

ASSIGNMENT-4.3

2303A51337

BATCH-10

❖ TASK-1:

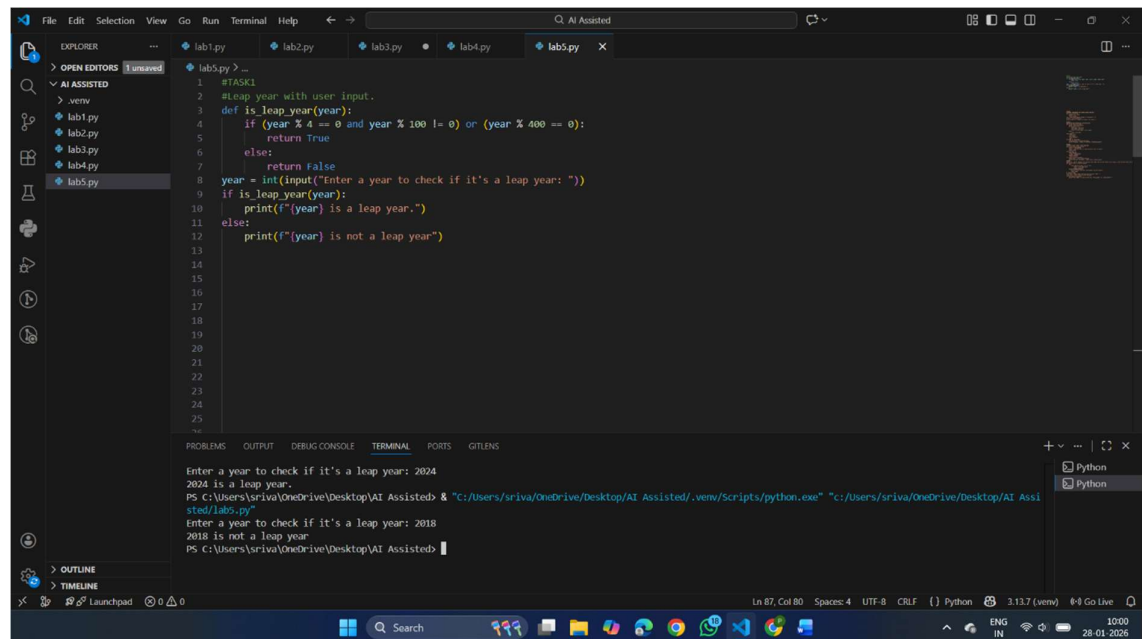
PROMPT:

#Leap year with user input.

CODE:

```
def is_leap_year(year):  
    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):  
        return True  
    else:  
        return False  
  
year = int(input("Enter a year to check if it's a leap year: "))  
if is_leap_year(year):  
    print(f"{year} is a leap year.")  
else:  
    print(f"{year} is not a leap year")
```

OUTPUT:



The screenshot shows a Visual Studio Code editor window with a dark theme. The Explorer sidebar on the left shows a file named 'lab5.py' selected. The main editor area displays the Python code for Task 1. The terminal at the bottom shows the output of the program, which prompts the user to enter a year and then prints whether it is a leap year or not. The output shows two test cases: 2024 is a leap year, and 2018 is not a leap year.

```
1 #TASK1  
2 #Leap year with user input.  
3 def is_leap_year(year):  
4     if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):  
5         return True  
6     else:  
7         return False  
8 year = int(input("Enter a year to check if it's a leap year: "))  
9 if is_leap_year(year):  
10     print(f"{year} is a leap year.")  
11 else:  
12     print(f"{year} is not a leap year")  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS GIT LENS

```
Enter a year to check if it's a leap year: 2024  
2024 is a leap year.  
PS C:\Users\sriya\OneDrive\Desktop\AI Assisted & "C:\Users\sriya\OneDrive\Desktop\AI Assi  
sted\lab5.py"  
Enter a year to check if it's a leap year: 2018  
2018 is not a leap year  
PS C:\Users\sriya\OneDrive\Desktop\AI Assisted
```

Ln 87, Col 80 Spaces: 4 UTF-8 CRLF Python 3.13.7 (venv) 6-9 Go Live

EXPLANATION:

A year is a leap year if:

1. It is divisible by 4
 2. BUT NOT divisible by 100,
 3. UNLESS it is also divisible by 400
- `year % 4 == 0` → year is divisible by 4
 - `year % 100 != 0` → year is not divisible by 100
 - `year % 400 == 0` → year is divisible by 400

If the condition is true, the function returns True (leap year).

Otherwise, it returns False (not a leap year).

❖ TASK-2

PROMPT:

#CONVERT CENTIMETER INTO INCHES USING FUNCTION.

CODE:

```
def cm_to_inches(cm):  
    inches = cm / 2.54  
    return inches  
  
cm = float(input("Enter length in centimeters: "))  
inches = cm_to_inches(cm)  
print(f"{cm} cm is equal to {inches:.2f} inches.")
```

OUTPUT:

```
19
20 #TASK2
21 #CONVERT CENTIMETER INTO INCHES USING FUNCTION.
22 def cm_to_inches(cm):
23     inches = cm / 2.54
24     return inches
25 cm = float(input("Enter length in centimeters: "))
26 inches = cm_to_inches(cm)
27 print(f"{cm} cm is equal to {inches:.2f} inches.")
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
...
```

```
std/1ab5.py
Enter length in centimeters: 10
10.0 cm is equal to 3.94 inches.
PS C:\Users\sriya\OneDrive\Desktop\AI Assisted>
```

EXPLANATION:

This program converts a length given in centimeters into inches using a function.

The function applies the standard conversion formula, where 1 inch equals 2.54 centimeters.

The user is asked to enter a value in centimeters, which is then passed to the function.

The function performs the calculation and returns the equivalent length in inches.

Finally, the result is displayed to the user, formatted to two decimal places for better readability.

❖ TASK-3:

PROMPT:

#Generate name formatting using function.

CODE:

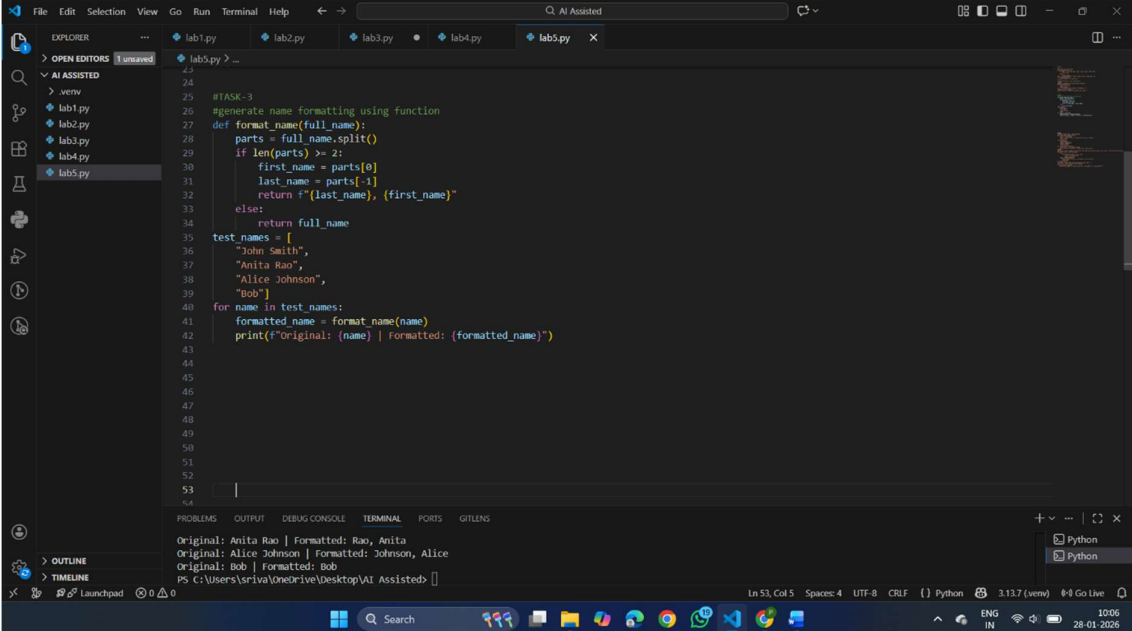
```
def format_name(full_name):
    parts = full_name.split()
    if len(parts) >= 2:
        first_name = parts[0]
        last_name = parts[-1]
        return f"{last_name}, {first_name}"
    else:
```

```

        return full_name
test_names = [
    "John Smith",
    "Anita Rao",
    "Alice Johnson",
    "Bob"]
for name in test_names:
    formatted_name = format_name(name)
    print(f"Original: {name} | Formatted: {formatted_name}")

```

OUTPUT:



The screenshot shows a VS Code editor with a Python file named `lab5.py`. The code defines a function `format_name` that takes a full name as input and splits it into individual parts using spaces. If the name contains two or more words, the function treats the first word as the first name and the last word as the last name, then rearranges them into the required format. If the name has only one word, it is returned unchanged. A list of sample names is used to test the function. Each name in the list is processed, and the program prints both the original name and the formatted version.

```

24
25 #TASK-3
26 #generate name formatting using function
27 def format_name(full_name):
28     parts = full_name.split()
29     if len(parts) >= 2:
30         first_name = parts[0]
31         last_name = parts[-1]
32         return f"{last_name}, {first_name}"
33     else:
34         return full_name
35 test_names = [
36     "John Smith",
37     "Anita Rao",
38     "Alice Johnson",
39     "Bob"]
40 for name in test_names:
41     formatted_name = format_name(name)
42     print(f"Original: {name} | Formatted: {formatted_name}")
43
44
45
46
47
48
49
50
51
52
53
54

```

The terminal output shows the following results:

```

Original: Anita Rao | Formatted: Rao, Anita
Original: Alice Johnson | Formatted: Johnson, Alice
Original: Bob | Formatted: Bob

```

EXPLANATION:

This program formats names into a “Last name, First name” style.

It defines a function that takes a full name as input and splits it into individual parts using spaces.

If the name contains two or more words, the function treats the first word as the first name and the last word as the last name, then rearranges them into the required format.

If the name has only one word, it is returned unchanged.

A list of sample names is used to test the function. Each name in the list is processed, and the program prints both the original name and the formatted version.

❖ TASK-4:

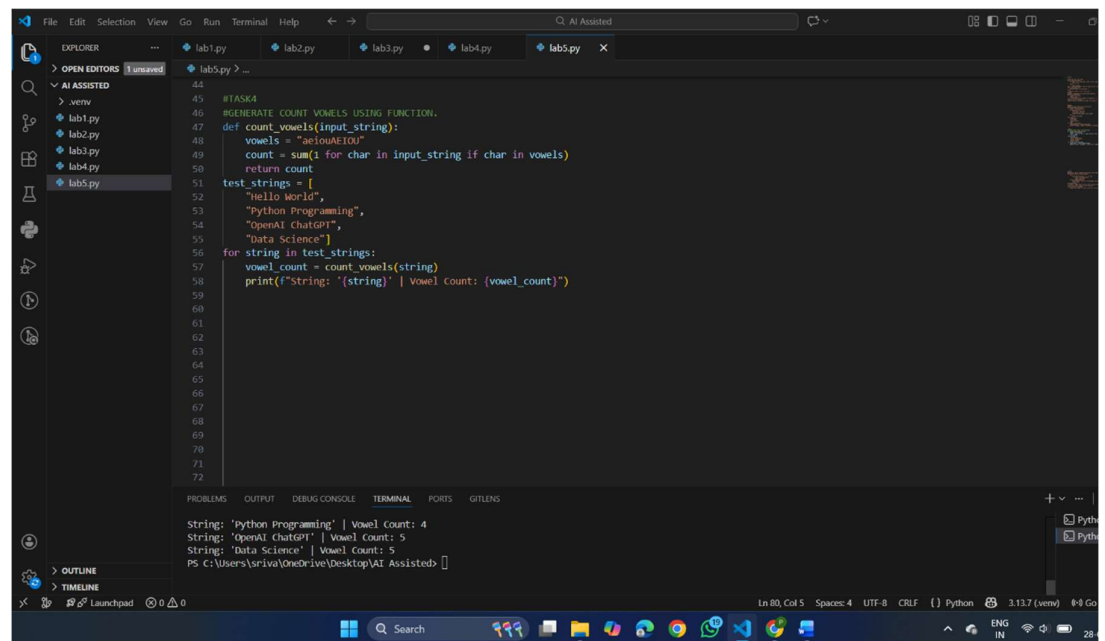
PROMPT:

#GENERATE COUNT VOWELS USING FUNCTION.

CODE:

```
def count_vowels(input_string):  
    vowels = "aeiouAEIOU"  
    count = sum(1 for char in input_string if char in vowels)  
    return count  
test_strings = [  
    "Hello World",  
    "Python Programming",  
    "OpenAI ChatGPT",  
    "Data Science"]  
for string in test_strings:  
    vowel_count = count_vowels(string)  
    print(f"String: '{string}' | Vowel Count: {vowel_count}")
```

OUTPUT:



The screenshot shows a Visual Studio Code editor with a Python file named lab5.py. The code defines a function count_vowels that takes an input string and returns the count of vowels (a, e, i, o, u) in both uppercase and lowercase. It then iterates over a list of test strings and prints the vowel count for each. The terminal output shows the results for the first two strings: 'Python Programming' with 4 vowels and 'OpenAI ChatGPT' with 5 vowels. The file explorer on the left shows a project structure with a .venv directory and several lab files (lab1.py to lab5.py). The status bar at the bottom indicates the current file is lab5.py, line 80, column 5, with 4 spaces, UTF-8 encoding, and CRLF line endings.

```
44  
45 #TASK4  
46 #GENERATE COUNT VOWELS USING FUNCTION.  
47 def count_vowels(input_string):  
48     vowels = "aeiouAEIOU"  
49     count = sum(1 for char in input_string if char in vowels)  
50     return count  
51 test_strings = [  
52     "Hello World",  
53     "Python Programming",  
54     "OpenAI ChatGPT",  
55     "Data Science"]  
56 for string in test_strings:  
57     vowel_count = count_vowels(string)  
58     print(f"String: '{string}' | Vowel Count: {vowel_count}")  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72
```

String: 'Python Programming' | Vowel Count: 4
String: 'OpenAI ChatGPT' | Vowel Count: 5
String: 'Data Science' | Vowel Count: 5
PS C:\Users\sriya\OneDrive\Desktop\AI Assisted >

EXPLANATION:

The program counts how many vowels (a, e, i, o, u) appear in each string (both uppercase and lowercase) and prints the result for every test string.

❖ TASK-5:

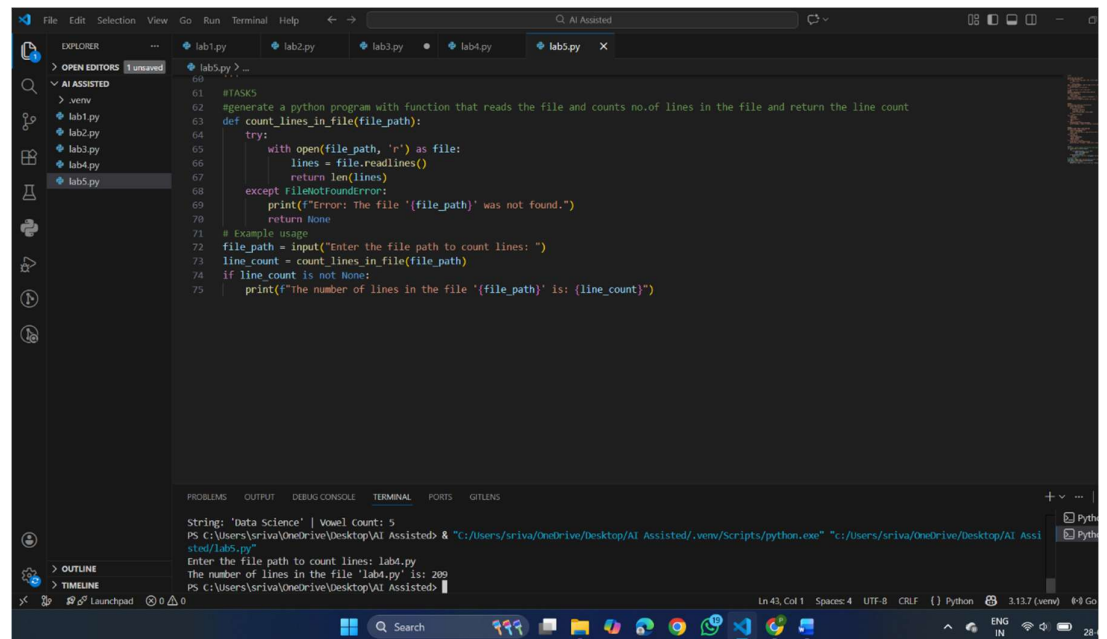
PROMPT:

#generate a python program with function that reads the file and counts no.of lines in the file and return the line count.

CODE:

```
counts no.of lines in the file and return the line count
def count_lines_in_file(file_path):
    try:
        with open(file_path, 'r') as file:
            lines = file.readlines()
            return len(lines)
    except FileNotFoundError:
        print(f"Error: The file '{file_path}' was not found.")
        return None
# Example usage
file_path = input("Enter the file path to count lines: ")
line_count = count_lines_in_file(file_path)
if line_count is not None:
    print(f"The number of lines in the file '{file_path}' is: {line_count}")
```

OUTPUT:



The screenshot shows a Visual Studio Code editor window with a Python script named `lab5.py` open. The script defines a function `count_lines_in_file` that reads a file and returns the number of lines. It includes an example usage section where it prompts the user for a file path and prints the line count. The terminal at the bottom shows the execution of the script, where the user enters `lab4.py` and the output is `The number of lines in the file 'lab4.py' is: 209`.

```
61 #TASK5
62 #generate a python program with function that reads the file and counts no.of lines in the file and return the line count
63 def count_lines_in_file(file_path):
64     try:
65         with open(file_path, 'r') as file:
66             lines = file.readlines()
67             return len(lines)
68     except FileNotFoundError:
69         print(f"Error: The file '{file_path}' was not found.")
70         return None
71 # Example usage
72 file_path = input("Enter the file path to count lines: ")
73 line_count = count_lines_in_file(file_path)
74 if line_count is not None:
75     print(f"The number of lines in the file '{file_path}' is: {line_count}")
```

String: 'Data Science' | Vowel Count: 5
PS C:\Users\sriva\OneDrive\Desktop\AI Assisted> "C:\Users\sriva\OneDrive\Desktop\AI Assisted\.venv\Scripts\python.exe" "C:\Users\sriva\OneDrive\Desktop\AI Assisted\lab5.py"
Enter the file path to count lines: lab4.py
The number of lines in the file 'lab4.py' is: 209
PS C:\Users\sriva\OneDrive\Desktop\AI Assisted>

EXPLANATION:

This defines a function that reads a text file and counts the number of lines in it.

The function accepts a file path as input and attempts to open the file in readmode.

If the file is successfully opened, it reads all the lines and returns the total number of lines present in the file.

The user is prompted to enter the file path, which is passed to the function. If a valid line count is returned, the program prints the total number of lines in the file.