CUSTOM PROGRAM DETAILED PLAN

I. Student Information

- Name: Truong Ngoc Gia Hieu

- Student ID (HCMC): SWS01217

- Student ID (Australia ID): 105565520

II. Custom Program Title: Tetris Game

III. Overview of Custom Program

The fascinating project Tetris game implementation showcases a digital version of the classic puzzle game. Moreover, the program challenge players to rotate and strategically place falling blocks (tetrominoes) to create complete horizontal lines, which then disappear, earning points. Furthermore, throughout the level, the game increases in speed and complexity as the player progresses, providing a continuous challenge. Notably, the game illustrates the understanding of four principles concepts of OOP including Abstraction, Encapsulation, Inheritance, and Polymorphism.

IV. Project Description Overvall for D Level

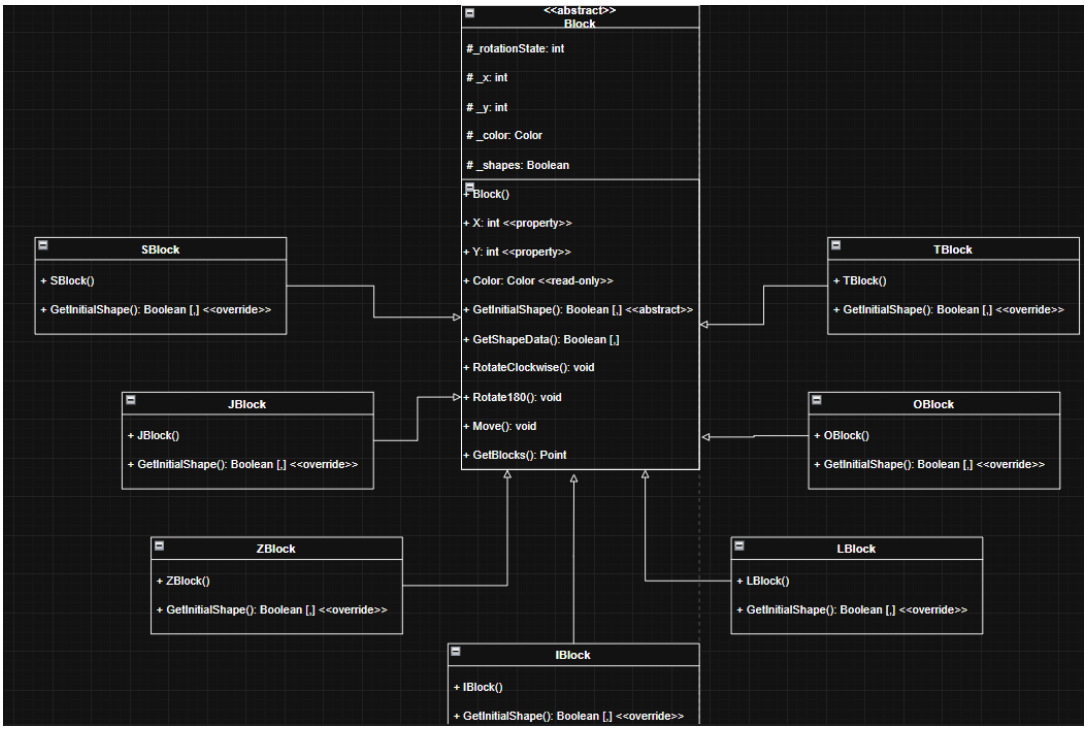
The game Tetris (Block game) consists of a grid include 22 rows (2 hidden rows) and 10 colunns. To achieve the D level, the game Tetris implements principles of Object-Oriented Programming (OOP). The gird will be in the center with black cells and white lines between cells. About the logic game, when start the program, the main Homepage will showcase which include the title “Welcome to the Tetris game” and buttons are below the text which include the Play button in the middle, Home button in the left side, and “Setting” button in the right side. When player click the button “Play”, the game will start. In the logic game, random blocks will drop down and player can utilize arrow keys on the keyboard to move blocks to left, right, and drop down faster. By the way, column on the left and right side will also work.

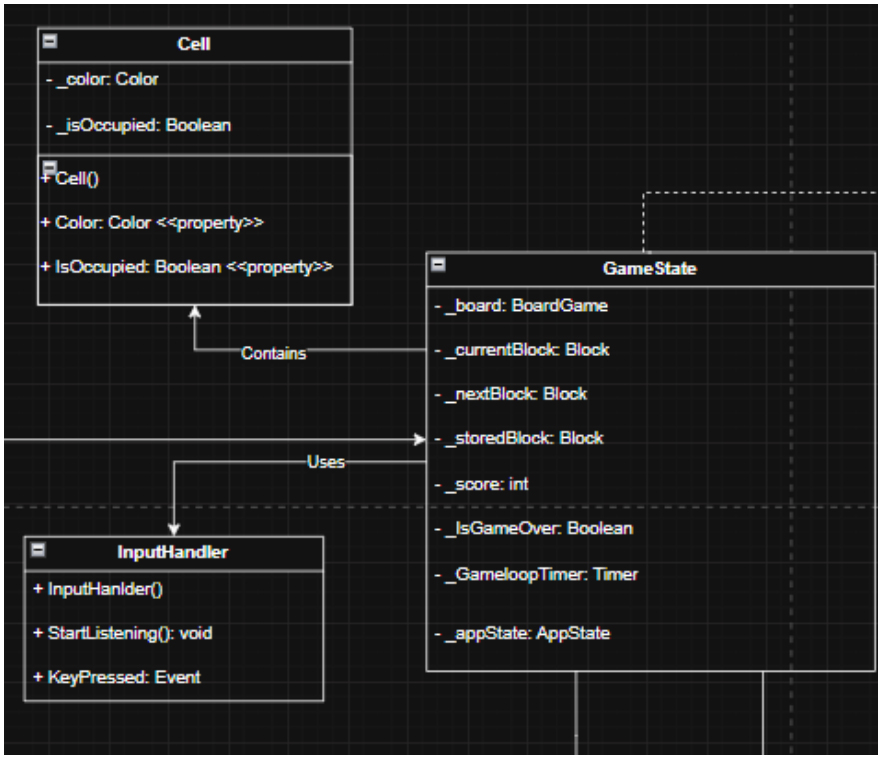
Moreover, on the left side, there will have test “Next” and under that word, a black rounded-corner rectangle will contains next random blocks. Notably, the game will have special function which allows the player to store the block if they want by pressing “S” (Store) key in the keyboard. As a result, the stored block and text “Store”will be put below the “Next”. Similarly, on the right side of the grid, the text “Current” will display and current block will be displayed in the black rounded-corner rectangle that is below the text. Furthermore, the Score will be also implemented too. In the logic for the D Level, the time will not limit and play until the blocks

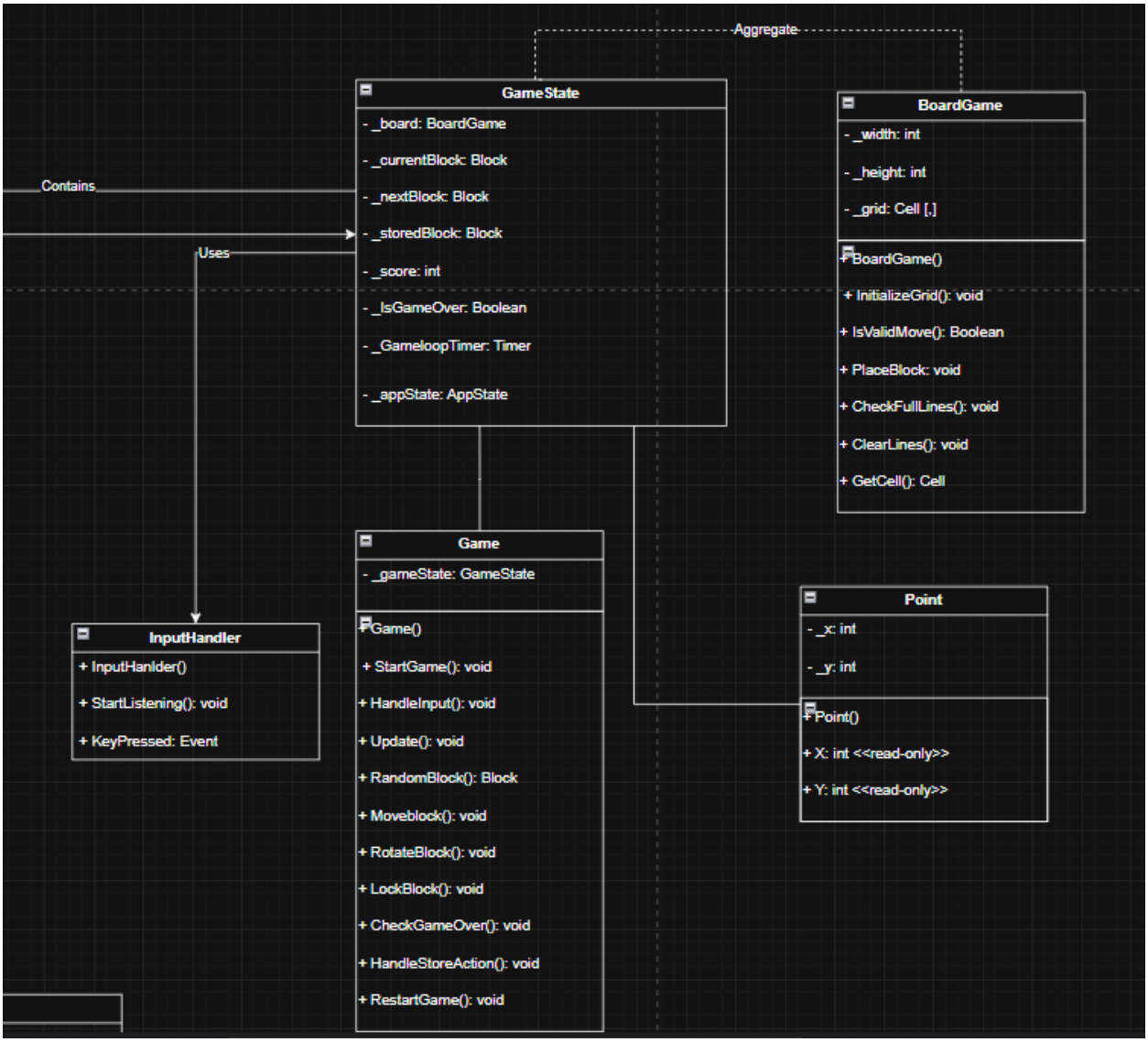
reach the top grid will stop. If reach top of the grid, the game will stop and ask the player wanna play again ot not. If yes, restart the game. Otherwise, return to Homepage

In summary, this is my initial plan for the custom program Tetris game which achieve level.

V. UML Diagram

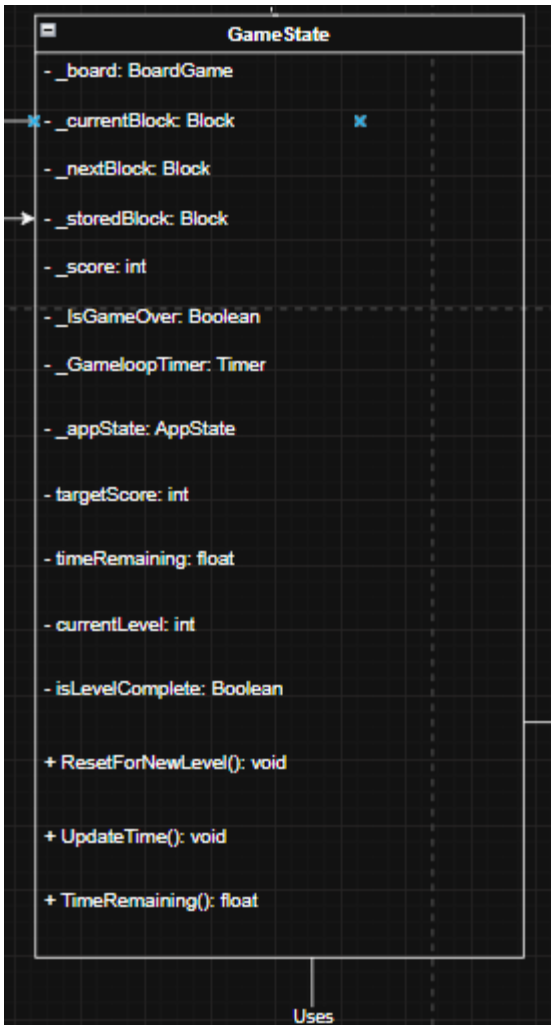


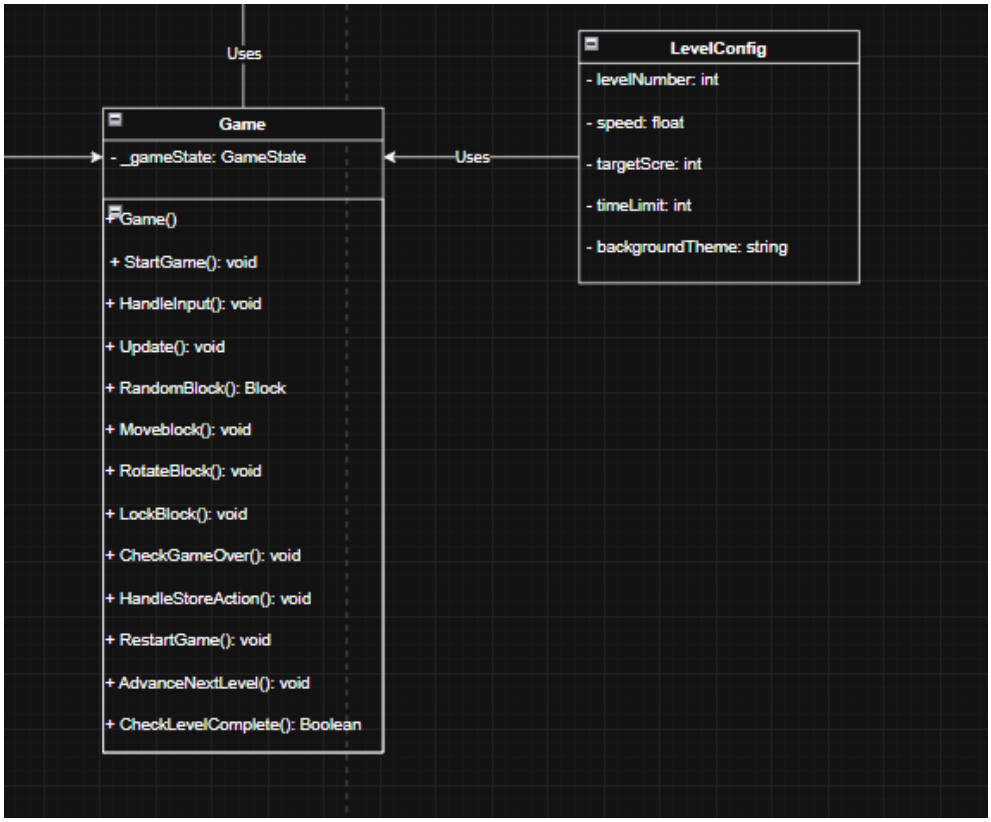


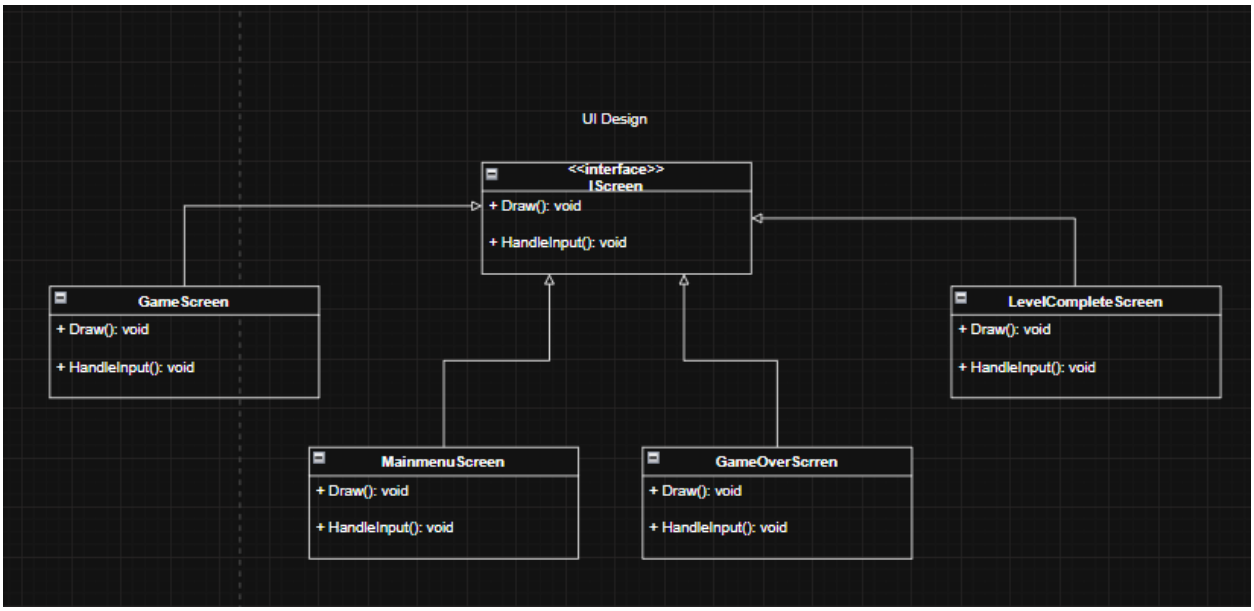


VI. Project Description Overvall for HD Level

As the description above, for the HD Level, I will implement different levels of game with different speed. For example. In the level I, time limit will be set and speed will be normally. In addtion, the player will play until they achieve the target score. After finishing Level I, the game will ask the player want to play next level or not. If No, returns to the Main Menu. If Yes, continue to Level which speed will be faster and target score will be higher with time smaller and loop will be repeat after finishing Level 2 and then following levels which background will be changed according to the level


GameState
- _board: BoardGame
- _currentBlock: Block
- _nextBlock: Block
- _storedBlock: Block
- _score: int
- _IsGameOver: Boolean
- _GameloopTimer: Timer
- _appState: AppState
- targetScore: int
- timeRemaining: float
- currentLevel: int
- isLevelComplete: Boolean
+ ResetForNewLevel(): void
+ UpdateTime(): void
+ TimeRemaining(): float
Uses


Uses
Game
- _gameState: GameState
+ Game()
+ StartGame(): void
+ HandleInput(): void
+ Update(): void
+ RandomBlock(): Block
+ Moveblock(): void
+ RotateBlock(): void
+ LockBlock(): void
+ CheckGameOver(): void
+ HandleStoreAction(): void
+ RestartGame(): void
+ AdvanceNextLevel(): void
+ CheckLevelComplete(): Boolean
Uses
LevelConfig
- levelNumber: int
- speed: float
- targetScre: int
- timeLimit: int
- backgroundTheme: string


UI Design
<<interface>>
IScreen
+ Draw(): void
+ HandleInput(): void
GameScreen
+ Draw(): void
+ HandleInput(): void
LevelCompleteScreen
+ Draw(): void
+ HandleInput(): void
MainmenuScreen
+ Draw(): void
+ HandleInput(): void
GameOverScrren
+ Draw(): void
+ HandleInput(): void