Documentation of Lab 1

1. An introduction where the problem is described. For example, what should the program do? Which classes do you have to define? Which methods do you have to implement for these classes?

The program should be able to create a turtle. This one, will have the possibility to draw anything thanks to some instructions already designed by the programmers. To make this possible, we have to define two classes: Instruction and Turtle.

On the one hand, the Instruction class has three different attributes:

- - word: String; name of the instruction.
- - minRange: int; minimum Range of the instruction.
- - maxRange: int; maximum Range of the instruction.

At the same time, it also contains several methods; both constructors and getters:

- Constructors:
  - + Instruction(initWord: String); constructor for instructions that are not within a range. If the introduced String does not have exactly three letters, it will not be saved.
  - + Instruction(initWord String, initMinRange: int, initMaxRange); constructor for instructions that are within a range. If the introduced String does not have exactly three letters, it will not be saved. Additionally, if the minimum range is equal or greater than the maximum range, they will not be saved neither.
- Getters:
  - + getWord(): String; returns the name of the instruction.
  - + getMinRange(): int; returns the minimum range of the instruction.
  - + getMaxRange(): int; returns the maximum range of the instruction.
  - + toString(): String; returns a string representation of the instruction.
  - + isParameterValid(): Boolean; returns whether parameter is in legal range or not.

On the other hand, the Turtle class has various attributes:

- - xPosition: int; current coordinate x of the turtle.
- - yPosition: int; current coordinate y of the turtle.
- - currentAngle: int; current angle of the turtle in degrees.

- - isPenActivated: boolean; whether the pen is currently activated (true) or not (false).

Meanwhile, it holds as well some methods; both constructors, getters and setters.

- Constructors:
  - + Turtle(); constructor to initialize the Turtle and sets all its attributes to 0.
- Getters:
  - + getXPosition(): int; returns the current x coordinate of the turtle.
  - + getYPosition(): int; returns the current y coordinate of the turtle.
  - + getAngle(): int; returns the current angle of the turtle.
  - + getIsPenActivated(): int; returns whether the turtle's pen is activated or not.
  - + toString(): returns a string representation of the current status of the turtle.
- Methods:
  - + moveForward(int distance): void; it moves forward the turtle the distance introduced as a parameter taking into account its angle.
  - + rotate(int degrees): void; rotates the turtle the amount of degrees introduced as a parameter. It always updates its current angle doing a modulus of 360, so it cannot exceed.
- Setters:
  - + setPen(int value): void; sets the pen to activated or not (true or false).

2. A description of possible alternative solutions that were discussed, and a description of the chosen solution and the reason for choosing this solution rather than others. It is also a good idea to mention the related theoretical concepts of object-oriented programming that were applied as part of the solution.

To start with, a possible alternative solution for the pen attribute in the Turtle class could be to initialize the variable isPenActivated as an integer. In our project, we have decided pen attribute to be a boolean so that we clearly can identify in which state the variable is found. Although our pen variable is a boolean, the method setPen receives an integer value which ranges between 0-1. Therefore, either programmers and the same program can validate the ranges of the PEN instruction.

Moreover, we discussed the isParameterValid method name. The reference solution set this name to isLegal, however we have considered the final name given that we believe that method names must describe the method function.

3. A conclusion that describes how well the solution worked in practice, i.e. did the tests show that the classes were correctly implemented? You can also mention any difficulties during the implementation as well as any doubts you might have had.

The TestLogo class is set to implement test cases for both Instruction and Turtle classes.

Firstly, we have initialized a move forward instruction with the name “FWD” which ranges from 1 to 500. In order to check if it works, we use the toString method to print its information. Additionally, we call the isParameterValid function with different values in and out the range to check if it works. On the one hand, with parameter set to 1, which fits into the range, the output is true. On the other hand, with parameter set to 512, clearly out of the bounds, the output is false.

To continue with, we have initialized a Turtle class called turtle, which all its attributes are set to 0. Then, we have changed certain attributes. Now, the toString method should print the attributes of the class changed. Indeed, it works. Moreover, we have defined some distance, and then we have moved the turtle with the pen activated. After the turtle has been moved, we have to check its attributes again. Again, it has worked.

To conclude, we can affirm that our solution to the seminar works in practice given that our tests have succeeded. Generally, we have not been in any difficulties during the implementation of the program.