

ASSIGNMENT-4.3

Prompt:

Generate a Python function that accepts a year as input and checks whether the given year is a leap year. The function should apply correct leap year conditions and return a suitable message or boolean result. Also provide sample input and output to demonstrate the function.

Code:

```
def is_leap_year(year):
    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
        return True
    else:
        return False

# Sample Input and Output
year = int(input("Enter a 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 code editor with the file '4-3lab.py' open. The code is identical to the one provided above. Below the code editor is a terminal window showing the execution of the script. The terminal output shows two test cases: entering '1010' which outputs '1010 is not a leap year.', and entering '2024' which outputs '2024 is a leap year.' The terminal interface includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS, and a sidebar with icons for powershell, Python, and Prettier.

```
4-3lab.py > ...
1 # Prompt:
2 # Generate a Python function that accepts a year as input and checks whether the given year is a leap year. The function should apply correct leap
year conditions and return a suitable message or boolean result. Also provide sample input and output to demonstrate the function.
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 # Sample Input and Output
9 year = int(input("Enter a year: "))
10 if is_leap_year(year):
11     print(f"{year} is a leap year.")
12 else:
13     print(f"{year} is not a leap year.")

PS C:\Users\HP\OneDrive\Desktop\ai assistent> [REDACTED]
Enter a year: 1010
1010 is not a leap year.
PS C:\Users\HP\OneDrive\Desktop\ai assistent> & c:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HP/OneDrive/Desktop/ai assistent/4-3lab.py"
PS C:\Users\HP\OneDrive\Desktop\ai assistent> [REDACTED]
Enter a year: 2024
2024 is a leap year.
PS C:\Users\HP\OneDrive\Desktop\ai assistent> [REDACTED]
```

Analysis:

The generated Python function correctly checks whether a given year is a leap year using proper conditions. It checks divisibility by 4, 100, and 400 to ensure accurate results. The function returns True for leap years and False for non-leap years. The sample input and output show that the program works correctly. This proves that zero-shot prompting can generate correct code without giving any examples.

Task 2:

Prompt:

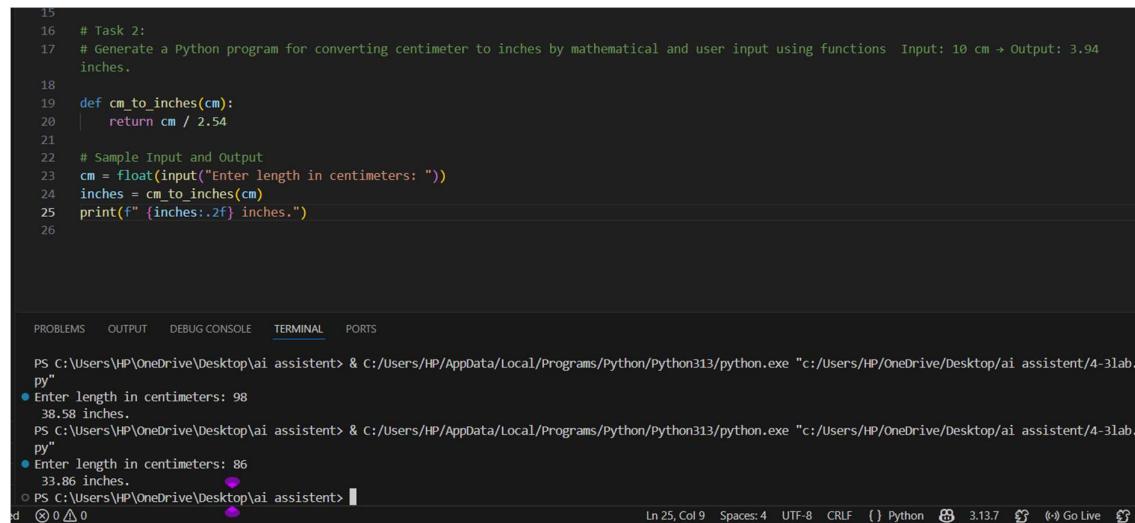
Generate a Python program for converting centimeter to inches by mathematical and user input using functions Input: 10 cm → Output: 3.94 inches.

Code:

```
def cm_to_inches(cm):
    return cm / 2.54

# Sample Input and Output
cm = float(input("Enter length in centimeters: "))
inches = cm_to_inches(cm)
print(f"{inches:.2f} inches.")
```

Output:



```
15
16  # Task 2:
17  # Generate a Python program for converting centimeter to inches by mathematical and user input using functions Input: 10 cm → Output: 3.94
18  # inches.
19  def cm_to_inches(cm):
20  |     return cm / 2.54
21
22  # Sample Input and Output
23  cm = float(input("Enter length in centimeters: "))
24  inches = cm_to_inches(cm)
25  print(f" {inches:.2f} inches.")

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\HP\OneDrive\Desktop\ai assistant> & c:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HP/OneDrive/Desktop/ai assistant/4-3lab.py"
● Enter length in centimeters: 98
38.58 inches.
PS C:\Users\HP\OneDrive\Desktop\ai assistant> & c:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HP/OneDrive/Desktop/ai assistant/4-3lab.py"
● Enter length in centimeters: 86
33.86 inches.
○ PS C:\Users\HP\OneDrive\Desktop\ai assistant>
```

Ln 25, Col 9 Spaces: 4 UTF-8 CRLF {} Python 3.13.7 ⚡ (↻) Go Live ⚡

Analysis:

The AI-generated Python function correctly converts centimeters to inches using the standard mathematical formula by dividing the value by 2.54. The function takes user input and returns the converted value accurately. The sample input of 98 cm produces the correct output of 38.58 inches. This shows that one-shot prompting helps the AI understand the task better by using a given example.

Task 3:

Develop a Python function that takes a full name as input and formats it in the form "Last, First". Examples: "John Smith" as "Smith, John" "Anita Rao" as "Rao, Anita" Based on these examples, show sample test cases.

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 "Invalid name format"

# Sample Test Cases
name1 = "John Smith"
name2 = "Anita Rao"
print(format_name(name1)) # Output: "Smith, John"
print(format_name(name2)) # Output: "Rao, Anita"
name3 = "Alice"
print(format_name(name3)) # Output: "Invalid name format"
```

Output:

The screenshot shows a code editor interface with a dark theme. At the top, there is a code block containing Python code for Task 3. Below the code, there is a terminal window showing the execution of the script and its output. The terminal window has tabs at the bottom labeled 'PROBLEMS', 'OUTPUT', 'DEBUG CONSOLE', 'TERMINAL' (which is selected), and 'PORTS'. The output in the terminal window is as follows:

```
py"
Enter length in centimeters: 86
33.86 inches.
PS C:\Users\HP\OneDrive\Desktop\ai assistent> & C:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HP/OneDrive/Desktop/ai assistent/4-3lab.py"
Smith, John
Rao, Anita
Invalid name format
PS C:\Users\HP\OneDrive\Desktop\ai assistent> 
```

At the bottom of the terminal window, there are status indicators: 'Ln 29, Col 153', 'Spaces: 4', 'UTF-8', 'CRLF', 'Python 3.13.7', and icons for Go Live and a refresh.

Task 4:

A)

Prompt:

Generate a Python function that takes a string as input and counts how many vowels are present in it. The function should return the vowel count and include sample input and output.

Code:

```
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = sum(1 for char in input_string if char in vowels)
    return count

# Sample Input and Output
test_string1 = "hello"
test_string2 = "education"

print(f"Number of vowels in '{test_string1}': {count_vowels(test_string1)}") # Output: 2
print(f"Number of vowels in '{test_string2}': {count_vowels(test_string2)}") # Output: 5
```

B)

Prompt:

Generate a Python function to count vowels in a string.Examples:Input: "hello" as Output: 2
Input: "education" as Output: 5 Using these examples, generate a clear and accurate
function with additional sample test cases.

Code:

```
def count_vowels(input_string):  
    vowels = "aeiouAEIOU"  
    count = sum(1 for char in input_string if char in vowels)  
    return count  
  
# Sample Test Cases  
  
test_string1 = "hello"  
test_string2 = "education"  
  
print(f"Number of vowels in '{test_string1}': {count_vowels(test_string1)}") # Output: 2  
print(f"Number of vowels in '{test_string2}': {count_vowels(test_string2)}") # Output: 5  
test_string3 = "Python Programming"  
print(f"Number of vowels in '{test_string3}': {count_vowels(test_string3)}") # Output: 4  
test_string4 = "OpenAI ChatGPT"  
print(f"Number of vowels in '{test_string4}': {count_vowels(test_string4)}") # Output: 5
```

Outputs:

The screenshot shows a Jupyter Notebook cell with the following Python code:

```
4-3lab.py > ...
48 # Generate a Python function that takes a string as input and counts how many vowels are present in it. The function should return the vowel count
49 def count_vowels(input_string):
50     vowels = "aeiouAEIOU"
51     count = sum(1 for char in input_string if char in vowels)
52     return count
53 # Sample Input and Output
54 test_string1 = "hello"
55 test_string2 = "education"
56 print(f"Number of vowels in '{test_string1}': {count_vowels(test_string1)}") # Output: 2
57 print(f"Number of vowels in '{test_string2}': {count_vowels(test_string2)}") # Output: 5
58 |
59 # Task4(B):
60 # Generate a Python function to count vowels in a string.Examples:Input: "hello" as Output: 2 Input: "education" as Output: 5 Using these
61 # examples, generate a clear and accurate function with additional sample test cases.
62 def count_vowels(input_string):
63     vowels = "aeiouAEIOU"
64     count = sum(1 for char in input_string if char in vowels)
65     return count
66 # Sample Test Cases
67 test_string1 = "hello"
68 test_string2 = "education"
69 print(f"Number of vowels in '{test_string1}': {count_vowels(test_string1)}") # Output: 2
70 print(f"Number of vowels in '{test_string2}': {count_vowels(test_string2)}") # Output: 5
71 test_string3 = "Python Programming"
72 print(f"Number of vowels in '{test_string3}': {count_vowels(test_string3)}") # Output: 4
73 test_string4 = "OpenAI ChatGPT"
74 print(f"Number of vowels in '{test_string4}': {count_vowels(test_string4)}") # Output: 5
```

The terminal output shows the results of running the code:

```
PS C:\Users\HP\OneDrive\Desktop\ai assistant> & C:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HP/OneDrive/Desktop/ai assistant/4-3lab.py"
Number of vowels in 'hello': 2
Number of vowels in 'education': 5
Number of vowels in 'hello': 2
Number of vowels in 'education': 5
Number of vowels in 'Python Programming': 4
Number of vowels in 'OpenAI ChatGPT': 5
PS C:\Users\HP\OneDrive\Desktop\ai assistant> []
```

Below the terminal output, the status bar indicates: Ln 58, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.7 ⚡ Go Live ⚡

Analysis:

In the zero-shot prompt, the AI generated a function to count vowels in a string without giving any examples. The function checks each character and counts vowels correctly. The sample inputs like “hello” and “education” gave correct results.

In the few-shot prompt, by giving examples, the AI generated the same function but added more test cases. The outputs were accurate and easy to understand. The few-shot method gave clearer results and better demonstration.

Overall, both prompts worked correctly, but few-shot prompting was more helpful because it provided more examples and clearer output.

Task5:

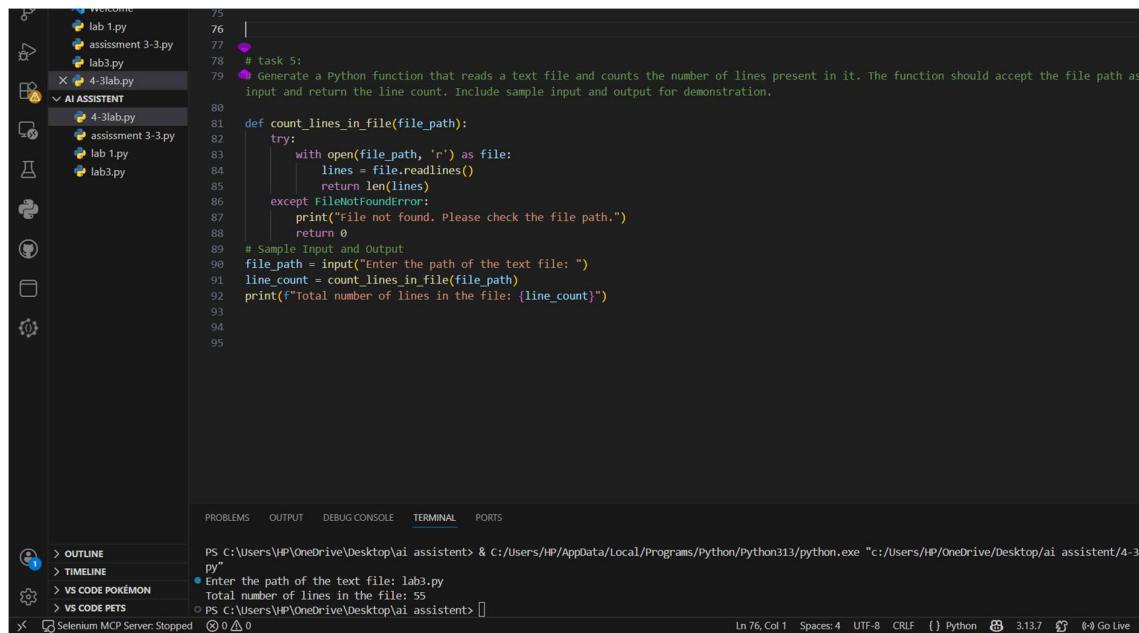
Prompt:

Generate a Python function that reads a text file and counts the number of lines present in it. The function should accept the file path as input and return the line count. Include sample input and output for demonstration.

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("File not found. Please check the file path.")
        return 0
# Sample Input and Output
file_path = input("Enter the path of the text file: ")
line_count = count_lines_in_file(file_path)
print(f"Total number of lines in the file: {line_count}")
```

Output:



The screenshot shows the Visual Studio Code interface with the AI Assistant extension active. The left sidebar displays a project structure with files like lab1.py, lab3.py, and assignment 3-3.py. The main editor area shows a Python function named `count_lines_in_file` that reads a file and counts its lines. A task description at the top of the code asks for a function that reads a text file and counts lines. The code also includes sample input and output code. The terminal at the bottom shows the command being run and the resulting output: "Total number of lines in the file: 55".

```
76 |
77 |
78 # task 5:
79 # Generate a Python function that reads a text file and counts the number of lines present in it. The function should accept the file path as
80 # input and return the line count. Include sample input and output for demonstration.
81 def count_lines_in_file(file_path):
82     try:
83         with open(file_path, 'r') as file:
84             lines = file.readlines()
85             return len(lines)
86     except FileNotFoundError:
87         print("File not found. Please check the file path.")
88         return 0
89 # Sample Input and Output
90 file_path = input("Enter the path of the text file: ")
91 line_count = count_lines_in_file(file_path)
92 print(f"Total number of lines in the file: {line_count}")
93
94
95
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

PS C:\Users\HP\OneDrive\Desktop\ai assistant> & C:/Users/HP/AppData/Local/Programs/Python/Python313/python.exe "c:/Users/HP/OneDrive/Desktop/ai assistant/4-3
ipy"
● Enter the path of the text file: lab3.py
Total number of lines in the file: 55
○ PS C:\Users\HP\OneDrive\Desktop\ai assistant>

Selenium MCP Server: Stopped ⑧ 0 △ 0 Ln 76, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.13.7 ⚡ Go Live

Analysis:

The AI-generated Python function correctly reads a text file and counts the number of lines present in it. It uses file handling to open the file and read all lines, then returns the total count. The function also handles the case where the file is not found by showing an error message. The sample input and output clearly demonstrate how the program works. This shows that few-shot prompting can generate accurate and practical file handling code.