.

**Fonctions de Jeu** : displayBoard, isStackBlocked, isGameBlocked : Fonctions de jeu.

1. ***impl.c :***

***Fonctions de Design :***

**setConsoleColor(int textColor, int bgColor)** : Modifie les couleurs du texte et de l'arrière-plan de la console selon les paramètres fournis.

**resetConsoleColor() :** Réinitialise les couleurs de la console aux valeurs par défaut (blanc sur noir).

***Fonctions de Carte (Card) :***

* **createCard(int num, char\* type)** : Crée une nouvelle carte avec un numéro et un type spécifié.
* **displayCard(Card\* card)** : Affiche une carte avec sa valeur et sa couleur dans la console.
* **isSameColor(Card\* card1, Card\* card2)** : Vérifie si deux cartes sont de la même couleur.

***Fonctions de Pile (StackCards) :***

* **createElement(Card\* card)** : Crée un nouvel élément (noeud) pour la pile avec une carte spécifiée.
* **createStack(char\* name)** : Crée une nouvelle pile avec un nom spécifié.
* **Empiler(StackCards\*\* pile, Card\* C)** : Empile une carte sur la pile spécifiée.
* **Depiler(StackCards\* pile)** : Dépile une carte de la pile spécifiée.
* **displayStack(StackCards\* pile)** : Affiche le contenu de la pile dans la console.

***Fonctions de Zone (Zone) :***

* **createZone(int num)** : Crée une nouvelle zone avec un nombre spécifié de piles.
* **createZone1()** : Crée la zone 1 avec huit piles pré-définies.
* **createZone2()** : Crée la zone 2 avec quatre piles prédéfinies.
* **createZone3()** : Crée la zone 3 avec quatre piles prédéfinies.
* **displayZone (Zone\* zone)** : Affiche le contenu de la zone dans la console.
* **isZoneEmpty (Zone\* zone) :** Vérifie si la zone est vide.
* **initZone1 (Zone\* zone1)** : Initialise la zone 1 avec des cartes dans un ordre aléatoire.

***Fonctions de Mouvement :***

* **move (StackCards\* depart, StackCards\* arrive) :** Déplace une carte d'une pile à une autre.
* **wantedStack (char \* text, Zone\* z1, Zone\* z2, Zone\* z3)** : Retourne la pile spécifiée par le texte dans les zones 1, 2 ou 3.
* **executeMove (Zone\* z1, Zone\* z2, Zone\* z3)** : Exécute un mouvement en demandant au joueur la pile de départ et d'arrivée.
* **isValideMove (StackCards\* depart, StackCards\* arrive)** : Vérifie si un mouvement est valide en respectant les règles du jeu.

***Fonctions de Jeu :***

* **displayBoard (Zone \*z1, Zone z2, Zone z3)** : Affiche l'état actuel du plateau de jeu.
* **isStackBlocked (StackCards\* pile, Zone\* z1, Zone\* z2, Zone\* z3)** : Vérifie si une pile spécifiée est bloquée et ne peut pas effectuer de mouvements valides.
* **isGameBlocked (Zone\* z1, Zone\* z2, Zone\* z3)** : Vérifie si le jeu est bloqué, c'est-à-dire si aucune pile ne peut effectuer de mouvements valides.

1. ***main.c***

* Initiation du jeu :
* Les trois zones du jeu (zone1, zone2, zone3) sont créées et initialisées.
* La fonction initZone1(zone1) remplit la zone 1 avec des cartes disposées aléatoirement.
* Début de la boucle de jeu (while(1)) :
* La boucle continue tant que le joueur n'a pas gagné ou que la partie n'est pas bloquée.
* La fonction system("cls") efface l'écran à chaque itération pour une mise à jour propre de l'affichage.
* Affichage du tableau de jeu :
* La fonction displayBoard(zone1, zone2, zone3) est appelée pour montrer l'état actuel du plateau de jeu dans la console.
* Conditions de Fin de Jeu :
* Si les zones 1 et 2 sont vides, le joueur a gagné. Un message de victoire est affiché en vert.
* Si la partie est bloquée (aucun mouvement valide possible), un message de défaite est affiché en rouge.
* Saisie et Exécution du Mouvement :
* La fonction executeMove(zone1, zone2, zone3) est appelée pour demander au joueur de saisir le mouvement.
* Fin de la Boucle :
* La boucle continue jusqu'à ce que le joueur gagne ou que la partie soit bloquée.
* Fin du Programme :
* La fonction return 0 et marque la fin du programme.

***Difficultés :***

Lors de l'implémentation de l'interface graphique, j'ai fait face à plusieurs difficultés avec différentes bibliothèques. Voici un résumé de mes expériences :

* **GTK4 :**

-Rencontré des difficultés d'installation et de liaison avec l'environnement.

-L'utilisation de Gobject était délicate.

-GTK4 avec Glade (fichiers Builder) : La manipulation des fichiers Builder contenant du XML avec le logiciel Glade s'est avérée complexe. Difficultés dans la gestion de l'interface utilisateur.

* **QT Creator :**

-Principalement basé sur C++, ce qui ne correspondait pas entièrement à mes besoins.

-Décision de ne pas utiliser QT Creator pour ce projet.

* **SDL (Simple DirectMedia Layer) :**

-Choix de cette bibliothèque pour l'interface graphique.

-Réalisation de l'interface initiale avec succès.

-Problèmes rencontrés lors de l'importation de la bibliothèque **SDL\_image**.

Difficulté à trouver les fichiers **include** et **lib** nécessaires pour la liaison.

* **Affichage vertical sur la console :**

Possibilité d'implémentation, mais l'esthétique a été compromis en raison du fond blanc des cartes. La visualisation rend le terminal moins organisé.