  You are working in an Engineering Squad for the  Melody Mars Mission, tasked with designing software to manage robots  and cool vehicles for space exploration!  

Your Task

Setting the Scene

You have been asked to create a program to move rovers around the surface of Mars! ✨

The surface of Mars is represented by a Plateau, you can make the assumption that the Plateau is a square/rectangular grid for the purpose of this task.

Rovers navigate the Plateau so they can use their special cameras  and robot arms to collect samples back to Planet Earth 

Representation of a Rover's Position on the Plateau

The Plateau is divided into a grid. A Rover's position is represented by x and y co-ordinates and the letters N, S, W, E to represent North, South, West, East (the four cardinal compass points) respectively.

Example

0 0 N

This means the Rover is at the bottom-left corner facing in the North direction.

N.B. Assume that the square directly North from (x, y) is (x, y+1), and the square directly East from (x, y) is (x + 1, y)

Instructing a Rover to Move Around the Plateau

�� To move a Rover around the Plateau, a string of letters is sent to a Rover.

Here are the letters and their resultant action:

Letter Action
L Spins the Rover 90 degrees ***Left*** without moving from the current coordinate point

	R Spins the Rover 90 degrees ***Right*** without moving from the current coordinate point
	M Moves the Rover forward by one grid point, maintaining the same heading/orientation

N.B. Assume that the square directly North from (x, y) is (x, y+1).

Inputs into the Program

First Line of Input to the Program

The first line inputted into the program represents the upper-right coordinates of the Plateau.

5 5

This Plateau has maximum (x, y) co-ordinates of (5, 5).

N.B. Assume that the lower-left coordinate is (0, 0).

Subsequent Lines of Input into the Program - Input to Rovers

This represents the instructions to move the rovers.

Each rover receives **two lines of input.**

First Line of Input to a Rover

The Rover's position is represented by two integers representing the X and Y coordinates and a letter representing where the Rover is facing (its orientation).

1 2 N

Second Line of Input to a Rover

A string of letters representing the instructions to move the Rover around the Plateau.

Movement Rules

Rovers move sequentially, this means that the first Rover needs to finish moving first before the next one can move.

Output

For each Rover, the output represents its final position (final coordinates and where it is facing).

Example Test Case

Lines of Input to the

Program: 5 5

1 2 N

LMLMLMLMM

3 3 E

MMRMMRMRRM

Expected Output:

1 3 N

5 1 E

Your Solution

�? Think about what features you can add to turn this into a proper "mini-application" rather than a simple input-output to a function.

�? Feel free to implement an approach that you feel comfortable with to receive input into your program e.g. feeding input values into unit tests; input via a console application; supplying input via a file etc.

�? We would like you to apply Test-Driven Development (TDD) to test-drive

your solution.

🔍 We would like to see production-quality code, this means you have thought carefully about your code design and that your code is clean and well-tested.

🔍 We will be assessing the quality of your codebase:

- Is your code readable?
- Have you split your code into a sensible/neat folder/file structure? ● **Separation of concerns** - is your UI code entwined with your Mars Rover logic? If so, how could you separate them out?

🔍 We'd love to see good unit test coverage and all unit tests passing.

🔍 **Top Tips**

- Sketch / plan out your ideas first.
- Think about which features you must include, and which you'd like to include if you have time.
- Imagine you're working on a team of developers on a growing project - what would your colleagues expect to see from you as you design and implement this codebase?
- Commit into your Github repository frequently and with descriptive commit messages.
- Aim for production-quality code: well-designed, easy to extend, readable, and well-tested.
- Write a descriptive README to document the key features of your solution, your assumptions, approaches and future thoughts. Look into the use of Markdown to write a professional-looking README.
- Note down future thoughts / considerations:

○ You can assume that the Plateau is rectangular, but be sure to have a think about how easily your program can be extended upon in the future to support a different shaped Plateau.

○ How might your Plateau support other vehicles and not just Rovers? ● Have fun with it! It’s not every day you get to put a Rover on Mars, get creative and enjoy!

 Once you've finished the task, if you want to extend your solution with a visual interface, a programmable Rover, obstacles, aliens, go for it!

Please submit a github link to your completed solution to the assignment by the deadline

This Mars Rover brief was inspired by

Mars Rover Kata.