CO322: Data Structure and Algorithms

Lab-3 : part 2

Analyze the algorithm of the simple filter code

This simple filter code scan the large file and give the whole words that have same or sub set of the command argument word. To run the code the command line argument method have used. For example to find the words that use letters as *'er'* , the command can be used as *'**py regEx.py word.txt er**'* (in windows). The result of this command can be generated on command prompt as follows.

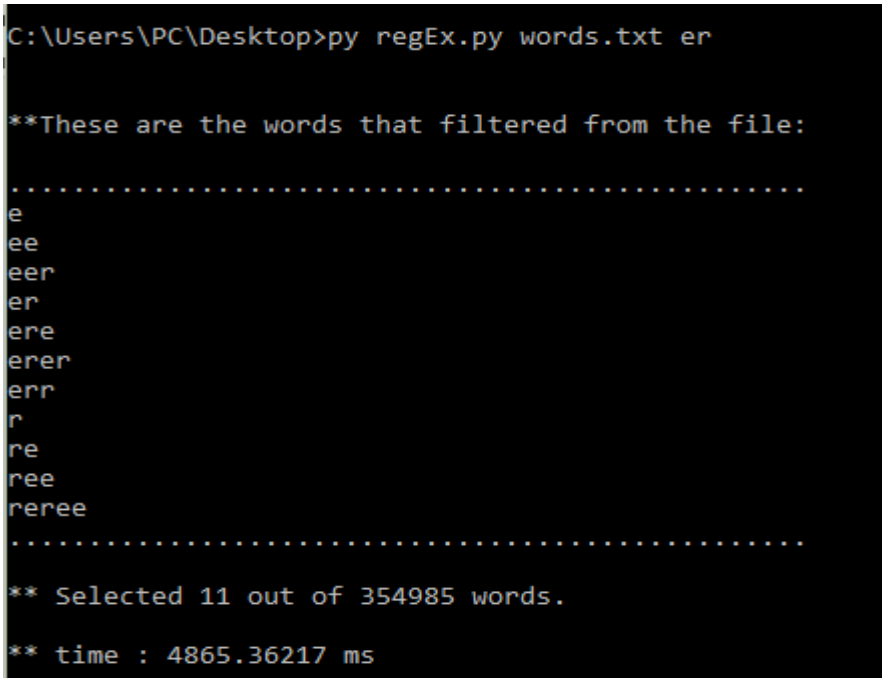

Figure 01

When consider about the algorithm part, it has two for loops to compare the two strings (command line string and the strings of the file) letter by letter. These strings are not in same length and the string length will affect to the complexity of the program.

Complexity of the simple filter code

Complexity of the given program will give the idea about the how do recourse requirement to run the program. Therefore, the complexity of the program will be directly affected to the performance of the computer. Mathematically complexity can be express as the big O (O) notation. Complexity can be divide as the runtime complexity and the space complexity.

1. Runtime complexity of the simple filter code: when consider about the runtime complexity of this code, it consists of two loops to compare the two strings. Then this strings are not always same in length. Let's assume that the command line argument string has *'m'* number of characters and the strings of the file has *'n'* number of characters, but this 'n' is completely depend on the *'len(list[i])'* code fragment. Therefore, *n* will vary always. And also, it should be compared all the string of the file to select the relevant string to print. Then when consider about the one string comparison it has *O(m.len(list[i]))* complexity, we can say normally algorithm has *O(mn)* complexity. When consider about input as *'aelp'*, it will print 97 elements within 6963.3 ms (this time as some variation because of variation of CPU utilization)

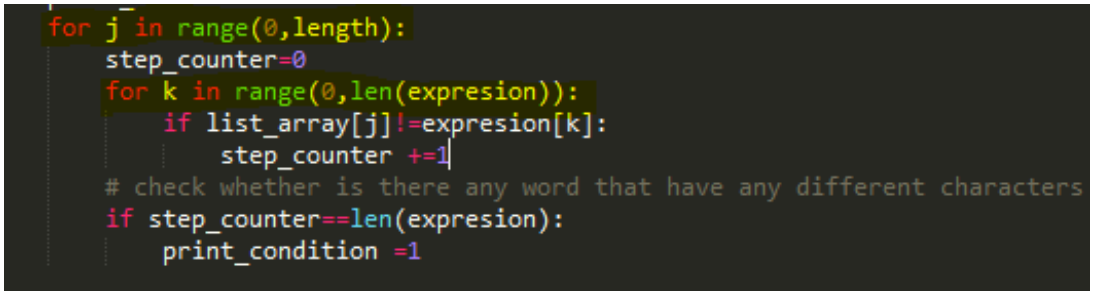

Figure 02

2. Space complexity of the algorithm: This is about the number of memory cells that utilize the program. In this method to store the all word a large list have used. Then this list will use in the program to compare the strings. This

method will fast because the program need not to scan the string from the *word.txt* file when comparing the strings.

There are some variables that have used to find any matching string according to the given conditions. In this algorithm, these variables are also important because they are always using in the two *for loops* and in the *if conditions.* And they are using in efficient manner as possible.