Team Member: Khai Phan, Mamadou Diallo, Tyrell Unser

**A) NAnt**

1. **Which file(s) did you have to examine when looking for command line switches?**

* <https://github.com/nant/nant/blob/master/src/NAnt.Core/CommandLineOptions.cs>

1. **What are your first reactions to these files when you examine them?**

* The file is written in C# and it is well documented. In each part of the Command Line Switch, there is a ShortName parameter which I believe it will be the command line switch when running on terminal. There is also a regular name along, which I believe, when taking account into command line switch, it will be –Name or –ShortName (--debug or –d)

1. **How is the code regarding command line switches organized at the method, class, and project level?**

* All the command line switch are stored under one class, Command Line Switch and it is storing under namespace of NAnt.Core. I believe doing this, the author can just do using namespace NAnt. Core in every file that contain working data for of Argument options and they can be able to work with all command line at the same time (Hope you understand, my reword is bad). Instead of breaking down each Command Line Switch into files that deal with each of them, doing this will save the space and easier for developer to understand.

1. **How are invalid filenames dealt with?**

* In each of the Command Line Switch, there is a default bool with value of false, which mean that whenever an invalid, wrong path, etc., parse in, the function is set to false, and nothing will be doing

1. **Describe two similarities in the code between your two chosen projects?**

* Between NAnt and Ant, they both have the Command Line Switch declared in one file and they are organized well and easy to read in developer perspective.

1. **Describe two differences in the code between your two chosen projects?**

* NAnt is more about .NET, C and C# compatible while Ant is more about just Java.
* NAnt is more about get and set data when a certain command is executed (default is false is data parsed is invalid/incorrect or the command line is not correct) and leave the program coding to another file to handle, meanwhile in Ant, most of the coding part are done within the Main, which is easy to keep track and understand each Command Line Switch is doing right on the file

**B) Apache (Ant)**

1. **Which file(s) did you have to examine when looking for command line switches?**

* https://github.com/apache/ant/blob/master/src/main/org/apache/tools/ant/Main.java

1. **What are your first reactions to these files when you examine them?**

* As this is written in Java, generally command line switches are found in the Main.java file. This file was easy to retrieve going into the “SRC” folder and alternatively could be found using the GitHub search feature. The initial reaction was how well the code was documented; every method and variable were easy to understand as the comments were properly explained.

1. **How is the code regarding command line switches organized at the method, class, and project level?**

* The code is organized in a manner that every command line switch is contained in the “Main” class. When Ant is being built, it calls a method called “start” which reads the incoming argument, properties and class loader. When initialized another method called “processArgs” & reads the argument then appropriately responds to that argument using multiple if statements that will execute different behavior based on that argument.

1. **How are invalid filenames dealt with?**

* When an invalid filename is found, a message returns “Unknown argument : \*Argument name\*”, Following that message a list of all the correct arguments is displayed.