Reflection Assignment #6

Testing Approach:

We first created a TreeMap of type String key and Integer value associated.
Our getWordsFromFile method takes a file from the user and makes a Scanner object from it. the method has a try catch block as to throw an exception in case there ever was an error with the file.

– The first loop/check is to see whether the Scanner has a next element. We then make a String from the next word and replace all the instances of the listed characters with a blank space. We use the toLowerCase method and trim to have a word that is a upper case version of a word out the same towards that key's value as the lower case version of that word.

– The second loop/check is to see if the our wordMap contains the key of the specific word. This allows for an increase of the value associated with the key. If the key exists n times, then the value of the respective key is n, signifying n occurrences.

– To account for the stop words from the stop words file, me created a file object based on the stop word file. We then created a scanner to scan the stop word file. The next loop is to see if the stop word document has a next element and while it does remove it from the wordMap.

Our start (displayWordCloud) method is within a try catch block as to throw an error when the input file does not exist.

– This method first takes in a file from the user and creates a file associated with the new input file. The next check is to see if the file exists in the first place, and if it does not then ask again for the file name. After all necessary checks, the getWordsFromFile method is used for the file f.

– Next comes the majority of the javafx work.

– the most important check that came from this section is to see if the wordMap was empty, and if it is not then continue with adding elements to the GridPane from the end of the wordMap.

– We then created the appropriate step as to have the word with the greatest value appear the largest.

– The next check/ if, else statement was to track the gridline row and column coordinates.

We do believe that in the end the appropriate checks were made as to run the program effectively and efficiently.

Reflection:

– Initially we had problems coming up with a solution as to insert the elements of the wordMap to the pane and at the same time have the words look aesthetically pleasing.

– The next problem was with finding the location of where to add the catch block as to throw an exception when necessary.

- The assignment says to have a method called display words that creates the word cloud. We couldn't figure out how to do that outside of a direct start method, i.e. the start method does what the displaywords method was supposed to do. JavaFX seems to force that to be the place where you launch the scene from. Additionally we couldn't figure out how to run the static methods from main. Creating a display words method with everything in it doesn't work with this current format because the stage and pane would be inaccessible.
- The map of the words is empty after the program runs, it wasn't specified whether or not it should have remained usable after running.