Rapport Projet : Tech-tris

MI1-F : Louis Khadri-Boulay, Dimitri Chau, Ibrahim Boualaoui

Sommaire :

1) Description du projet :

Notre groupe MI1-F, composé de Louis Khadri-Boulay, Dimitri Chau et de Ibrahim Boualaoui, a décidé de choisir le projet Tech-Tris consistant à réaliser une version plus statique du Tetris. Maniant gestion de fichiers et tableaux 2D, l'utilisateur aura la possibilité de placer une pièce, pour laquelle il aura choisi la rotation, dans une grille de 10 par 10, selon une limite de temps. Cette pièce est contenue dans un tableau de pièces, obtenu depuis un fichier texte contenant les pièces. La particularité du projet est que les pièces sont dessinées par l'utilisateur dans ce fichier texte selon différentes règles. De même que le jeu initial, un score s'incrémente à la réalisation d'une ligne de blocs dans la grille.
Les variantes du projet qui nous ont été imposées sont : les pièces à jouer sont choisies aléatoirement ; il y a exactement 7 pièces à dessiner ; ces pièces sont stockées ensemble dans un fichier unique.
Face à cette version statique, nous avons décidé de rendre le jeu davantage dynamique, se rapprochant de la version originale de Tetris tout en ajoutant des fonctionnalités inédites et en respectant le cahier des charges.
Pour rappel, on nous demandait de faire en sorte que le joueur choisisse où il veut poser la pièce. Ainsi nous avons fait en sorte que les pièces puissent chuter dans la grille et que nous puissions tourner la pièce, accélérer sa descente ou encore, la faire chuter d'un coup.
Nous avons même inventé des options inexistantes dans le jeu Tetris original, comme une sauvegarde de la partie en cours et l'utilisation d'un atelier afin de créer ses propres pièces au lieu de modifier directement dans le fichier texte. La réalisation de ces pièces doit respecter une mise en forme dans le fichier texte "piecesmodifiees.txt", 1 pour des blocs, 0 pour le vide et des # comme séparateur. Ces pièces sont dessinées dans un tableau de 5 par 5, elles doivent être composées de 5 blocs ou moins, dont le bloc central du tableau est forcément placé.
Nous avons aussi ajouté un affichage coloré à partir d'émoticônes et un système de difficulté qui accélère la chute de la pièce.

Le flux de travail s'est organisé de manière progressive : une base fonctionnelle a d'abord été mise en place (création et affichage des pièces, grille de jeu, déplacement), avant d'y intégrer des modules plus avancés

(rotation, sauvegarde, atelier, score). Chacun a pu travailler sur une partie bien définie tout en s'adaptant aux évolutions du projet. Grâce à une bonne répartition des tâches et l'instauration d'une date butoir par semaine, l'avancée a pu être efficace. L'intégration des différentes fonctionnalités s'est faite de manière continue, en testant les fonctionnalités avec le programme principal. Pour permettre la transmission d'informations, un groupe Discord et Whatsapp a été créé.

2) Répartitions des tâches :

- Ibrahim a pu gérer la partie stockage des informations des joueurs et également l'affichage des couleurs dans le terminal, permettant un jeu plus esthétique. Il a aussi proposé plusieurs alternatives à son code pouvant s'adapter à la structure principale du code. Pour finir, Ibrahim s'est occupé du README disponible sur GitHub. Soucieux des erreurs, Ibrahim s'est occupé davantage de tester chaque version du programme pour découvrir les différentes erreurs et dysfonctionnements afin d'améliorer la robustesse de ce dernier.
- Louis s'est occupé du squelette du jeu, tel que la rotation des pièces, leur création, ou bien l'affichage sur le terminal avec des hashtags. Étant donné que Louis a codé une grande partie du code, il a aussi pu aider au débogage de celui-ci.
- Dimitri s'est occupé des fichiers de sauvegarde d'une partie et du score. Il est l'inventeur et le concepteur de son atelier permettant de modifier des pièces, tout en imposant les règles de réalisation de pièces à l'utilisateur. Dimitri a joué un rôle majeur dans l'adaptabilité et la vérification des codes réalisé par chacun.

3) Problèmes majeurs rencontrés :

L'un des problèmes auxquels nous avons été confrontés fut celui de la rotation des tétrominos. En effet, une première version du code fut écrite par Dimitri, qui consistait en un simple algorithme de rotation de matrice (les blocs pleins étant représentés par des 1 et les vides par des 0). Cependant, le problème était que la manière dont les tétrominos étaient implémentés dans la grille de jeu ne correspondait pas à ce code (les tétrominos sont représentés par un double tableau contenant les coordonnées de chaque bloc). Louis essaya tout d'abord de simplement adapter le programme pour qu'il prenne en argument la structure Tetromino au lieu d'une matrice, mais le programme était fait pour appliquer une rotation dans une grille de 5 par 5 par rapport à son centre (le centre de cette grille étant le centre du tétromino). L'adapter dans la grille de Tetris faisait appliquer une rotation par rapport au centre de cette même grille (qui n'est alors plus forcément le centre du tétromino), ne donnant pas le résultat attendu. Il fut alors décidé d'utiliser de la trigonométrie et les coordonnées polaires, en appliquant une rotation de $\pi/2$ par rapport au centre du tétromino. Mais après de nombreux tests et cas problématiques, la trigonométrie n'était pas non plus concluante. Louis pensa alors à revenir au code de départ (celui de la rotation dans une grille de 5 par 5). Si le problème était que le tétromino ne se trouvait pas dans une grille comme telle, alors il suffisait de faire comme si on déplaçait le tétromino dans une grille de 5 par 5 : en utilisant les coordonnées du centre du tétromino (le tableau contenant celles-ci est toujours le premier du double tableau des blocs), on connaît sa distance par rapport aux coordonnées (2,2) (centre d'une grille de 5×5). On a alors juste à soustraire cette distance sur chaque bloc du tétromino pour le "déplacer" en haut à gauche de la grille afin que les coordonnées correspondent à celles d'une grille de 5×5, on applique la rotation, puis on redéplace le tétromino à l'endroit où il était précédemment. Le code fonctionnait, le problème était résolu.

Un deuxième problème en lien avec la rotation est apparu : celui des collisions. Au départ, pour éviter que le tétromino sorte de la grille lors d'une

rotation, une condition avait été ajoutée pour que la rotation ne se fasse pas quand le tétromino collait une bordure. Mais cela ne réglait qu'à moitié le problème, car le tétromino pouvait quand même traverser les autres tétrominos "morts" au sol. Après quelques recherches, le procédé des wall kicks utilisé dans le jeu original fut découvert par Louis. On décida alors de l'implémenter dans le code à notre manière. Ce procédé est composé de plusieurs étapes : on applique d'abord la rotation sur un tableau temporaire et on vérifie si elle est valide. Si c'est le cas, on copie les coordonnées de ce tableau dans celles du vrai tétromino. Sinon, on applique plusieurs wall kicks (des vecteurs de 1 allant dans une direction unique : haut, bas, etc.) et on vérifie encore si la rotation est valide pour ensuite l'appliquer. Si ce n'est toujours pas le cas, alors on ne tourne pas la pièce. L'implémentation de ce procédé fut réalisée, résolvant ainsi le problème.

Finalement, pour s'assurer d'avoir un programme robuste, nous avons passé une journée à provoquer des erreurs dans notre jeu (scanf, lecture de fichiers...). Chaque fois que notre programme plantait, nous passions un moment à résoudre le problème et s'assurer que l'erreur soit traitée correctement. L'une des erreurs qui fut la plus difficile à gérer était la lecture de fichiers corrompus. En effet, nous modifions manuellement les fichiers textes afin de provoquer les erreurs. De cette manière, nous avons pu traiter différents cas de figures (mauvais séparateur, nombres de pièces différents de 7, mauvais caractère, mauvais indicateur...) et améliorer grandement la robustesse de notre programme.

4) Conclusion :

Le projet TechTris a abouti à un jeu fonctionnel enrichi par plusieurs fonctionnalités : pièces choisis par le programme, coloration du terminal, système de sauvegarde, ou encore un atelier de création de pièces personnalisées. Les principales fonctionnalités de Tetris sont opérationnelles, y compris la rotation des pièces avec la gestion des collisions. L'objectif du groupe a été atteint, réaliser une version de Tetris similaire à la toute première version de Tetris.