

**Github link :**

[https://github.com/sandeepvemulakonda/CS\\_5103\\_ion054/tree/master](https://github.com/sandeepvemulakonda/CS_5103_ion054/tree/master)

**Strings and Words** : This project is about writing a program to perform various word statistics of a given document (as a string). We will give input as a document and the output will be a list of the frequencies of each unique word, replacing an existing word with a given keyword and returning the lines in the document that contains at least one occurrence of the keyword.

In this project the initial requirement is to count the frequency of each unique word. Provided with the second requirement to replace the occurrences of the existing word to a given replacement word. The third requirement is to add a feature called grepline. This feature will take a document and keyword as its input and return the lines in the document that contain at least one occurrence of the keyword.

**Unit Tests:**

I wrote 4 unit tests for my project and all the 4 tests ran successfully. The 4 test cases include, one for Unique Word Frequency and one for Replacing a Keyword and two for the grepline function to print the line count of the keyword. All the tests ran successfully. I have provided the screenshots of the unit tests below. Also I have pushed the unittest file to my github repository.

```
CS_5103_ion054  UnitTests.py
StringUtil.py x UnitTests.py x
No Python interpreter configured for the module
1  import unittest
2  import StringUtil as Str
3
4
5  class MyTestCase(unittest.TestCase):
6      str = Str.StringUtil()
7      fileContent = ' Hi everyone. Hi again, this is Sandeep '
8      def test_uniqueWordCount(self):
9          actualOutput = self.str.uniqueWordCount(self.fileContent)
10         expectedOutput = {"Hi": 2,
11                           "everyone.": 1,
12                           "again,": 1,
13                           "this": 1,
14                           "is": 1,
15                           "Sandeep": 1}
16         self.assertEqual(len(actualOutput), len(expectedOutput))
17
18
19     def test_replaceWord(self):
20         findBy = 'Hi'
21         replaceBy = 'Hello'
22         actualOutput = self.str.replaceWord(self.fileContent, findBy, replaceBy)
23         expectedOutput = ' Hello everyone. Hello again, this is Sandeep '
24         self.assertEqual(actualOutput, expectedOutput)
25
26
27     def test_grepline_match(self):
28         keyword = 'Sandeep'
29         actualOutput = len(self.str.grepline(self.fileContent, keyword))
30         expectedOutput = 1
31         self.assertEqual(actualOutput, expectedOutput)
32
33
34     def test_grepline_no_match(self):
35         keyword = 'UTSA'
36         actualOutput = len(self.str.grepline(self.fileContent, keyword))
37         expectedOutput = 0
38         self.assertEqual(actualOutput, expectedOutput)
39
40
41     if __name__ == '__main__':
42         unittest.main()
```

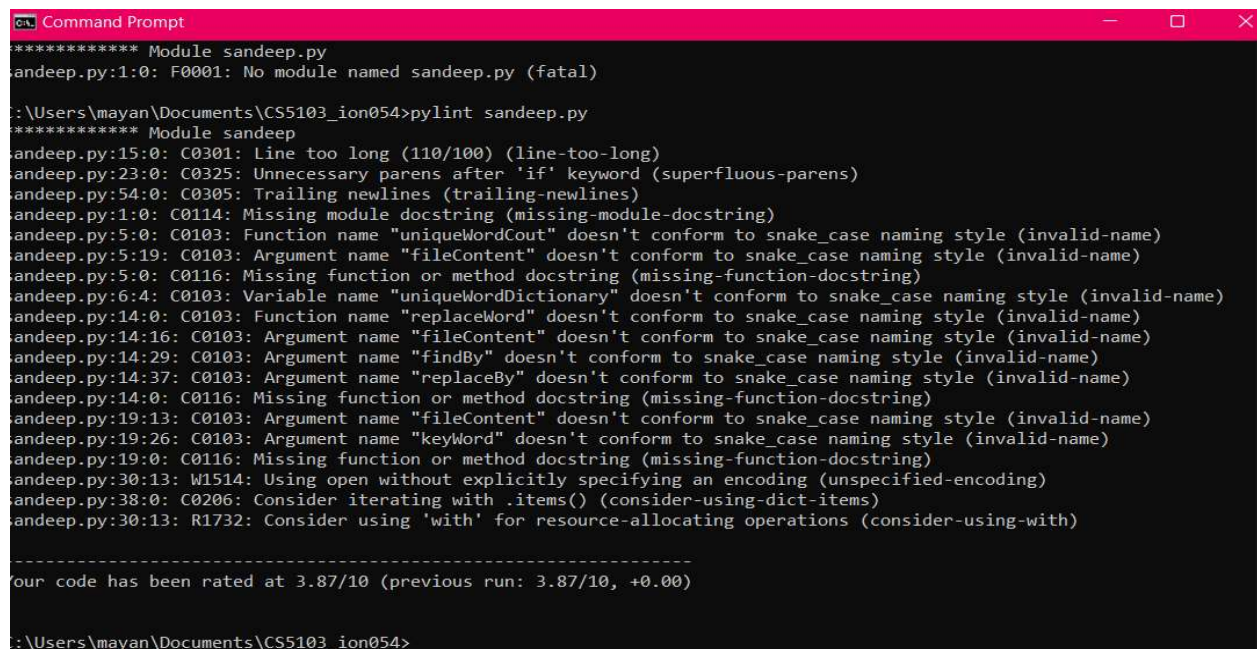
```
Ran 4 tests in 0.001s
Hi everyone. Hi again, this is Sandeep

keyword line
OK

Process finished with exit code 0
```

## Tool Application Report:

I have used the Pylint tool to check my code for any bugs or errors. Pylint is static analysis tool. I have attached the screenshot of the result after running my code with Pylint.



```
Command Prompt
***** Module sandeep.py
sandeep.py:1:0: F0001: No module named sandeep.py (fatal)

.: \Users\mayan\Documents\CS5103_ion054>pylint sandeep.py
***** Module sandeep
sandeep.py:15:0: C0301: Line too long (110/100) (line-too-long)
sandeep.py:23:0: C0325: Unnecessary parens after 'if' keyword (superfluous-parens)
sandeep.py:54:0: C0305: Trailing newlines (trailing-newlines)
sandeep.py:1:0: C0114: Missing module docstring (missing-module-docstring)
sandeep.py:5:0: C0103: Function name "uniqueWordCout" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:5:19: C0103: Argument name "fileContent" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:5:0: C0116: Missing function or method docstring (missing-function-docstring)
sandeep.py:6:4: C0103: Variable name "uniqueWordDictionary" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:14:0: C0103: Function name "replaceWord" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:14:16: C0103: Argument name "fileContent" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:14:29: C0103: Argument name "findBy" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:14:37: C0103: Argument name "replaceBy" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:14:0: C0116: Missing function or method docstring (missing-function-docstring)
sandeep.py:19:13: C0103: Argument name "fileContent" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:19:26: C0103: Argument name "keyWord" doesn't conform to snake_case naming style (invalid-name)
sandeep.py:19:0: C0116: Missing function or method docstring (missing-function-docstring)
sandeep.py:30:13: W1514: Using open without explicitly specifying an encoding (unspecified-encoding)
sandeep.py:38:0: C0206: Consider iterating with .items() (consider-using-dict-items)
sandeep.py:30:13: R1732: Consider using 'with' for resource-allocating operations (consider-using-with)

-----
Your code has been rated at 3.87/10 (previous run: 3.87/10, +0.00)

.: \Users\mayan\Documents\CS5103_ion054>
```

I have also used the Pep8 online tool to check the format of my code and also to see if there are any redundancies and also helped me to check the code against some style conventions.

# Check results

[Save](#) ▾[Share](#)

Code	Line	Column	Text
<b>E501</b>	15	80	line too long (110 > 79 characters)
<b>E303</b>	28	1	too many blank lines (3)
<b>E501</b>	51	80	line too long (84 > 79 characters)
<b>E231</b>	53	11	missing whitespace after ','
<b>W391</b>	54	1	blank line at end of file

## Design Change Report:

In the code I have included the first function as **UniqueWordCount** which returns the frequency of unique words. I created an empty word dictionary to store and update word count for each unique word. Second function as **ReplaceWord** which replaces the existing word with the given new word.

We will give input as a sample text file and the output will be a list of the frequencies of words. Also we need to give the existing and the new word, to replace the existing word with a new word.

For the third requirement, I added a new function called **grepline**, in which the file content is splitted and stored in the variable lines. Then we will search for the line in which the given keyword is available and then print the entire line.

## Implementation:

I have used the PyCharm editor to run my python code. The name of the file is **sandeep.py** and also I have created a **sample** text file in which we will store the string. I have uploaded the code in the github account as a public repository. Open the Python file(saved as .py file) and the sampleText file from the github repository, download it and run it.