To compile using terminal:

1. Go to the directory the file is located in
2. To compile run: g++ main.cpp
   1. if there is a warning message regarding auto start = std::chrono::high\_resolution\_clock::now();Please ignore it, it should not affect DFS search algorithm in maze).
3. To run application: ./a.out input.txt

Main Function:

1. use fstream to load input file.

2.Create the maze by calling Graph.create\_maze function, passing in the number of rows and columns find in the txt file.

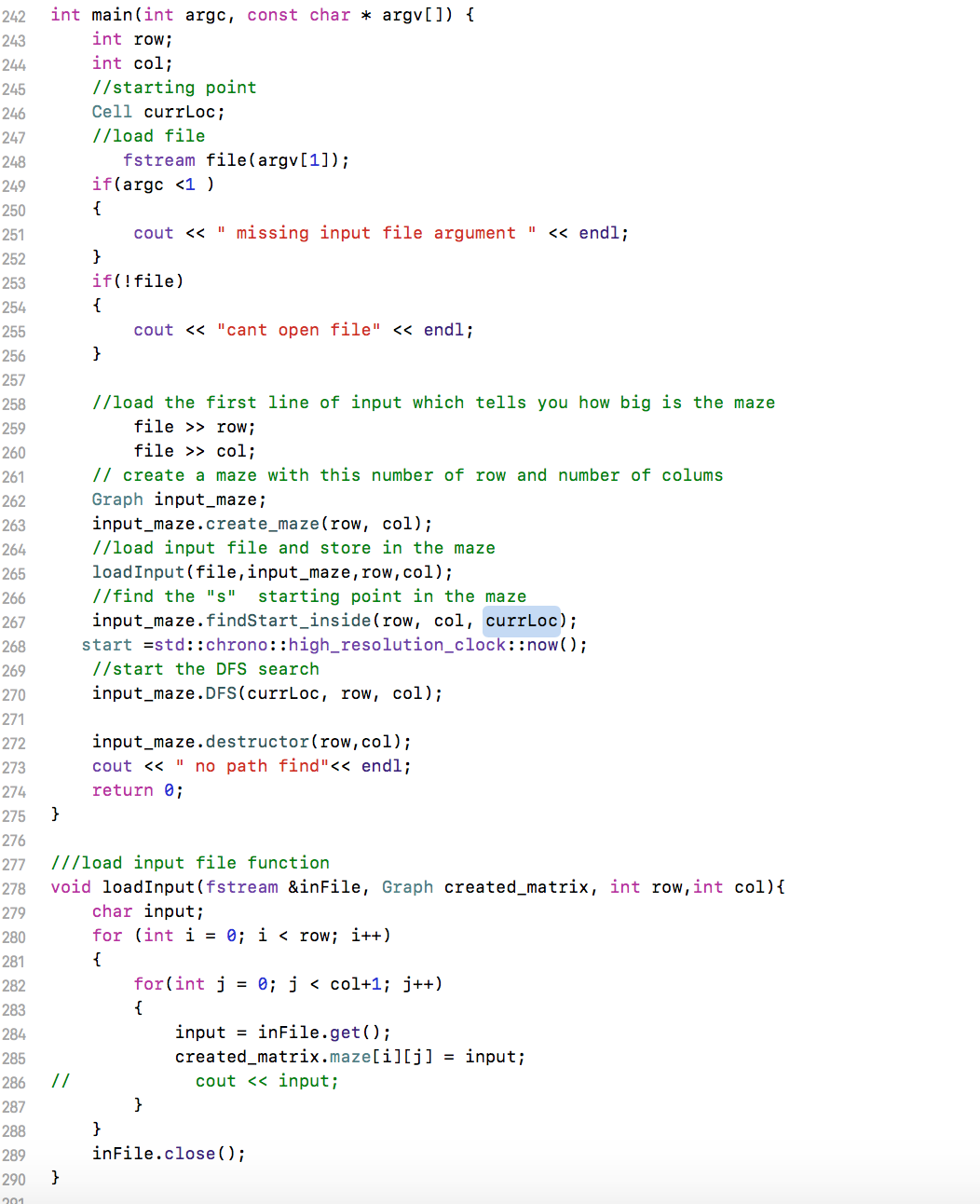
3. Call loadInput function to load the maze from the file.

4. use Graph.findStart\_inside function to find the starting point in the maze and store the point into a Cell object currLoc.

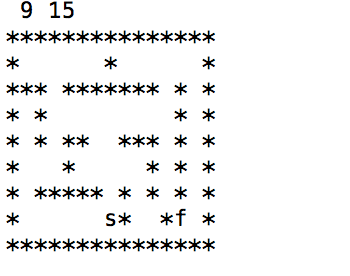
5.Call DFS passing in currLoc and number of rows and columns in the maze to find path.

6.If a path is found, it will get displayed by the display function inside DFS. The program is exit form DFS

7. If not found, return to main and finish executing.

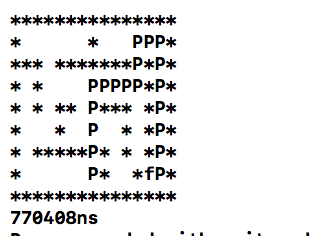


Input file example :



The input maze file is in txt . And the first row is number of rows and number of columns in the maze. “\*” represent walls and space represent path. “s” represent start point and “f” represents finish point

Output example:



In the output, “p” represent the path the algorithm found using DFS. If there is “.” appears in the maze, it is the mark representing this cell is visited it is not the path found using DFS.