Various tools were used for the task of updating legacy Fortran 77 code to Java. Eclipse’s IDE was used to write and test the Java code. GitHub was used as the source control tool which tracked all changes to the project. Using an Eclipse plug-in known as EGit was attempted in the initial stages of this project. The EGit plug-in would have allowed syncing of the GitHub project directly from Eclipse. However, due to issues faced with EGit, the plug-in was not used. The GitHub desktop was then used to sync changes with the remote repository. An Eclipse plug-in known as JAutoDocs was used to automatically generate comments for the classes and functions in the project. These comments were then updated to add additional information about each class and function. The JavaDocs tool was used to automatically generate web pages based on the comments around each class and function.

The main purpose of the project was to understand the workings of the Fire Danger System using the documentation and Fortran 77 code provided, and upgrade the system by creating it using Java instead of Fortran 77. The documentation provided included a brief description of the system, an explanation and formulas for deriving various values used by the system, a flow chart to map to the overall flow of the system and a copy of the Fortran 77 code. Much of the Fortran 77 code could be read by someone who did not know Fortran 77 but knew other programming languages. However, certain syntaxes of the Fortran 77 code were sufficiently different and warranted further research. In order to successfully port the code over, I had to learn the syntax of the if and the do loops in Fortran.

While writing code in Java, if was extremely helpful to have a set of coding standards to follow. For example, all functions dealing with user input were a part of their own class, while all functions dealing with calculations were in their own class. This approach not only allows for an increase in reuse of the existing code, it also allows for better documentation since each function performs one and only one action. JAutoDocs was used to automatically generate comments for each class and function. These comments were then updated with additional information regarding the classes and functions. For a class, information regarding its general use as well the author was added. For functions, additional information regarding its functionality, input parameters and output was added. In addition to this, comments were added throughout the code to reflect the flow chart and formulas provided as a part of the documentation.

The process of updating the code to Java was not without its limitations; I faced many issues. One of the major issues I faced was with learning how to use GitHub correctly (despite having taken a course on Code Academy). I initially tried to use EGit, but was unable to add a new project into a remote repository using EGit! As an alternative solution, I attempted to use the GitHub desktop. The initial repository created and synced using the GitHub desktop was corrupted; I could not open my project on another machine by simply downloading it. This may have been due to presence of conflicting EGit configuration in the project. After recreating the project from scratch and then adding it via the GitHub desktop, everything seemed to work fine. There were also some issues with the documentation provided regarding the Fire Danger System. Some of the formulas provided were incorrectly formatted and thus hard to understand. I resorted to using the formulas in Fortran code for Fine Fuel Moisture and Build Up Index. The documentation also missed some important details. For example, the documentation did not list that grass and timber indexes should not be greater than 99. It also did not have a description of how the drying factor was determined. Finally, there were also some conflicts between the code and the documentation. The documentation stated that indexes should equal 1 when FFM and ADFM were greater than 33%. However, the limit implemented in the code was 30% and not 33%.