

AI-Assisted Coding

Week-2-Assignment 3.5:

Name:V.siddartha

Date:23-01-2026

Hall.no:2303A52062

#Question1

Question 1: Zero-Shot Prompting (Leap Year Check)

Write a zero-shot prompt to generate a Python function that checks whether a given year is a leap year.

Task:

- Record the AI-generated code.
- Test with years like 1900, 2000, 2024.
- Identify logical flaws or missing conditions.

#prompt

#write a python program to find the given three years are leap years or not. Accept user input for three years. Check and display whether each year is a leap year.

#Code

```
def is_leap_year(year):  
    """Check if a given year is a leap year.
```

Parameters:

year (int): The year to be checked.

Returns:

bool: True if the year is a leap year, False otherwise.

....

```
if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
```

```
    return True
```

```
else:
```

```
    return False
```

```
# Accept user input for three years
```

```
years = []
```

```

for i in range(3):
    year = int(input(f"Enter year {i+1}: "))
    years.append(year)
# Check and display whether each year is a leap year
for year in years:
    if is_leap_year(year):
        print(f"{year} is a leap year.")
    else:
        print(f"{year} is not a leap year.")

```

#Output

```

PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> & c:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/Users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"
Enter year 1: 1900
Enter year 2: 2000
Enter year 3: 2024
1900 is not a leap year.
2000 is a leap year.
2024 is a leap year.
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> []

```

#REVIEW

This program is clear and easy to understand. The logic for checking leap years is correct and well organized using a function. Taking input in a loop makes the code neat and avoids repetition. Overall, it is a clean and beginner-friendly solution.

#QUESTION 2

Question 2: One-Shot Prompting (GCD of Two Numbers)

Write a one-shot prompt with one example to generate a Python function that finds the Greatest Common Divisor (GCD) of two numbers.

Example:

Input: 12, 18 → Output: 6

Task:

- Compare with a zero-shot solution.
- Analyze algorithm efficiency.

#Prompt

write a python program to find the GCD of two numbers take user input

Compare with a zero-shot solution. Analyze algorithm efficiency.

#code

```

def gcd(a, b):
    """Compute the Greatest Common Divisor (GCD) of two numbers using the Euclidean algorithm.

```

Parameters:

```
a (int): First number.  
b (int): Second number.
```

Returns:

int: GCD of the two numbers.

.....

while b:

 a, b = b, a % b

return a

```
# Accept user input for two numbers
```

```
num1 = int(input("Enter the first number: "))
```

```
num2 = int(input("Enter the second number: "))
```

```
# Calculate and display the GCD
```

```
result = gcd(num1, num2)
```

```
print(f"The GCD of {num1} and {num2} is: {result}")
```

```
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding & C:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/Users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"  
Enter the first number: 12  
Enter the second number: 18  
The GCD of 12 and 18 is: 6
```

```
# Zero-shot solution for GCD without function
```

```
a = int(input("Enter the first number: "))
```

```
b = int(input("Enter the second number: "))
```

while b:

 a, b = b, a % b

```
print(f"The GCD of the two numbers is: {a}")
```

#OUTPUT

```
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding & C:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/Users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"  
Enter the first number: 12  
Enter the second number: 18  
The GCD of the two numbers is: 6
```

Algorithm Efficiency Analysis:

The Euclidean algorithm used in both implementations has a time complexity of $O(\log(\min(a, b)))$, where a and b are the two input numbers. This is because with each iteration, the size of the numbers reduces significantly.

Both implementations are efficient, but the function-based approach offers better readability and reusability, making it more suitable for larger applications.

#QUESTION 3

Few-Shot Prompting (LCM Calculation)

Write a few-shot prompt with multiple examples to generate a Python function that computes the Least Common Multiple (LCM).

Examples:

- Input: 4, 6 → Output: 12
- Input: 5, 10 → Output: 10
- Input: 7, 3 → Output: 21

Task:

- Examine how examples guide formula selection.
- Test edge cases.

#prompt

write a python program to find LCM in functions that takes input as (4,6) and multiple inputs and take input from user for multiple pairs of numbers

Function to compute LCM of two numbers

#Code

```
def lcm(a, b):  
    #Compute the Least Common Multiple (LCM) of two numbers.  
  
    #Parameters:  
    #a (int): First number.  
    #b (int): Second number.  
  
    #Returns:  
    #int: LCM of the two numbers.  
    return abs(a * b) // gcd(a, b)  
# Function to compute GCD (used in LCM calculation)  
def gcd(a, b):  
    while b:  
        a, b = b, a % b  
    return a  
# Accept user input for multiple pairs of numbers  
num_pairs = int(input("Enter the number of pairs of numbers: "))  
for i in range(num_pairs):  
    num1 = int(input(f"Enter the first number of pair {i+1}: "))  
    num2 = int(input(f"Enter the second number of pair {i+1}: "))  
    result = lcm(num1, num2)  
    print(f"The LCM of {num1} and {num2} is: {result}")
```

#OUTPUT

```
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> & C:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/Users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"
Enter the number of pairs of numbers: 3
Enter the first number of pair 1: 4
Enter the second number of pair 1: 6
The LCM of 4 and 6 is: 12
Enter the first number of pair 2: 5
Enter the second number of pair 2: 10
The LCM of 5 and 10 is: 10
Enter the first number of pair 3: 7
Enter the second number of pair 3: 3
The LCM of 7 and 3 is: 21
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding>
```

#REVIEW

The program efficiently calculates the LCM of multiple pairs of numbers using well-defined functions for both LCM and GCD. It enhances readability and reusability, making it suitable for various applications.

#QUESTION-4

Question 4: Zero-Shot Prompting (Binary to Decimal Conversion)

Write a zero-shot prompt to generate a Python function that converts a binary number to decimal.

Task:

- Test with valid and invalid binary inputs.
- Identify missing validation logic

#prompt

write a python program to convert binary number to decimal number using functions and take user input 1)• Test with valid and invalid binary inputs.2)• Identify missing validation logic

#code

```
def binary_to_decimal(binary_str):
    #Convert a binary string to its decimal equivalent.
```

#Parameters:

#binary_str (str): The binary string to be converted.

#Returns:

#int: The decimal equivalent of the binary string.

try:

```
    decimal_value = int(binary_str, 2)
```

```
    return decimal_value
```

except ValueError:

```
    return None
```

```

# Accept user input for binary number
binary_input = input("Enter a binary number: ")
# Validate input and convert to decimal
decimal_output = binary_to_decimal(binary_input)
if decimal_output is not None:
    print(f"The decimal equivalent of binary {binary_input} is: {decimal_output}")
else:
    print("Invalid binary input. Please enter a valid binary number.")

```

#output

```

PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> & C:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/Users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"
Enter a binary number: 101
The decimal equivalent of binary 101 is: 5
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> & C:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/Users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"
Enter a binary number: 1201
Invalid binary input. Please enter a valid binary number.
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> 

```

#REVIEW

The program effectively converts binary strings to decimal while incorporating error handling for invalid inputs. However, it could benefit from additional validation logic to check for non-binary characters before conversion.

#QUESTION-5

One-Shot Prompting (Decimal to Binary Conversion)

Write a one-shot prompt with an example to generate a Python function that converts a decimal number to binary.

Example:

Input: 10 → Output: 1010

Task:

- Compare clarity with zero-shot output.
- Analyze handling of zero and negative numbers.

#prompt

write a python program to convert decimal number to binary number using functions and take user input Analyze handling of zero and negative numbers.

#code

```

def decimal_to_binary(decimal_num):
    #Convert a decimal number to its binary string equivalent.

```

#Parameters:

#decimal_num (int): The decimal number to be converted.

#Returns:

```

#str: The binary string equivalent of the decimal number.

if decimal_num < 0:
    return None # Negative numbers are not handled
return bin(decimal_num).replace("0b", "")
# Accept user input for decimal number
decimal_input = int(input("Enter a decimal number: "))
# Validate input and convert to binary
binary_output = decimal_to_binary(decimal_input)
if binary_output is not None:
    print(f"The binary equivalent of decimal {decimal_input} is: {binary_output}")
else:
    print("Negative numbers are not supported. Please enter a non-negative integer.")

```

#output

```

PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> & C:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/Users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"
Enter a decimal number: 99
The binary equivalent of decimal 99 is: 1100011

```

#REVIEW

The program correctly converts zero to its binary equivalent, which is '0'. However, it does not handle negative numbers, returning None and prompting the user to enter a non-negative integer. This limitation should be addressed if negative binary representations are required.

#QUESTION-6

Few-Shot Prompting (Harshad Number Check)

Write a few-shot prompt to generate a Python function that checks whether a number is a Harshad (Niven) number.

Examples:

- Input: 18 → Output: Harshad Number
- Input: 21 → Output: Harshad Number
- Input: 19 → Output: Not a Harshad Number

Task:

- Test boundary conditions.
- Evaluate robustness

#prompt

write a pythonprogram to Write a few-shot prompt to generate a Python function that checks whether a number is a Harshad (Niven) number.Input: 18 → Output: Harshad Number • Input: 21 → Output: Harshad Number • Input: 19 → Output: Not a Harshad Number Test boundary conditions. • Evaluate robustness take user input

#CODE

```
def is_harshad_number(num):
    #Check if a number is a Harshad (Niven) number.

    #Parameters:
    #num (int): The number to be checked.

    #Returns:
    #str: "Harshad Number" if the number is a Harshad number, otherwise "Not a Harshad Number".

    if num <= 0:
        return "Input must be a positive integer."
    digit_sum = sum(int(digit) for digit in str(num))
    if num % digit_sum == 0:
        return "Harshad Number"
    else:
        return "Not a Harshad Number"

# Accept user input for a number
user_input = int(input("Enter a positive integer: "))
# Check and display whether the number is a Harshad number
result = is_harshad_number(user_input)
print(f"The number {user_input} is: {result}")

# Test boundary conditions
test_values = [18, 21, 19, 0, -5]
for value in test_values:
    print(f"Input: {value} → Output: {is_harshad_number(value)}")
```

#OUTPUT

```
Input: 5 → Output: Not a Harshad Number
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> & C:\Users\sidda\AppData\Local\Programs\Python\Python314\python.exe "c:/users/sidda/OneDrive/Documents/Desktop/AI - Assisted coding/AI-assist 3.5.py"
Enter a positive integer: 12
The number 12 is: Harshad Number
Input: 18 → Output: Harshad Number
Input: 21 → Output: Harshad Number
Input: 19 → Output: Not a Harshad Number
Input: 0 → Output: Input must be a positive integer.
Input: -5 → Output: Input must be a positive integer.
PS C:\Users\sidda\OneDrive\Documents\Desktop\AI - Assisted coding> █
```

#REVIEW

The function handles positive integers correctly and provides appropriate messages for zero and negative inputs, ensuring robustness

