**Instructions to run the java program: FileWordFrequency**

**Assumptions:**

1. Base OS (Windows, MaxOSX, Linux) already installed in the workstation
2. User has basic understanding to run commands in Terminal/command Prompt

**Steps:**

1.Install Java.

Download JDK (Latest preferred) from Oracle website: https://www.oracle.com/technetwork/java/javase/downloads/index.html

Accept the Oracle Binary Code License Agreement for Java SE.

Choose the correct OS depending on ur workstation and install the JDK.

2. Copy the Java source code to your machine in a specific directory.

3. Compile the program:

For this, open command prompt (cmd) on Windows, if you are Mac OS then open Terminal.

To compile the program, navigate to the directory where **FileWordFrequency.java** resides and type the following command and hit enter.

**javac FileWordFrequency.java**

You may get this error when you try to compile the program: “javac’ is not recognized as an internal or external command, operable program or batch file“. This error occurs when the java path is not set in your system

If you get this error then you first need to set the path before compilation.

**Set Path in Windows**:

Open command prompt (cmd), go to the place where you have installed java on your system and locate the bin directory, copy the complete path and write it in the command like this.

set path=C:\Program Files\Java\jdk1.8.0\_121\bin

Note: Your jdk version may be different. Since I have java version 1.8.0\_121 installed on my system, I mentioned the same while setting up the path.

**Set Path in Mac OS X**

Open Terminal, type the following command and hit return.

export JAVA\_HOME=/Library/Java/Home

Type the following command on terminal to confirm the path.

echo $JAVA\_HOME

That’s it.

4. After compilation the .java file gets translated into the .class file(byte code). To run the program, type the following command and hit enter:

**java FileWordFrequency**

**About the program:**

**To** handle 1000’s of lines of relative files and counting words, multi-threading has been used. The maximum number of threads in the pool is 30 (rough assumption). Each worker thread will be maintained through an executor service and process one file at a time for counting words.

I have used two hashMaps: inner Hashmap to store word and count in <key,value> pair and outer HashMap to store filename as the Key and inner HashMap as value so that the it can take advantage of already computed word counts in each file. The program has been implemented to ensure interaction between the user and the program itself so that a. the user gets a FileChooser dialog to select the input file and also have the option to select a word and see which files that word appears and also the invidual count and total count in all files.