**Mars Rovers Development Overview**

MARS ROVERS

A squad of robotic rovers are to be landed by NASA on a plateau on Mars. This plateau, which is curiously rectangular, must be navigated by the rovers so that their on-board cameras can get a complete view of the surrounding terrain to send back to Earth.

A rover's position and location is represented by a combination of x and y co-ordinates and a letter

representing one of the four cardinal compass points. The plateau is divided up into a grid to simplify

navigation. An example position might be 0, 0, N, which means the rover is in the bottom left corner and

facing North.

In order to control a rover, NASA sends a simple string of letters. The possible letters are 'L', 'R' and 'M'.

'L' and 'R' makes the rover spin 90 degrees left or right respectively, without moving from its current spot. 'M' means move forward one grid point, and maintain the same heading.

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

INPUT:

The first line of input is the upper-right coordinates of the plateau, the lower-left coordinates are

assumed to be 0,0. The rest of the input is information pertaining to the rovers that have been deployed. Each rover has two lines of input. The first line gives the rover's position, and the second line is a series of instructions telling the rover how to explore the plateau.

The position is made up of two integers and a letter separated by spaces, corresponding to the x and y

co-ordinates and the rover's orientation. Each rover will be finished sequentially, which means that the second rover won't start to move until the first one has finished moving.

OUTPUT

The output for each rover should be its final co-ordinates and heading.

INPUT AND OUTPUT

Test Input:

5 5

1 2 N

LMLMLMLMM

3 3 E

MMRMMRMRRM

Expected Output:

1 3 N

5 1 E

**Features of MarsRover Application:**

* It gets initialized with the input plateau size.
* After initialization of the plateau, the application initializes the given rovers.
* Initialization is done for valid sets of inputs.
* Then the rovers are moved according to the input commands for each rover.
* Any failures/errors are reported to the console as well as to the log file.

**Design:**

* Simple Object oriented solution with the following objects
  + Plateau
    - With max coordinates
  + Rover
    - With deployment coordinates
    - Heading
    - Command sequence
* The service methods to operate on the above objects to deploy and move the rovers in the given plateau.
* Avoided use of open source utility libraries to prevent the code from become vulnerable to

**Assumptions:**

* For initializing the system, input is provided as a txt file. (input.txt) placed within the application root folder.

**Future enhancements:**

* More tests could be added for more code coverage and covering as many scenarios as possible.
* Data driven tests could be added to test various rover movements and failures as well.
* Avoid reading input from local storage and instead read from trusted/hosted source.