ENSET
ECOLE NORMALE SUPÉRIEURE DE L'ENSEIGNEMENT TECHNIQUE DE MOHAMMEDIA
UNIVERSITÉ HASSAN II DE CASABLANCA
المدرسة العليا لأساتذة التعليم التقني المحمدية

Rapport sur le Projet "TÉTRISTE" en Langage CPP

BENABID Ibtissam AMOUHAL Nouhayla

1. Introduction :

Ce rapport présente le projet de développement d'un jeu en C++, intitulé : “Tetriste", réalisé en utilisant la bibliothèque SFML (Simple and Fast Multimedia Library). L'objectif principal de ce projet est de créer un jeu de puzzle original, inspiré du concept classique de Tetris, où les joueurs doivent former des ensembles de trois pièces ayant soit la même couleur, soit la même forme. Sur un plateau de jeu, des pièces de différentes formes et couleurs descendent progressivement. Les joueurs doivent les manipuler et les positionner stratégiquement afin de créer des ensembles cohérents de trois pièces horizontalement. Ces ensembles disparaissent lorsqu'ils sont formés, laissant de la place pour de nouvelles pièces. Tetriste vise à offrir une expérience de jeu stimulante, mettant l'accent sur la réflexion stratégique et la planification, tout en mettant en pratique les concepts de programmation orientée objet en C++ et l'utilisation de la bibliothèque SFML pour la gestion des graphismes et des événements utilisateur. Ce rapport détaille les différentes composantes du projet, y compris les structures de données, les fonctionnalités implémentées, et les perspectives d'amélioration pour le jeu.

2. Méthodes et Bibliothèques Utilisées :

Ce projet fait usage de différentes bibliothèques et méthodes pour sa réalisation. Voici une description détaillée de celles-ci :

- Bibliothèques Utilisées :

- **SFML (Simple and Fast Multimedia Library)** : Cette bibliothèque est utilisée pour la gestion des fenêtres, le rendu graphique, ainsi que la gestion des événements utilisateur.

SFML offre une interface simple et efficace pour la création de jeux et d'applications multimédias interactives en C++.

- **<vector>** : La bibliothèque `<vector>` est utilisée pour la gestion dynamique des pièces sur le plateau de jeu. Elle permet de stocker et de manipuler facilement des collections de pièces sans avoir à spécifier une taille fixe à l'avance, ce qui est essentiel pour la flexibilité et la gestion efficace des pièces dans le jeu Tetriste.

- **<cstdlib> et <ctime>** : Ces bibliothèques standard de C++ sont utilisées pour la génération de nombres aléatoires. En combinant `rand()` avec `srand(time(nullptr))`, elles permettent de créer des séquences de nombres aléatoires pour la génération aléatoire des formes et des couleursdes pièces dans le jeu.

- **<iostream>** : Cett e bibliothèque est utilisée pour l'entrée et la sortie standard. Bien qu'elle ne soit pas directement impliquée dans la logique du jeu, elle est utilisée pour afficher des messages à l'utilisateur, tels que "Game Over!" ou d'autres informations pertinentes pendant le déroulement du jeu.

- **<algorithm>** : Cette bibliothèque standard de C++ est utilisée pour les opérations algorithmiques, telles que `std::swap()`. Dans le contexte de ce projet, `std::swap()` est utilisé pour échanger deux éléments dans le vecteur de pièces, ce qui est utile pour la manipulation des pièces lors du déplacement dans le jeu Tetriste.

Ces bibliothèques offrent un ensemble de fonctionnalités essentielles qui sont utilisées pour créer et gérer efficacement le jeu Tetriste, en assurant la robustesse, la performance et la flexibilité nécessaires à son développement.

3. Structures de Données :

Dans ce projet, différentes structures de données sont utilisées pour représenter les pièces du jeu et le plateau de jeu lui-même.

- Enums :

- `ShapeType` : Cet enum définit les différentes formes que peuvent prendre les pièces sur le plateau de jeu. Il comprend des valeurs telles que Rectangle, Triangle, Losange et Circle, permettant ainsi de diversifier les formes des pièces.

- `Color` : Cet enum définit les différentes couleurs que peuvent avoir les pièces sur le plateau de jeu. Il comprend des valeurs telles que Red, Blue, Green et Yellow, offrant ainsi une variété de couleurs pour les pièces.

- Structures :

- `Piece` : Cette structure représente une pièce individuelle sur le plateau de jeu. Chaque pièce est caractérisée par sa forme (`ShapeType`) et sa couleur (`Color`). En associant une forme et une couleur à chaque pièce, cette structure permet de définir et de manipuler efficacement les différentes pièces présentes dans le jeu Tetriste.

4. La Classe `Gameboard` :

La classe `Gameboard` est au cœur de la logique du jeu Tetriste. Elle gère le plateau de jeu, les pièces qui s'y trouvent et les interactions avec le joueur.

- Attributs :

- **`BOARD_WIDTH` et `BOARD_HEIGHT` :** Ces constantes définissent respectivement la largeur et la hauteur du plateau de jeu.
- **`board` :** Ce vecteur contient les pièces actuellement présentes sur le plateau de jeu. Il est utilisé pour stocker et manipuler dynamiquement les pièces pendant le déroulement du jeu.
- `timeSinceLastPiece` : Ce flottant représente le temps écoulé depuis l'ajout de la dernière pièce sur le plateau de jeu. Il est utilisé pour contrôler la génération périodique de nouvelles pièces.
- **`clock` :** Cette horloge est utilisée pour mesurer le temps écoulé dans le jeu, notamment pour le contrôle de la génération de nouvelles pièces.
- **`selectedPieceIndex` :** Cet entier représente l'index de la pièce actuellement sélectionnée sur le plateau de jeu. Il est utilisé pour suivre la pièce que le joueur a sélectionnée pour manipulation.

- Méthodes :

- **`Gameboard()` :** Le constructeur de la classe qui initialise les attributs de la classe.
- **`~Gameboard()` :** Le destructeur de la classe qui ne réalise aucune action spécifique.
- **`initialize_Game()` :** Cette méthode initialise le plateau de jeu en générant les premières pièces.
- **`generate_Next_Shape()` :** Cette méthode génère une nouvelle pièce et l'ajoute au plateau de jeu.

- `draw()` : Cette méthode dessine le plateau de jeu et les pièces sur la fenêtre SFML.
- `handleInput()` : Cette méthode gère les entrées utilisateur pour déplacer les pièces sur le plateau de jeu.
- `canPlacePiece()` : Cette méthode vérifie si une pièce peut être placée à une position donnée sur le plateau de jeu.
- `removeMatches()` : Cette méthode supprime les pièces correspondantes du plateau de jeu si elles forment des ensembles cohérents.
- `isGameOver()` : Cette méthode vérifie si le jeu est terminé en vérifiant si le plateau de jeu est plein.

5. Fonction Principale (`main()`) :

La fonction principale `main()` du programme est responsable de démarrer le jeu Tetriste en initialisant la fenêtre SFML et en créant une instance de la classe `Gameboard`. Elle implémente ensuite la boucle principale du jeu, gérant les événements, la mise à jour de l'état du jeu et le rendu graphique.

- **Initialisation de la Fenêtre SFML :** La fonction principale démarre en créant une fenêtre SFML avec une taille prédéfinie et un titre spécifié.
- **Création de l'Instance de `Gameboard` :** Ensuite, une instance de la classe `Gameboard` est créée. Cela initialise le jeu en générant les premières pièces sur le plateau de jeu.
- **Boucle Principale du Jeu :** La boucle principale du jeu commence, où les événements sont gérés, l'état du jeu est mis à jour et le rendu graphique est effectué en continu. Cette boucle continue jusqu'à ce que l'utilisateur ferme la fenêtre de jeu.

6. Conclusion :

En conclusion, le projet Tetriste a été développé avec succès en utilisant la bibliothèque SFML et le langage de programmation C++. Les fonctionnalités principales du jeu ont été implémentées, notamment la génération et le déplacement des pièces sur le plateau de jeu, ainsi que la vérification des ensembles cohérents de pièces pour leur suppression.

Ce jeu offre une expérience de jeu divertissante et stimulante, mettant en avant la réflexion stratégique et la prise de décision rapide. Pour les améliorations futures, plusieurs fonctionnalités peuvent être envisagées, telles que l'ajout de niveaux de difficulté, de bonus spéciaux, ou même d'un mode multijoueur. En résumé, Tetriste représente un projet réussi qui démontre les compétences en programmation orientée objet en C++ et l'utilisation efficace de la bibliothèque SFML pour le développement de jeux.