**Design change report for CS 5103 Course Project: Software Engineering Practice : word statistics (word count , line count, character count and word replacement project)**

1. **Introduction**

The word statistics project has three phases, the first phase is to count the frequency of each unique word. The code should support combinations of space, tab, and newline characters as separators. The second phase i.e first requirement change is to Provide two more features: Counting the number of lines (LineCount) and Counting the number of characters (CharCount). The third and the final phase 3 i.e the second requirement change is to allow replacement of all occurrences of a given word to a given replacement word. Note that the replacement happens only when the given pattern word matches with a whole word. For example, for text “ab cd ef”, replace “a” with “b” will result in no change, while replace “ab” with “cd” will result in “cd cd ef”.

1. **Design changes**

There is no specific architecture followed however there are code changes.

* 1. **Code changes**

I have used the python3 programming language in visual studio code editor environment to implement the project. For the first phase (initial set of requirements) the class get\_word\_freq in the code only counts the words. So we need to open the text file containing the string to be analyzed and pass it to the function so that it can create a list of unique words and it’s count. To support the first requirements change the original code is modified to accommodate splitting the lines of the text file so that we can get the number of lines and the characters count in the text file before creating the word list.

Intital code (refer it as Existing code going forward in document) :

|  |
| --- |
| def get\_word\_freq(ip\_string): |
|  |  |
|  | # convert the input string into a list of words |
|  | #input\_string\_list = re.split('\n|\t| ', input\_string) |
|  | #delimiters = " ", "\n", "\t" |
|  | #regex\_pattern = '|'.join(map(re.escape, delimiters)) |
|  | #regex\_pattern |
|  | #input\_string\_list = re.split(regex\_pattern, input\_string) |
|  | #input\_string\_list = input\_string.split() |
|  | input\_string\_list = ip\_string.strip().split() |
|  | #list(map(lambda x:x.strip(),input\_string\_list)) |
|  |  |
|  | #input\_string\_list1 = input\_string\_list.split('\n') |
|  | unique\_str\_list = [] |
|  |  |
|  | # iterate the input string list and find unique words |
|  | for i in input\_string\_list: |
|  |  |
|  | # test for duplicate values |
|  | if i not in unique\_str\_list: |
|  |  |
|  | # add unique words to second list |
|  | unique\_str\_list.append(i) |
|  |  |
|  | print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*") |
|  | print("unique\_string\_list = ", unique\_str\_list) |
|  | print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\n") |
|  |  |
|  | print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*") |
|  | for i in range(0, len(unique\_str\_list)): |
|  |  |
|  | # compute word frequency in input string |
|  | print('Word Frequency [{}]: {}'.format(unique\_str\_list[i], input\_string\_list.count(unique\_str\_list[i]))) |
|  |  |
|  | print("\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*") |
|  |  |
|  | def Driver(): |
|  | text = open("sample.txt", "r") |
|  | ip\_string = text.read() |
|  | """ python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yespython csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes python csharp javascript php python javascript csharp python csharp php\tmahi\nmahi please test c,c \ |
|  | yes""" |
|  | print(ip\_string) |
|  | get\_word\_freq(ip\_string) |
|  |  |
|  | if \_\_name\_\_=="\_\_main\_\_": |
|  | Driver() # call Driver() function |

Made below changes to the existing code : added the extra lines as below

#Added :

if len(unique\_str\_list) == 0:

      print("The input string is empty")

#also added :

text1 = open("sample.txt", "r")

   lines = len(text1.readlines())

   print("The number of lines in the input file is: \n\n", lines)

For the last batch of requirements, I have added a new function to replace word and removed the old ones.

def get\_word\_replacement(ip\_string):

    if len(ip\_string) == 0:

       print("The input string is empty")

    else:

       # Accept the input from the user for the given word and the replacement word

       old\_word = input("Provide the old word to be replaced :")

       new\_word = input("Provide the new word to be replaced with :")

       if len(old\_word) == 0:

          print("the word is not passed it's spaces. please start again ")

       else:

          # Replace all occurrences of old word with the new word

          replaced   = re.sub(r'(?<!\S)' + re.escape(old\_word) + r'(?!\S)', new\_word, ip\_string)

          # Write the modified contents to the file

          if (ip\_string == replaced):

             print("Nothing changed becuase word not found")

          with open('sample1.txt', 'w') as file:

               file.write(replaced)

               print("modified/replaced text file contents : ", replaced)