MARS ROVER API---MRA

Develop an API that moves a rover around a planet. The planet is represented as a grid with *x* and *y* coordinates. The rover has also a direction that it is facing. The direction can be either north (*N*), south (*S*), west (*W*) or east (*E*). The input received by the rover is a string (i.e., an array of character), representing the command -- simple or in series (see the user stories) --, it needs to execute.

Instructions:

- **KEEP TRACK** of the time (expressed in **MINUTES**) it takes to fulfill the assignment. Be honest, you will not be evaluated based on the completion time---in other words, it does not matter if you complete the assignment in 2 hours, rather than in 4 hours, and so on.
- **DEVELOP** the MRA program by following TDD and remember to **APPLY** your design knowledge (e.g., refactor your code when needed). You are allowed to use the IDE you prefer.
- Once the assignment is completed (i.e., you are confident that all the stories are correctly implemented), **FILL OUT** the post-mortem survey available at https://forms.gle/L4C3vngyMqgMwtuB7, and then **SEND ME** your implementation of MRA (e.g., the Eclipse project). The **DEADLINE** to deliver your implementation **May, the 27th at 11:59 (a.m.)**.

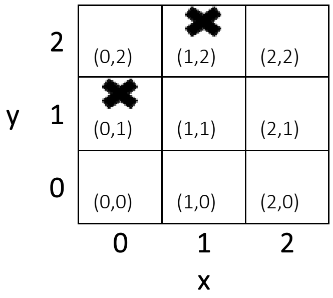
Constraints:

- You **MUST** follow **TDD** to develop MRA.
- The MRA program **MUST** be developed in **Java**.
- You **CANNOT** use Eclipse plug-ins like EclEmma or Pit (or similar tools) when carrying out this assignment.

User Stories

1. The planet

The planet where the rover moves is represented as a square grid, with *x* and *y* coordinates indicating each cell. Obstacles might be present on the planet. For example, the following figure shows a planet of 3x3 size with two obstacles in (0,1) and (1,2), respectively.



	2	(0,2)	✖ (1,2)	(2,2)
y	1	✖ (0,1)	(1,1)	(2,1)
	0	(0,0)	(1,0)	(2,0)
		0	1	2
			x	

Requirement: Define a planet with x and y coordinates (where x=y). Define obstacles on the planet using the same coordinate system.
Example: A planet has size 100x100. The cell at (45,67) is an obstacle.

2. Landing

The rover lands on the planet. It starts its journey at the start of the grid -- i.e., (0,0) -- and faces north.
Requirement: Define a planet as a square grid. When the rover lands on the planet its status is (0,0) facing north.
Example: An empty command (i.e., an empty string) to the rover results in returning his landing status, i.e., the string *"(0,0,N)"*.

3. Turning

The rover turns right or left. It remains on the same grid's cell. Its orientation changes accordingly.
Requirement: Compute the position of the rover after turning left *l* or right *r*.
Example: A rover in position (0,0,N) returns the string *"(0,0,E)"* -- i.e., its new position – after executing the command *"r"*. On the other hand, a rover in position (0,0,N) returns the string *"(0,0,W)"* after executing the command *"l"*.

4. Moving forward

The rover moves forward receiving an *"f"* command.
Requirement: Compute the position of the rover after moving forward of one cell forward on its direction on the grid.
Example: A rover in position (7,6,N) arrives at (7,7,N) -- i.e., it returns the string *"(7,7,N)"* -- after executing an *"f"* command.

5. Moving backward

The rover moves backward after receiving a *"b"* command.
Requirement: Compute the position of the rover after moving it one cell backward on its direction on the grid.
Example: A rover in position (5,8,E) arrives at (4,8,E) -- i.e., it returns the string *"(4,8,E)"* -- after executing a *"b"* command.

6. Moving combined

The rover is able to execute a sequence of command *"f", "b", "l", "r"*.
Requirement: Compute the position of the rover after executing a command in series.
Example: A rover in position (0,0,N) arrives in position (2,2,E) -- i.e., it returns the string *"(2,2,E)"* -- after executing the command *"ffrff"*.

7. Wrapping

Since the planet is a sphere the rover wraps at the opposite edge once it moves over it.
Requirement: Compute the position of the rover moving beyond the edges. The rover should spawn on the opposite side.
Example: A rover on a planet of 100x100 size, moves backward after landing (position (0,0,N)) and arrives in position (0,99,N) -- i.e., it returns the string *"(0,99,N)"*.

8. Single obstacle

The rover might encounter an obstacle. When this happens, the rover cannot pass through the obstacle, thus it reports the encountered obstacle and continues executing the remaining commands. If the same obstacle is encountered twice or more, it is reported only once.
Requirement: Compute the position of a rover encountering an obstacle, and report the obstacle.
Example: A rover is just landed (position (0,0,N)). There is one obstacle on coordinates (2,2) of the planet. The rover executes *"ffrfff"* and returns the string *"(1,2,E)(2,2)"*. *Notice that the same obstacle is encountered twice but reported only once*.

9. Multiple obstacles

The rover might encounter multiple obstacles. When this happens, it reports all of them once, and in the order they have been encountered.
Requirement: Compute the position of the rover encountering obstacles, and report the obstacles encountered in the correct order.
Example: A rover is just landed (position (0,0,N)). There are two obstacles on coordinates (2,2) and (2,1) of the planet. The rover executes *"ffrfffrflf"* and returns the string *"(1,1,E)(2,2)(2,1)"*. *Notice that the first obstacle is encountered twice but reported only once*.

10. Wrapping and obstacles

The rover encounters an obstacle when trying to wrap around the opposite edge of the planet.

Requirement: The rover tries to wrap at the opposite edge of the planet when moving beyond it, but there is an obstacle which prevents it from wrapping.

Example: On a planet of 10x10 size, there is one obstacle on (0,9). The rover is just landed (position (0,0,N)) and tries to execute *"b"*. The rover returns the string *"(0,0,N)(0,9)"*.

10. A tour around the planet

The rover goes on a tour around the planet encountering several obstacles, and warping in both dimensions.

Requirement: Compute the position of a rover that executes a series of commands that result in moving along both axes in both directions, encountering several obstacles and wrapping from both edges of the planet.

Example: The rover lands on a planet of 6x6 size with obstacles on (2,2), (0,5) and (5,0). It executes the command *"ffrfffrbbblllfrfrbbl"* and returns the string *"(0,0,N)(2,2)(0,5)(5,0)"*.

Congratulations, you are done!