Backend Engineer Assignment

Introduction

You will create a service that navigates an imaginary robotic hoover (much like a Roomba) through an equally imaginary room based on:

- room dimensions as X and Y coordinates, identifying the top right corner of the room rectangle. This room is divided up in a grid based on these dimensions; a room that has dimensions X: 5 and Y: 5 has 5 columns and 5 rows, so 25 possible hoover positions. The bottom left corner is the point of origin for our coordinate system, so as the room contains all coordinates its bottom left corner is defined by X: 0 and Y: 0.
- locations of patches of dirt, also defined by X and Y coordinates identifying the bottom left corner of those grid positions.
- an initial hoover position (X and Y coordinates like patches of dirt)
- driving instructions (as cardinal directions where e.g. N and E mean "go north" and "go east" respectively)

The room will be rectangular, has no obstacles (except the room walls), no doors and all locations in the room will be clean (hoovering has no effect) except for the locations of the patches of dirt presented in the program input.

Placing the hoover on a patch of dirt ("hoovering") removes the patch of dirt so that patch is then clean for the remainder of the program run. The hoover is always on - there is no need to enable it.

Driving into a wall has no effect (the robot skids in place).

Goal

The goal of the service is to take the room dimensions, the locations of the dirt patches, the hoover location and the driving instructions as input and to then output the following:

- The final hoover position (X, Y)
- The number of patches of dirt the robot cleaned up

Your goal is to verify whether the provided implementation behaves according to specification.

Service specification

Input

Program input will be received in a json payload with the format described here.

Example:

```
{
  "roomSize" : [5, 5],
  "coords" : [1, 2],
  "patches" : [
    [1, 0],
    [2, 2],
    [2, 3]
  ],
  "instructions" : "NNESEESWNWW"
}
```

Output

Service output is returned as a json payload.

Example (matching the input above):

```
{
  "coords" : [1, 3],
  "patches" : 1
}
```

Where `coords` are the final coordinates of the hoover and `patches` is the number of cleaned patches.

Deliverable

The service:

- is implemented using Java 11 or later
- must run on Mac OS X or Linux (x86-64)
- can make use of any existing open source libraries that don't directly address the problem statement (use your best judgement).

Send us:

- The full source code, including any code written which is not part of the normal program run (e.g. scripts)
- Clear instructions on how to obtain and run the service and its tests
- Please provide any deliverable and instructions using a public Github (or similar) Repository as several people will need to inspect the solution

Evaluation

The point of the exercise is for us to see:

- How you approach API and system design
- How you structure your code and tests
- How you explain the approach you took and the assumptions you made
- How you deal with uncertainty and contribute to requirements specification

We will especially consider:

- Code organisation
- Code readability
- Quality of instructions
- Quality of tests