Programming

New Austrian Coding School

week 4 - reflection

You are allowed to use internet resources but you're not allowed to see other participants' code.
The evaluation has a duration of 1 hour and a total of 2 points.

Exercise 1 (1 point)

Derek's robotic company has managed to secretly steal a prototype robodog from their competitor's lab. They want to reverse engineer it so they can create a better one and beat them in the market. They have been able to access its encrypted instructions file that shows all the instructions that the robodog performed.
Your mission is to decrypt its information and figure out how many unique instructions the robodog can perform and display them.
Follow the instructions below:

- Use the file *robodog.txt* in the GitHub repository. Download it, don't copy paste it from the browser.
- Reuse the *FileReader* class from exercise 9. Don't copy it, just reuse it by importing it.
- You need the following classes: *RobodogApplication*.
- In the *main* method of the *RobodogApplication*, read the *robodog.txt* file, decrypt its lines into instructions, figure out the unique instructions, display how many there are and which ones are there.

Hints:

- ➢ *Every line in robodog.txt is one instruction polluted with many unnecessary exclamation marks.*
- ➢ *Lists, Sets and replaceAll will be very useful.*
- ➢ *Keep it simple stupid. Don't overdo things, just stick to what the exercise asks for.*

```
Number of unique instructions: 5
Commands: [bark, move right, step backwards, move left, step forward]
```

Exercise 2 (1 point)

The design team of Derek's robotic company wants to know where to put more effort, so they need to know how often the robodog performs those instructions and also which one is the most used one.
Your mission is to figure out how many times each instruction is used, and afterwards display the one that is used the most.
Follow the instructions below:

- Reuse the *RobodogApplication* from exercise 1.
- Add this new code in the *main* method of the *RobodogApplication*.

Hints:

- ➢ *Maps and Collections.sort(entries, Comparator.comparing(Map.Entry::getValue)) will be very useful.*
- ➢ *Keep it simple stupid. Don't overdo things, just stick to what the exercise asks for.*

```
Number of unique instructions: 5
Commands: [bark, move right, step backwards, move left, step forward]
Frequency: {step forward=14, move left=5, bark=8, move right=13, step backwards=10}
Most used command: step forward | 14 times
```