What Is the Toy Robot Challenge (or Toy Robot Simulation)?

Description

The application is a simulation of a toy robot moving on a square tabletop, of dimensions 5 units x 5 units. There are no other obstructions on the table surface.

The robot is free to roam around the surface of the table, but must be prevented from falling to destruction. Any movement that would result in the robot falling from the table must be prevented, however further valid movement commands must still be allowed.

Create an application that can read in commands of the following form:

- PLACE X, Y, FACING
- MOVE
- LEFT
- RIGHT
- REPORT

PLACE will put the toy robot on the table in position X, Y and facing NORTH, SOUTH, EAST or WEST.

The origin (0,0) can be considered to be the SOUTH WEST most corner.

The first valid command to the robot is a PLACE command, after that, any sequence of commands may be issued, in any order, including another PLACE command. The application should discard all commands in the sequence until a valid PLACE command has been executed.

MOVE will move the toy robot one unit forward in the direction it is currently facing. LEFT and RIGHT will rotate the robot 90 degrees in the specified direction without changing the position of the robot.

REPORT will announce the X, Y and FACING of the robot. This can be in any form, but standard output is sufficient.

A robot that is not on the table can choose the ignore the MOVE, LEFT, RIGHT and REPORT commands.

Input can be from *a file*, or from standard input, as the developer chooses.

Provide test data to exercise the application.

Delivery

Please complete the exercise in **Ruby**, using practices which represent the quality of code you are proud of.

If you don't know Ruby, then Java, Smalltalk or *another O-O language is acceptable*. If want to show us an implementation in your personal favorite language (e.g. functional) then please include that for bonus marks, but we'll need to ALSO see a solution in an O-O language to proceed to tech interview.

Example Input and Output

```
PLACE 0,0, NORTH
MOVE
REPORT
```

Output: 0,1,NORTH

```
PLACE 0,0, NORTH
LEFT
REPORT
```

Output: 0,0,WEST

```
PLACE 1,2, EAST
MOVE
MOVE
LEFT
MOVE
REPORT
```

Output: 3,3,NORTH