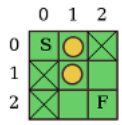
# ☆ Bob's and Alice's Chance

Bob and Alice have teamed up on a game show. They won the first round, allowing them access to a maze with hidden gold. If Bob can collect all the gold coins and deliver them to Alice's position, they can split the gold. Bob can move North⇆South or East⇆West as long as he stays in the maze and the cell is not blocked. The task is to determine the shortest path Bob can follow to collect all gold coins and deliver them to Alice. If it is not possible, return *-1*.

There is an *n* × *m* array where each of the values ∈ *{0, 1, 2}* representing *open, blocked* and *open with a gold coin*. Alice's position is given as *(x,y) = (row, column)*. Bob starts at the top left, at cell *(0, 0)*.

For example, *maze = [[0,2,1],[1,2,0],[1,0,0]]* with Alice at *(2,2)* is represented as follows:

```
0 2 1
1 2 0 →
1 0 0
```




Alice's position is marked with an *F* for *Finish*. Bob, starting at *(0,0)*, has two paths to Alice of length *4*.

**Function Description**

Complete the function *minMoves* in the editor below. The function must return the integer length of Bob's shortest path, or *-1* if it's not possible.

minMoves has the following parameter(s):

*maze[maze[0][0],...maze[n-1][m-1]]:* a 2D array of integers
*x:* an integer denoting Alice's row coordinate
*y:* an integer denoting Alice's column coordinate

**Constraints**

- *1 ≤ n, m ≤ 100*
- *0 ≤ the number of coins ≤ 10*
- *1 ≤ x < n*
- *1 ≤ y < m*

### Input Format For Custom Testing

The first line contains an integer *n*, the numbers of rows in *maze*.
The second line contains an integer *m*, the number of columns in *maze*.
Each of the next *n* lines contains *m* space-separated integers that describe the cells of each row in *maze*.
The next line contains an integer *x*.
The next line contains an integer, *y*.

### Sample Case 0

**Sample Input 0**

```
3
3
0 2 0
0 0 1
1 1 1
1
1
```

**Sample Output 0**

```
2
```

**Explanation 0**

The shortest path Bob can take is *(0, 0) → (0, 1) → (1, 1)*.

### Sample Case 1

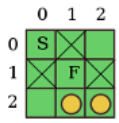
**Sample Input 1**

```
3
3
0 1 0
1 0 1
0 2 2
1
1
```

**Sample Output 1**

```
-1
```

**Explanation 1**




It is not possible for Bob to reach Alice, so return *–1*.

### Sample Case 2

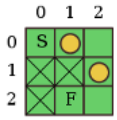
**Sample Input 2**

```
3
3
0 2 0
1 1 2
1 0 0
2
1
```

**Sample Output 2**

```
5
```

**Explanation 2**




The shortest path Bob can take is *(0, 0) → (0, 1) → (0, 2) → (1, 2) → (2, 2) → (2, 1)*.

YOUR ANSWER

We recommend you take a quick tour of our editor before you proceed. The timer will pause up to 90 seconds for the tour. Start tour

Saving draft.. View Code Diff Java 8

```
 1  import java.io.*;…
14
15  class Result {
16
17      /*
18       * Complete the 'minMoves' function below.
```

Mindtree
Welcome to possible

Mindtree Ltd Full-stack Developer ...

03 : 56
to test end

6d8ab688c3

```
23        *  2. INTEGER x
24        *  3. INTEGER y
25        */
26
27       public static int minMoves(List<List<Integer>> maze, int x, int y) {
28       // Write your code here
29
30       }
31
32   }
33
34   public class Solution {⋯
```

Line: 14 Col: 1

Test against custom input

Run Code | Submit code & Continue

(You can submit any number of times)

Download sample test cases *The input/output files have Unix line endings. Do not use Notepad to edit them on windows.*