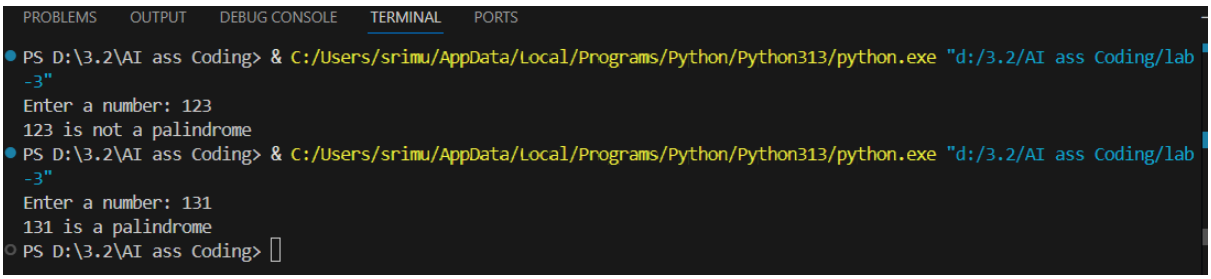
ASSIGNMENT -3

Hall ticket :2303A54038
Batch :48

Question 1: Zero-Shot Prompting (Palindrome Number Program)
Write a zero-shot prompt (without providing any examples) to generate
a Python function that checks whether a given number is a palindrome.
Task:
• Record the AI-generated code.
• Test the code with multiple inputs.
• Identify any logical errors or missing edge-case handling.
Prompt :

give me the python code to check wether the given number is palindrome or not

```
lab-3 > ...
1   # Get input from user
2   num = int(input("Enter a number: "))
3
4   # Convert to string and remove negative sign if present
5   num_str = str(abs(num))
6
7   # Compare string with its reverse
8   if num_str == num_str[::-1]:
9       print(f"{num} is a palindrome")
10  else:
11      print(f"{num} is not a palindrome")
12
```



Code Explanation :

The palindrome checker code converts a number into a string, then compares it with its reverse using Python's slice notation [::-1]. If the original string matches the reversed string, the number is a palindrome (reads the same forwards and backwards), otherwise it's not. For example, 121 reversed is still 121, so it's a palindrome, but 123 reversed becomes 321, so it's not a palindrome

Question 2: One-Shot Prompting (Factorial Calculation)
Write a one-shot prompt by providing one input-output example and
ask the AI to generate a Python function to compute the factorial of a
given number.
Example:
Input: 5 → Output: 120
Task:
• Compare the generated code with a zero-shot solution.
• Examine improvements in clarity and correctness.

Prompt:

give me the pyhton code for factorial

```python
num = int(input("Enter a number: "))
fact = 1

for i in range(1, num + 1):
    fact = fact * i

print(f"Factorial of {num} is {fact}")
```

```
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab
-3"
Enter a number: 6
Factorial of 6 is 720
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab
-3"
Enter a number: 5
Factorial of 5 is 120
```

Code Explanation :

The code multiplies a number by all positive integers below it. It initializes fact = 1, loops from 1 to the input number, and multiplies fact by each loop value. For example, 5! = 1×2×3×4×5 = 120.

Question 3: Few-Shot Prompting (Armstrong Number Check)
Write a few-shot prompt by providing multiple input-output examples
to guide the AI in generating a Python function to check whether a
given number is an Armstrong number.
Examples:
• Input: 153 → Output: Armstrong Number
• Input: 370 → Output: Armstrong Number
• Input: 123 → Output: Not an Armstrong Number
Task:
• Analyze how multiple examples influence code structure and accuracy.

• **Test the function with boundary values and invalid inputs.**

give me the pyhton code for armstrong number

```python
num = int(input("Enter a number: "))
digits = str(num)
n = len(digits)
sum_of_powers = 0

for digit in digits:
    sum_of_powers = sum_of_powers + int(digit) ** n

if sum_of_powers == num:
    print(f"{num} is an Armstrong number")
else:
    print(f"{num} is not an Armstrong number")
```

```
PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  PORTS
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab
-3"
Enter a number: 123
123 is not a palindrome
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab
-3"
Enter a number: 131
131 is a palindrome
PS D:\3.2\AI ass Coding>
```

Code Explanation :

The code takes a number as input, converts it to a string to get individual digits, counts the number of digits, then raises each digit to the power of the total digit count and sums them. If the sum equals the original number, it's an Armstrong number; otherwise, it's not. For example, 153 has 3 digits, so we calculate $1^3 + 5^3 + 3^3 = 1 + 125 + 27 = 153$, which equals the original number, so 153 is an Armstrong number.

Question 4: Context-Managed Prompting (Optimized Number Classification)

Design a context-managed prompt with clear instructions and

constraints to generate an optimized Python program that classifies a
number as prime, composite, or neither.
Task:
• Ensure proper input validation.
• Optimize the logic for efficiency.
• Compare the output with earlier prompting strategies.
Question 5: Zero-Shot Prompting (Perfect Number Check)
Write a zero-shot prompt (without providing any examples) to generate a Python function that checks whether a given number is a
perfect number.
Task:
• Record the AI-generated code.
• Test the program with multiple inputs.
• Identify any missing conditions or inefficiencies in the logic.

Generate an optimized Python program that classifies a number as prime, composite, or neither

```
num = int(input("Enter a number: "))
if num <= 1:
    print(f"{num} is neither prime nor composite")
else:
    for i in range(2, int(num**0.5) + 1):
        if num % i == 0:
            print(f"{num} is composite")
            break
    else:
        print(f"{num} is prime")
```

```
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab
-3"
Enter a number: 3
3 is prime
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab
-3"
Enter a number: 0
0 is neither prime nor composite
```

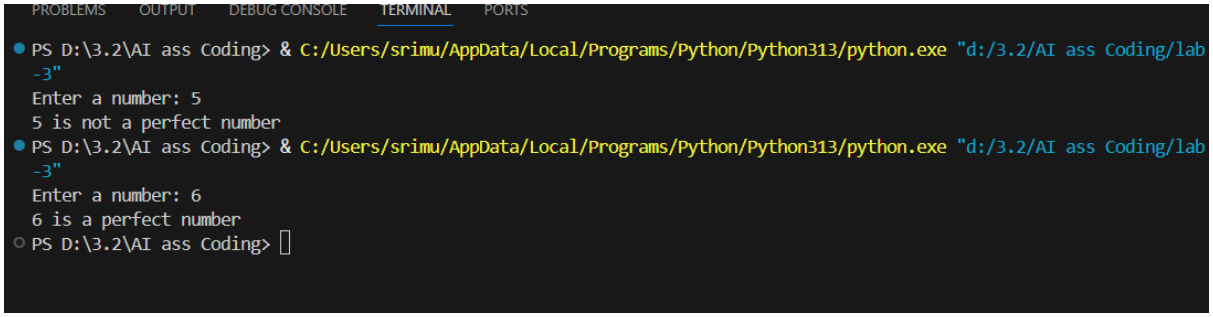
Code Explanation :

This program checks whether a given number is **prime or composite**. First, it takes an integer input from the user. If the number is **0 or 1**, it prints that the number is neither prime nor composite. Otherwise, it runs a loop from **2 up to the square root of the number**, checking if the number is divisible by any value. If it finds a divisor, the number is **composite** and the loop stops. If the loop finishes without finding any divisor, the `else` part of the loop runs and the number is **prime**

Question 5: **Zero-Shot Prompting (Perfect Number Check)**
Write a zero-shot prompt (without providing any examples) to generate a Python function that checks whether a given number is a
perfect number.
Task:
• Record the AI-generated code.
• Test the program with multiple inputs.
• Identify any missing conditions or inefficiencies in the logic

Prompt :
Generate a Python function that checks whether a given number is a perfect number.

```
num = int(input("Enter a number: "))
divisor_sum = 0
for i in range(1, num):
    if num % i == 0:
        divisor_sum += i
if divisor_sum == num:
    print(f"{num} is a perfect number")
else:
    print(f"{num} is not a perfect number")
```



Code Explanation :

This program checks whether a given number is a **perfect number**. It first takes a number as input and initializes a variable to store the sum of its divisors. Then it loops from **1 to one less than the number**, and whenever a value divides the number exactly, it adds it to the sum. After the loop, if the sum of all proper divisors is **equal to the number itself**, the program prints that the number is a perfect number; otherwise, it prints that it is not.

Question 6: Few-Shot Prompting (Even or Odd Classification with
Validation)

Write a few-shot prompt by providing multiple input-output examples to guide the AI in generating a Python program that determines whether a given number is even or odd, including proper
input validation.

Examples:

• Input: 8 → Output: Even
• Input: 15 → Output: Odd
• Input: 0 → Output: Even

Task:

• Analyze how examples improve input handling and output clarity.
• Test the program with negative numbers and non-integer inputs.

Prompt :

Generate a Python program that etermines whether a given number is even or odd, including proper

```python
#a Python program that etermines whether a given number is even or odd, including proper
num = int(input("Enter a number: "))
if num % 2 == 0:
    print(f"{num} is even")
else:
    print(f"{num} is odd")
```

```
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab-3
"
Enter a number: 5
5 is odd
PS D:\3.2\AI ass Coding> & C:/Users/srimu/AppData/Local/Programs/Python/Python313/python.exe "d:/3.2/AI ass Coding/lab-3
"
Enter a number: 6
6 is even
PS D:\3.2\AI ass Coding>
```

Code Explanation :

This program checks whether a given number is **even or odd**. It takes a number as input and uses the modulo (`%`) operator to see if the number is divisible by 2. If the remainder is **0**, the number is **even**; otherwise, it is **odd**, and the result is printed accordingly.