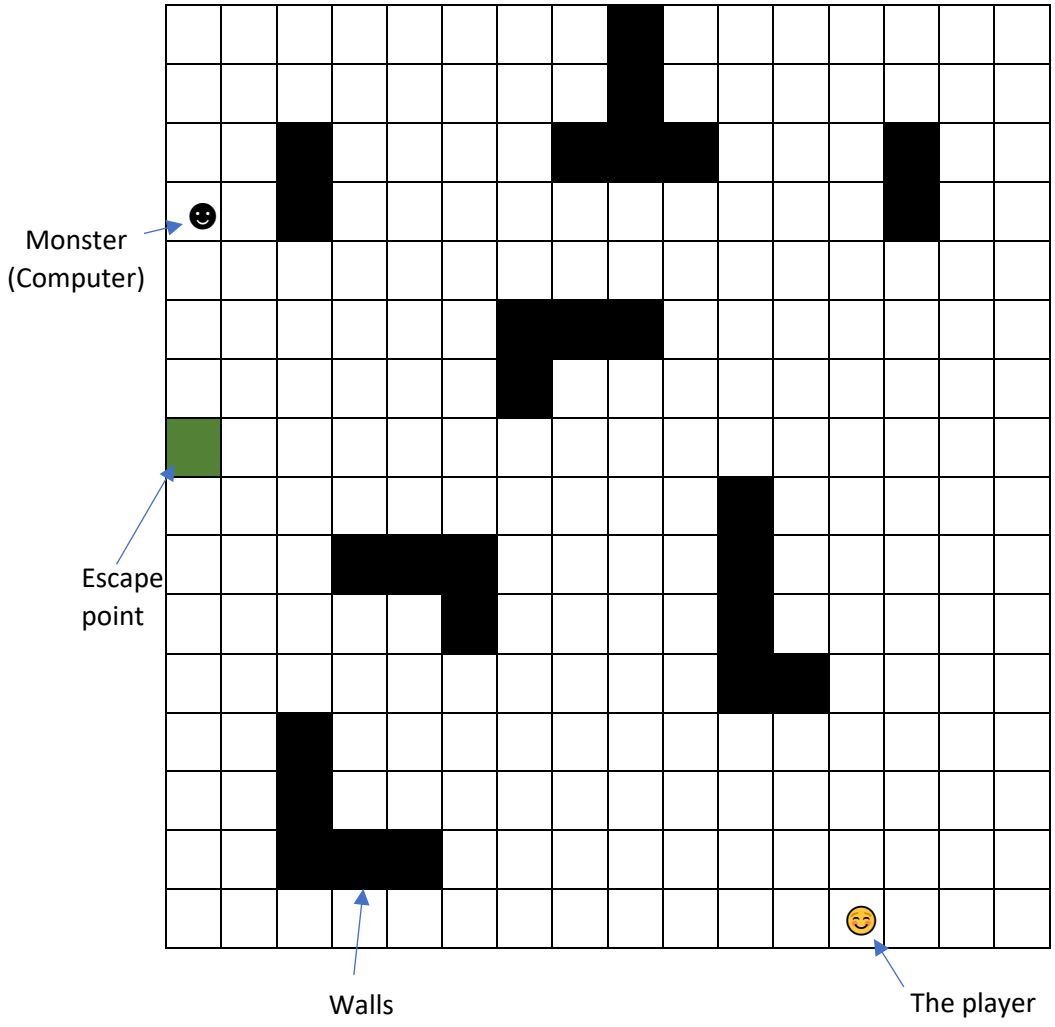
EEF110E 2021-2022 Spring 1st Homework

In this homework, you will be programming a game played against a computer. The game will be played on a 16x16 grid maze as shown below. You have to place the walls within the maze as shown below.

You will be trying to escape from the maze at the escape point before being caught by a monster (the computer) chasing you.



The player and monster are allowed to move up, down, left and right. No diagonal moves are allowed. The player and the monster cannot go through the walls and must move around them.

The game is played in turns. The player makes his/her move and then the computer makes its move.

The escape point will be on the 8th row and 1th column of the maze as shown in the figure. If the player reaches this square, the player wins the game. If, otherwise, the monster reaches the square where the player currently resides, the game is lost.

At the beginning of the game, the player and the monster are randomly placed on the maze not closer than **16 squares** from each other and the thief not closer than **16 squares** from the escape point in **Manhattan distance**.

The Manhattan distance between two squares at coordinates (x1, y1) and (x2, y2) is defined as |x2-x1| + |y2-y1|

The game will have two difficulty levels: easy and hard. In the easy level, the monster will make a random move to one of the feasible squares at each turn. A feasible square is a square that is adjacent to the current square the monster resides and that doesn't have a wall in it.

In the hard level, the computer will be smarter and will make a move to an **optimal** feasible square. An optimal feasible square is the square that makes the distance to the thief the shortest. If two or more squares have an equal distance, the square to move to will be picked randomly among the optimal feasible squares. You may use any algorithm you desire to find the optimal feasible squares. Hint: Don't let the monster fall into an infinite loop by checking the last position the monster was at before.

The game will first ask for the difficulty level. After the difficulty level is selected, the game will start with the player's move.

When the current game ends, a message that tells "You lost!" or "You won!" will be displayed on the screen depending on the outcome. Then, the player will be asked if she/he wants to play the game once more or wants to exit the game. If the player chooses to play once more, the above actions will repeat starting from choosing the difficulty level again. If the player decides to exit, the program will end.

Homework Deliverables

1. You should submit your source code. Your source code should be properly commented. Also write a comment to explain how to compile and run the source code.

2. Although not needed, you may include a binary, but your code will be tested whether it compiles and runs correctly.

Tips for the Homework

1. You may use ASCII characters _, |, W, P, T to represent maze square boundaries, walls, the monster and the thief, respectively when drawing the game platform to screen.

2. You can represent the maze by a 16x16 two-dimensional array. Each position in the array may represent either an empty square, a wall, the monster or the thief.

3. You can use system("clear") command to clear the screen when refreshing the maze