Pierrick Delrieu

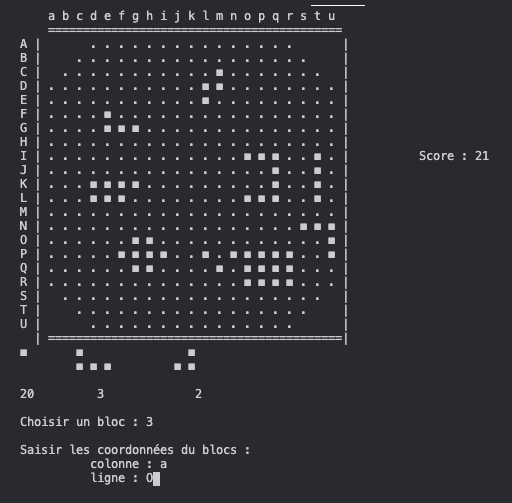
Romain Joreau

L1 - Groupe B

Projet de programmation en C

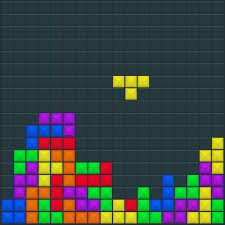
2019-2020

**COMME UN AIR DE TETRIS**



**SOMMAIRE**

1. L'architecture du projet
2. Le menu de présentation et de paramétrage du jeu
3. Les plateaux de jeu (cercle, losange et triangle)
4. Gestion de la création des blocs
5. Affichage des blocs en fonctions de la politique choisie
6. Saisie des coordonnées et placement du bloc choisi
7. Gestion du plateau lors de l’annulation d’une ligne
8. L’architecture du main



**Le jeu Tetris** est un jeu vidéo sortie il y a plus de 40 ans. C’est un jeu de puzzle. C’est puzzle soumis à la gravité tombe de manière aléatoire et plus en plus fréquente. Le but du jeu et de survivre le plus longtemps possible en gardant le plateau non- remplit. Les puzzles peuvent être contrôlé de droite à gauche le temps de leur descente. Lorsqu’une ligne est pleine celle-ci est détruite. Le but du jeu est donc de placer les blocs de telle sorte à remplir les lignes.

L’objectif du projet était de réaliser un jeu similaire à celui-ci avec des conditions de jeu légèrement différentes. Le jeu que nous avons créé porte le nom de **"Comme un air de Tetris"**.

**Les règles du jeu :**

Le jeu "Comme un air de Tetris" se joue sur un plateau en cercle, en losange ou en triangle de minimum 19 cases de largeur. Le but du jeu est de placer des formes sur un plateau afin d'annuler des lignes et des colonnes. Contrairement au jeu Tetris classique, les blocs ne subissent pas la gravité. Si une ligne est annulée alors toutes les cases pleines au-dessus de la ligne descende d'un cran (il est possible d'annuler plusieurs lignes en même temps). Une ligne ou une colonne est annulée si et seulement si suite au placement d'une forme la ligne est complète (attention si la descente des blocs remplis une ligne ou une colonne, celle-ci n'est pas annulé). Le score correspond donc au nombre de cases annulées. Vous aurez seulement trois essais pour saisir des coordonnés valides pour le bloc choisi. Une fois ces trois chances épuisé la partie se finir et vous aurez atteint votre score maximal.

1. L'architecture du projet

Une image contenant capture d’écran, moniteur, noir, téléphone

Description générée automatiquementAu fur et à mesure du projet nous avons dû placer avec soin nos différents fichiers.

Le dossier source contient tous les fichiers de code C. Chaque fichier contient les fonctions qui lui corresponde. Le fichier "main.c" contient la fonction principale du programme.

Le dossier blocs contient trois fichiers de type ".csv" qui contiennent dans un tableau les différents blocs en fonctions de la forme du plateau.

Le fichier Makefile permet de faciliter la compilation d'un fichier exécutable en tapant simplement make dans le terminal car il est nécessaire de compiler en même temps les 5 fichiers .c. Le fichier exécutable main se situe alors au même endroit que ce dernier.

Le Readme.md est un dossier git qui permet de mettre quelques informations nécessaires au programme. Le fichier "tetris.code-worspace" est un fichier propre a Visual Studio Code afin d'ouvrir plus facilement le projet sur cet éditeur.

1. Le menu de présentation et de paramétrage du jeu

Tout d’abord, nous avons réalisé un menu de présentation lorsque le joueur commence le jeu.

**Le menu de présentation se situe à chaque début de partie et contient :**

* Le nom du jeu avec le choix d'affichage des règles du jeu
* Le choix de la forme du plateau
* Le choix de la taille du plateau
* Le choix de la politique de suggestion des blocs

Chacun des quatre affichages ci-dessus contient une fonction dédiée :

void ecran\_accueil(void);

void choix\_plateau(int \* choix\_forme);

void saisie\_taille(int\* taille, int choix\_plateau);

void choix\_politique\_suggestion\_blocs(int\* choix);

L'affichage au niveau de la console se fait de manière brute avec des printf.

Les fonctions de retourne rien. La valeur saisie par l'utilisateur est enregistrée dans le paramètres modifié de la fonction. Ceci permet de réduire le nombre de variable nécessaire au programme. La saisie de l'utilisateur est sécurisé grâce a des boucles "faire tant que" (do while) et grâce a la réinitialisation du buffer d'entrée (stdin) par la fonction prédéfinie fflush(stdin);. Si l'utilisateur échoue la saisie alors un message d'erreur est affiché et une nouvelle saisie lui ai demandé.

**On supprime la console** à la fin de chaque fonction pour que le programme soit plus "esthétique". Afin de rendre le projet portable, il faut d'abord connaitre le système d'exploitation dans lequel on se situe. Ceci se réalise dans le fichier "fonction.h" à l'aide de condition de préprocesseur :

#ifdef \_WIN32

#define SYSTEME\_EXPLOITATION "win" *//windows*

#elif \_\_linux\_\_

#define SYSTEME\_EXPLOITATION "linux"

#elif \_\_APPLE\_\_

#define SYSTEME\_EXPLOITATION "mac"

#else

#define SYSTEME\_EXPLOITATION "unk" *//systéme inconnu*

#endif

La fonction "supr\_console" utilise donc la macro-constante définie en fonction du système d'exploitation pour effectuer la bonne saisie dans le terminal pour effacer la console :

void supr\_console(void)

{

system(SYSTEME\_EXPLOITATION == "win" ? "cls" : "clear");

}