Rapport

Notre projet consistait à coder un jeu Tetris en C. Pour cela, nous avons commencé par créer un dépôt GitHub, puis nous avons entamé le développement du programme. Ce dernier a été réalisé sur Replit, car c’est un bon compilateur pour coder en C et importer des fichiers.

À partir de là, nous avions les bases nécessaires pour commencer le code. Nous avons d'abord codé la matrice en utilisant une double boucle afin de l'initialiser conformément aux critères.

Créer les pièces n’a pas été très compliqué, car sur Replit, nous pouvons importer des fichiers .txt. Nous avons donc créé un grand fichier texte contenant plusieurs pièces. Pour l’apparence des pièces, nous avons utilisé des "1".

Nous avons ensuite codé le placement des pièces. Pour faire en sorte que celles-ci descendent, nous avons utilisé une fonction qui modifie les lignes et les colonnes.

Au début, nous avions du mal à positionner correctement les pièces dans les colonnes souhaitées : elles ne tombaient pas à l’endroit prévu. Nous avons donc ajusté le code en ajoutant des variables supplémentaires pour mieux gérer le placement par ligne et par colonne. Cela a permis de stabiliser le jeu.

La rotation a également posé problème. La première méthode que nous avons utilisée ne fonctionnait pas correctement. Nous avons donc opté pour l’utilisation d’angles fixes (90°, 180°, 270°), ce qui a rendu la gestion des rotations plus simple et plus efficace.

Pour supprimer les lignes pleines, nous avons utilisé une boucle qui vérifie si une ligne contient un zéro. Si c’est le cas, la ligne n’est pas supprimée. Si elle est entièrement remplie (composée uniquement de "1"), elle est vidée en la remplaçant par des zéros.

Nous avons également dû lire des fichiers .txt dans un dossier, mais nous ne savions pas comment faire. Nous nous sommes donc renseignés et avons utilisé les fonctions file open pour ouvrir les fichiers. Nous avons aussi utilisé la fonction dir avec un pointeur vers le dossier, ce qui nous a permis de parcourir les fichiers .txt et de compter leur nombre. Ensuite, nous avons utilisé les fonctions str, ce qui nous a permis de rechercher une sous chaîne dans une chaîne de caractères afin d’identifier et de compter les fichiers .txt.

Après cela, nous avons retravaillé notre affichage. À l'origine, il affichait des "0" pour les emplacements vides et des "1" pour les pleins. Nous l’avons amélioré dans un but esthétique : les "0" sont devenus invisibles, tandis que les "1" ont été conservés pour représenter les blocs.

Enfin, nous avons rencontré des problèmes concernant le placement des pièces dans la matrice. Lors de leur positionnement, nous comptions tous les zéros, ce qui créait un décalage à cause de ces zéros pris en compte. Nous avons donc ajusté notre logique pour éviter ces erreurs.

Pour terminer, la partie la plus complexe a été la gestion du placement de la matrice dans le jeu. Concernant les erreurs de dépassement de colonnes, nous avions d’abord envisagé d’afficher une erreur lorsqu’un chiffre dépassait le nombre de colonnes disponibles. Finalement, nous avons décidé que si un chiffre dépassait, la pièce apparaîtrait soit complètement à gauche, soit complètement à droite de la matrice.

En ce qui concerne l’architecture du projet, nous avons organisé notre travail de manière modulaire en séparant les fonctionnalités dans plusieurs fichiers. Le fichier main.c contient la boucle principale du jeu et appelle toutes les

fonctions nécessaires. Nous avons créé plusieurs fichiers d’en-tête, comme tetris.h pour la gestion de la grille, piece.h pour le chargement, la rotation et le placement des pièces, et score.h pour la gestion des scores. Les différentes pièces sont stockées dans des fichiers .txt situés dans un dossier spécifique. Cette organisation rend le code plus lisible.