

## School of Computer Science and Artificial Intelligence

### Lab Assignment # 3

Name of Student : k.saikiran  
Enrollment No. : 2303A51539  
Batch No. : 22

#### 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.

Code:-

```
Assignment3.py >-
1  #write a program that checks whether a given number is palindrome or not, show me output as true or false
2  def is_palindrome(n):
3      str_n = str(n)
4      return str_n == str_n[::-1]
5  #Example usage:
6  number = input("Enter a number: ")
7  number = int(number)
8  result = is_palindrome(number)
9  print(f'Is the number {number} a palindrome? {result}')
```

OUTPUT:

```
Enter a number: 121
Is the number 121 a palindrome? True
PS C:\Users\DELL\Desktop\PYTHON> & C:/U

Enter a number: 120
Is the number 120 a palindrome? False
PS C:\Users\DELL\Desktop\PYTHON> |
```

**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:**input: 5 => output: 120 write a function to calculate factorial of a number

**Code:-**

```
#:input: 5 => output: 120 write a function to calculate factorial of a number
def factorial(n):
    if n == 0 or n == 1:
        return 1
    else:
        return n * factorial(n - 1)
#Example usage:
number = input("Enter a number: ")
number = int(number)
result = factorial(number)
print(f'The factorial of {number} is {result}')
```

**Output:-**

```
Enter a number: 5
The factorial of 5 is 120
PS C:\Users\DELL\Desktop\PYTHON>
```

**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.

**(Optional Extension)**

**Prompt:** input: 370 => output: Armstrong write a function to check if a number is an Armstrong number.

**Code:-**

```
#Armstrong write a function to check if a number is an Armstrong number.  
def is_armstrong(n):  
    num_str = str(n)  
    num_digits = len(num_str)  
    sum_of_powers = sum(int(digit) ** num_digits for digit in num_str)  
    return sum_of_powers == n  
  
#Example usage:  
number = input("Enter a number: ")  
number = int(number)  
result = is_armstrong(number)  
print(f'Is the number {number} an Armstrong number? {result}')
```

**Output:-**

```
Enter a number: 14  
Is the number 14 an Armstrong number? False  
PS C:\Users\DELL\Desktop\PYTHON> [
```



**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.

**Prompt:** write a program on a context-managed that classifies number as prime, composite or neither.

**Code:-**

```
#write a program on a context-managed that classifies number as prime, composite or neither
class NumberClassifier:
    def __init__(self, number):
        self.number = number
        self.classification = None

    def __enter__(self):
        if self.number <= 1:
            self.classification = 'neither prime nor composite'
        else:
            for i in range(2, int(self.number**0.5) + 1):
                if self.number % i == 0:
                    self.classification = 'composite'
                    break
                else:
                    self.classification = 'prime'
        return self

    def __exit__(self, exc_type, exc_value, traceback):
        pass
#Example usage:
number = input("Enter a number: ")
number = int(number)
with NumberClassifier(number) as classifier:
    print(f'The number {number} is classified as: {classifier.classification}')
```

**Output:-**

```
Enter a number: 134
The number 134 is classified as: composite
PS C:\Users\DELL\Desktop\PYTHON> & C:/Users/DELL/AppData/Local/Temp/134.py

Enter a number: 12
The number 12 is classified as: composite
PS C:\Users\DELL\Desktop\PYTHON> & C:/Users/DELL/AppData/Local/Temp/134.py

Enter a number: 1
The number 1 is classified as: neither prime nor composite
PS C:\Users\DELL\Desktop\PYTHON> []
```

---

**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.

**Code:-**

```
#write a python program to generate a Python function that checks whether a given number is a perfect number:
def is_perfect_number(n):
    if n <= 1:
        return False
    divisors_sum = sum(i for i in range(1, n) if n % i == 0)
    return divisors_sum == n
#Example usage:
number = input("Enter a number: ")
number = int(number)
result = is_perfect_number(number)
print(f'Is the number {number} a perfect number? {result}')
```

**Output:-**

```
Enter a number: 15
Is the number 15 a perfect number? False
PS C:\Users\DELL\Desktop\PYTHON> & C:/Users

Enter a number: 18
Is the number 18 a perfect number? False
PS C:\Users\DELL\Desktop\PYTHON> & C:/Users

Enter a number: 6
Is the number 6 a perfect number? True
PS C:\Users\DELL\Desktop\PYTHON>
```

---

**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:**input: 8 ->  
output: Even  
input: 15 -  
> output: Odd  
input: 0 -  
> output: Even  
write a function to check if a number is even or odd.

**CODE:**

```
#write a python program Even write a function to check if a number is even or odd.
def is_even_or_odd(n):
    return "Even" if n % 2 == 0 else "Odd"
#Example usage:
number = input("Enter a number: ")
number = int(number)
result = is_even_or_odd(number)
print(f'The number {number} is {result}')
```

**Output:-**

```
Enter a number: 3
The number 3 is Odd
PS C:\Users\DELL\Desktop\PYTHON> & C

Enter a number: 2
The number 2 is Even
PS C:\Users\DELL\Desktop\PYTHON> & C

Enter a number: 7
The number 7 is Odd
PS C:\Users\DELL\Desktop\PYTHON>
```