Course Title : AI Assisted Coding

Name : B.Thanmai

Batch : 33

Hall No : 2303a52126

Lab 4: Advanced Prompt Engineering – Zero-shot, One-shot, and Few-shot Techniques

Task Description-1

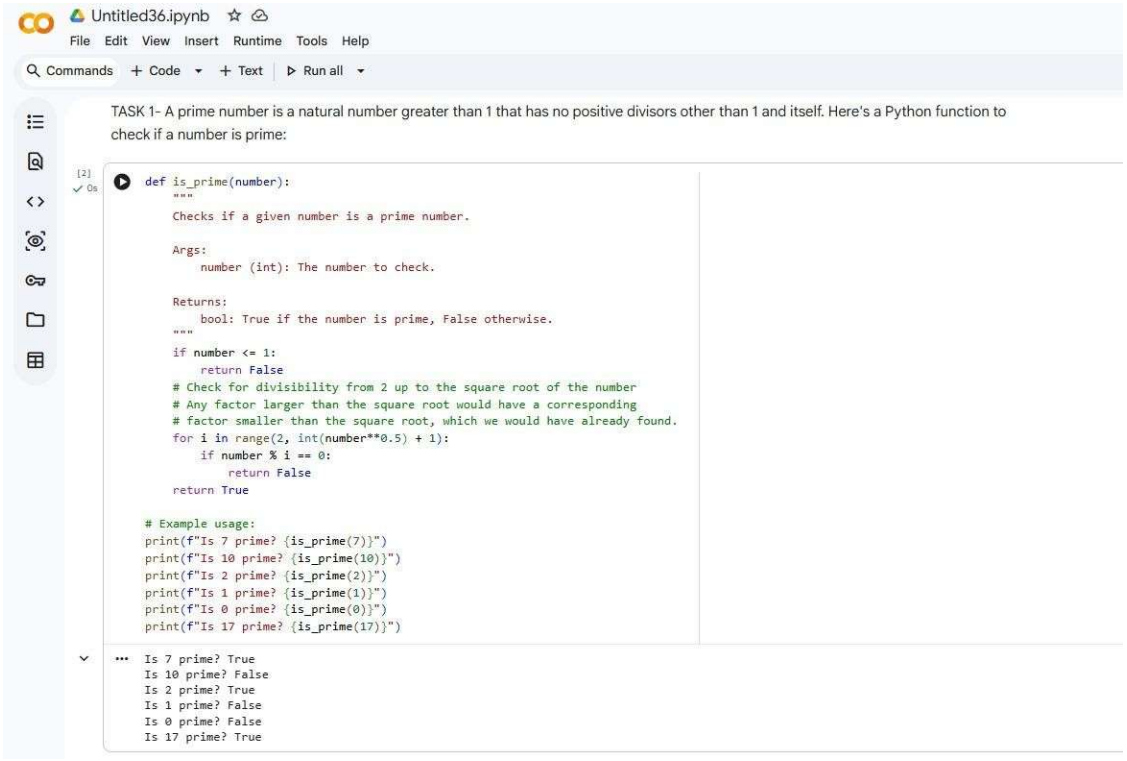
• Zero-shot: Prompt AI with only the instruction. Write a Python function to determine

whether a given number is prime

Expected Output-1

• A basic Python function to check if a number is prime, demonstrating correct logical

conditions without relying on examples or additional context



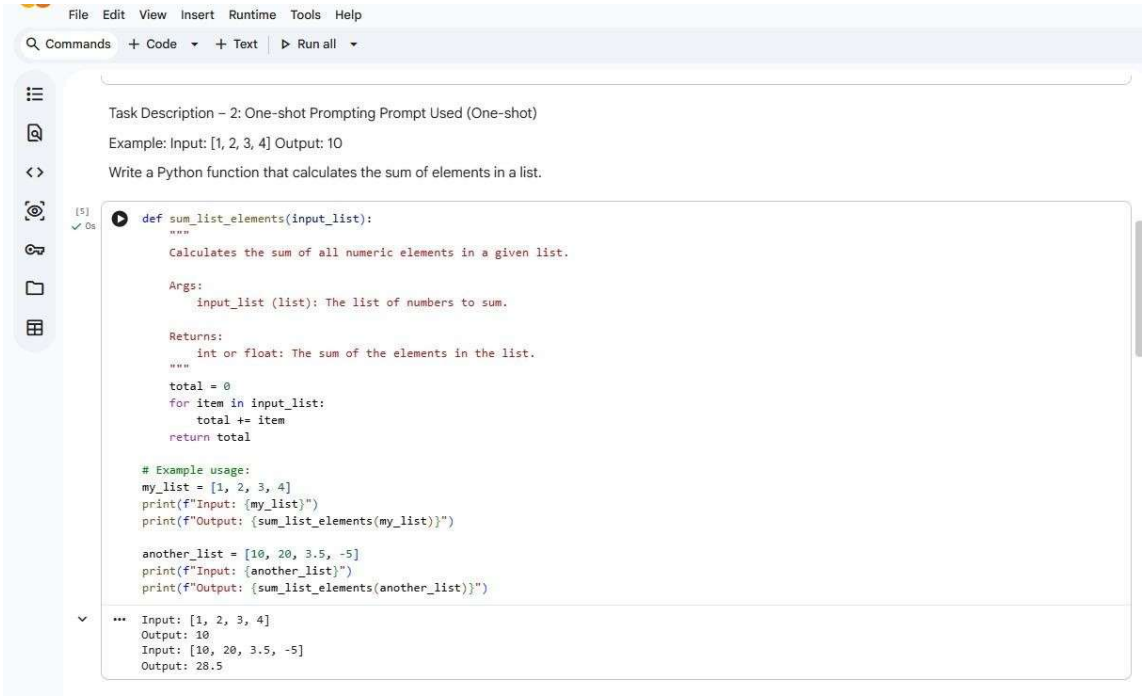
Task Description-2

• One-shot: Provide one example: Input: [1, 2, 3, 4], Output: 10 to help AI generate a

function that calculates the sum of elements in a list.

Expected Output-2

• A correct conversion function guided by the single example.



Task Description-3

• Few-shot: Give 2–3 examples to create a function that extracts digits from an

alphanumeric string.

Expected Output-3

• Accurate function that returns only the digits from alphanumeric string.

Commands + Code + Text ▷ Run all

Task Description – 3: Few-shot Prompting

Prompt Used (Few-shot)

Examples: Input: "a1b2c3" → Output: "123" Input: "x9y8" → Output: "98" Input: "abc123" → Output: "123"

Write a function to extract digits from an alphanumeric string.

[11] ✓ 0s

```
def extract_digits(input_string):
    """
    Extracts all digits from an alphanumeric string.

    Args:
        input_string (str): The alphanumeric string to process.

    Returns:
        str: A string containing only the digits found in the input string.
    """
    digits = ""
    for char in input_string:
        if char.isdigit():
            digits += char
    return digits

# Example usage:
print(f'Input: "a1b2c3" → Output: "{extract_digits("a1b2c3")}"')
print(f'Input: "x9y8" → Output: "{extract_digits("x9y8")}"')
print(f'Input: "abc123" → Output: "{extract_digits("abc123")}"')
print(f'Input: "NoDigitsHere" → Output: "{extract_digits("NoDigitsHere")}"')
print(f'Input: "12345" → Output: "{extract_digits("12345")}"')
print(f'Input: "" → Output: "{extract_digits("")}"')
```

```
Input: "a1b2c3" → Output: "123"
Input: "x9y8" → Output: "98"
Input: "abc123" → Output: "123"
Input: "NoDigitsHere" → Output: ""
Input: "12345" → Output: "12345"
Input: "" → Output: ""
```

Task Description-4

- **Compare zero-shot vs few-shot prompting for generating a function that counts the**

number of vowels in a string.

Expected Output-4

- **Output comparison + student explanation on how examples helped the model.**

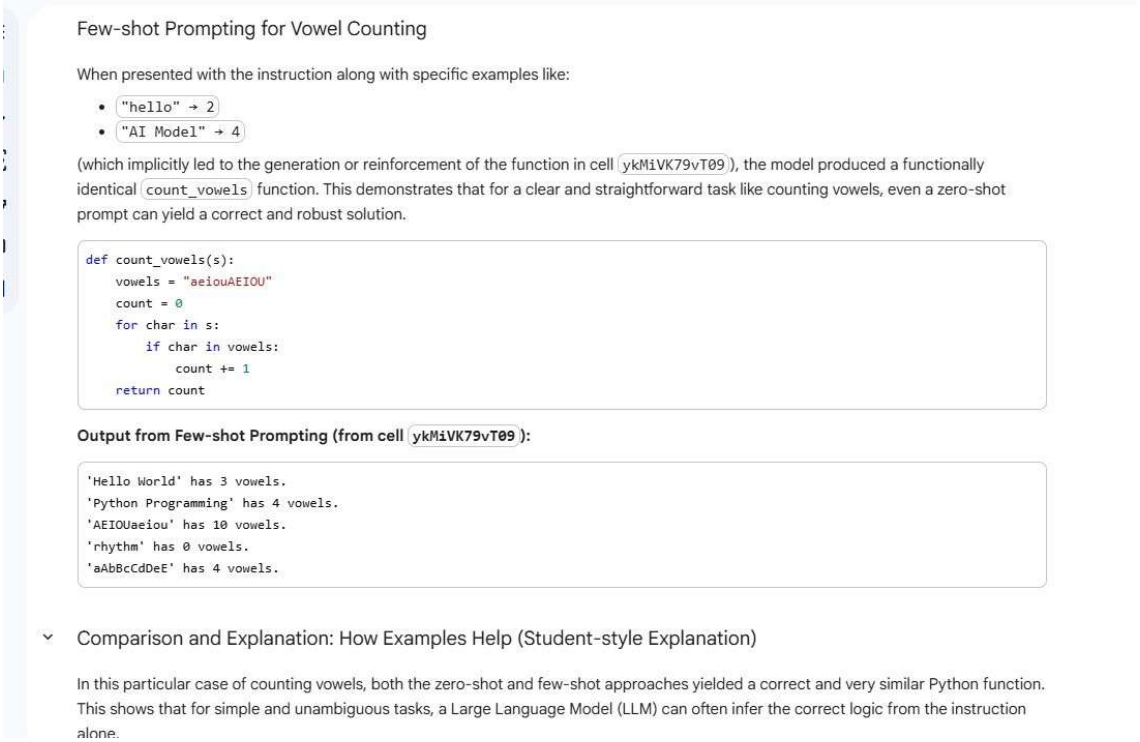
Zero-shot Prompting for Vowel Counting

When given only the instruction "Write a Python function to count the number of vowels in a string" (a zero-shot prompt, meaning no examples were provided), the model generated the following function (as seen in cell `33a537c9`):

```
def count_vowels(input_string):
    vowels = "aeiouAEIOU"
    count = 0
    for char in input_string:
        if char in vowels:
            count += 1
    return count
```

Output from Zero-shot Prompting (from cell `33a537c9`):

```
'Hello World' has 3 vowels.
'Python Programming' has 4 vowels.
'AEIOUaeiou' has 10 vowels.
'rhythm' has 0 vowels.
```



Task Description-5

• Use few-shot prompting with 3 sample inputs to generate a function that determines

the minimum of three numbers without using the built-in min() function.

Expected Output-5

• A function that handles all cases with correct logic based on example patterns.

Commands + Code + Text ▷ Run all

Task Description – 5: Few-shot Prompting (Minimum of Three Numbers)

Prompt Used (Few-shot)

Examples: (3, 5, 1) → 1 (10, 2, 7) → 2 (4, 4, 9) → 4

Write a function to find the minimum of three numbers without using `min()`.

[32] ✓ 0s

```python
def find_minimum_of_three(a, b, c):
    """
    Finds the minimum of three numbers without using the built-in min() function.

    Args:
        a (int or float): The first number.
        b (int or float): The second number.
        c (int or float): The third number.

    Returns:
        int or float: The minimum of the three numbers.
    """
    minimum = a
    if b < minimum:
        minimum = b
    if c < minimum:
        minimum = c
    return minimum

# Example usage based on the prompt:
print(f'({3}, {5}, {1}) → {find_minimum_of_three(3, 5, 1)}')
print(f'({10}, {2}, {7}) → {find_minimum_of_three(10, 2, 7)}')
print(f'({4}, {4}, {9}) → {find_minimum_of_three(4, 4, 9)}')

# Additional examples:
print(f'({-1}, {0}, {1}) → {find_minimum_of_three(-1, 0, 1)}')
print(f'({5.5}, {2.1}, {8.0}) → {find_minimum_of_three(5.5, 2.1, 8.0)}')
print(f'({7}, {7}, {7}) → {find_minimum_of_three(7, 7, 7)}')
```

```
(3, 5, 1) → 1
(10, 2, 7) → 2
(4, 4, 9) → 4
(-1, 0, 1) → -1
(5.5, 2.1, 8.0) → 2.1
(7, 7, 7) → 7
```