**My Robot Simulator**

Remember to git add && git commit && git push each exercise!

We will execute your function with our test(s), please DO NOT PROVIDE ANY TEST(S) in your file

For each exercise, you will have to create a folder and in this folder, you will have additional files that contain your work. Folder names are provided at the beginning of each exercise under submit directory and specific file names for each exercise are also provided at the beginning of each exercise under submit file(s).

| **My Robot Simulator** |  |
| --- | --- |
| Submit directory | ex00 |
| Submit file | my\_robot\_simulator\* |
| Languages | javascript => .js, python => .py, ruby => .rb, c => .c, ... |

**Description**

Write a robot simulator.

++ Instructions:

The robot factory manufactures robots that have three possible movements:

* turn right
* turn left
* advance

Robots are placed on a hypothetical infinite grid, facing a particular direction (north, east, south, or west) at a set of {x,y} coordinates, e.g., {3,8}.

Robots can pivot left and right.

The factory's test facility needs a program to verify robot movements.

The floor of the room is a grid, each square of which measures 1 square RU (Robot Unit).

The rooms are always oriented so that each wall faces east, south, west, and north.

The test algorithm is to place a robot at a coordinate in the room, facing in a particular direction.

The robot then receives a number of instructions, at which point the testing facility verifies the robot's new position, and in which direction it is pointing.

The robot factory's test facility has a simulator which can take a string of letters and feed this into a robot as instructions.

* The letter-string "RAALAL" means:
  + Turn right
  + Advance twice
  + Turn left
  + Advance once
  + Turn left yet again
* Say a robot starts at {7, 3} facing north. Then running this stream of instructions should leave it at {9, 2} facing west.

++ Your job

Write a function that takes a string of instructions as parameters and return string with the following format:

"{x: X, y: Y, bearing: DIRECTION}"

X and Y with last coordinates of the robot. DIRECTION will be the orientation (north, west, south, east)

Robot always starts in 0, 0 facing north.

Going south will increase Y. Going west will decrease X.

The grid is unlimited, positions can be negative.

**Example 00**

Input: "RAALALL"

Output:

Return Value: "{x: 2, y: -1, bearing: 'south'}"

**Example 01**

Input: "AAAA"

Output:

Return Value: "{x: 0, y: -4, bearing: 'north'}"

**Example 02**

Input: "RAARA"

Output:

Return Value: "{x: 2, y: 1, bearing: 'south'}"

**Example 03**

Input: ""

Output:

Return Value: "{x: 0, y: 0, bearing: 'north'}"