Lab-4.2

Course : AI Assisted Coding

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

Name	: Mohammed Sabir
Hall Ticket No	: 2303A51506
Batch	: 22

Task-01

Final Optimal Prompt:

write a python code to check the number is prime or not, but the starting number to ending number is user should give and print the prime numbers between the range

also handle the error when user give neagative input or string input

---

Code Screenshot :

```python
# also handle the error when user give neagative input or string input
import sys
def is_prime(n: int) -> bool:
    if n <= 1:
        return False
    for i in range(2, int(n**0.5) + 1):
        if n % i == 0:
            return False
    return True
try:
    start = int(input("Enter the starting number: "))
    end = int(input("Enter the ending number: "))
    prime_count = 0
    for i in range(start, end + 1):
        if is_prime(i):
            prime_count += 1
            print(f"{i} is a prime number.")
        else:
            print(f"{i} is not a prime number.")
    print(f"Total prime numbers between {start} and {end} are {prime_count}.")
except ValueError:
    print("Invalid input. Please enter a valid integer.")
    sys.exit(1)
except Exception as e:
    print(f"An error occurred: {e}")
    sys.exit(1)
```

Output Screenshot:

```
Enter the starting number: 5
Enter the ending number: 20
5 is a prime number.
6 is not a prime number.
7 is a prime number.
8 is not a prime number.
9 is not a prime number.
10 is not a prime number.
11 is a prime number.
12 is not a prime number.
13 is a prime number.
14 is not a prime number.
15 is not a prime number.
16 is not a prime number.
17 is a prime number.
18 is not a prime number.
19 is a prime number.
20 is not a prime number.
Total prime numbers between 5 and 20 are 6.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>
```

Explanation/Justification/Observation (100 words / 5 – 6 sentence) :

The program checks and counts prime numbers within a given range. The is_prime() function efficiently determines whether a number is prime by checking divisibility from 2 up to the square root of the number, reducing unnecessary iterations. If a divisor is found, it returns False; otherwise, it returns True. The main block takes starting and ending values as input and iterates through the range, printing whether each number is prime or not while counting total prime numbers. Exception handling ensures invalid inputs are managed properly. Overall, the program demonstrates efficient logic, looping, conditional checks, and proper error handling.

Task-02

Final Optimal Prompt:

write a python program to calculate a sum in a given list

also handle the error when user give string input or negative input or tell the user to give correct input

example: [1, 2, s, 4, 5]

output: Invalid input. Please enter a valid integer.

Code Screenshot :

```python
def sum_of_list(lst):
    try:
        return sum(lst)
    except TypeError:
        return "Invalid input. Please enter a valid integer."
if __name__ == "__main__":
    lst = list[int](map[int](int, input("Enter a list of integers separated by spaces: ").split()))
    print(f"The sum of the list is {sum_of_list(lst)}.")
    try:
        lst = list[int](map[int](int, input("Enter a list of integers separated by spaces: ").split()))
        print(f"The sum of the list is {sum_of_list(lst)}.")
    except ValueError:
        print("Invalid input. Please enter a valid integer.")
    except Exception as e:
        print(f"An error occurred: {e}")
```

Output Screenshot:

```
OneDrive/Desktop/Ai-Assistant/lab5.py
Enter a list of integers separated by spaces: 5 4 3 8 9 -6
The sum of the list is 23.
Enter a list of integers separated by spaces: 6 3 4 85 6
The sum of the list is 104.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>
```

Explanation/Justification/Observation (100 words / 5 – 6 sentence) :

The program calculates the sum of a list of integers entered by the user. It uses Python's built-in sum() function inside the sum_of_list() function to compute the total efficiently. A try-except block is included to handle potential TypeError exceptions if invalid data types are passed. In the main block, user input is converted into integers using map() and split(), ensuring numeric processing. Additional exception handling manages ValueError for invalid integer inputs and catches unexpected errors. Overall, the program demonstrates proper use of built-in functions, input validation, and structured exception handling for robust execution.

Task-03

Final Optimal Prompt:

write a python program to extract the digits from a given alphanumeric string

example: "a1b2c3" -> [1, 2, 3]

example: "Abc12=+-569()" -> [1, 2, 5, 6, 9]

Code Screenshot :

```python
import sys
def extract_digits(s: str) -> list:
    return [int(char) for char in s if char.isdigit()]
if __name__ == "__main__":
    try:
        s = input("Enter a string: ")
        print(f"The digits in the string are {extract_digits(s)}.")
    except ValueError:
        print("Invalid input. Please enter a valid string.")
        sys.exit(1)
    except Exception as e:
        print(f"An error occurred: {e}")


Ctrl+I to chat, Ctrl+K to generate
```

Output Screenshot:

```
The digits in the string are [1, 2, 3, 4, 5].
Enter a string: fgdufgu6454
The digits in the string are [6, 4, 5, 4].
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & C:\Python314\python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab5.py
Enter a string: +96 jhsdw
The digits in the string are [9, 6].
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>
```

Explanation/Justification/Observation (100 words / 5 – 6 sentence) :

The program extracts all numeric digits from a given string and returns them as a list of integers. The extract_digits() function uses a list comprehension to iterate through each character in the string and checks whether it is a digit using the isdigit() method. If true, it converts the character into an integer and includes it in the resulting list. The main block handles user input and prints the extracted digits. Exception handling is included to manage

unexpected errors gracefully. Overall, the program demonstrates effective string processing, list comprehension usage, and structured error handling in Python.

Task-04

Final Optimal Prompt:

write a python program to count the number of vowels in a given string

write a python program to count no of vowels in a given string

example: "Hello World" -> 3

example: "Python is fun" -> 3

example: "AEIOUaeiou" -> 10

example: "1234567890" -> 0

example: "!@#$%^&*()" -> 0

example: " " -> 0

example: "" -> 0

example: "1234567890" -> 0

example: "!@#$%^&*()" -> 0

example: " " -> 0

example**: "" -> 0**

Code Screenshot :

```python
'''
import sys
def count_vowels(s: str) -> int:
    return sum(1 for char in s if char.lower() in 'aeiou')
if __name__ == "__main__":
    try:
        s = input("Enter a string: ")
        print(f"The number of vowels in the string are {count_vowels(s)}.")
    except ValueError:
        print("Invalid input. Please enter a valid string.")
        sys.exit(1)
    except Exception as e:
        print(f"An error occurred: {e}")


'''
```

```python
import sys
def count_vowels(s: str) -> int:
    return sum(1 for char in s if char.lower() in 'aeiou')
if __name__ == "__main__":
    try:
        s = input("Enter a string: ")
        print(f"The number of vowels in the string are {count_vowels(s)}.")
    except ValueError:
        print("Invalid input. Please enter a valid string.")
        sys.exit(1)
    except Exception as e:
        print(f"An error occurred: {e}")
Ctrl+L to chat, Ctrl+K to generate
```

Output Screenshot:

```
Problems   Output   Debug Console   Terminal   Ports   Azure

PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & C:\Python314\python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab5
Enter a string: sakir is bad
The number of vowels in the string are 4.
Enter a string: king of underground worls
The number of vowels in the string are 7.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>

Ctrl+K to generate command
```

Explanation/Justification/Observation (100 words / 5 – 6 sentence) :

Aspect	Zero-Shot Version	Few-Shot Version
Prompt Style	Only instruction given	Instruction + multiple examples

Examples Included	❌ No examples	☑ Multiple input-output examples
Code Logic	Uses count_vowels() with generator expression	Uses same count_vowels() logic
Handling Edge Cases	Implicitly handled	Explicitly demonstrated through examples
Output Format	Simple print statement	Same print statement
Clarity of Requirement	General	Very clear due to sample cases
Guidance to AI	Minimal	Strong guidance with test scenarios
Accuracy Expectation	Good for simple tasks	More reliable for edge cases
Complexity	Simple and direct	Slightly more descriptive prompt
Learning Pattern	Based on model knowledge	Based on provided patterns

---

Tack-05

Final Optimal Prompt:

write a python program to take 3 sample inputs and find the minimum no amoung three inputs without using build in function

example: 10, 20, 30 -> 10

example: 10.5, 20.5, 30.5 -> 10.5

example: "10", "20", "30" -> 10

example: "10.5", "20.5", "30.5" -> 10.5

example: "10", "20.5", "30" -> 10

example: "10.5", "20", "30.5" -> 10.5

Code Screenshot :

```python
import sys
def find_minimum(a, b, c):
    if a < b and a < c:
        return a
    elif b < a and b < c:
        return b
    else:
        return c
if __name__ == "__main__":
    try:
        a = float(input("Enter first number: "))
        b = float(input("Enter second number: "))
        c = float(input("Enter third number: "))
        print(f"The minimum number is {find_minimum(a, b, c)}.")
    except ValueError:
        print("Invalid input. Please enter a valid number.")
        sys.exit(1)
    except Exception as e:
        print(f"An error occurred: {e}")
        sys.exit(1)
```

Output Screenshot:

```
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> ^C
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant> & C:\Python314\python.exe c:/Users/sakir/OneDrive/Desktop/Ai-Assistant/lab5.py
Enter first number: 10
Enter second number: 90
Enter third number: 3
The minimum number is 3.0.
PS C:\Users\sakir\OneDrive\Desktop\Ai-Assistant>
```

Explanation/Justification/Observation (100 words / 5 – 6 sentence) :

The program determines the minimum of three numbers using conditional statements. The find_minimum() function compares the three inputs using if-elif-else conditions and returns the smallest value. It checks whether a is smaller than both b and c, then checks b, and otherwise returns c. The main block handles user interaction by taking numeric input and converting it to float to allow decimal values. Exception handling is implemented using try-except blocks to manage invalid inputs and unexpected errors gracefully. Overall, the program demonstrates clear conditional logic, input validation, and structured error handling in Python.