Castle on the Grid ☆

247.86 more points to get your next star!

Rank: 280130 | Points: 227.14/475

Problem Solving ★★★

Problem Submissions Leaderboard Editorial

You are given a square grid with some cells open (**.**) and some blocked (**X**). Your playing piece can move along any row or column until it reaches the edge of the grid or a blocked cell. Given a grid, a start and an end position, determine the number of moves it will take to get to the end position.

For example, you are given a grid with sides $n = 3$ described as follows:

```
...
.X.
...
```

Your starting position $(startX, startY) = (0, 0)$ so you start in the top left corner. The ending position is $(goalX, goalY) = (1, 2)$. The path is $(0, 0) \rightarrow (0, 2) \rightarrow (1, 2)$. It takes 2 moves to get to the goal.

Function Description

Complete the minimumMoves function in the editor. It must print an integer denoting the minimum moves required to get from the starting position to the goal.

minimumMoves has the following parameter(s):

- grid: an array of strings representing the rows of the grid
- startX: an integer
- startY: an integer
- goalX: an integer
- goalY: an integer

Input Format

The first line contains an integer n, the size of the array grid.

Each of the next n lines contains a string of length n.

The last line contains four space-separated integers, $\mathbf{startX}$, $\mathbf{startY}$, $\mathbf{goalX}$, $\mathbf{goalY}$

Constraints

- $1 \leq n \leq 100$
- $0 \leq startX,\ startY,\ goalX,\ goalY < n$

Output Format

Print an integer denoting the minimum number of steps required to move the castle to the goal position.

Sample Input

```
3
.X.
.X.
...
0 0 0 2
```

Sample Output

```
3
```

Explanation

Here is a path that one could follow in order to reach the destination in 3 steps:

$(0,0) \rightarrow (2,0) \rightarrow (2,2) \rightarrow (0,2)$.

Current Buffer (saved locally, editable)

Java 7

```
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;

public class Solution {

    // Complete the minimumMoves function below.
    static int minimumMoves(String[] grid, int startX, int startY, int goalX, int goalY) {


    }

    private static final Scanner scanner = new Scanner(System.in);

    public static void main(String[] args) throws IOException {
        BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT_PATH")));

        int n = scanner.nextInt();
        scanner.skip("(\r\n|[\n\r\u2028\u2029\u0085])?");

        String[] grid = new String[n];

        for (int i = 0; i < n; i++) {
            String gridItem = scanner.nextLine();
            grid[i] = gridItem;
        }

```

```
32          String[] startXStartY = scanner.nextLine().split(" ");
33
34          int startX = Integer.parseInt(startXStartY[0]);
35
36          int startY = Integer.parseInt(startXStartY[1]);
37
38          int goalX = Integer.parseInt(startXStartY[2]);
39
40          int goalY = Integer.parseInt(startXStartY[3]);
41
42          int result = minimumMoves(grid, startX, startY, goalX, goalY);
43
44          bufferedWriter.write(String.valueOf(result));
45          bufferedWriter.newLine();
46
47          bufferedWriter.close();
48
49          scanner.close();
50      }
51  }
52
```

Line: 1 Col: 1

Upload Code as File Test against custom input Run Code Submit Code