Refactorings is a very useful practice which every programmer should integrate within their code development cycle. Refactoring refers to systematically reorganising code to improve readability and structure of the code. This makes things like finding bugs and adding new functionality much easier.

Nullness Annotations give us a way to analyze code and enable us to identify and flag possible instances of Null Pointer Exceptions during compilation of the code. We used specific compiler tools like the Checker Framework, which is programmed to automatically analyze annotated code. Annotations are an extremely useful tool which allows users to prevent pushing out possibly broken code which compiles correctly but may cause runtime errors for particular instances of the program.

The goal of our research project is to integrate refactoring tools in Eclipse which will allow users to insert Nullness Annotations in their code. To a large extent, this will allow users to automatically insert annotations without having to explicitly understand how the Checker Framework analyses the annotations. To develop such a tool, we first need to explore different uses of annotations in different contexts, understand where and why annotations are required, and how the Checker Framework uses these annotations to flag cases of
possible exceptions. Towards this end, we have spent quite a bit of time inserting annotations within code and analysing annotated code, looking through warnings that the Checker Framework threw at us to see why exactly were those warnings generated. Once we had grasped the basics of how the annotations worked, we employed our skills in annotating open source Java libraries like the Apache Commons library. Currently, we are working on comparing the annotations inserted by NIT and Julia, two open source programs which analyse code and automatically generate annotations to be inserted within the code. This allows us to learn how existing automated inference tools work and how they analyse code to insert annotations. Also, creating a detailed report of our comparisons is a good way to contribute towards research conducted in this field. This allows other researchers and developers to fix bugs in existing code and improve the tools available to make automated insertion of annotations more accurate.

Besides Nullness annotations, the Checker Framework supports other annotations which are used to verify correctness of other aspects of code. Over the next month, we aim to expand our research by looking at other annotations, and understand how they work. I am very interested in continuing this research, and plan to take it up as independent study in the next semester. Our next step will be to create a few refactoring tools within eclipse which will allow users to automatically insert annotations in their code.