**HW2**

**ITEC 2150**

**Fall 2019**

Due Date: Sunday, October 13, 2019 11:30 PM- Questions to instructor should be emailed no later than 4 PM on Friday before. You must upload all classes and text files to D2L

In this program we are going to read and write to text files. You are provided with a text file that is a text file of a novel which is no longer under copyright and obtained from the Guttenberg project. The program will prompt for the name of the text file to read. Exception handling should be used to force the user to reenter the file name until the file is found in the default file location. In other words, the program should work on any text file not just the one provided.

The default file location for Eclipse is the project directory which is the parent directory for the src directory. If using Dr. Java, the default file location is the same directory as the Java and class files.

We are interested in finding out information about:

1. How many words are in this text file?
2. How many instances of the word “the” existing in this text file? (regardless of upper/lower case)
3. How many instances of the word “a” or “an” exist in this text file? (one statistic counts both of these words)

These statistics will then be written to a different text file named “stats.txt” in the following format (xxxx is used to indicate something the program must calculate and print or get from program) Except for the xxxx your program should print exactly what is shown:

The file name is: xxxxx.txt

The total number of words is: xxxx

The total number of “the” is : xxxx

The total number of “a” or “an” is : xxxxx

How to read only words from a text file.

In order for this program to work, a regular expression must be used to filter out punctuation and blank spaces as we are only interested in reading and counting words. The following code shows how to set that regular expression and read only words.

Scanner in = new Scanner(new File(filename));

// Use any characters other than a-z or A-Z as delimiters

in.useDelimiter("[^a-zA-Z]+");

while (in.hasNext())

{

String aWord = in.next();

// convert to lower case

aWord = aWord.toLowerCase();

// process the word

}

Grading Criteria (total 100 points):

|  |  |
| --- | --- |
| Follows coding standards | 10 |
| Uses appropriate exception handling | 10 |
| Properly inputs all data from file | 20 |
| Properly stores requested data in program | 10 |
| Properly collects statistics on desired words | 15 |
| Correctly outputs statistics to a second file | 20 |
| Outputs are neat and easily read | 15 |