PLUM GUIDE

Coding Exercise

Before you begin

Let us walk you through the assessment to better prepare you for it.

- The assessment should take around 2.5 hours to complete - we appreciate your time!
- The assessment is designed to help you showcase your coding skills and work style, and ability to architect solutions.

There are three parts to this assessment.

1. Pluto Rover Coding Exercise - recommended 2 hours
2. Architectural Design - Recommended 30 minutes
3. Final thoughts and feedback survey - 10 minutes

For the coding exercise, you will be assessed based on the following,

- **Project Structure** - Does the code have a clear and concise project structure with defining boundaries?
- **Code Structure** - Does the code adhere to SOLID principles?
- **Maintainability** - Is the solution presented easy to extend for future features?
- **Tests** - Are there a good set of tests for the given scenarios?
- **Command of language** - Display of clean and non-verbose code
- **Documentation** - Clear documentation has been provided (please provide a README.md)

1. Pluto Rover Coding Exercise

After NASA's New Horizon successfully flew past Pluto, they now plan to land a Pluto Rover to further investigate the surface. You are responsible for developing an API that will allow the Rover to move around the planet. As you won't get a chance to fix your code once it is onboard, you are expected to use test driven development.

To simplify navigation, the planet has been divided up into a grid. The rover's position and location is represented by a combination of x and y coordinates and a letter representing one of the four cardinal compass points. An example position might be 0, 0, N, which means the rover is in the bottom left corner and facing North. Assume that the square directly North from (x, y) is (x, y+1).

In order to control a rover, NASA sends a simple string of letters. The only commands you can give the rover are 'F','B','L' and 'R'

- Implement commands that move the rover forward/backward (‘F’,’B’). The rover may only move forward/backward by one grid point, and must maintain the same heading.
- Implement commands that turn the rover left/right (‘L’,’R’). These commands make the rover spin 90 degrees left or right respectively, without moving from its current spot.

Here is an example:

- Let's say that the rover is located at 0,0 facing North on a 100x100 grid.
- Given the command "FFRFF" would put the rover at 2,2 facing East.

Tips!

- Don't worry about the structure of the rover. Let the structure evolve as you add more tests.
- Start simple. For instance you might start with a test that if at 0,0,N with command F, the robot's position should now be 0,1,N.
- Don't start up/use the debugger, use your tests to implement the kata. If you find that you run into issues, use your tests to assert on the inner workings of the rover (as opposed to starting the debugger).

2. Architectural Design

You're now responsible for architecting a rover guidance system, it's going to be used by mission control to log the history and current movements of the rover. The system you design needs to take into consideration the following requirements,

- The data is going to be stored in some form of database. What would you choose and where would you host?
- We also want to introduce a dashboard that allows users to view rovers' live movements as well as historic movements.

Can you outline an architectural diagram of what technologies, patterns you would use for that data to be updated in the data store as well as viewed by a user. Include details of any frontend and backend considerations.

Tips!

- The diagram can be created in a tool of your choice, if you’re struggling for options you can try draw.io
- Don't worry about the cleanliness of the diagram, simple boxes and arrows will do

3. Final thoughts & Feedback Survey

Kindly add your responses to the Feedback Survey.