

AI Assisted Coding

P.Manikanta | | 2303A51271 | | Batch:- 8

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 with a Python script named `lab 4.3.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 then prompts the user for a year and prints whether it is a leap year or not. The terminal below shows the script being run and a non-leap year being checked.

```
lab 4.3.py > ...
1  def is_leap_year(year):
2      if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
3          return True
4      else:
5          return False
6
7
8  if __name__ == "__main__":
9      year = int(input("Enter a year: "))
10
11     if is_leap_year(year):
12         print(f"{year} is a leap year.")
13     else:
14         print(f"{year} is not a leap year.")

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS    +
```

- PS C:\Users\porika manikanta\OneDrive\Desktop\ai assistant> & "C:/Program Files/Python311/python.exe" "c:/Users/porika manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py"
Enter a year: 2026
2026 is not a leap year.
- PS C:\Users\porika manikanta\OneDrive\Desktop\ai assistant>

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

Scenario

One-shot prompting guides AI using a single example.

Code:

```
5      # Generate a python program to calculate Centimeters to Inches
6
7      # Example:
8      # Input: 10 cm
9      # Output: 3.94 inches
10
11     def cm_to_inches(cm):
12         inches = cm / 2.54
13         return inches
14
15
16
17     if __name__ == "__main__":
18         cm = float(input("Enter length in centimeters: "))
19         inches = cm_to_inches(cm)
20         print(f"{cm} cm is equal to {inches:.2f} inches.")
21
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
C:\Users\porika manikanta\OneDrive\Desktop\ai assistant> & "C:/Program Files/Python310/python.exe" "c:/Users/porika manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py"
Enter length in centimeters: Traceback (most recent call last):
File "c:/Users/porika manikanta/OneDrive\Desktop\ai assistant\lab 4.3.py", line 19
    cm = float(input("Enter length in centimeters: "))
                                ^
KeyboardInterrupt
C:\Users\porika manikanta\OneDrive\Desktop\ai assistant> & "C:/Program Files/Python310/python.exe" "c:/Users/porika manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py"
Enter length in centimeters: 10
10.0 cm is equal to 3.94 inches.
C:\Users\porika manikanta\OneDrive\Desktop\ai assistant>
```

Task 3: Few-Shot Prompting – Name Formatting

Scenario

Few-shot prompting improves accuracy by providing multiple examples. **Code:**

```
31  # Generate a python program to format string output as "lastname, firstname"
32
33
34  # Example:
35  # Input: "John Smith"
36  # Output: "Smith, John"
37
38  def format_name(full_name):
39      first_name, last_name = full_name.split()
40      return f"{last_name}, {first_name}"
41
42
43  if __name__ == "__main__":
44      name = input("Enter full name (First Last): ")
45      formatted_name = format_name(name)
46      print(f"Formatted name: {formatted_name}")
47
48
```

The screenshot shows a terminal window with the following content:

- Terminal tab is selected.
- Command: PS C:\Users\porika manikanta\OneDrive\Desktop\ai assistant> & "C:/Program Files/Python311/python "c:/Users/porika manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py"
- Output:
 - File "c:/Users/porika manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py", line 39, in format_name
 first_name, last_name = full_name.split()
 ^^^^^^^^^^
 - ValueError: not enough values to unpack (expected 2, got 1)
 - PS C:\Users\porika manikanta\OneDrive\Desktop\ai assistant> & "C:/Program Files/Python311/python "c:/Users/porika manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py"
 - Enter full name (First Last): porika manikanta
 - Formatted name: manikanta, porika
- PS C:\Users\porika manikanta\OneDrive\Desktop\ai assistant>

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

Scenario

Different prompt strategies may produce different code quality.

Code:

```
8 # 1) One using zero-shot prompting
9 # 2) One using few-shot prompting with examples
0
1 # Example:
2 # Input: "hello"
3 # Output: 2
4
5 # Example:
6 # Input: "AEIOU"
7 # Output: 5
8
9
0 # Zero-shot function
1 def count_vowels_zero_shot(s):
2     vowels = "aeiouAEIOU"
3     count = 0
4     for ch in s:
5         if ch in vowels:
6             count += 1
7     return count
8
9
0 # Few-shot function
1 def count_vowels_few_shot(s):
2     vowels = "aeiouAEIOU"
3     count = 0
4     for ch in s:
5         if ch in vowels:
6             count += 1
7     return count
8
9
0 if __name__ == "__main__":
1     input_string = input("Enter a string: ")
2
3     vowel_count_zero_shot = count_vowels_zero_shot(input_string)
```

```
lab 4.3.py > count_vowels_few_shot
61  def count_vowels_zero_shot(s):
62      return count
63
64
65  # Few-shot function
66  def count_vowels_few_shot(s):
67      vowels = "aeiouAEIOU"
68      count = 0
69
70      for ch in s:
71          if ch in vowels:
72              count += 1
73
74  return count
75
76
77
78
79
80 if __name__ == "__main__":
81     input_string = input("Enter a string: ")
82
83     vowel_count_zero_shot = count_vowels_zero_shot(input_string)
84     vowel_count_few_shot = count_vowels_few_shot(input_string)
85
86     print(f"Zero-shot vowel count: {vowel_count_zero_shot}")
87     print(f"Few-shot vowel count: {vowel_count_few_shot}")
88
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
"c:/Users/porika/manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py"
Enter a string: AI Assisted coding
Zero-shot vowel count: 7
"c:/Users/porika/manikanta/OneDrive/Desktop/ai assistant/lab 4.3.py"
Enter a string: AI Assisted coding
Zero-shot vowel count: 7
Zero-shot vowel count: 7
Few-shot vowel count: 7
PS C:\Users\porika\manikanta\OneDrive\Desktop\ai assistant>
```

```
Few-shot vowel count: 7
```