Report of Homework 2

In this homework, we were asked to make a game. First, I

called srand(time(NULL)) to generate random numbers each time. If this function is not called and we only use time(NULL), we would get the same random numbers each run.

This program checks if there is a generated file called **game_state.txt**, which holds the data of width, height, and the coordinates of the player and the door. If no such file exists, the program asks for width and height to generate a new file. To achieve this, I first tried to open **game_state.txt** in “r” (read) mode. The fopen function tries to open the file ONLY if it exists; otherwise, it returns NULL. This is why I used it to check if the file exists. If the file does not exist, my program tries to open and create the file in “w” (write) mode.

The user is then asked to input width and height. For example, if the user inputs “5 5”, the program generates a 2D board that is 5x5, with the xcoordinate ranging from 0 to width - 1 and the y-coordinate ranging from 0 to height - 1.

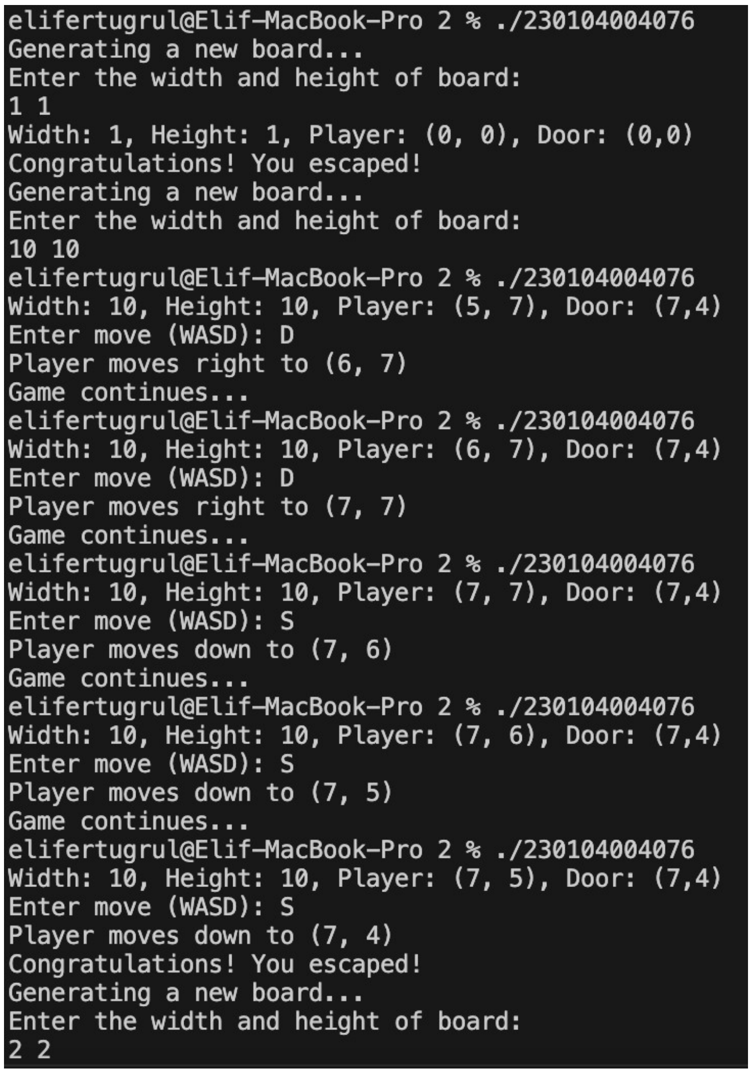
The player uses WASD to move across the board: ‘W’ for moving up, ‘A’ for moving left, ‘S’ for moving down, and ‘D’ for moving right. These controls mean: ‘W’ increases the y-coordinate, ‘A’ decreases the x-coordinate, ‘S’ decreases the y-coordinate, and ‘D’ increases the xcoordinate of the player.

On each run, the program checks if the player is hitting a wall by using if (playerY < 0 || playerY > height - 1)or if (playerX < 0 || playerX > width - 1). If this condition is true, my program undoes the move and does not update the **game_state.txt** file.

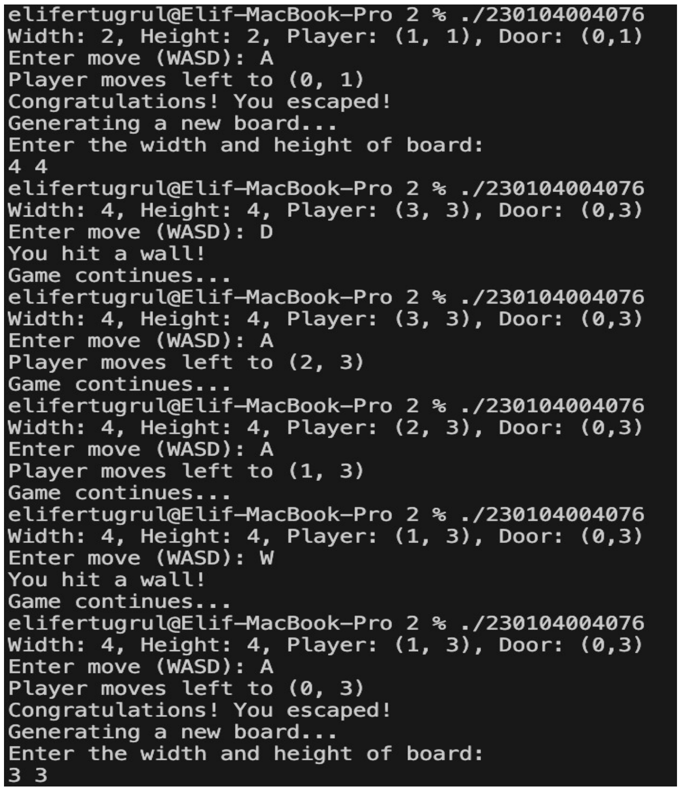
Lastly, the program checks if the user wins using the statement if (playerX == doorX && playerY == doorY). If this condition is true, the game ends, and a congratulations message is printed. After that, the program asks the user to input width and height to generate a new board.

Another thing of file operations in my program is that I need to close the file after overwriting its contents. If I don't, the "w" mode does not recursively overwrite its contents; it only does so once, and the rest of the data gets written after the existing contents. So, fclose is very useful for handling this problem.

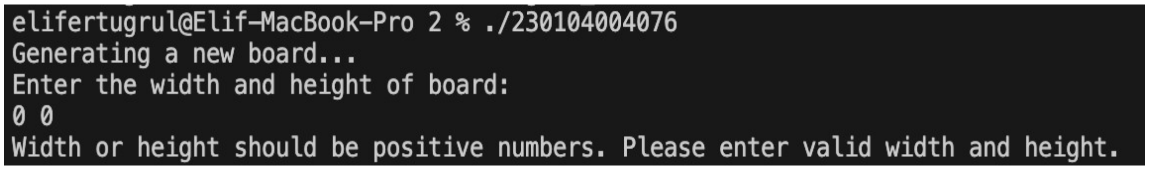
Here are some outputs of my homework:



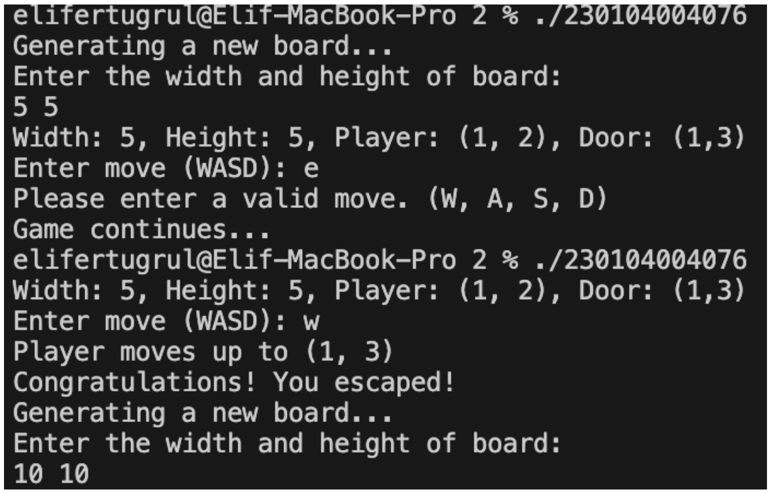
```
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Generating a new board...
Enter the width and height of board:
1 1
Width: 1, Height: 1, Player: (0, 0), Door: (0,0)
Congratulations! You escaped!
Generating a new board...
Enter the width and height of board:
10 10
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 10, Height: 10, Player: (5, 7), Door: (7,4)
Enter move (WASD): D
Player moves right to (6, 7)
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 10, Height: 10, Player: (6, 7), Door: (7,4)
Enter move (WASD): D
Player moves right to (7, 7)
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 10, Height: 10, Player: (7, 7), Door: (7,4)
Enter move (WASD): S
Player moves down to (7, 6)
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 10, Height: 10, Player: (7, 6), Door: (7,4)
Enter move (WASD): S
Player moves down to (7, 5)
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 10, Height: 10, Player: (7, 5), Door: (7,4)
Enter move (WASD): S
Player moves down to (7, 4)
Congratulations! You escaped!
Generating a new board...
Enter the width and height of board:
2 2
```



```
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 2, Height: 2, Player: (1, 1), Door: (0,1)
Enter move (WASD): A
Player moves left to (0, 1)
Congratulations! You escaped!
Generating a new board...
Enter the width and height of board:
4 4
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 4, Height: 4, Player: (3, 3), Door: (0,3)
Enter move (WASD): D
You hit a wall!
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 4, Height: 4, Player: (3, 3), Door: (0,3)
Enter move (WASD): A
Player moves left to (2, 3)
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 4, Height: 4, Player: (2, 3), Door: (0,3)
Enter move (WASD): A
Player moves left to (1, 3)
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 4, Height: 4, Player: (1, 3), Door: (0,3)
Enter move (WASD): W
You hit a wall!
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 4, Height: 4, Player: (1, 3), Door: (0,3)
Enter move (WASD): A
Player moves left to (0, 3)
Congratulations! You escaped!
Generating a new board...
Enter the width and height of board:
3 3
```



```
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Generating a new board...
Enter the width and height of board:
0 0
Width or height should be positive numbers. Please enter valid width and height.
```



```
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Generating a new board...
Enter the width and height of board:
5 5
Width: 5, Height: 5, Player: (1, 2), Door: (1,3)
Enter move (WASD): e
Please enter a valid move. (W, A, S, D)
Game continues...
elifertugrul@Elif-MacBook-Pro 2 % ./230104004076
Width: 5, Height: 5, Player: (1, 2), Door: (1,3)
Enter move (WASD): w
Player moves up to (1, 3)
Congratulations! You escaped!
Generating a new board...
Enter the width and height of board:
10 10
```