**Project 11: 2048 Game**

A screenshot of a game

Description automatically generated

**Introduction**

In this project, you will create 2048 games using Python. 2048 is a single player sliding block puzzle game. The game's objective is to slide numbered tiles on a grid to combine them to create a tile with the number 2048. The game is played on a 4x4 grid, and begins with two randomly placed numbers (2 or 4). The player can slide the tiles in any of the four directions: up, down, left, or right. All tiles slide as far as possible in the chosen direction until they are stopped by either another tile or the edge of the grid. If two tiles of the same number collide while moving, they will merge into a tile with the total value of the two tiles that collided. The resulting tile cannot merge with another tile again in the same move.

Every turn, a new tile will randomly appear in an empty spot on the board with a value of either 2 or 4. The game is over when the board is filled and no moves are left that can merge tiles, resulting in a game over. The player's score starts at zero and is increased whenever two tiles combine, by the value of the new tile.

**Tasks**

The following tasks need be completed to develop 2048 game:

1. Game Board Design: Create a 4x4 grid using a suitable data structure like a 2D list. This will serve as the game board.
2. Game Initialization: At the start of the game, two random positions on the grid should be filled with 2 or 4.
3. User Input: Implement functionality to capture user input. This will be the four arrow keys for moving the tiles up, down, left, and right.
4. Tile Movement Logic: Implement the logic for moving the tiles according to user input. If two tiles of the same number collide during movement, they should merge into a tile with the total value of the two tiles.
5. Random Tile Generation: After each turn, a new tile with a value of 2 or 4 should appear at a random empty position on the grid.
6. Game Over Condition: The game should end when there are no empty spaces left on the grid and no adjacent tiles have the same value, meaning no more moves are possible.
7. Scoring System: Implement a scoring system. The player's score starts at zero and is increased whenever two tiles combine, by the value of the new tile.
8. Blocks: Randomly put blocks/ obstacles in the board at random place. This block will stop the tiles from sliding and adding up. These blocks will act like a wall keeping a separation.
9. Obstacles can be broken when user reaches a very specific number in any tile on board. This can be a random number set by the program at the start of each game
10. User Interface (Optional): Although not a requirement, having a graphical user interface using a library like Tkinter or Pygame would greatly enhance the user experience.

* Tkinter is the standard Python interface to the Tk GUI toolkit. It is the most commonly used method for creating graphical user interfaces with Python. With Tkinter, you can create windows, labels, buttons, menus, textboxes, canvas for drawing, and many other widgets.
* For further reading go to https://docs.python.org/3/library/tkinter.html

For further questions you can contact Wu LIN at cswlin@comp.polyu.edu.hk & CC to muhammad.tayyab@connect.polyu.hk