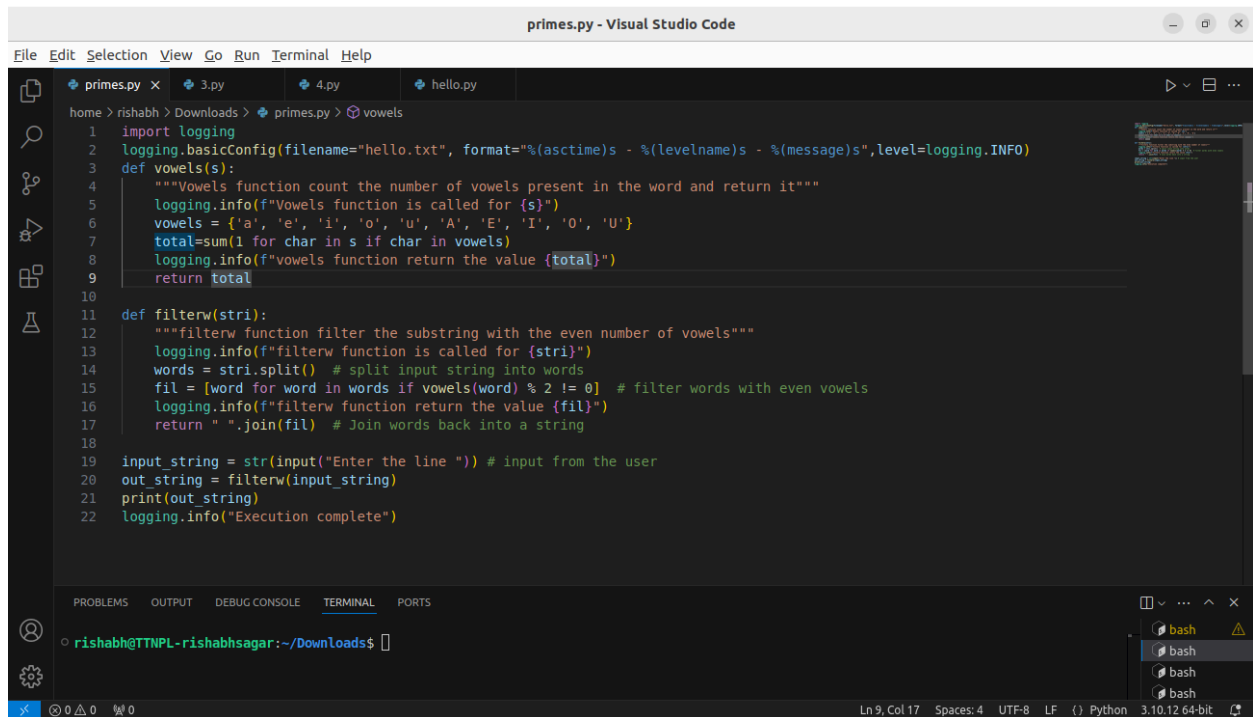


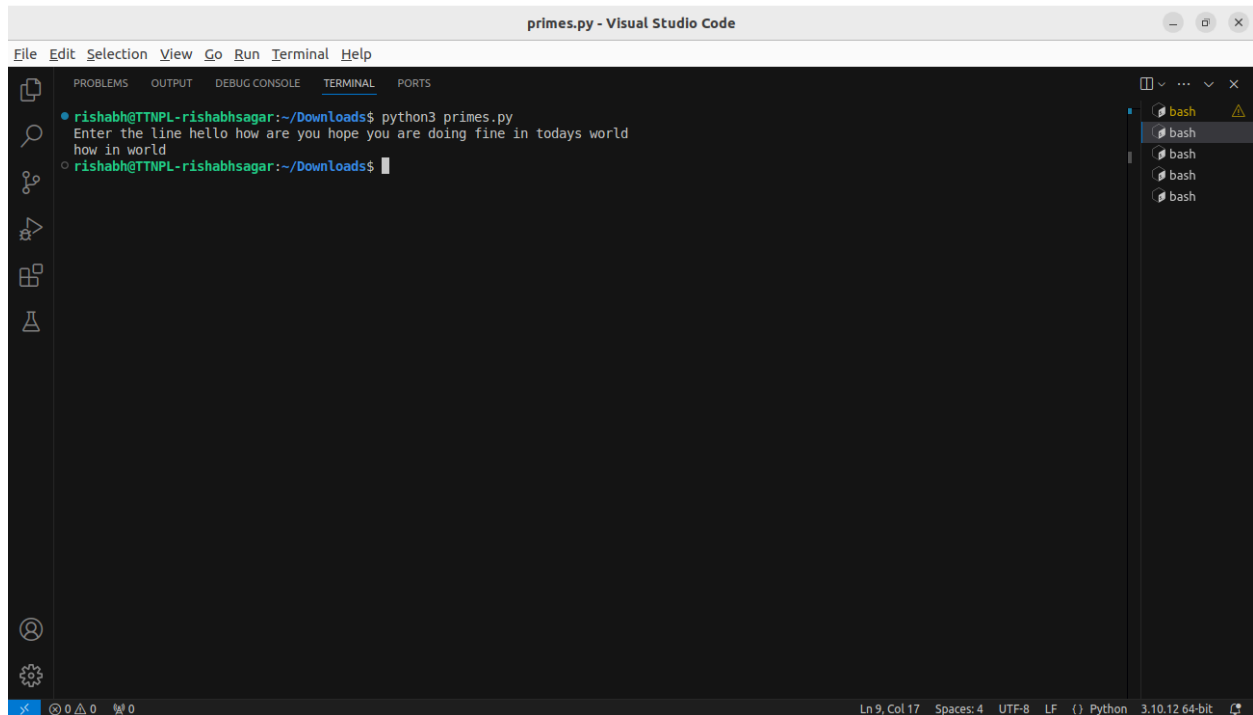
## Control Flow and Built in Data Structures

Q1) Write a code to filter all sub-strings which have an even number of vowels? example Input: "I have an input string which contains even and odd numbers of vowels aA aa aaa ae aeo" output: I an string which contains and odd of aaa aeo

Ans.



```
primes.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
primes.py x 3.py 4.py hello.py
home > rishabh > Downloads > primes.py > vowels
1 import logging
2 logging.basicConfig(filename="hello.txt", format="%asctime)s - %(levelname)s - %(message)s", level=logging.INFO)
3 def vowels(s):
4     """Vowels function count the number of vowels present in the word and return it"""
5     logging.info(f"Vowels function is called for {s}")
6     vowels = {'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'}
7     total = sum(1 for char in s if char in vowels)
8     logging.info(f"vowels function return the value {total}")
9     return total
10
11 def filterw(str1):
12     """filterw function filter the substring with the even number of vowels"""
13     logging.info(f"filterw function is called for {str1}")
14     words = str1.split() # split input string into words
15     fil = [word for word in words if vowels(word) % 2 != 0] # filter words with even vowels
16     logging.info(f"filterw function return the value {fil}")
17     return " ".join(fil) # Join words back into a string
18
19 input_string = str(input("Enter the line ")) # input from the user
20 out_string = filterw(input_string)
21 print(out_string)
22 logging.info("Execution complete")
```



```
primes.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
rishabh@TTNPL-rishabhsagar:~/Downloads$ python3 primes.py
Enter the line hello how are you hope you are doing fine in todays world
how in world
rishabh@TTNPL-rishabhsagar:~/Downloads$
```



```
1 2025-02-04 16:33:07,291 - INFO - filterw function is called for hello how are you hope you are doing fine in todays world
2 2025-02-04 16:33:07,291 - INFO - filterw function return the value ['how', 'in', 'world']
3 2025-02-04 16:33:07,291 - INFO - Execution complete|
```

The screenshot shows a code editor window with the title 'hello.txt' and a path '~Downloads'. The editor contains three lines of log output. The status bar at the bottom indicates 'Plain Text', 'Tab Width: 8', 'Ln 3, Col 52', and 'INS'.

**Q2)Programing: From a multi-words and multi-line string, prepare a dict with key as "word" and value as occurence of word. Example Input: astring = """Python Multiline String Using Triple-Quotes Using the triple quotes style is one of the easiest and most common ways to split a large string into a multiline Python string. Triple quotes ("" or ''''') can be used to create a multiline string. It allows you to format text over many lines and include line breaks. Put two triple quotes around the multiline Python string, one at the start and one at the end, to define it.""" output (Partical): {'the': 5, 'to': 4, 'Python': 3, 'quotes': 3, 'one': 3, 'and': 3, 'a': 3, 'multiline': 3, 'Using': 2, 'triple': 2, 'string.': 2, 'at': 2, 'Multiline': 1, 'String': 1, 'Triple-Quotes': 1, 'style': 1, 'is': 1, 'of': 1, 'easiest': 1, 'most': 1, 'common': 1, 'ways': 1, 'split': 1, 'large': 1, 'string': 1, 'into': 1, 'Triple': 1, '("": 1, 'or': 1, ''''')': 1, 'can': 1, 'be': 1, 'used': 1, 'create': 1, 'It': 1, 'allows': 1, 'you': 1, 'format': 1, 'text': 1, 'over': 1, 'many': 1, 'lines': 1, 'include': 1, 'line': 1, 'breaks.': 1, 'Put': 1, 'two': 1, 'around': 1, 'string.': 1, 'start': 1, 'end.': 1, 'define': 1, 'it.': 1}**

Ans.

hello.py - Visual Studio Code

File Edit Selection View Go Run Terminal Help

4.py 3.py hello.py x

home > rishabh > Downloads > hello.py > count

```
1 import logging
2 logging.basicConfig(filename="hi.txt", format="%{asctime}s - %{levelname}s - %{message}s", level=logging.INFO)
3 dic = {}
4 def count(stri):
5     """Count function count the frequency of the word in the string and print it"""
6     logging.info(f'Inside the function for the {stri}')
7     sp = stri.split() # split the line into the list
8     for word in sp:
9         if word in dic:
10             dic[word] += 1
11         else:
12             dic[word] = 1
13     logging.info("Function execution complete")
14     sorted_dict = dict(sorted(dic.items(), key=lambda item: item[1], reverse=True)) # make the final result by sorting
15     print(sorted_dict)
16
17 count('Python Multiline String Using Triple-Quotes Using the triple quotes style is one of the easiest and most common way
18 string into a multiline Python string. Triple quotes (\'\'\' or \'\')
19 can be used to create a multiline string. It allows you to format text
20 over many lines and include line breaks. Put two triple quotes around the
21 multiline Python string, one at the start and one at the end, to define it.\'\'')
22
```

Ln 12, Col 26 Spaces: 4 UTF-8 LF Python 3.10.12 64-bit

The screenshot displays the Visual Studio Code interface. The top menu bar includes File, Edit, Selection, View, Go, Run, Terminal, and Help. The left sidebar shows icons for Explorer, Search, Source Control, Run and Debug, and Extensions. The main editor area is split into two panes. The left pane shows a file named `hello.py` with the following Python code:

```
python3 hello.py
{
  'the': 5, 'to': 4, 'Python': 3, 'quotes': 3, 'one': 3, 'and': 3, 'a': 3, 'multiline': 3, 'Using': 2, 'triple': 2, 'string.': 2, 'at': 2,
  'Multiline': 1, 'String': 1, 'Triple-Quotes': 1, 'style': 1, 'is': 1, 'of': 1, 'easiest': 1, 'most': 1, 'common': 1, 'ways': 1, 'split':
  1, 'large': 1, 'string': 1, 'into': 1, 'Triple': 1, 'or': 1, 'can': 1, 'be': 1, 'used': 1, 'create': 1, 'It': 1,
  'allows': 1, 'you': 1, 'format': 1, 'text': 1, 'over': 1, 'many': 1, 'lines': 1, 'include': 1, 'line': 1, 'breaks': 1, 'Put': 1, 'two':
  1, 'around': 1, 'string.': 1, 'start': 1, 'end.': 1, 'define': 1, 'it.': 1
}
```

The right pane shows the output of the script, which is a JSON object containing word frequency counts. The bottom status bar indicates the current file is `hi.txt` at line 7, column 20, using UTF-8 encoding and LF line endings, with a Python 3.10.12 64-bit interpreter.

Below the main editor, a separate window titled `hi.txt` is open, showing the output of the script as a log of messages:

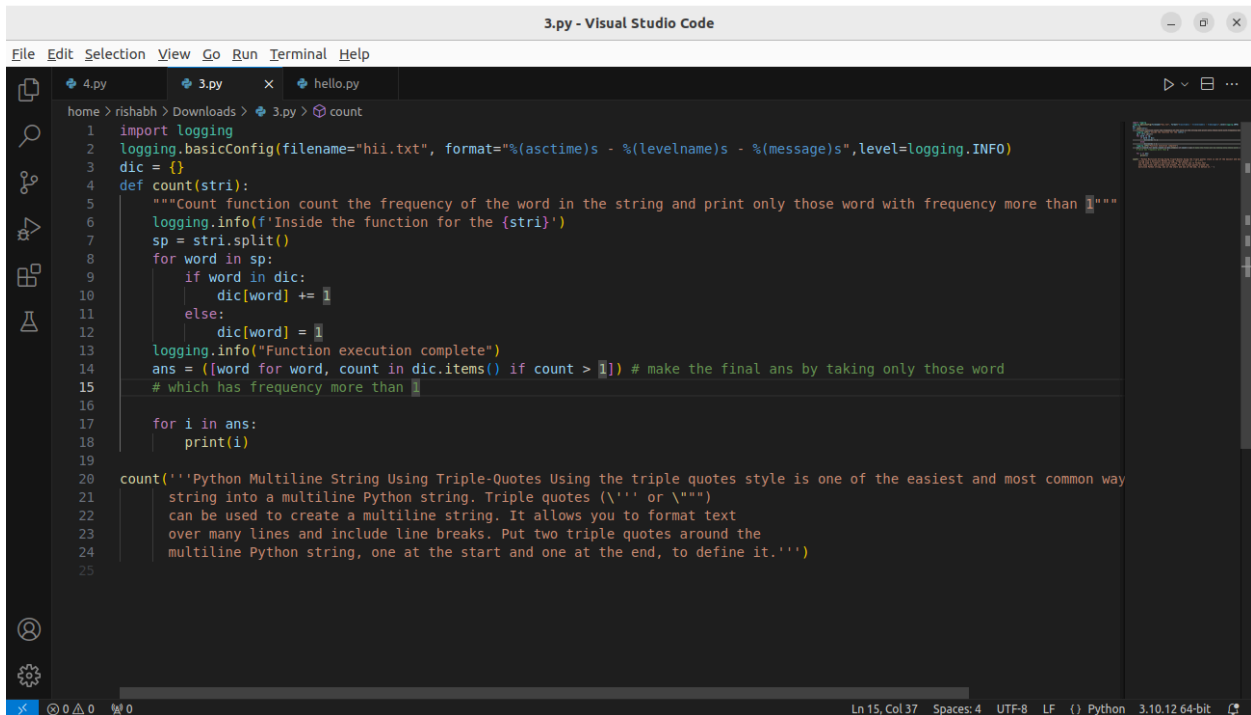
```
1 2025-02-04 16:36:41,912 - INFO - Inside the function for the Python Multiline String Using Triple-Quotes Using the triple quotes style is
one of the easiest and most common ways to split a large
2 string into a multiline Python string. Triple quotes (''' or ''')
3 can be used to create a multiline string. It allows you to format text
4 over many lines and include line breaks. Put two triple quotes around the
5 multiline Python string, one at the start and one at the end, to define it.
6 2025-02-04 16:36:41,913 - INFO - Function execution complete
```

The bottom status bar of this window shows 'Plain Text', 'Tab Width: 8', 'Ln 6, Col 61', and 'INS'.

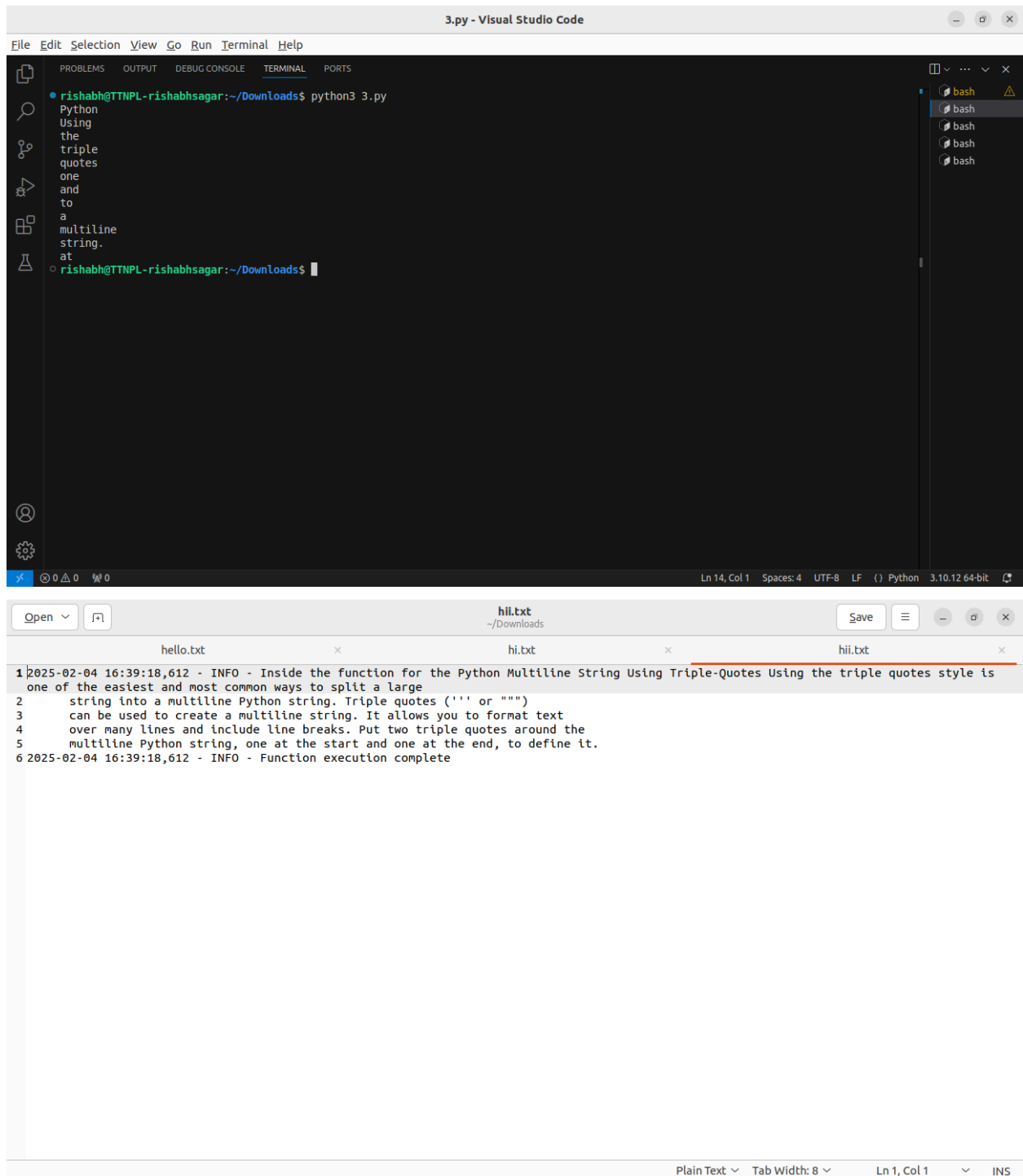
**Q3)Programming: From a multi-words and multi-line string, prepare a filter for the list of words which have multiple occurrences in the string. Example Input: astring = """Python Multiline String Using Triple-Quotes Using the triple quotes style is one of the easiest and most common ways to split a large string into a multiline Python string. Triple quotes ("" or """) can be used to create a multiline string. It allows you to format text over many lines and include line breaks. Put two triple quotes around the multiline Python string,**

one at the start and one at the end, to define it.""" output: Python Using the triple quotes one and to a multiline string.

Ans.



```
3.py - Visual Studio Code
File Edit Selection View Go Run Terminal Help
4.py 3.py x hello.py
home > risabh > Downloads > 3.py > count
1 import logging
2 logging.basicConfig(filename="hii.txt", format="%asctimes - %(levelname)s - %(message)s",level=logging.INFO)
3 dic = {}
4 def count(str1):
5     """Count function count the frequency of the word in the string and print only those word with frequency more than 1"""
6     logging.info(f'Inside the function for the {str1}')
7     sp = str1.split()
8     for word in sp:
9         if word in dic:
10             dic[word] += 1
11         else:
12             dic[word] = 1
13     logging.info("Function execution complete")
14     ans = ([word for word, count in dic.items() if count > 1]) # make the final ans by taking only those word
15     # which has frequency more than 1
16
17     for i in ans:
18         print(i)
19
20 count('Python Multiline String Using Triple-Quotes Using the triple quotes style is one of the easiest and most common way
21 string into a multiline Python string. Triple quotes (\'\'\' or \\'\'\'\'\'
22 can be used to create a multiline string. It allows you to format text
23 over many lines and include line breaks. Put two triple quotes around the
24 multiline Python string, one at the start and one at the end, to define it.\'\'\'')
25
Ln 15, Col 37 Spaces: 4 UTF-8 LF () Python 3.10.12 64-bit
```

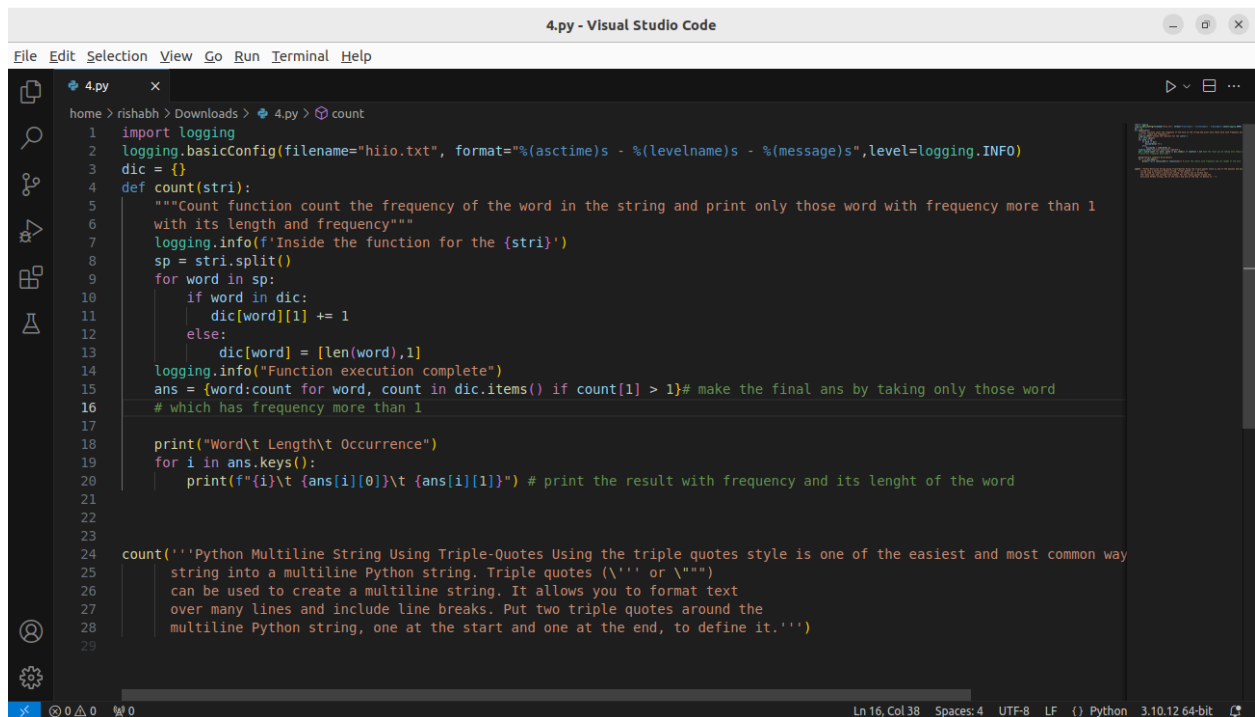


The screenshot displays the Visual Studio Code interface. The top panel shows the 'TERMINAL' tab with the command `python3 3.py` executed. The bottom panel shows the 'hii.txt' file, which contains the output of the script. The output is a list of words and their lengths, sorted by length in descending order.

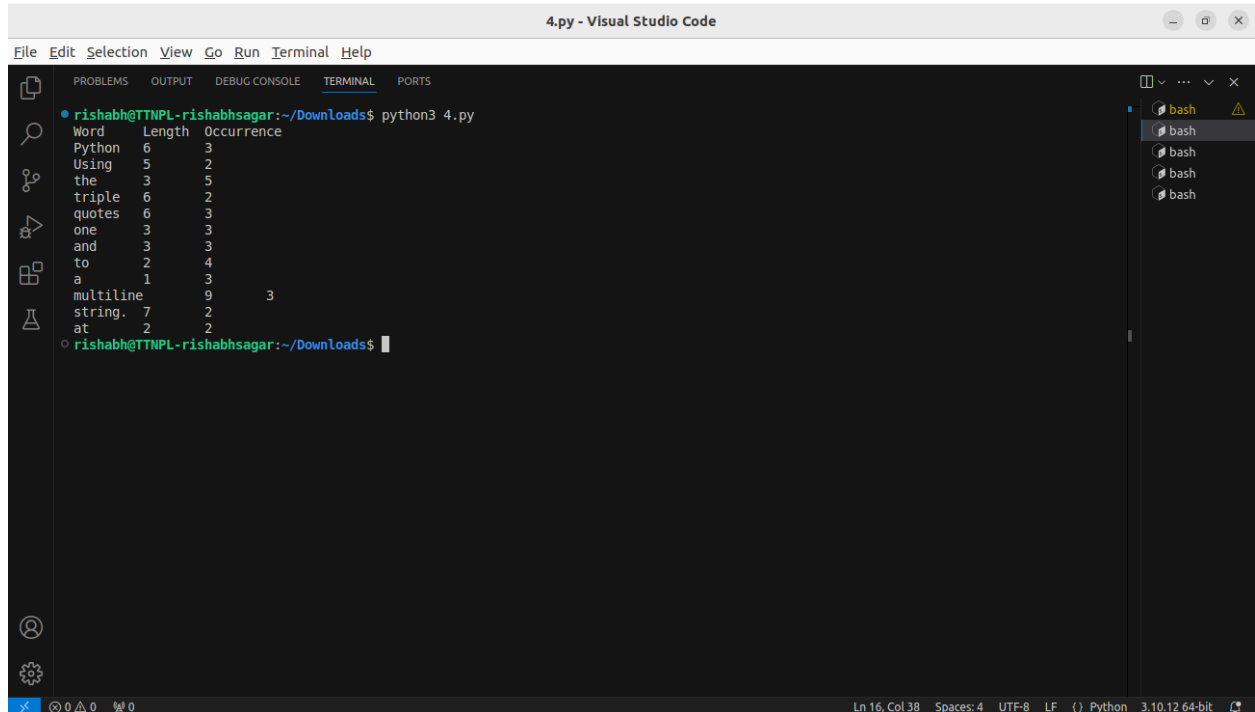
```
1 2025-02-04 16:39:18,612 - INFO - Inside the function for the Python Multiline String Using Triple-Quotes Using the triple quotes style is
2 one of the easiest and most common ways to split a large
3 string into a multiline Python string. Triple quotes (''' or """)
4 can be used to create a multiline string. It allows you to format text
5 over many lines and include line breaks. Put two triple quotes around the
6 multiline Python string, one at the start and one at the end, to define it.
7 2025-02-04 16:39:18,612 - INFO - Function execution complete
```

**Q4)Programing: From a multi-words and multi-line string, display list of words and word's length with occurrence more than 1 in sorted order Example Input: astring =  
"""Python Multiline String Using Triple-Quotes Using the triple quotes style is one of the easiest and most common ways to split a large string into a multiline Python string. Triple quotes ("" or \"""") can be used to create a multiline string. It allows you to format text over many lines and include line breaks. Put two triple quotes around the multiline Python string, one at the start and one at the end, to define it.""" Word Length**

Occurrence Python 6 3 Using 5 2 the 3 5 triple 6 2 quotes 6 3 one 3 3 and 3 3 to 2 4 a 1 3  
multiline 9 3 string. 7 2 at 2 2  
Ans.



```
1 import logging
2 logging.basicConfig(filename="hiio.txt", format="%(asctime)s - %(levelname)s - %(message)s", level=logging.INFO)
3 dic = {}
4 def count(stri):
5     """Count function count the frequency of the word in the string and print only those word with frequency more than 1
6     with its length and frequency"""
7     logging.info(f'Inside the function for the {stri}')
8     sp = stri.split()
9     for word in sp:
10         if word in dic:
11             dic[word][1] += 1
12         else:
13             dic[word] = [len(word), 1]
14     logging.info("Function execution complete")
15     ans = {word:count for word, count in dic.items() if count[1] > 1} # make the final ans by taking only those word
16     # which has frequency more than 1
17
18     print("Word\t Length\t Occurrence")
19     for i in ans.keys():
20         print(f"{i}\t {ans[i][0]}\t {ans[i][1]}") # print the result with frequency and its lenght of the word
21
22
23
24 count('Python Multiline String Using Triple-Quotes Using the triple quotes style is one of the easiest and most common way
25 string into a multiline Python string. Triple quotes (\'\'\' or \\'\'\'')
26 can be used to create a multiline string. It allows you to format text
27 over many lines and include line breaks. Put two triple quotes around the
28 multiline Python string, one at the start and one at the end, to define it.\'\'\'')
29
```



```
Word Length Occurrence
Python 6 3
Using 5 2
the 3 5
triple 6 2
quotes 6 3
one 3 3
and 3 3
to 2 4
a 1 3
multiline 9 3
string. 7 2
at 2 2
```

Open

hilo.txt  
~/Downloads

Save

hello.txt × hi.txt × hii.txt × hilo.txt ×

1

2025-02-04 16:53:02,152 - INFO - Inside the function for the Python Multiline String Using Triple-Quotes Using the triple quotes style is one of the easiest and most common ways to split a large

2 string into a multiline Python string. Triple quotes (''' or """)

3 can be used to create a multiline string. It allows you to format text

4 over many lines and include line breaks. Put two triple quotes around the

5 multiline Python string, one at the start and one at the end, to define it.

6 2025-02-04 16:53:02,152 - INFO - Function execution complete

Loading file "/home/rishabh/Downloads/hilo.txt" ...

Plain Text

Tab Width: 8

Ln 1, Col 1

INS