Jordi Alfonso Poza
Eric Navarro Lopez

Lab 1 Object oriented programming [Report]

The aim of this document is to engage the implementation process of the multiple structures we required to develop with in regard to the Object Oriented programming first seminar; and describe the different design proposals and respective hardships the team had to endure and overcome.

First thing is first, we were asked to provide a general description of the behaviour of the overall program, taking into account its classes along with the design and structure that constitute them. After the overall development and completion of the code we are able to summarize this aspect by saying that the program´s main goal is to elaborate and accumulate a series of commandments that will be returned in the form of an ordered and well structured list of said instructions.

Yet, this was not so simple; since we were required to develop different strategies in order to succeed with the development of the code for this seminar. We concluded that the best feasible strategy was to implement two main classes. Said main classes would be the so called “Program” along with the “Instruction”.

Class Program:

For the class “Program” we used the following methods:
“**getName**”(to return a string containing the said name),”**goToStartLoop**”(which basically resets the loop),”**getNextInstruction**” (This returns the following to the current instruction the program is on),”**addInstruction**”(which as the name states adds a commandment to the list of instructions),”**hasFinished**”(This checks for the state of the overall instructions and returns accordingly to said state),”**restart**”(quite literally restarts the overall loop of instructions),”**isCorrect**”(for displaying the situation of the instruction), and last but not least: “**printErrors**”(used for displaying the main errors the code may have with regards to the instructions given),

Class Instruction

Addressing the class Instruction the methods applied were the following:
“**getCode**” and “**getParam**” (which work in a pretty similar way by obtaining a specific portion of information from the program in which they specialize**),**”**isRepInstruction**”(Consisting of a boolean that returns true or false given the case that repetition was found among the instructions provided),”**errorCode**”(Which nature consists about notifying the user if there was an unexpected behaviour of any kind regarding the list)**,”isCorrect**”(A boolean to state if the proper execution of the list can be attained) and finally “**info**”(Which consists on returning a string).

Jordi Alfonso Poza
Eric Navarro Lopez

Despite us wanting to attain a pretty straightforward approach to the overall code, we must note that we hit some slight bumps and even roadblocks when it came to the development of the methods. Some of them were minute and solved with ease but some others caused some trouble and even made us get completely stuck in the mud. Being so, the case of the methods belonging to the class instruction.

Originally we were planning on compressing some methods into only one since we thought that this would make the code more efficient. It was not long until we found out that the loop,both the iteration and instruction methods would not be possible to be integrated into the “addInstruction”. Such as that the restarting method was not compatible with the error code.

Obviously, this approach was everything but clean,efficient; or simply feasible.
It was not long until we decided to step back on it and keep a more simplistic approach to our work by defining multiple separated methods for each task. It must be noted that one of the major encouragers of this change was the program blowing up and not compiling despite the best of our efforts and prayers nonetheless…

After many tests we were able to come up with a bit of guidance to the final product of the code, yet it was not always that straightforward and on multiple occasions we would feel a little bit lost and have to halt the trial process of implementations to reconsider our approach.

One significant (if not the most) significant issue we experienced was the one we had with the repetitions. It cannot be stated enough the trouble this part of the code had given to us along all of the project. The core part of the problem was that we were unable to come up with a proper implementation that would not completely destroy the program’s execution when reached. In addition to this it must be said that we felt lost within the time we worked on the issue and we even had to resort to advice in order to comprehend how to develop it.

Overall we believe our approach to be quite the average work given our skills and the current conditions the team has faced for this lab but we are nonetheless proud of our work, such as the efforts done in order to comprehend the nature of Java by ourselves. Perhaps being a step closer to being able to develop propper simple software.