Final Project

- 2048 -

Objective

Create a 4x4 grid where players continuously move and merge numbers. The game ends either when a cell reaches the value 2048, or when the board is filled and no further moves are possible.

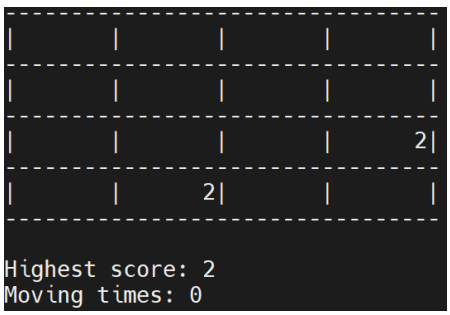
Rules

1. The game board must be a 4x4 grid.
2. The keys W, S, A, D represent up, down, left, and right movements.
3. At the start, two '2' tiles appear randomly. Each move spawns a new '2' or '4' tile.
4. Every 10 moves, a special tile '[/2]' appears, which removes adjacent tiles upon merging.
5. All tiles move to the farthest position in the chosen direction and merge if they are the same value.
6. A special rule for '[2]' interactions: When adjacent to a '[/2]' tile, '[/2]' disappears. If merged with '2,' the number remains unchanged; if merged with '4,' it becomes '2'; and so on.
7. The game ends if the board is filled without valid moves, resulting in a loss.
8. The game is won if any tile reaches 2048.
9. Display the highest tile value and total moves taken.
10. Show a success or failure screen at the game's end.

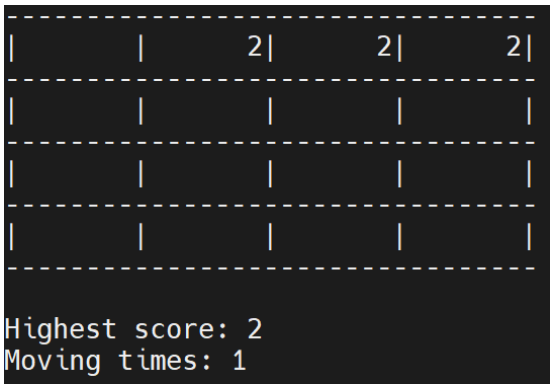
Examples

The diagram below is for reference only; you are encouraged to design your own.

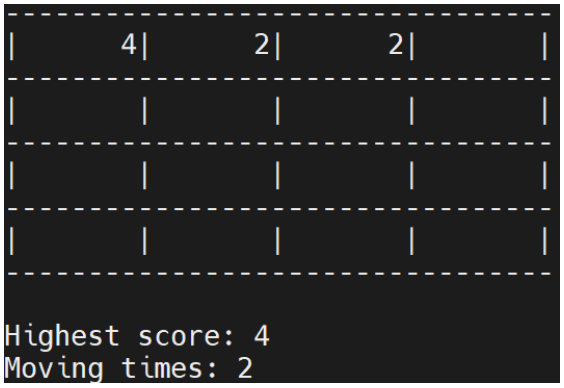
1. Initial game screen: Two random tiles with the number "2" appear.



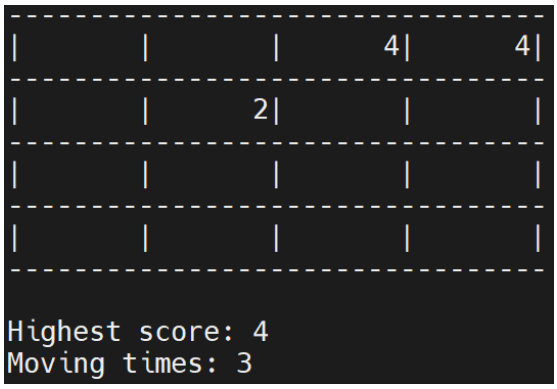
2. After selecting the upward movement, all numbers shift to the top, and a new tile with the number “2” is randomly generated.



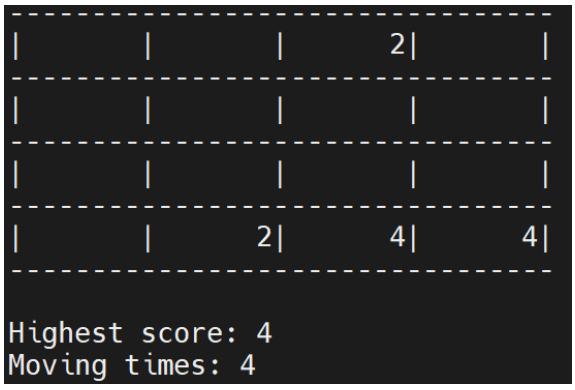
3. After selecting the leftward movement, all numbers shift to the far left, and when two tiles with "2" meet, they merge into "4." A new tile with the number "2" is then randomly generated.



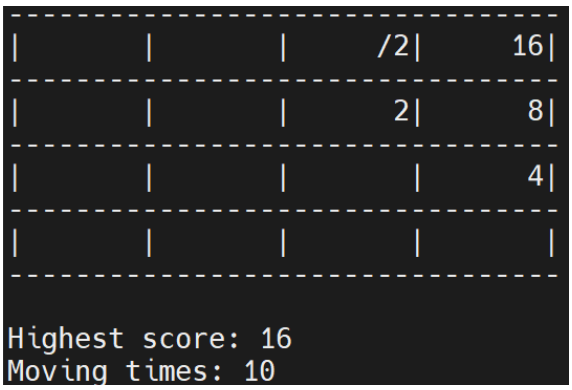
4. After selecting the downward movement, all numbers shift to the bottom, and a new tile with the number "2" is randomly generated.

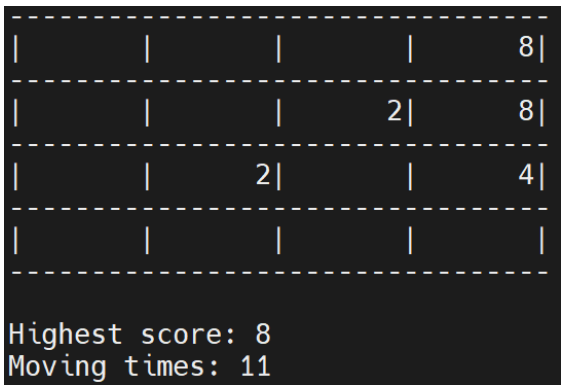


5. After selecting the downward movement, all numbers shift to the bottom, and a new tile with the number "2" is randomly generated.

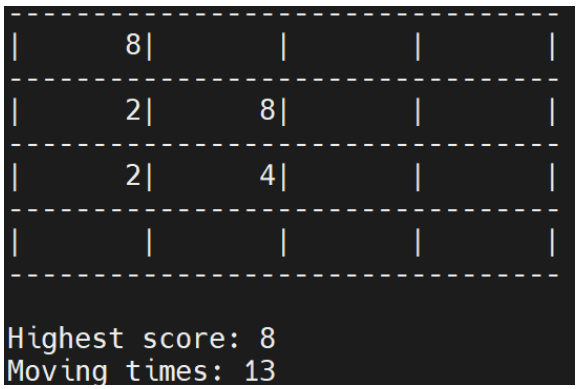


6. After selecting the rightward movement, all numbers shift to the far right. If "2" and "16" meet, they merge into "2," and a new tile with the number "8" is randomly generated.

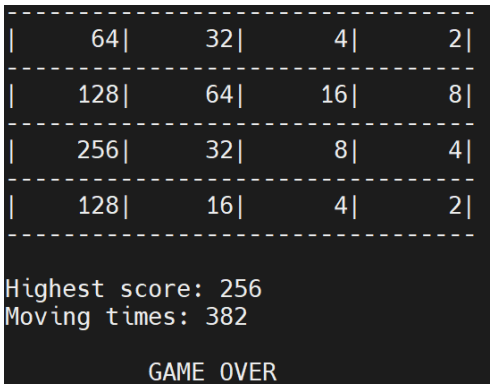




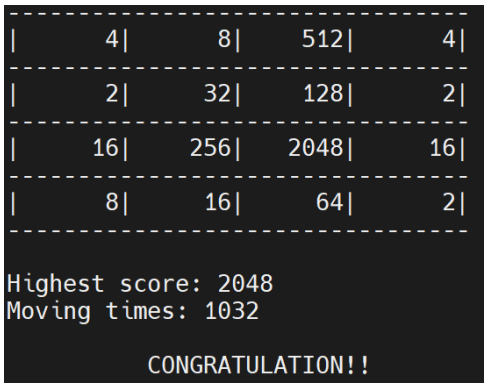
7. When the player makes an invalid move, the move count **does not increase**. For example, in the situation shown in the diagram, if the player chooses to move left but no updates occur, it is considered an invalid move, so the move count remains at 13.



8. When all tiles are filled and no adjacent numbers are the same, making further moves impossible, the game ends.



9. When a tile reaches the value of 2048, the challenge is successfully completed, and the game ends.



Scoring (70%)

1. Follows the game rules.
2. Displays the correct highest score and move count.
3. Shows an end screen.

Bonus (30%)

Extra points can be earned for enhancements like

1. improved graphics (5%)
2. an undo button (10%)
3. viewing game history (10%)
4. calculating total game time (5%)
5. other advanced features.

Submission Format

1. Source Files
   - Include all CPP files with detailed comments.
   - Include executable files (if required).
   - Ensure all necessary files for running the project are included.
2. Flowchart
   - Provide a flowchart that illustrates the game's logic and structure.
     - Alternatively, you can describe the flow in text if you prefer a written explanation over a diagram.
3. Project Report (Chinese is acceptable.)

Disclaimer

1. TA reserves the right to change any content in this project.
2. TA reserves the right to determine the final score.
3. TA reserved the right to modify this disclaimer.

TA

1. 鄭律恒 (mike.ee13@nycu.edu.tw)
2. 黃尹蓁 (yz.ee13@nycu.edu.tw)
3. 葉承岳 (jasonyeh0514@gmail.com)

Plagiarism is strictly prohibited.
Code from the Internet cannot be used directly.
If found, the score will be 0.
We will report to the professor and handle the matter according to the relevant regulations.