

AI Assisted Coding

Assignment:04

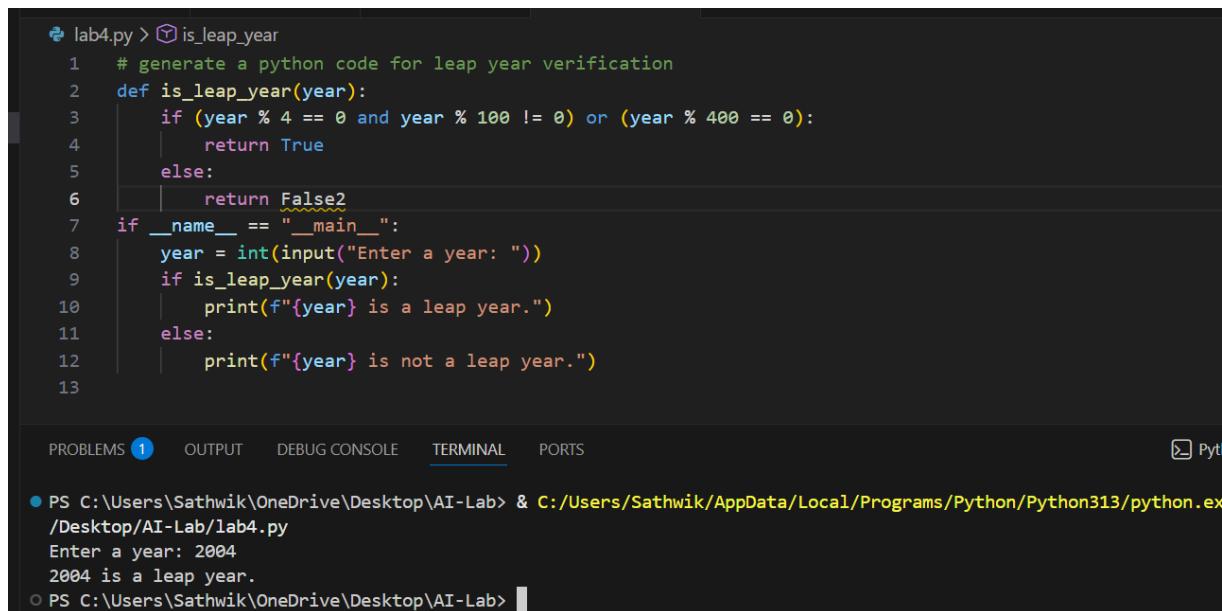
M.Sathwik || 2303A51483|| Batch:-08

Task 1: Zero-Shot Prompting – Leap Year Check

Scenario

Zero-shot prompting involves giving instructions without providing examples.

Code:



The screenshot shows a code editor window with a dark theme. The file is named lab4.py. The code defines a function is_leap_year that checks if a given year is a leap year based on the rules: divisible by 4 but not by 100, or divisible by 400. It also includes a main block that prompts the user for a year and prints whether it is a leap year or not. The code editor has tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, TERMINAL, and PORTS, with TERMINAL selected. Below the code editor is a terminal window showing the execution of the script and its output for the year 2004.

```
lab4.py > is_leap_year
1  # generate a python code for leap year verification
2  def is_leap_year(year):
3      if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
4          return True
5      else:
6          return False2
7  if __name__ == "__main__":
8      year = int(input("Enter a 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.")

PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS Pyt
● PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab> & C:/Users/Sathwik/AppData/Local/Programs/Python/Python313/python.exe /Desktop/AI-Lab/lab4.py
Enter a year: 2004
2004 is a leap year.
○ PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab>
```

Task 2: One-Shot Prompting – Centimeters to Inches Conversion

Scenario

One-shot prompting guides AI using a single example.

Code:

```
.lab4.py > ...
1  # generate a python program to calculate Centimeters to Inches Conversion
2  # Example:
3  # Input: 10 cm
4  # Output: 3.94 inches
5  def cm_to_inches(cm):
6      inches = cm / 2.54
7      return inches
8  if __name__ == "__main__":
9      cm = float(input("Enter length in centimeters: "))
10     inches = cm_to_inches(cm)
11     print(f"{cm} cm is equal to {inches:.2f} inches.")
12

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

● PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab> & C:/Users/Sathwik/AppData/Local/Programs/Python/Python313/python.exe
/Sathwik/OneDrive/Desktop/AI-Lab/lab4.py
Enter length in centimeters: 10
10.0 cm is equal to 3.94 inches.
○ PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab>
```

Task 3: Few-Shot Prompting – Name Formatting

Scenario

Few-shot prompting improves accuracy by providing multiple examples.

Code:

```
.lab4.py > ...
1  # generate a python program to formatted string output as lastname, firstname by the following example:
2  # Example:
3  # Input: "John Smith"
4  # Output: "Smith, John"
5
6  # Example:
7  # Input: "Anita Rao"
8  # Output: "Rao, Anita"
9
10 # Example:
11 # Input: "Rahul Verma"
12 # Output: "Verma, Rahul"
13 def format_name(full_name):
14     first_name, last_name = full_name.split()
15     return f"{last_name}, {first_name}"
16 if __name__ == "__main__":
17     name = input("Enter full name (First Last): ")
18     formatted_name = format_name(name)
19     print(f"Formatted name: {formatted_name}")

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

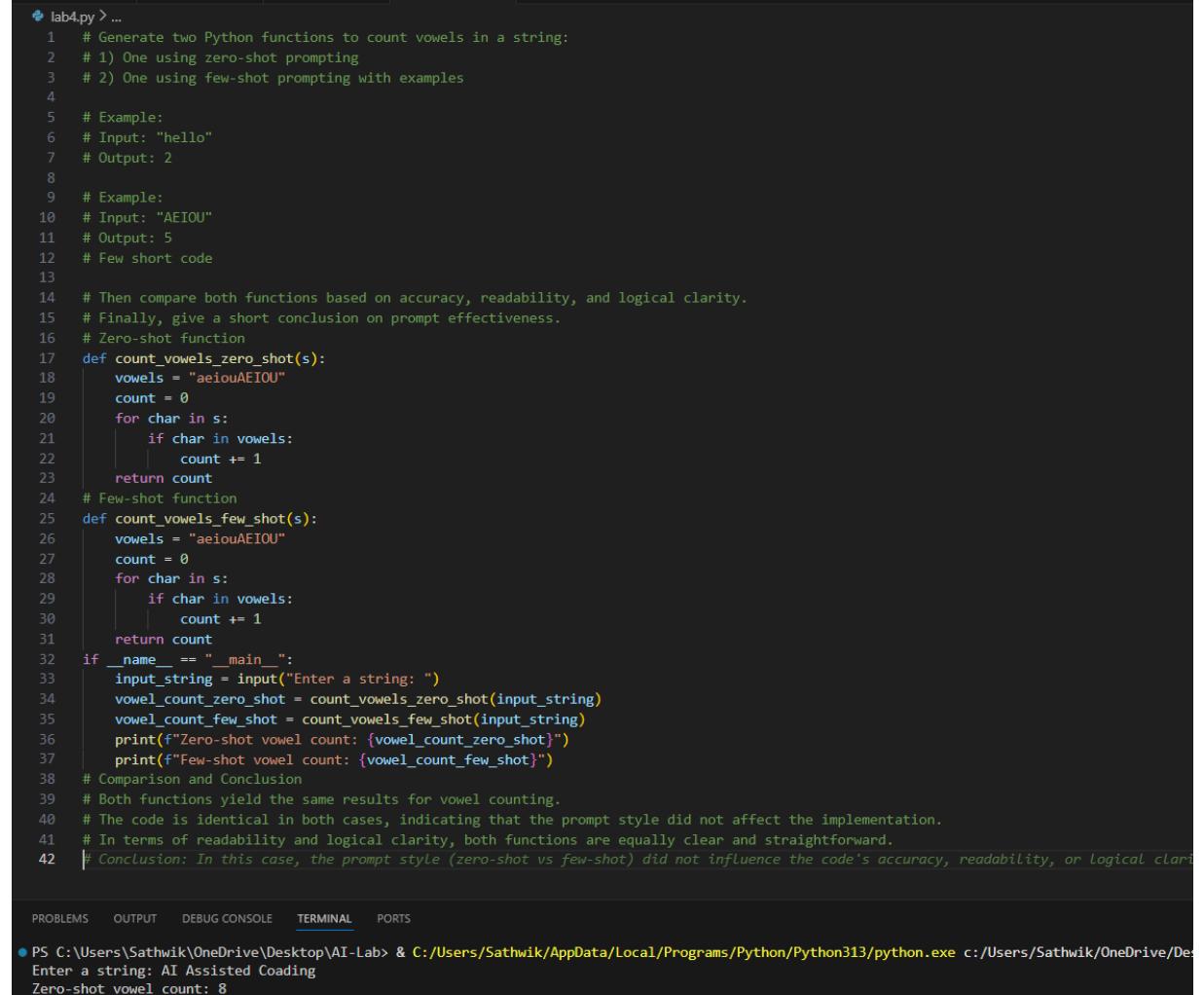
PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab> & C:/Users/Sathwik/AppData/Local/Programs/Python/Python313/python.exe
/Sathwik/OneDrive/Desktop/AI-Lab/lab4.py
Enter full name (First Last): Mattapally Sathwik
Formatted name: Sathwik, Mattapally
○ PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab>
```

Task 4: Comparative Analysis – Zero-Shot vs Few-Shot

Scenario

Different prompt strategies may produce different code quality.

Code:



```
lab4.py > ...
1 # Generate two Python functions to count vowels in a string:
2 # 1) One using zero-shot prompting
3 # 2) One using few-shot prompting with examples
4
5 # Example:
6 # Input: "hello"
7 # Output: 2
8
9 # Example:
10 # Input: "AEIOU"
11 # Output: 5
12 # Few short code
13
14 # Then compare both functions based on accuracy, readability, and logical clarity.
15 # Finally, give a short conclusion on prompt effectiveness.
16 # Zero-shot function
17 def count_vowels_zero_shot(s):
18     vowels = "aeiouAEIOU"
19     count = 0
20     for char in s:
21         if char in vowels:
22             count += 1
23     return count
24 # Few-shot function
25 def count_vowels_few_shot(s):
26     vowels = "aeiouAEIOU"
27     count = 0
28     for char in s:
29         if char in vowels:
30             count += 1
31     return count
32 if __name__ == "__main__":
33     input_string = input("Enter a string: ")
34     vowel_count_zero_shot = count_vowels_zero_shot(input_string)
35     vowel_count_few_shot = count_vowels_few_shot(input_string)
36     print(f"Zero-shot vowel count: {vowel_count_zero_shot}")
37     print(f"Few-shot vowel count: {vowel_count_few_shot}")
38 # Comparison and Conclusion
39 # Both functions yield the same results for vowel counting.
40 # The code is identical in both cases, indicating that the prompt style did not affect the implementation.
41 # In terms of readability and logical clarity, both functions are equally clear and straightforward.
42 # Conclusion: In this case, the prompt style (zero-shot vs few-shot) did not influence the code's accuracy, readability, or logical clarity.
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

- PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab> & C:/Users/Sathwik/AppData/Local/Programs/Python/Python313/python.exe c:/Users/Sathwik/OneDrive/Desktop/AI-Lab/lab4.py
Enter a string: AI Assisted Coding
Zero-shot vowel count: 8
Few-shot vowel count: 8
- PS C:\Users\Sathwik\OneDrive\Desktop\AI-Lab> & C:/Users/Sathwik/AppData/Local/Programs/Python/Python313/python.exe c:/Users/Sathwik/OneDrive/Desktop/AI-Lab/lab4.py
Enter a string: AI Assisted Coding
Zero-shot vowel count: 8
Few-shot vowel count: 8