MAZE GAME WITH MONSTER

152120141003 Şafak AKINCI

Eskişehir Osmangazi University
Faculty of Engineering and Architecture
Department of Computer Engineering
Computer Programming Lab Term Project

Lecturer Yıldıray ANAGÜN

APRIL 2019

SECTION 1

INTRODUCTION

Summary Of The Project

A two dimensional matrix whose sizes are taken from the user is used for the maze. Both row and column must be between 10 and 100. For example, if row and column are both 10 then the starting and final point should be (0, 0) and (9, 9) respectively. The maze is firstly filled by random numbers between 0 and 4. The explanation of these numbers can be seen at Figure 1. Since the number are randomly chosen, we can not guarantee that there will always be a path filled by 0 from starting point to the final point. In order to satisfy this I wrote a function called createPath which takes two arguments, starting point and final point. In this function, I randomly get one of the direction either DOWN or RIGHT since the final point can only be reached like that. I added those points into a vector then wrote 0 to those points in the maze matrix. The collected golds are counted and stored in a global counter called collectedGolds. But there is a limitation, if the collected gold is counted one it should NOT be recounted when user tries to collect it again. I used another vector called collectedGoldPoints to know the location of the collected golds. By doing this, I satisfy that the gold only be counted ONCE if the user collects it. Whenever user moves to monster the game is restarted and all the collectedGold is reset. If user achieve the final point, the movements (points which user visited) are saved along with the collected gold number in a text file called "game_scores.txt".

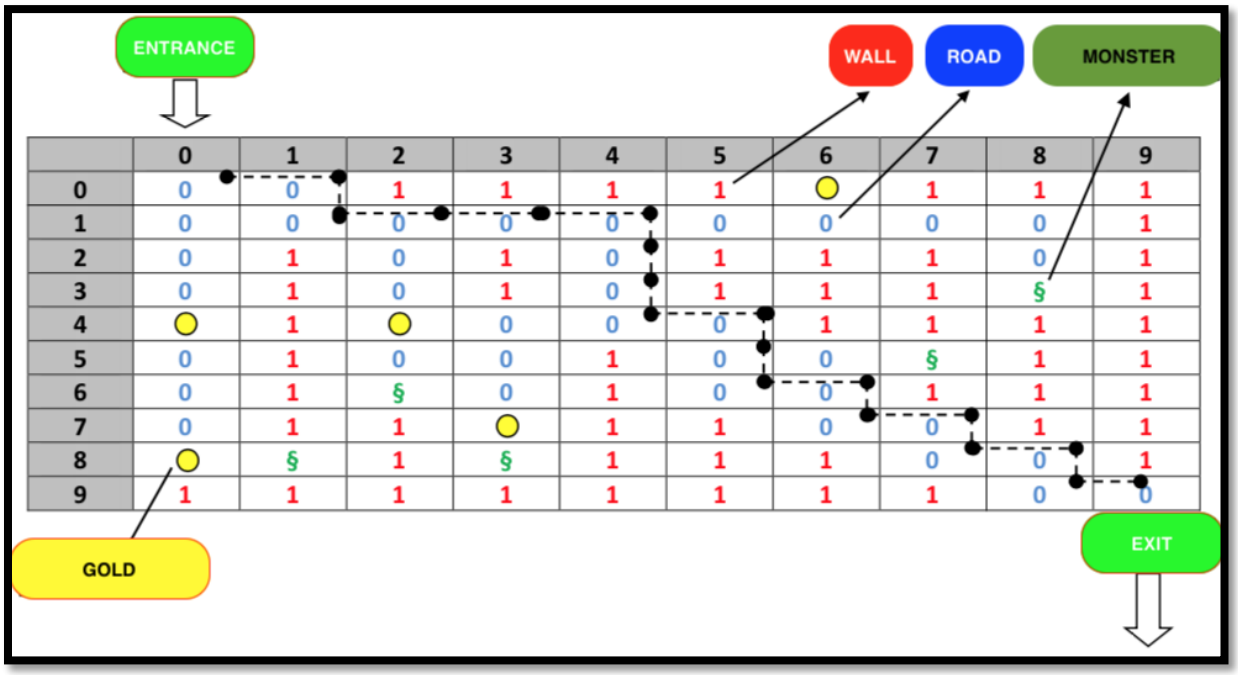

Figure 1. Maze and object explanations

```cpp
int main() {

    // initialize random seed:
    srand(time(NULL));


    getRowAndColumn();
    int** maze = getMazeMatrix();


    int randomNumberFrom = 0, randomNumberTo = 4;
    fillMazeMatrix(maze, randomNumberFrom, randomNumberTo);


    tuple<int, int> startPosition = make_tuple(0, 0);
    tuple<int, int> finalPosition = make_tuple(rowSize - 1, colSize - 1);
    vector<tuple<int, int>> path  = createPath(startPosition, finalPosition);
    drawPath(maze, path, ROAD);


    explainAim();
    explainObjects();


    player = whoIsPlayer();
    player == 'U' ? user(maze) : computer(maze);


    printScores();
    writeScoresToFile();


    system("pause");
    return 0;
}
```

Figure 2. Main function

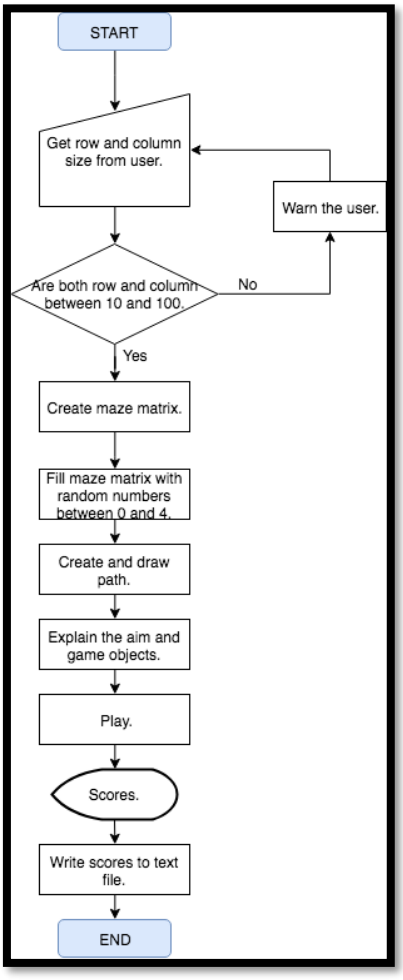

Figure 3. Flow diagram

Screenshots

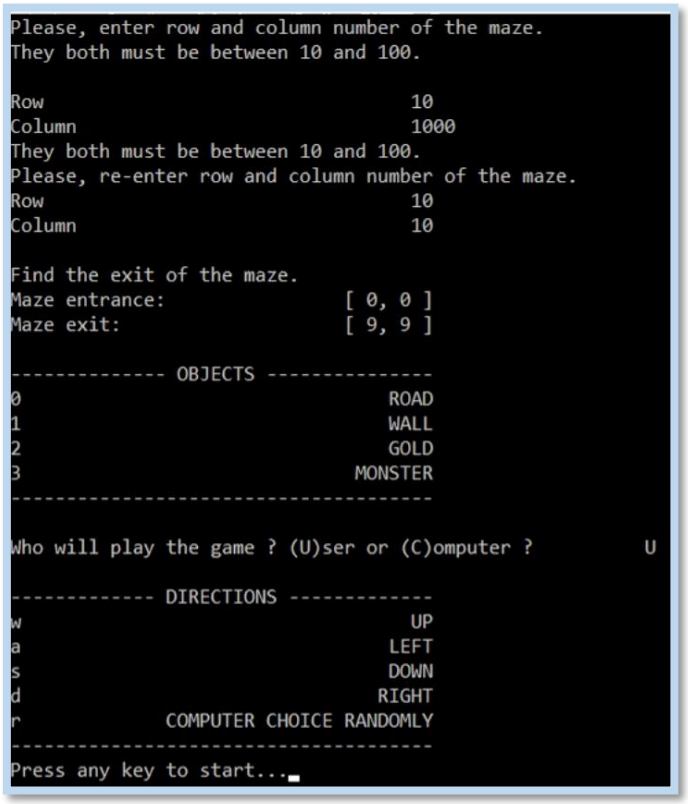

Figure 4. Row and column control

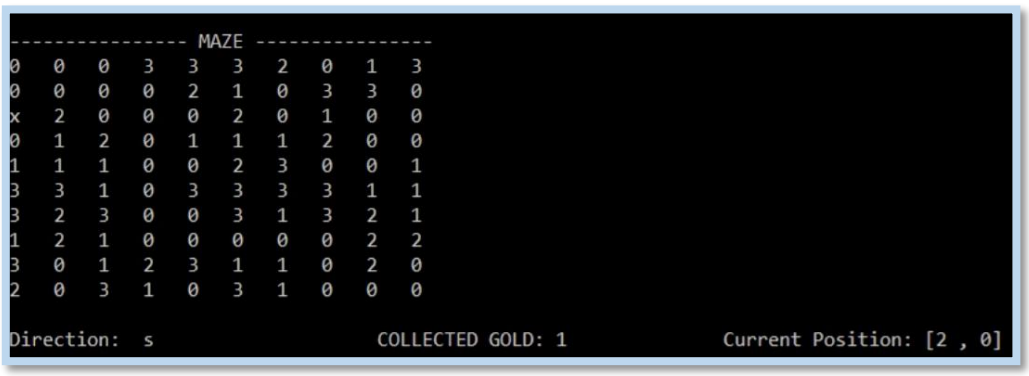

Figure 5. User plays

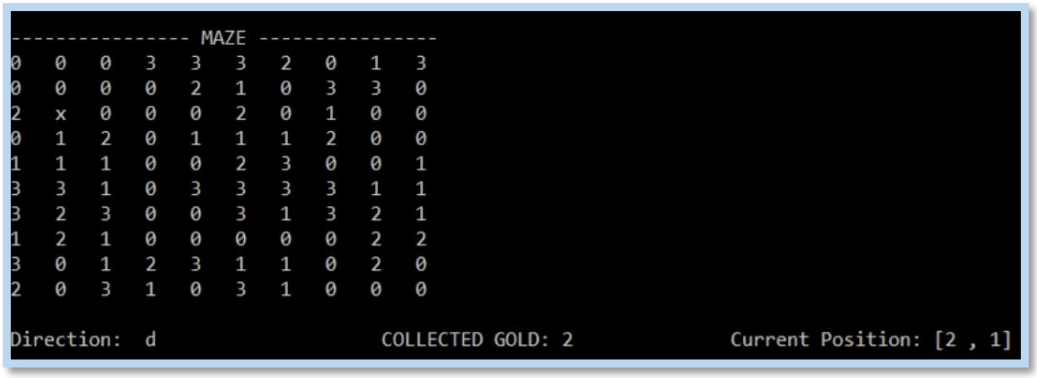

Figure 6. User collects gold

Figure 7. User is caught by monster

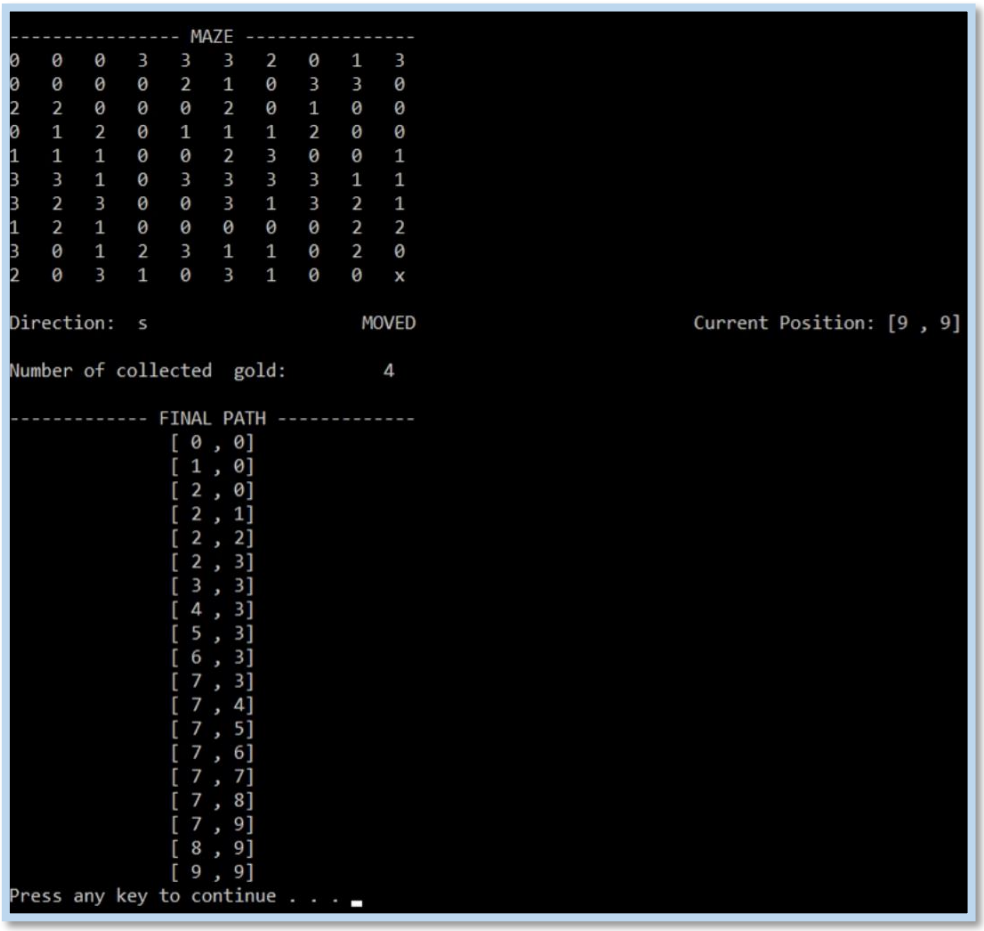

Figure 8. Print scores and path

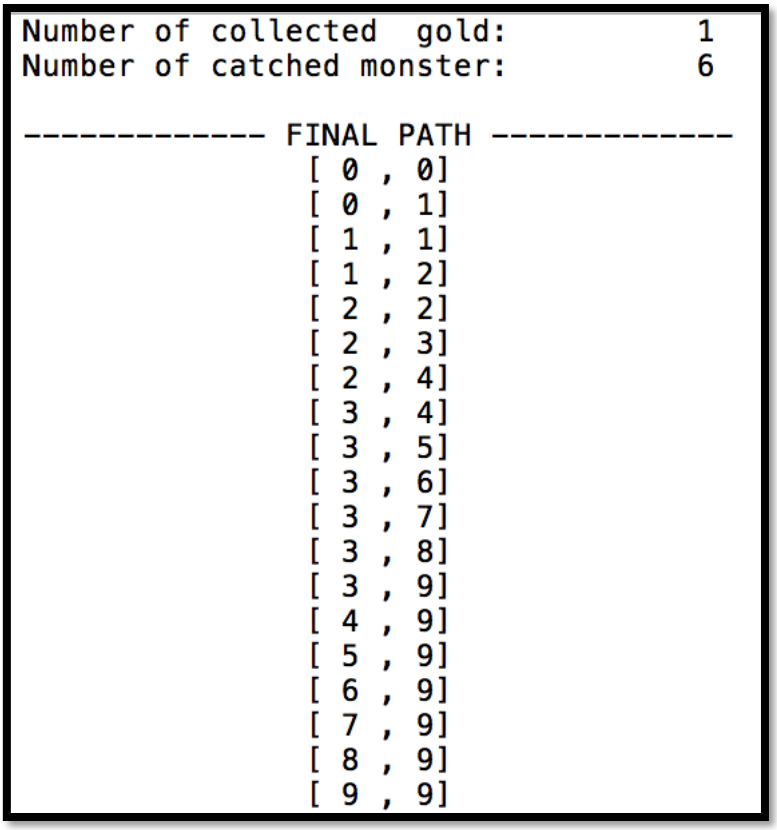

Figure 9. Scores and path in file