

# Lab Assignment- 3.1

AI Assisted Coding  
A Harshita – 2303A52211

## Question1: Zero-Shot Prompting (Palindrome Number Program)

**Prompt:** generate a Python code, function that checks whether a given number is a palindrome. And input is given by the user

The screenshot shows the Visual Studio Code interface with an AI assistant sidebar. The main editor has the following code:

```
def is_palindrome():
    """function to check if a number is a palindrome"""
    num_str = str(num)
    return num_str == num_str[::-1]

def main():
    user_input = input("Enter a number: ")
    try:
        number = int(user_input)
        if is_palindrome(number):
            print(f"{number} is a palindrome")
        else:
            print(f"{number} is not a palindrome")
    except ValueError:
        print("Invalid input. Please enter a valid integer.")

if __name__ == "__main__":
    main()
```

The terminal shows the program running and accepting user input. An error message is visible in the status bar indicating a maximum recursion depth exceeded.

- It generates correct logic but does not handle negative numbers explicitly.\

## Question 2: One-Shot Prompting (Factorial Calculation)

**Prompt:** generate a python code, to get the factorial number of a given number.  
for example, i/p: 5, o/p:120

The screenshot shows the Visual Studio Code interface with an AI assistant sidebar. The main editor has the following code:

```
def factorial(n):
    """function to calculate factorial of a number"""
    if n <= 0:
        return "Invalid input"
    result = 1
    for i in range(1, n + 1):
        result *= i
    return result

number = int(input("Enter a number: "))
print(factorial(number))
```

The terminal shows the program running and accepting user input. It handles edge cases like zero and negative numbers.

- When compared to zero- short prompting this prompt improves correctness and handles edge cases like zero and negative values.

### Question 3: Few-Shot Prompting (Armstrong Number Check)

**Prompt:** Generate a Python code to check whether a number is an Armstrong number.

Examples: i/p: 153, o/p: Armstrong Number

i/p: 370, o/p: Armstrong Number

i/p: 123, o/p: Not an Armstrong Number

The screenshot shows the Visual Studio Code interface with a Python file named Lab3.2236.py open. The code is as follows:

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n-1)

def is_armstrong(num):
    digits = str(num)
    power = len(digits)
    total = 0

    for d in digits:
        total += int(d) ** power

    return total == num

number = int(input("Enter a number: "))
print(is_armstrong(number))
```

The terminal window shows the execution of the script and some test cases:

```
PS C:\Users\kotas\OneDrive\AI\assistant> & C:/Users/kotas/AppData/Local/Microsoft/WindowsApps/python3.11.exe "C:/Users/kotas/OneDrive/AI/assistant/Lab3.2236.py"
Enter a number: 153
153 is an Armstrong Number
Enter a number: 370
370 is an Armstrong Number
Enter a number: 123
123 is not an Armstrong Number
PS C:\Users\kotas\OneDrive\AI\assistant>
```

- Few-shot prompting improves structural accuracy and produces organized logic.

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

**Prompt:** Write an optimized Python program to classify a number as prime, composite, or neither. Ensure input validation and efficient logic.

The screenshot shows the Visual Studio Code interface with a Python file named Lab4.2236.py open. The code is as follows:

```
def classify_number(num):
    if num < 2:
        return "Neither Prime nor Composite"
    for i in range(2, int(num ** 0.5) + 1):
        if num % i == 0:
            return "Composite Number"
    return "Prime Number"

number = int(input("Enter a number: "))
print(classify_number(number))
```

The terminal window shows the execution of the script and some test cases:

```
PS C:\Users\kotas\OneDrive\AI\assistant> & C:/Users/kotas/AppData/Local/Microsoft/WindowsApps/python3.11.exe "C:/Users/kotas/OneDrive/AI/assistant/Lab4.2236.py"
Enter a number: 47999999
47999999 is not a palindrome.
Enter a number: 23
Not an Armstrong Number
Enter a number: 21
21 is not a palindrome.
Enter a number: 94
Not an Armstrong Number
Enter a number: 2342
Composite Number
PS C:\Users\kotas\OneDrive\AI\assistant>
```

- Context-managed prompts produce optimized and validation-aware solutions suitable for real-world applications.

## Question 5: Zero-Shot Prompting (Perfect Number Check)

**Prompt:** Generate a Python function to check whether a given number is a perfect number.

The screenshot shows the AI Assistant interface with a task titled "generate a Python code function that ...". The code editor contains the following Python script:

```
def is_perfect(num):
    if num <= 0:
        return False

    total = 0
    for i in range(1, num):
        if num % i == 0:
            total += i

    return total == num

number = int(input("Enter a number: "))
if is_perfect(number):
    print("Perfect Number")
else:
    print("Not a Perfect Number")
```

The terminal window shows the execution of the script and some user interactions:

```
PS C:\Users\kotas\OneDrive\AI assist> python Lab5.1_2256.py
Enter a number: 543
Not an Armstrong Number
Enter a number: 243
243 is not a palindrome.
Enter a number: 12
47900150
Enter a number: 333
Not an Armstrong Number
Enter a number: 33
Prime Number
Enter a number: 343
Not a Perfect Number
```

The screenshot shows the AI Assistant interface with a task titled "generate a Python code function that ...". The code editor contains the same Python script as the previous screenshot:

```
def is_perfect(num):
    if num <= 0:
        return False

    total = 0
    for i in range(1, num):
        if num % i == 0:
            total += i

    return total == num

number = int(input("Enter a number: "))
if is_perfect(number):
    print("Perfect Number")
else:
    print("Not a Perfect Number")
```

The terminal window shows the execution of the script and some user interactions:

```
PS C:\Users\kotas\OneDrive\AI assist> python Lab5.1_2256.py
Enter a number: 345
345 is not a palindrome.
Enter a number: 22
1326988727776987688000
Enter a number: 34
Not an Armstrong Number
Enter a number: 23
Prime Number
Enter a number: 6
Perfect Number
PS C:\Users\kotas\OneDrive\AI assist>
```

- Zero-shot prompting works but is less optimized due to unnecessary full-range iteration.

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

**Prompt:** Generate a Python program to check whether a number is even or odd with input validation.

## Examples:

I/p: 8, O/p: Even

I/p: 15, O/p: Odd

I/p: 0, O/p: Even

- Few-shot prompting significantly improves the quality of AI-generated code by providing clear input–output examples.
  - The generated program handles input validation effectively, correctly classifies even and odd numbers, and manages negative and non-integer inputs more reliably compared to zero-shot prompting.