HW5-README

Paula Charles

November 2021

In this problem, I am reading a maze and finding a solution to go through it, from entrance to exit, knowing that there is an entrance at the top row of the maze and an exit at the bottom of it. I design this solution using the right hand following: I start with my hand on the right wall and I follow the wall with my hand up until the exit.

To do that, I designed a C++ file which tackles this issue in several steps:

1. I first compute the walls into a static array: the value of this array are set to 0 for the size of the maze (after verifying that the size of the maze is smaller than the size of the static array. I switch these values to 1 for every coordinate of a wall.
2. Knowing where the walls are, I start going through the maze: I find out where the entrance is. Going from there, I define a direction: it is the direction I am facing and it will enable me to know where my right is and thus to know how to move. This direction is updated every time I take a step.
3. Once I know the direction I am facing, I try to go to my right. If I can't (because there is a wall), I try to go straight. If I can't again because of a wall, I try to go left. Finally, if I can't go left, I go backwards.
4. I repeat this step of moving and modifying my direction up until I am in the last row: that is where I exit. This should enable me to follow the right hand solution.

This solution is valid only if the maze has the right format: there is only one entrance, the exit is on the last row (and not on the sides) and it is smaller than my static array (201x201, but can be modified in the code).

In the python file used to check if our C++ solution is valid, I created several functions which each are looking for a reason that would invalidate my solution: if they return True, it means my solution is valid for that issue.

- The first function checks if the solution is going through any wall. For that, I created 2 sets: one with the wall coordinates, one with the path coordinates : if they intersect, my solution does not work.

- The second function checks if the path is only moving one step at a time by comparing a set of coordinates for the path with the following one.
- The third function is checking if the solution is staying inside the maze boundaries. It is not needed if the maze is defined properly.
- The last function is checking if we are entering in the first row and exiting in the last one.

If every function states that the solution is correct, an output will appear on the screen claiming that this solution is correct.