

Escape Mines

A turtle must walk through a minefield. Write a program that will read the initial game settings and one or more sequences of moves, then for each move sequence, the program will output if the sequence leads to the success or failure of the little turtle. The program should also handle the scenario where the turtle doesn't reach the exit point or doesn't hit a mine.

Notes

The first line of the file contains the board settings.

The second line of the file contains a list of mines.

The third line of the file contains the exit point.

The fourth line of the file contains the starting position of the turtle.

From the fifth line to the end of the file are specified the sequences of moves.

Example:

5 4

1,1 1,3 3,3

4 2

0 1 N

R M L M M

R M M M

Where

R = rotate 90 degrees to the right	N = north direction
L = rotate 90 degrees to the left	S = south direction
M = move	E = east direction
	W = west direction

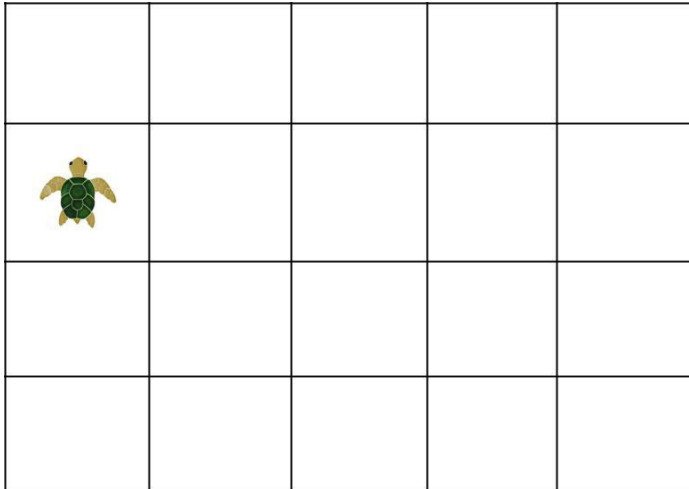
Inputs

The board is a square of n by m number of tiles:

5x4 Board

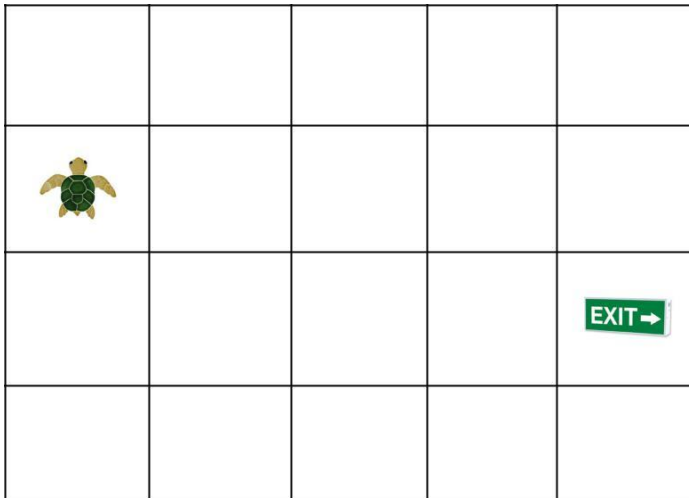
The starting position is a tile (x,y) and the initial direction the turtle is facing (that is: north, east, south, west):

Starting position: $x = 0$, $y = 1$, dir = North

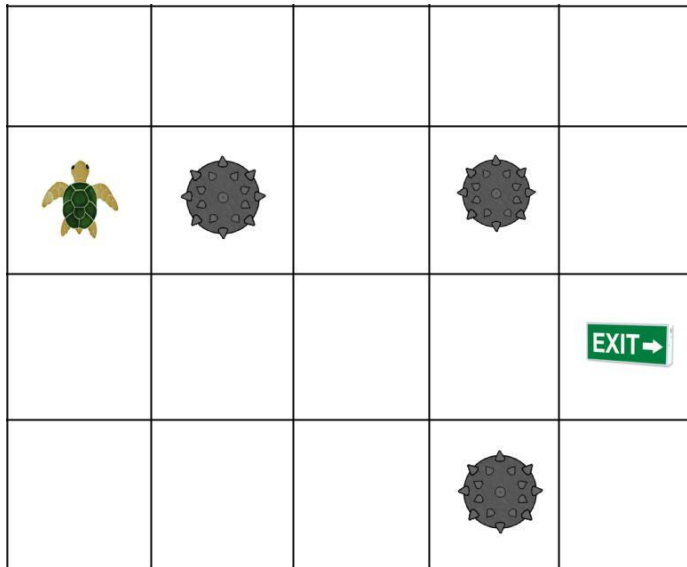


The exit point is a tile (x,y)

Exit point: $x = 4$, $y = 2$



The mines are defined as a list of tiles (x,y).



Turtle actions can be either a **move** one tile forward, **rotate(r)** 90 degrees to the right or **rotale(l)** 90 degrees to the left.

Deliverables

You are required to submit a solution written in C# or F# containing a program that models the problem.

You will be evaluated on your ability to decompose this problem. The breakdown and approach to the problem is as important / more important than it compiling and running. We expect quality over quantity; your code should be good enough to be shipped to production. This implies that the business logic needs to be optimized for performance and fully unit tested.

Any use of frameworks, other than for testing, is strongly discouraged.

While implementing your solution keep in mind that we do not like mutants and mutators.

Example

Given a file containing the board size, starting point and direction, exit point, mines and sequences of moves called "game-settings"

When I run the program passing the filename as a parameter, the program will print out the result for each sequence in the "game-settings" file.

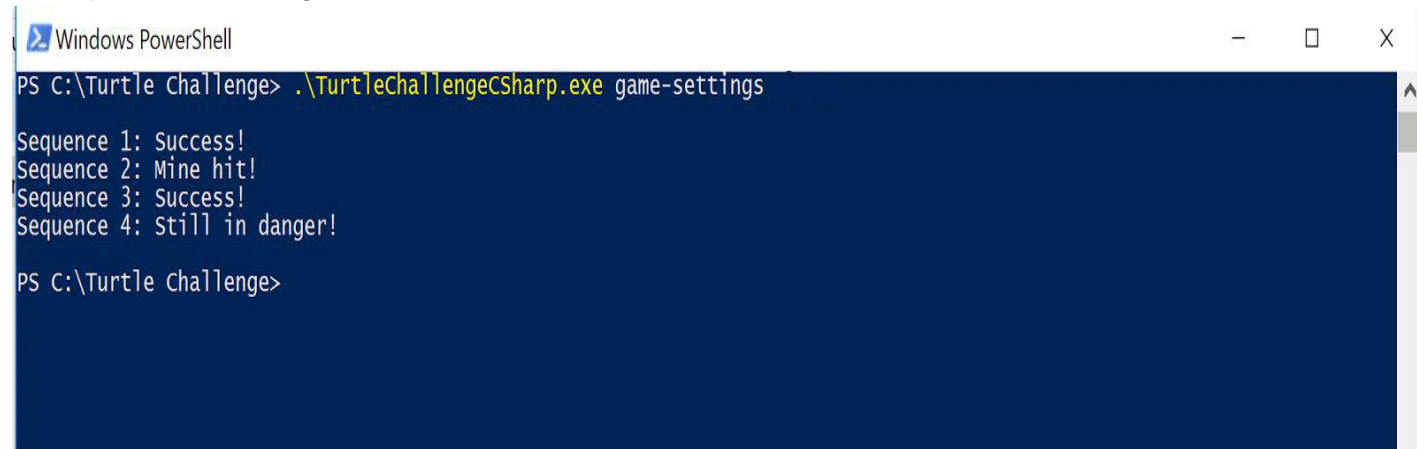
Results can be:

- Success!, if the turtle found the exit point
- Mine hit!, if turtle hit a mine
- Still in danger!, if the turtle did not exit nor hit a mine

For the file described below

```
5 4
1,1 1,3 3,3
4 2
0 1 N
M R M M M M R M M
R M M M
L L M L M M M M
M R M M M
```

we expect the following result

A screenshot of a Windows PowerShell window with a dark blue background. The title bar at the top reads "Windows PowerShell" and includes standard window control buttons (minimize, maximize, close). The command prompt shows the user is in the directory "C:\Turtle Challenge" and has executed the command ".\TurtleChallengeCSharp.exe game-settings". The output of the command is displayed in white text: "Sequence 1: Success!", "Sequence 2: Mine hit!", "Sequence 3: Success!", and "Sequence 4: Still in danger!". The prompt then returns to "PS C:\Turtle Challenge>".

```
PS C:\Turtle Challenge> .\TurtleChallengeCSharp.exe game-settings
Sequence 1: Success!
Sequence 2: Mine hit!
Sequence 3: Success!
Sequence 4: Still in danger!
PS C:\Turtle Challenge>
```