Mr. Kramer – CS 1030 – Python Project #4

Before starting to code your solutions, write pseudocode and/or draw flowcharts. Do not start coding until you have a deeper understanding of what the solutions should look like. Write pseudocode from which you can directly code your program. I am deliberately not giving step-by-step specifications as it is good practice for you to do that on your own. However, feel free to contact me for help or clarifications. See “What and how to submit” near the end of this document for file naming conventions.

**1. Scrabbletm – File, Loop and Dictionary Practice**

Write a program that reads a list of words and calculates the Scrabbletm point value of each word. If the word has 0 characters or 10 characters or more, assign a point value of 0. The words are read one per line from file

1030 Project 04 01 Words.txt

which is included with this specification. Convert lower case letters to upper case with a statement like

letter = character.upper()

and ignore all characters that aren’t letters. You can test if a character is a letter with

if letter.isalpha()

Point values are based on this table which can be conveniently implemented with a dictionary:

|  |  |
| --- | --- |
| # points | Letters |
| 1 | A, E, I, L, N, O, R, S, T, U |
| 2 | D, G |
| 3 | B, C, M, P |
| 4 | F, H, V, W, Y |
| 5 | K |
| 8 | J, X |
| 10 | Q, Z |

An actual Scrabbletm board includes some squares that multiply the value of a letter or the value of an entire word. For this problem, ignore these squares. The output should be in a list with the word and its points, with the point total displayed at the end of the program. For example:

Word Points

CAT 3

DOG 5

FOX 13

HIPPO 12

PLATYPUS 15

Total 48

This problem is derived from *The Python Workbook*, by Ben Stephenson, page 66, exercise 137.

**2. Letter Frequencies – File, Loop and Dictionary Practice**

Write a program that reads sentences from file

1030 Project 04 02 Sentences.txt

(included with this specification) and calculates letter frequencies (total counts of how often each letter appears) for all sentences in the file. Print the original lines as you read them. Then, convert lower case letters to upper case and ignore all characters that aren’t letters. Frequency analysis is one approach to decrypting messages. The letters E and T are the most common letters in the English language, so a frequency analysis of a long message would likely show high counts for letters E and T.

After processing all lines in the input file, display on the screen and write to output file

FirstnameLastname (Your section # here) 04 02 Output.txt

the letters and their frequencies. Here’s an example. Given lines

See Spot.

See Spot code in Python.

The screen display and output file would look something like:

Letter Frequency

A 0

B 0

…. ….

E 5

…. ….

O 4

P 1

…. ….

S 4

etc. etc.

This problem is derived from *The Python Workbook*, by Ben Stephenson, page 71 exercise 146.

**3. Concatenate Multiple Files – File and List Practice**

Write a program that concatenates (combines) one or more files, writing them to an output file:

1. Reads input file 1030 Project 04 03 Files.txt which itself is a list of files

2. Read each of the files, writing them to output file

FirstnameLastname (Your section # here) 04 03 Output.txt

For example, say input file 1030 Project 04 03 Files.txt has these three lines:

1030 Project 04 03 File 1.txt

1030 Project 04 03 File 2.txt

1030 Project 04 03 File 3.txt

The first file has these lines:

Line 1

Line 2

The second one has this line:  
Line 3

The third file has these lines:

Line 4

Line 5

Line 6

The output file, FirstnameLastname (Your section # here) 04 03 Output.txt, would look like this:

Line 1

Line 2

Line 3

Line 4

Line 5

Line 6

This problem is derived from *The Python Workbook*, by Ben Stephenson, page 70, exercise 143.

**What and how to submit**

You have now created these files:

FirstnameLastname (Your section # here) 04 01.py

FirstnameLastname (Your section # here) 04 02.py

FirstnameLastname (Your section # here) 04 02 Output.txt

FirstnameLastname (Your section # here) 04 03.py

FirstnameLastname (Your section # here) 04 03 Output.txt

**Note you do not submit the input files.** If your IDE supports it, turn on line numbering before printing. Submit your programs and accompanying files together in the order shown above. You will lose points if you don’t follow this submission guide.

**How your programs are evaluated**

1. Do they work according to the specifications?
2. Are the input files and prompts correct?
3. Are the programs documented?
4. Do the programs follow Python and file naming standards?