CSD2221 Individual Programming Assignment 2016

Deadline for submission 17:00, Friday, 9th December, 2016

General information

You are required to submit your work via the dedicated Unihub assignment link in the 'week 10' folder by the specified deadline. Submission should comprise a single 'ZIP' file. This file should contain a separate, cleaned, NetBeans project for each of the tasks described below. The work will be compiled and run in a Windows environment, so it is strongly advised that you test your work in the University labs prior to cleaning and submission.

In addition to the NetBeans projects, you are required to include a single, MS Word 'solution' document into which you should copy and paste the output from your solutions to the tasks as described below. Name the Word document using your student number.

For this assignment, four text files are provided as test files. These must not be altered in any way. The markers will use their own copies of these test files when checking your work.

Task 1 (15 marks)

Inspect the test file called *sonnet1-a.txt*. This file contains one of Shakespeare's famous sonnets (a type of poem with specifically 14 lines), but it has been simplified because there is no punctuation. The edited sonnet only contains words using lower and uppercase letters.

Write a Java program (as a NetBeans project) that gathers some basic statistics about the words in this file. In particular, you are required to write a Java program that opens up the sonnet file and calculates four properties about the sonnet:-

- The total number of words in the sonnet
- The total number of letters in the sonnet
- The total number of words with an even number of letters
- The total number of words with an odd number of letters

(a)

On building and running, your program should have the following output:-

```
File analysed: sonnet1-a.txt

There were W words and X letters

There were Y even words, and Z odd words
```

Where the values W, X, Y and Z are replaced with the values that your program has calculated

Having calculated these statistics your program should close the file properly.

Your program should handle exceptions robustly.

For part (a) described above, your statistical count should include all occurrences of any given word and its letters. So, for example, if the word 'And' occurs several times, then it should be counted that many times in the overall word count (and similarly with regard to the letters).

Copy and paste this console output to your solution document.

(b)

The program should also store the sets of odd words and even words, where each word is translated into an upper-case equivalent i.e., ‘thee’ becomes ‘THEE’ and ‘Thou’ becomes ‘THOU’.

However, for these two sets of words, each unique word should only be stored once. For example, even if the word ‘And’ occurs several times, it should only be stored once in the set of odd words.

The program should print both sets of unique words (the odd set and the even set) to the NetBeans console ***in alphabetical order.***

Copy and paste the two sets of odd and even words to your solution document.

(c)

In addition to the console output, your program should also save all of the output to parts (a) and (b) above to a text file in the NetBeans project root folder. Call the file *results.txt*.

Task 2   (10 marks)

Inspect the test file called *sonnet1-b.txt*. This is the same Shakespeare sonnet, only this time the sonnet contains the correct punctuation.

Write an extended version of the program you designed for Task 1 above, but this time the statistics should be gathered taking into account correct punctuation. In addition to white space, words should be delimited using punctuation marks with the exception of the apostrophe character. That is to say, for the sake of this exercise, words such as *dimm’d* and *summer’s* should be treated as a single word.

Also note that when an apostrophe does appear in a word, the apostrophe should not be included in the letter count. For example, the word *dimm’d* should be counted as a single word with 5 letters. Your program will need to account for this.

Apart from this extension, the program should operate in the same way as that for Task 1, parts a, b, and c above.

As before, all console outputs should be copied and pasted to your solution document, and saved to the *results.txt* file.

Additional task

Test your extended program with the test file *sonnet2.txt*, which contains another Shakespeare sonnet.

Note that the phrase *world-without-end* should be treated as three separate words.

All outputs for this second test should also be copied and pasted to your solution document.

Task 3 (15 marks)

Write another version of the statistics gathering Java program (using a different NetBeans project), but this time two further pieces of statistical information are required:-

- How many times each unique word appears in the sonnet (ignoring case)
- A list of all unique words that appear four or more times

You are required to test your solution to this task using the test file, ‘sonnet3.txt’.

This program should account for punctuation in the same way as Task 2 above.

The output to this program should start similar to above i.e.,

```
File analysed: sonnet3.txt

There were W words and Y letters

There were Y even words, and Z odd words
```

This should be followed by a mapping of unique words to their frequency as follows:-

```
WORD1 [FREQUENCY]

WORD2 [FREQUENCY]

Etc.
```

This should be then followed by the list of most frequent words (i.e., those that occur at least 4 times)

Copy and paste the output of this program to your solution document