In this project, I acquired several insight concepts pertaining to refactoring and detecting design smells in a Java source code. I utilized the Junit test to conduct test on the Monopoly source code and assessed the results to understand the purpose of conducting Junit tests. I learned to alleviate bad design smell by renaming class fields. I learned how to view and change a class hierarchy in a given code. The hierarchy view allowed me to move methods either down or up on the hierarchical structure of the program.

I also learned the purpose of interfacing classes of methods. This refactoring technique allow testers to encapsulate a method with another method to prevent design flaws in the original source code. I also learned how to extract methods from methods contain too much functionalities. The aforementioned factor helps scale the source code into a more efficient and faster program by minimizing the implemented functionality.

I liked the variety of refactoring methods the Eclipse IDE provides to ensure developers are running reliable code. These assurance methods limit the amount of design flaws in a program. It may also increase the security of the application under development. JDeodorant seemed like a very easy plug-in to understand. The plug-in provided visual representation for the implement. It provided an effective refactoring method for users to understand.

Unit tests are essential for the refactoring process because it ensure that the refactoring session implemented feasible changes. Unit tests examines how the modification generated by refactoring affect the entire program and provides feedback on whether the implemented change generates an error. It also ensures that all changes are logical and requires the developer to intricately examine the source code.