

ASSIGNMENT-4.3

2303A51546

BATCH-10

❖ TASK-1:

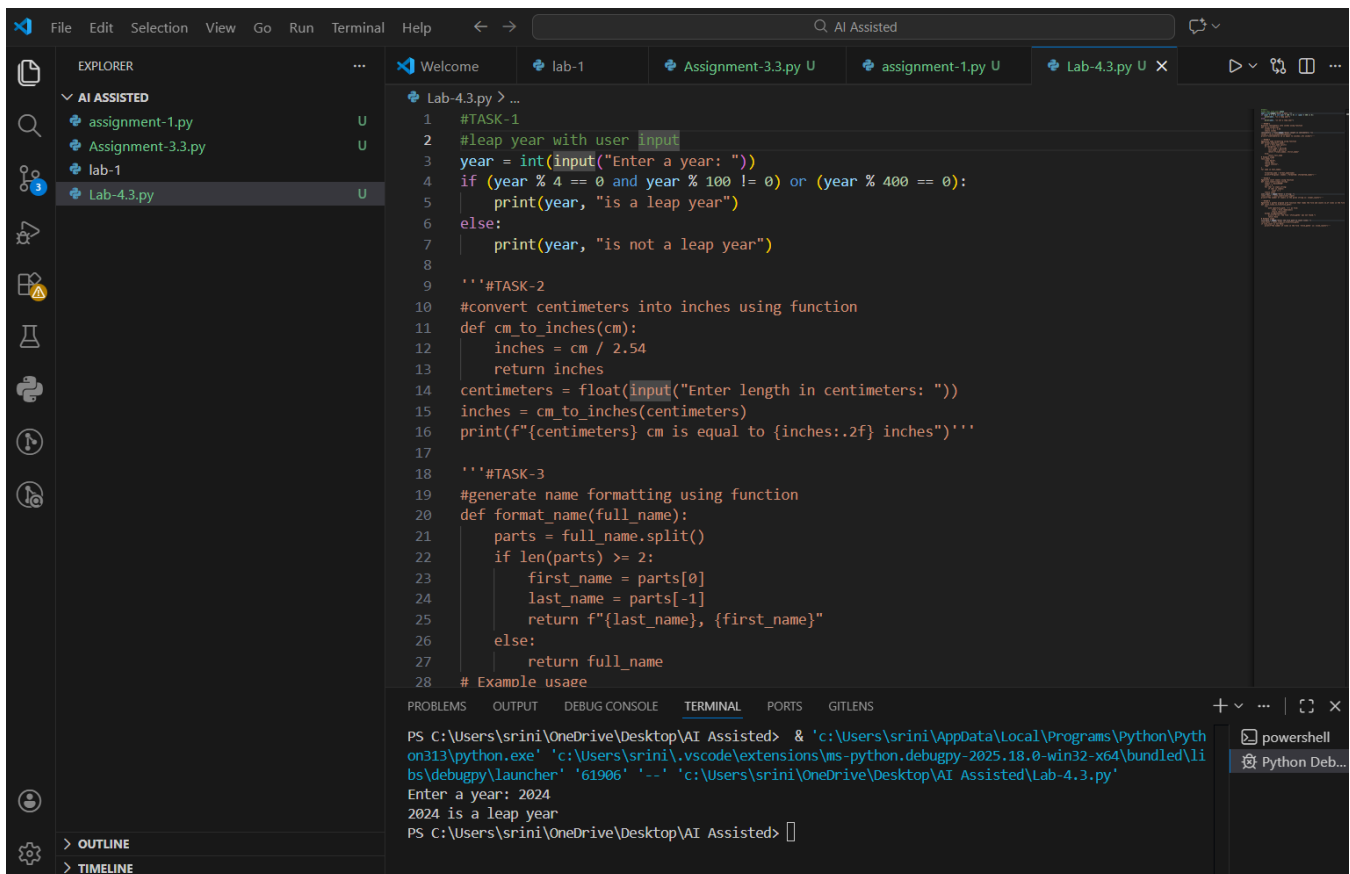
PROMPT:

LEAP YEAR WITH USER INPUT

CODE:

```
year = int(input("Enter a year: "))
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
    print(year, "is a leap year")
else:
    print(year, "is not a leap year")
```

OUTPUT:



The screenshot shows a VS Code editor with a Python file named 'Lab-4.3.py'. The code implements a function to check if a year is a leap year based on the prompt. The terminal output shows the program running and asking for a year, with '2024' entered as input, resulting in the output '2024 is a leap year'.

```
1 #TASK-1
2 #leap year with user input
3 year = int(input("Enter a year: "))
4 if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
5     print(year, "is a leap year")
6 else:
7     print(year, "is not a leap year")
8
9 '''TASK-2
10 #convert centimeters into inches using function
11 def cm_to_inches(cm):
12     inches = cm / 2.54
13     return inches
14 centimeters = float(input("Enter length in centimeters: "))
15 inches = cm_to_inches(centimeters)
16 print(f"{centimeters} cm is equal to {inches:.2f} inches")'''
17
18 '''TASK-3
19 #generate name formatting using function
20 def format_name(full_name):
21     parts = full_name.split()
22     if len(parts) >= 2:
23         first_name = parts[0]
24         last_name = parts[-1]
25         return f"{last_name}, {first_name}"
26     else:
27         return full_name
28 # Example usage
```

Terminal Output:

```
PS C:\Users\sriini\OneDrive\Desktop\AI Assisted> & 'c:\Users\sriini\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\sriini\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundle\libs\debugpy\launcher' '61906' '--' 'c:\Users\sriini\OneDrive\Desktop\AI Assisted\Lab-4.3.py'
Enter a year: 2024
2024 is a leap year
PS C:\Users\sriini\OneDrive\Desktop\AI Assisted>
```

EXPLANATION:

- The program asks the user to type a year and stores it as a number.
- It checks the leap year rule:

divisible by 4 **and not** divisible by 100, **or** divisible by 400.

- If the rule is true, it prints that the year is a leap year.
- If not, it prints that the year is not a leap year.

❖ TASK-2:

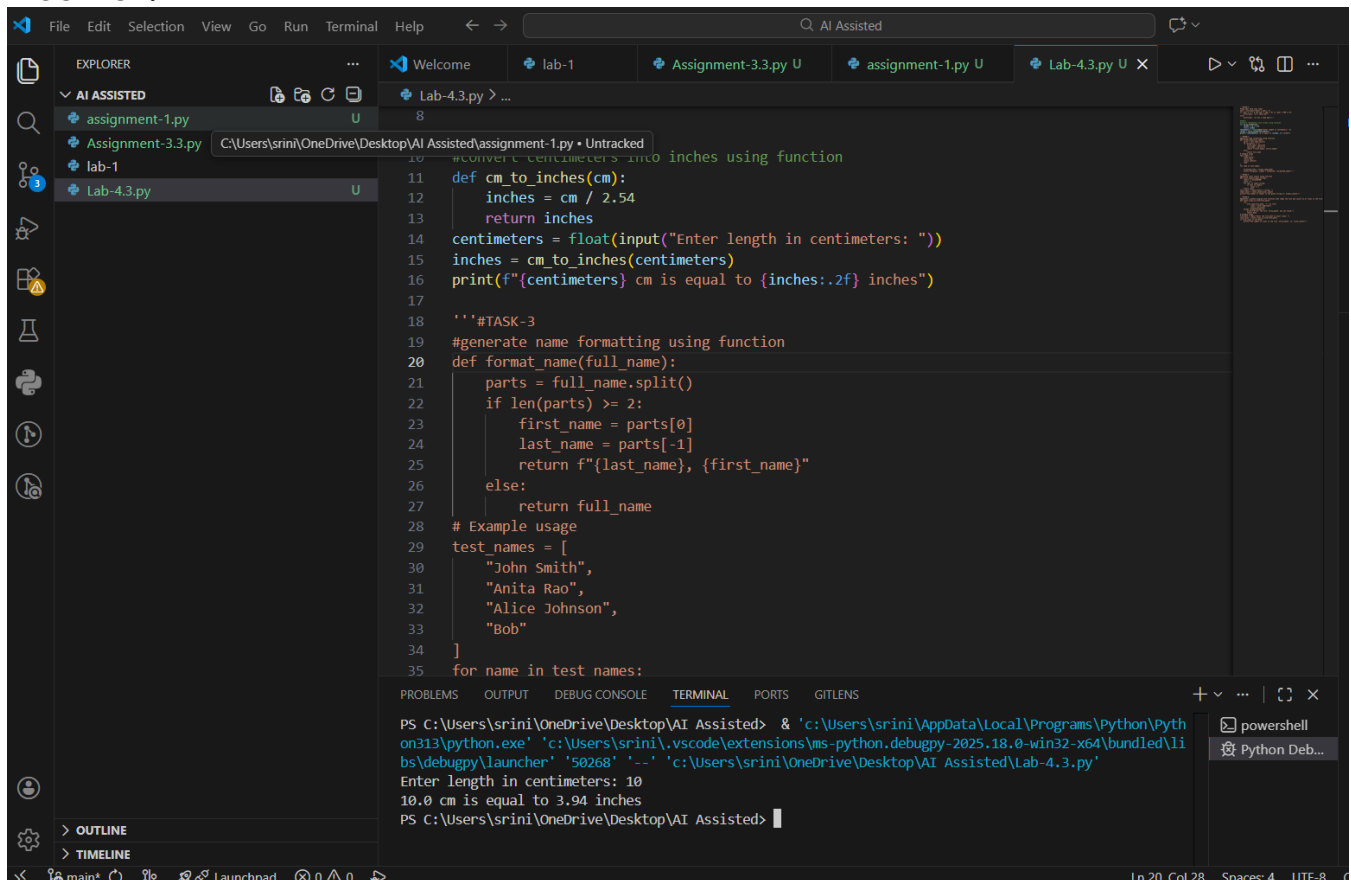
PROMPT:

convert centimeters into inches using function

CODE:

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

OUTPUT:



The screenshot shows a Visual Studio Code editor with a Python file named 'Lab-4.3.py' open. The code in the file is as follows:

```
10 #convert centimeters into inches using function  
11 def cm_to_inches(cm):  
12     inches = cm / 2.54  
13     return inches  
14 centimeters = float(input("Enter length in centimeters: "))  
15 inches = cm_to_inches(centimeters)  
16 print(f"{centimeters} cm is equal to {inches:.2f} inches")  
17  
18 '''#TASK-3  
19 #generate name formatting using function  
20 def format_name(full_name):  
21     parts = full_name.split()  
22     if len(parts) >= 2:  
23         first_name = parts[0]  
24         last_name = parts[-1]  
25         return f"{last_name}, {first_name}"  
26     else:  
27         return full_name  
28 # Example usage  
29 test_names = [  
30     "John Smith",  
31     "Anita Rao",  
32     "Alice Johnson",  
33     "Bob"  
34 ]  
35 for name in test_names:
```

The terminal at the bottom shows the execution of the script:

```
PS C:\Users\sri\OneDrive\Desktop\AI Assisted> & 'c:\Users\sri\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\sri\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '50268' '--' 'c:\Users\sri\OneDrive\Desktop\AI Assisted\Lab-4.3.py'  
Enter length in centimeters: 10  
10.0 cm is equal to 3.94 inches  
PS C:\Users\sri\OneDrive\Desktop\AI Assisted>
```

EXPLANATION:

- The function `cm_to_inches(cm)` uses the formula `cm / 2.54` to convert centimeters into inches.
- The program asks the user to enter a length in centimeters and stores it as a floating-point number.
- That value is passed into the function, which calculates and returns the result in inches.
- Finally, the program prints the conversion result, showing the answer rounded to two decimal places.

❖ TASK-3:

PROMPT:

Generate name formatting using function

CODE:

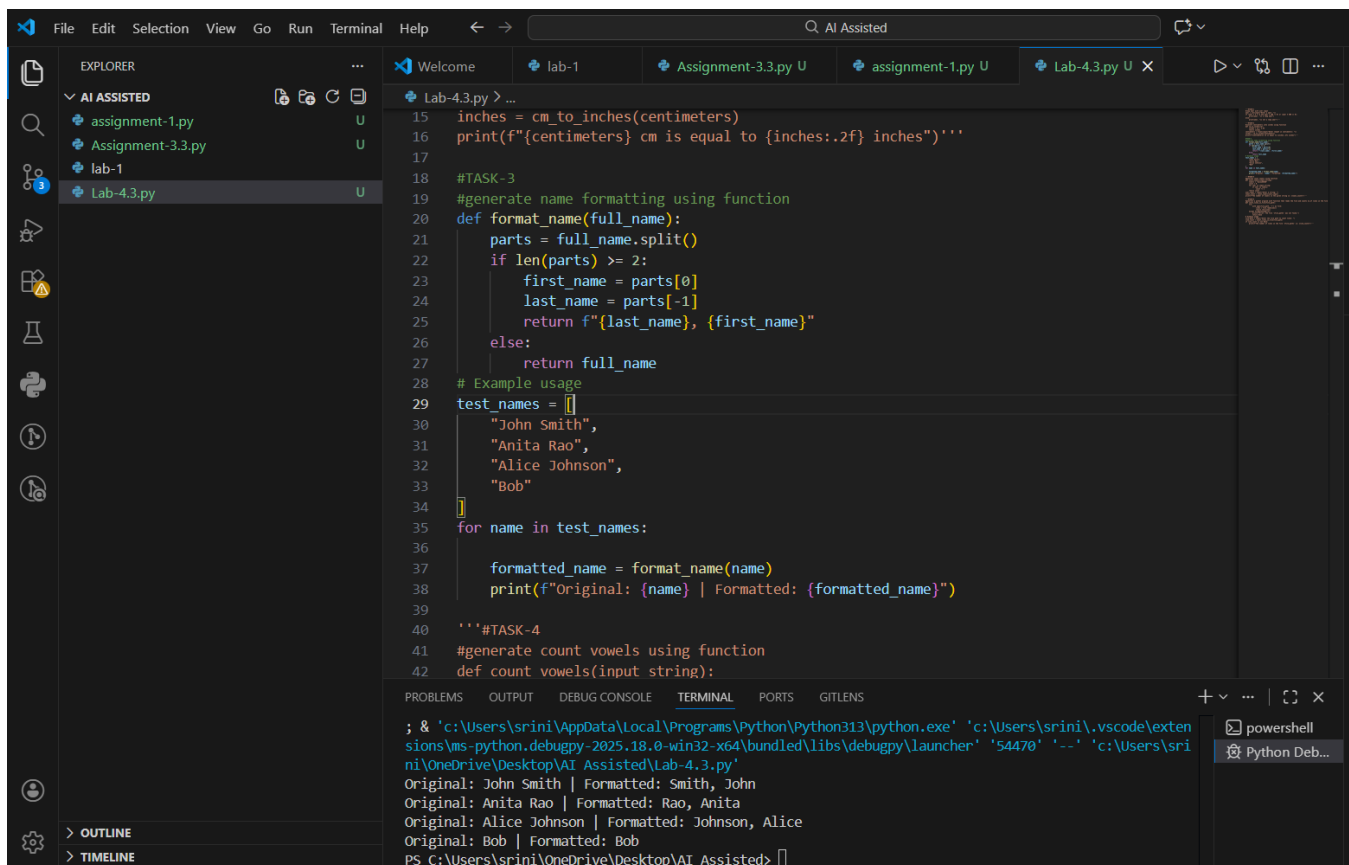
```
def format_name(full_name):
    parts = full_name.split()
    if len(parts) >= 2:
        first_name = parts[0]
        last_name = parts[-1]
        return f"{last_name}, {first_name}"
    else:
        return full_name

# Example usage
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:



```
15 inches = cm_to_inches(centimeters)
16 print(f"{centimeters} cm is equal to {inches:.2f} inches")'''
17
18 #TASK-3
19 #generate name formatting using function
20 def format_name(full_name):
21     parts = full_name.split()
22     if len(parts) >= 2:
23         first_name = parts[0]
24         last_name = parts[-1]
25         return f"{last_name}, {first_name}"
26     else:
27         return full_name
28 # Example usage
29 test_names = []
30     "John Smith",
31     "Anita Rao",
32     "Alice Johnson",
33     "Bob"
34 ]
35 for name in test_names:
36
37     formatted_name = format_name(name)
38     print(f"Original: {name} | Formatted: {formatted_name}")
39
40 '''#TASK-4
41 #generate count vowels using function
42 def count_vowels(input_string):
```

```
& 'c:\Users\sri\OneDrive\Desktop\AI Assisted\Lab-4.3.py'
Original: John Smith | Formatted: Smith, John
Original: Anita Rao | Formatted: Rao, Anita
Original: Alice Johnson | Formatted: Johnson, Alice
Original: Bob | Formatted: Bob
PS C:\Users\sri\OneDrive\Desktop\AI Assisted>
```

EXPLANATION:

- The function `format_name(full_name)` splits the name into words using spaces.
- If there are at least two words, it takes the first word as the first name and the last word as the last name, then rearranges them into "LastName, FirstName".
- If there is only one word (like "Bob"), it just returns the same name without changes.
- The program tests this function on a list of names and prints both the original and formatted versions.

❖ TASK-4:

PROMPT:

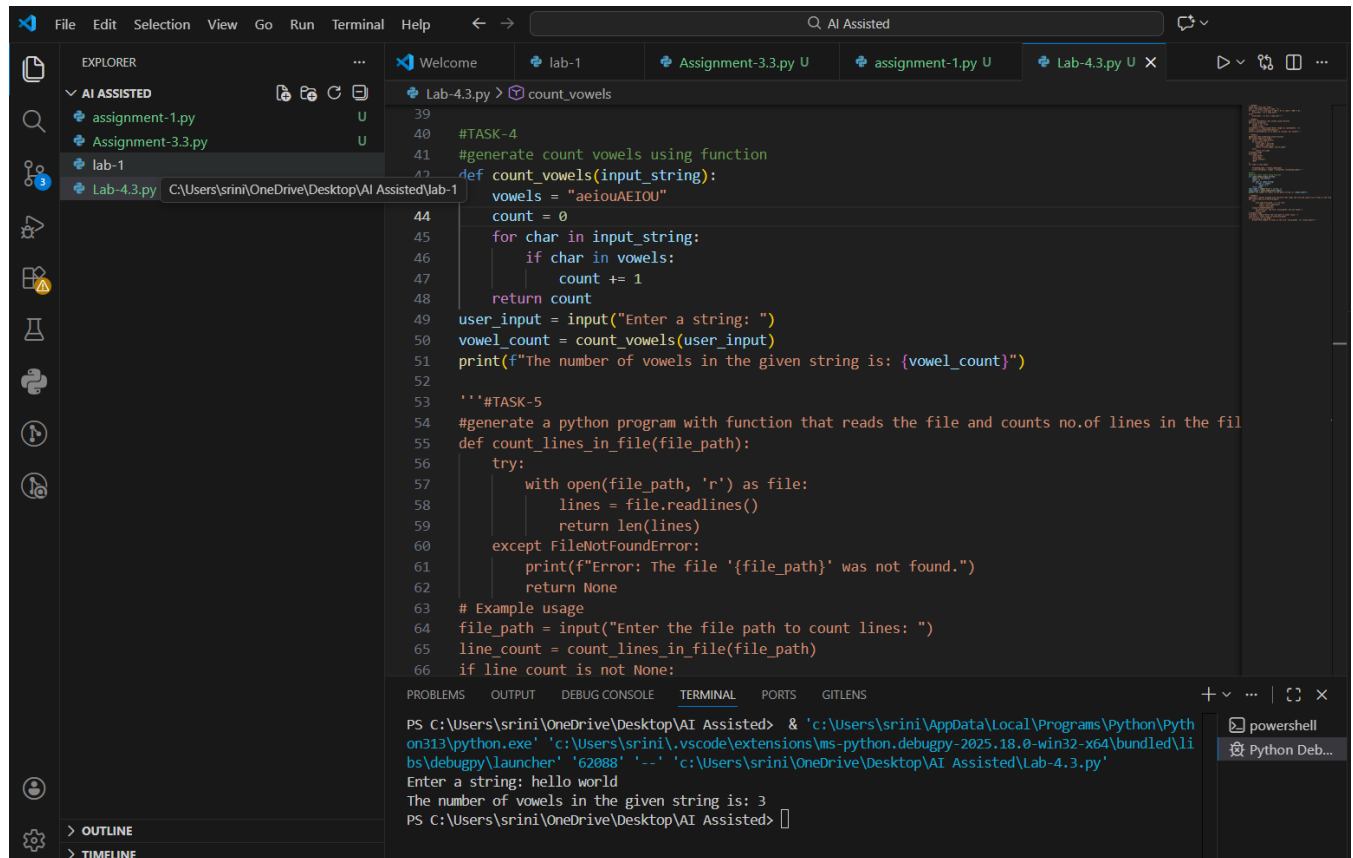
Generate count vowels using function

CODE:

```
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in input_string:
        if char in vowels:
            count += 1
    return count
user_input = input("Enter a string: ")
vowel_count = count_vowels(user_input)
```

```
print(f"The number of vowels in the given string is: {vowel_count}")
```

OUTPUT:



The screenshot shows a VS Code editor with a Python file named 'Lab-4.3.py'. The code defines a function 'count_vowels' that takes an input string and returns the count of vowels. It also includes a task for counting lines in a file. The terminal at the bottom shows the command prompt running the script, which prompts for a string and outputs the vowel count for 'hello world' as 3.

```
39
40 #TASK-4
41 #generate count vowels using function
42 def count_vowels(input_string):
43     vowels = "aeiouAEIOU"
44     count = 0
45     for char in input_string:
46         if char in vowels:
47             count += 1
48     return count
49 user_input = input("Enter a string: ")
50 vowel_count = count_vowels(user_input)
51 print(f"The number of vowels in the given string is: {vowel_count}")
52
53 '''#TASK-5
54 #generate a python program with function that reads the file and counts no.of lines in the file
55 def count_lines_in_file(file_path):
56     try:
57         with open(file_path, 'r') as file:
58             lines = file.readlines()
59             return len(lines)
60     except FileNotFoundError:
61         print(f"Error: The file '{file_path}' was not found.")
62         return None
63 # Example usage
64 file_path = input("Enter the file path to count lines: ")
65 line_count = count_lines_in_file(file_path)
66 if line_count is not None:
```

```
PS C:\Users\sринi\OneDrive\Desktop\AI Assisted> & 'c:\Users\sринi\AppData\Local\Programs\Python\Python313\python.exe' 'c:\Users\sринi\.vscode\extensions\ms-python.debugpy-2025.18.0-win32-x64\bundled\libs\debugpy\launcher' '62088' '--' 'c:\Users\sринi\OneDrive\Desktop\AI Assisted\Lab-4.3.py'
Enter a string: hello world
The number of vowels in the given string is: 3
PS C:\Users\sринi\OneDrive\Desktop\AI Assisted>
```

EXPLANATION:

- The function `count_vowels(input_string)` goes through each character of the given string.
- It checks if the character is a vowel (a, e, i, o, u in both uppercase and lowercase).
- If it is a vowel, the counter increases by 1.
- Finally, the program prints the total number of vowels found in the user's input 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:

```
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 with a Python file named `lab5.py` open. The code defines a function `count_lines_in_file` that reads a file and returns the number of lines. It includes an example usage section where the user is prompted to enter a file path. The terminal at the bottom shows the execution of the script, where the user entered `lab4.py` and the program correctly reported 200 lines.

```

60
61 #TASKS
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: 200
PS C:\Users\sriva\OneDrive\Desktop\AI Assisted>

```

EXPLANATION:

- The function `count_lines_in_file(file_path)` opens the file in read mode and reads all its lines.
- It counts the number of lines using `len(lines)` and returns that value.
- If the file is not found, it shows an error message instead of crashing.
- Finally, the program prints the total number of lines in the file entered by the user.

