Jordi Alfonso Poza
Eric Navarro López

Object Oriented Programming

Report Lab 2:

The aim of this lab session was to implement the design of Seminar 2 consisting of a complete Logo application. In order to do so we were asked to add a new method to Program.java called getCurrentInstruction() that returned the instruction on the current line the program was in. In addition,we were also asked to start implementing the different methods of Turtle and Logo classes such as Turtle forward and Turtle turn.
As a matter of fact, it was also required to add the Logo execute method to the logo class which would also have added it´s respective pen operations so that the turtle could go forward without drawing.

With regard to the graphical part,we were also tasked with incorporating the code into a graphical user interface (GUI)(which we were helped with by the Lab guide with a complete guide on how to do so.)
Aside from that, we were also tasked with the implementation of some changes that were needed in the main logo class, such as:
-Adding a Logo variable named logo.
-Initializing it in the LogoWindow constructor.
-Changing the initial window size setting using the width and height getters from Logo class.
-Declaring a Program variable called prog.
-Initializing the program in the LogoWindow constructor.
-Adding the necessary instructions to the program for drawing a square.
-Adding the function paint as specified in the code below.

As opposed to the previous lab, in this one we encountered different problems, It is true it was harder than the last time to fix them since their cause was somewhat unexpected by none of the members of the team.

The first of the inconveniences was that the “execute” function was not working and it caused compiler errors, same as forward and turn functions.
It must be said that we tried many different implementations and even re-wrote then once and again checking why they did not work.

After some thought, we arrived at the conclusion that perhaps the issue was with the functions from the previous lab and that we needed to check those. Later on, with some advice given at the previous lab by our teacher , we were able to deduce that the methods “hasFinished”, “restart”,and “isCorrect” were given an erroneous implementation that was the main cause of the incompatibilities with the new code.

1

Jordi Alfonso Poza
Eric Navarro López

Later on, we were able to find out what was the error beneath the “forward” and “turn” functions, despite our many failed attempts to formulate any unforeseeable cause.
It came to our surprise to discover that the origin of our troubles was just a mere mathematical miscalculation or spelling mistake hidden beneath their code.

On the other hand, the “execute” function was with no doubt the most troublesome asset of this Lab. We tried many different implementations of said function but, nonetheless we were unable to make it work whatsoever. At this point, and facing more and more complicated resolutions we decided not to stick with our exponentially complex implementation and we reached the conclusion that It would be a better idea to just try to redo from scratch the entire Program.java.

At the end, it came to our surprise how simple the issue would have been if we had just strayed from our complex and stubborn approach, since we managed to make the function work by just redesigning the class Program.java.

As a matter of fact, “execute started working as soon as we got some work done with the Program.java.

Last but not least, we encountered some problems with “turn” and “forward” methods.
Yet, those were at the end some minor issues that could be fixed by re-expressing the formulas associated to each of them and making sure that they were reliable regarding what they were supposed to do.

Overall, this has been a more complex lab that has reminded us the importance of questioning our previous work, such as reminding us that sometimes the best way to advance is to go back and redo what is wrong. Aside from it´s poetical meaning it really is a useful principle when dealing with code and with multiple implementations.
We still have a long way to go in order to be able to fully comprehend Java, but we belive that this experience got us one step closer to reaching that goal-

2