Summary

We are creating a program that will transform the way books are published. A high-powered professional grammar program is the goal. It will revolutionize the way people read. The product will be user friendly. The goal of the program is to analyze individual books or a set of documents.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

The program will use the results of ***matt\_wc.py*** and ***matt\_wc2.py.***

The program ***matt\_wc.py*** calculates unique words sorted by frequency.

This will be called the **Vocabulary List**.

The program ***matt\_wc2.py*** calculates phrases that appear more than once.

This will be called the **Phrase List**.

The Vocabulary List and the Phrase List will be input for the next phase of the program. \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Next Steps**:

The program will analyze *letter frequencies* for the entire document:

{‘a’: 75003, ‘b’: 16286, ‘c’: 20355, ‘d’: 34206, ‘e’: 110455, ‘f’: 16183, ‘g’: 22554, ‘h’: 47539, ‘i’: 63197, ‘j’: 2136, ‘k’: 12845, ‘l’: 40522, ‘m’: 28624, ‘n’: 59483, ‘o’: 83633, ‘p’: 14940, ‘q’: 724, ‘r’: 48730, ‘s’: 55743, ‘t’: 90398, ‘u’: 27667, ‘v’: 8373, ‘w’: 23594, ‘x’: 1626, ‘y’: 21810, ‘z’: 557}

This will be known as the **Letter Frequency Chart**.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Top 20 List** is the vocabulary list created in this program not to be confused with the Vocabulary List used as input. The Top 20 List is determined on a case-by case basis. That is for each phrase.

A ***Bank*** is created with the Letter Frequency Chart.

A ***Pool*** will be created with each phrase on the Phrase List.

This will create so many ***Pools***.

The program will determine the top 20 unique words for each phrase analyzed sorted by frequency ( Most frequent to least frequent.)

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Methods**

Import data from matt\_wc.py and matt\_wc2.py.

Calculate the letter frequencies in Typoglycemia.docx.

Given the letter frequencies you count how many times you may make a phrase given the Bank of Letters.

This will create a Pool.

There will be many Pools as there are many phrases.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

With each Pool you will analyze the vocabulary. The number of times you may make a word will determine the vocabulary list for each pool. The Pool will reset for each word checked.

You may do this by creating a list of letter counts for each word. Then you just multiply the letter counts until it fills the Pool. This will determine the frequency of the word in the Pool. Reset the Pool value after you check each word.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

After the Pools’ vocabulary is determined the top 20 phrases will be extracted. This will be done for each Pool. The whole list will be consolidated. The word frequency for each Pool will be combined and added together if there is a duplicate word. The list should not exceed 500 pages. If the list is longer I will adjust the requirements of the program, perhaps selecting only the top 10 words for each Pool.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Output**

A list of vocabulary will be created. The list will be sorted by frequency (Most frequent to least frequent.)

**Things I need to do**

Try to find a way to incorporate all the data collected and add to the program. Help develop a method to help run the program. Provide specifications which are manageable. For example we would not want the program to run too long. Create a Typoglycemic version of the file. Review files such as the output for the Word Frequency and Phrase Frequency lists. Decide how I may incorporate the star data into the program. Label each word in my book with the part(s) of speech. Study vectors.

**Things you need to do**

Allow for modifications. Program the program. Keep in contact by email. Choose a method I may understand. Such as how big the data is. Include suggestions. Include problems. Reply to all emails as needed or reply as time allows you to in your schedule. (Optional) Create a GitHub account. Explain to me how to use the GitHub account. Create a List of Top 20 Vocabulary for each Pool analysis. Let me know how the computer program works in ordinary terms.

**Some Questions**

How does your program work? The definition of phrase is? Should we be considering phrases that appear only once? Should we consider changing the definition of phrase? Should you consider taking advantage of the ability to distinguish between lower and upper case letters? How may we incorporate the star data? Too much data? What are 26-dimensional vector values?

**Goals**

The goal is to narrow down the solution. The solution will be equal letter counts for Old Book to New Book. By creating the most common word combinations to be found in the text we will be able to limit the amount of processing the program has to do.