Robot in Maze

Description

Imagine that we have an **N-by-M** grid, such that some squares on the grid are filled with obstacles (marked as 'x'). A robot located at in the bottom left corner of the grid, and wishes to move to the top right corner by a sequence of moves (go up, go right, go diagonally up and to the right) while avoiding squares that contain obstacles. Design an efficient algorithm to determine the number of ways a robot may accomplish this task?

Complexity

complexity of your algorithm should be **bounded by O(N x M)**

Function: **Implement it!**

```
public static long RequiredFunction(char[,] grid)
```

PROBLEM_CLASS.cs includes this method.

Example

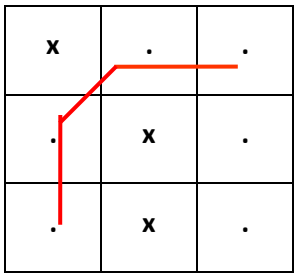
- **Example 1: No Path**

.	x	.
.	x	x
.	.	.

- Here, no possible path from the bottom-left corner to the top-right corner

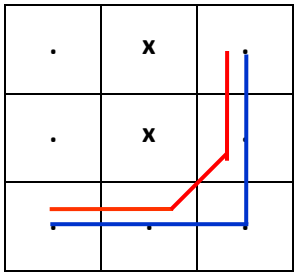
---

- **Example 2: One possible Path**



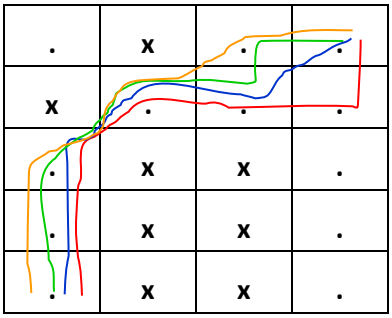
- Here, there's ONLY one path from the bottom-left corner to the top-right corner as shown.

- **Example 3: Two possible Path**



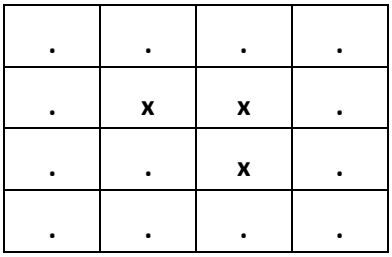
- Here, there're TWO paths from the bottom-left corner to the top-right corner as shown.

Example 4: Multiple Paths



- Here, there're FOUR possible paths from the bottom-left corner to the top-right corner as shown.

Example 5: Multiple Paths



- Here, there're FOUR possible paths from the bottom-left corner to the top-right corner.

C# Help

Getting the size of 1D array

```
int size = array1D.GetLength(0);
```

Getting the size of 2D array

```
int size1 = array2D.GetLength(0);
```

```
int size2 = array2D.GetLength(1);
```

Creating 1D array

```
int [] array1D = new int [size]
```

Creating 2D array

```
int [,] array2D = new int [size1, size2]
```

Sorting single array

Sort the given array "items" in ascending order

```
Array.Sort(items);
```

Sorting parallel arrays

Sort the first array "master" and re-order the 2nd array "slave" according to this sorting

```
Array.Sort(master, slave);
```