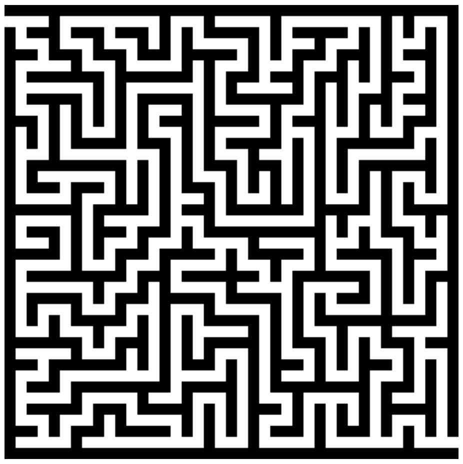
Today, you will construct an algorithm finding a shortest path through the binary maze. To this end, provide a private method `LoadMaze` which reads the text file, and creates the array of integers, with -1 representing the maze wall, 0 representing the maze corridor, 1 is the starting point, -2 is the finish. The walls in the text file are marked as W, the corridors as C, the start is S, the finish is F. The characters are separeted by the tabulator.



In order to find the solution, construct the simple (but not efficient) algorithm, exploring all possible paths, and then backtracking the shortest path. You will be able to move only in four directions, left, right, up, down (not in the diagonal directions). Perform your operations using the array cunstructed by the `LoadMaze`. You start with the node 1 , which is the start, and mark the nodes, reached after all possible single moves, as 2. Then you start with each node marked as 2 and then mark the nodes, reached after all possible single moves, as 3 (you need to exclude the walls, and the nodes that were visited already). To this end, create a private method `NumberAdjacent`, that for the node n numbers the potential adjacent nodes as $n + 1$. Create the public method `MakePaths`, that explores the maze, using the `NumberAdjacent`, until the final point is reached. Then, you can backtrack the maze, moving from the end to the beginning , at each move decreasing the node number. Create the public method `Backtrack`, which performs the backtracking and which creates the array holding the shortest path. Create the necessary methods to display your results (the maze and the shortest path).