

AI-Assisted Coding

Assignment-3.5

Name: sai akshith

Batch:45

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.

The screenshot shows a code editor with a Python file named `assignment_3.5.py`. The code defines a function `is_leap_year` that takes a year as input and returns a boolean. The function uses the following logic: a year is a leap year if it is divisible by 4 but not by 100, or if it is divisible by 400. The code includes docstrings and example usage.

```
1 def is_leap_year(year: int) -> bool:
2     """
3     Args:
4         year (int): The year to check.
5
6     Returns:
7         bool: True if the year is a leap year, False otherwise.
8     """
9     if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
10        return True
11    else:
12        return False
13
14 # Example usage:
15 year = int(input("Enter a year: "))
16 if is_leap_year(year):
17     print(f"{year} is a leap year.")
18 else:
19     print(f"{year} is not a leap year.")
```

The terminal output shows the execution of the code for three different years: 1900, 2000, and 2024. The output for 1900 is "1900 is not a leap year.", for 2000 is "2000 is a leap year.", and for 2024 is "2024 is a leap year.".

The chat window on the right shows a prompt: "Write a Python function named is_leap_year that accepts an integer representing a year. The function should return True if the year is a leap year according to the Gregorian calendar rules (divisible by 4, but not by 100 unless also divisible by 400), and False otherwise. Include type hinting and a docstring." The AI response is the code shown in the editor.

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

The screenshot shows a VS Code editor with a Python file named `assignment_3.5.py`. The code defines a function `find_gcd` that uses Euclid's algorithm to find the GCD of two integers. The function takes two arguments, `a` and `b`, and returns the GCD. The code also includes a main block that prompts the user to enter two numbers and prints the result.

```
23 def find_gcd(a: int, b: int) -> int:
24     """
25     Calculates the Greatest Common Divisor (GCD) of two integers using Euclid's algorithm.
26
27     Args:
28         a (int): The first integer.
29         b (int): The second integer.
30
31     Returns:
32         int: The GCD of a and b.
33     """
34     while b != 0:
35         a, b = b, a % b
36     return a
37
38 # Example usage:
39 num1 = int(input("Enter the first number: "))
40 num2 = int(input("Enter the second number: "))
41 gcd = find_gcd(num1, num2)
42 print(f"The GCD of {num1} and {num2} is {gcd}.")
```

The chat window on the right shows a prompt for writing a Python function named `find_gcd` that calculates the GCD of two integers. The prompt includes an example input and output: `find_gcd(12, 18)` returns `6`. The chat also shows the function definition and a main block that prompts the user to enter two numbers and prints the result.

```
def find_gcd(a: int, b: int) -> int:
    """
    Calculates the Greatest Common Divisor (GCD) of two integers using Euclid's algorithm.

    Args:
        a (int): The first integer.
        b (int): The second integer.

    Returns:
        int: The GCD of a and b.
    """
    while b != 0:
        a, b = b, a % b
    return a

# Example usage:
num1 = int(input("Enter the first number: "))
num2 = int(input("Enter the second number: "))
gcd = find_gcd(num1, num2)
print(f"The GCD of {num1} and {num2} is {gcd}.")
```

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

The screenshot shows a VS Code editor with a file named `assignment_3.5.py`. The code defines a function `calculate_lcm(a: int, b: int) -> int:` and includes a main block that takes user input and prints the LCM. The terminal output shows the function being tested with inputs (4, 6), (5, 10), and (7, 3), resulting in outputs 12, 10, and 21 respectively. A chat window on the right shows a prompt asking for a few-shot prompt to generate the function, and the response provides the function definition and a few-shot prompt with the same examples as the task description.

```
def calculate_lcm(a: int, b: int) -> int:
    """
    The LCM is the smallest positive integer that is a multiple of both a and b.

    Args:
        a (int): The first integer.
        b (int): The second integer.

    Returns:
        int: The LCM of a and b.
    """
    if a == 0 or b == 0:
        return 0 # LCM is undefined for zero, but per examples, assume positive integers
    gcd = find_gcd(a, b)
    return abs(a * b) // gcd

a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))
lcm = calculate_lcm(a, b)
print(f"The LCM of {a} and {b} is {lcm}."
```

```
PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter the first number: 4
Enter the second number: 6
The LCM of 4 and 6 is 12.

PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter the first number: 5
Enter the second number: 10
The LCM of 5 and 10 is 10.

PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter the first number: 7
Enter the second number: 3
The LCM of 7 and 3 is 21.
```

Chat prompt: LEAP YEAR FUNCTION IN PYTHON WIT... correctly.

Chat response: # ...existing code...
def calculate_lcm(a: int, b: int) -> int:
 """
 Calculates the Least Common Multiple (LCM) of two integers a and b.

 The LCM is the smallest positive integer that is a multiple of both a and b.

 Args:
 a (int): The first integer.
 b (int): The second integer.

 Returns:
 int: The LCM of a and b.
 """
 if a == 0 or b == 0:
 return 0 # LCM is undefined for zero, but per examples, assume positive integers
 gcd = find_gcd(a, b)
 return abs(a * b) // gcd

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.

The screenshot shows a VS Code editor with a Python file named `assignment_3.5.py`. The code defines a function `decimal_to_binary` that converts a decimal number to a binary string. It includes a docstring with a return value description and an example usage. The terminal shows the execution of the script, which prompts the user to enter a decimal number and displays the binary equivalent. A chat window on the right contains a prompt asking for a Python function to convert decimal to binary, with an example input and output.

```
def decimal_to_binary(decimal: int) -> str:
    """
    Returns:
    str: The binary representation as a string.
    """
    if decimal == 0:
        return "0"
    binary = ""
    while decimal > 0:
        binary = str(decimal % 2) + binary
        decimal //= 2
    return binary

# Example usage:
decimal_num = int(input("Enter a decimal number: "))
binary_rep = decimal_to_binary(decimal_num)
print(f"The binary representation of {decimal_num} is {binary_rep}.")
```

Terminal Output:

```
PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter the second number: 3
The LCM of 7 and 3 is 21.
PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter a decimal number: 1010
The decimal equivalent of 1010 is 10.
PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter a decimal number: 10
The binary representation of 10 is 1010.
PS D:\AI assisted coding>
```

Chat Prompt:

Write a Python function named `decimal_to_binary` that accepts an integer and returns its binary representation as a string. Use the example below to understand the expected output format.

Example: Input: `decimal_to_binary(10)`
Output: "1010"

Task: Ensure the function handles the integer 0 correctly and returns a clean string without Python's default prefix.

Question 6: Few-Shot Prompting (Harshad Number Check)

Write a few-shot prompt to generate a Python function that checks

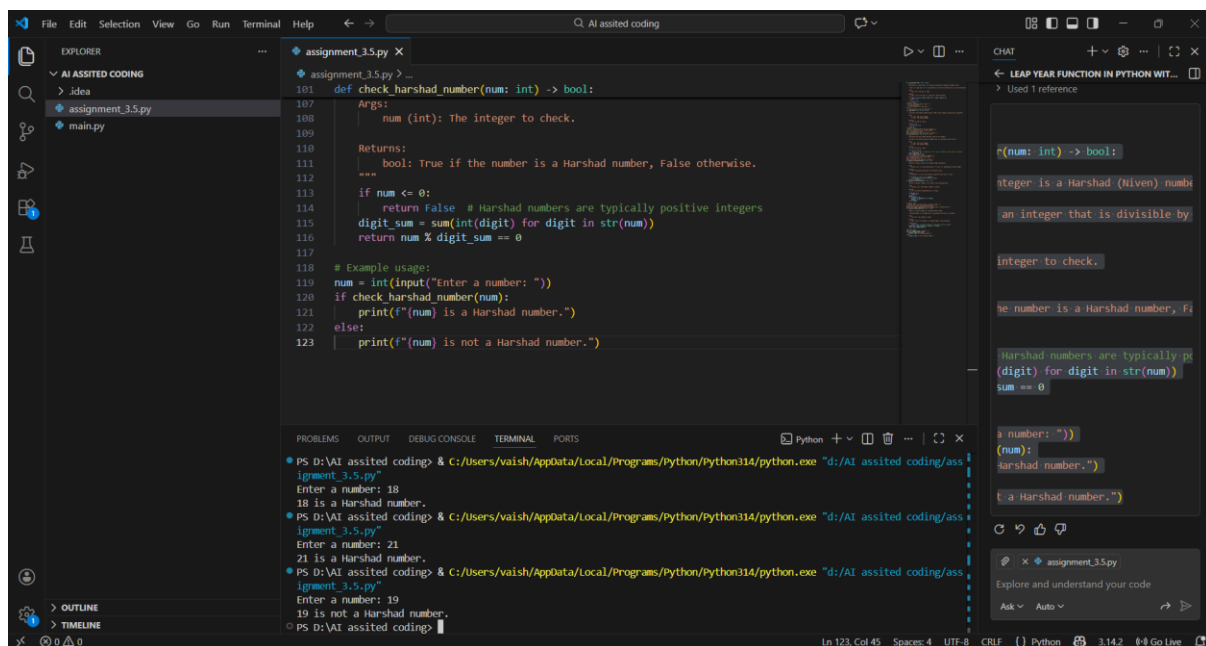
whether a number is a Harshad (Niven) number.

Examples:

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

Task:

- Test boundary conditions.
- Evaluate robustness



The screenshot shows a VS Code editor with a Python file named `assignment_3.5.py`. The code defines a function `check_harshad_number` that takes an integer `num` and returns a boolean. The function checks if the number is divisible by the sum of its digits. Comments explain that Harshad numbers are typically positive integers. The code includes an example usage section that prompts the user to enter a number and prints the result.

```
101 def check_harshad_number(num: int) -> bool:
102     Args:
103         num (int): The integer to check.
104
105     Returns:
106         bool: True if the number is a Harshad number, False otherwise.
107     """
108     if num <= 0:
109         return False # Harshad numbers are typically positive integers
110     digit_sum = sum(int(digit) for digit in str(num))
111     return num % digit_sum == 0
112
113 # Example usage:
114 num = int(input("Enter a number: "))
115 if check_harshad_number(num):
116     print(f"{num} is a Harshad number.")
117 else:
118     print(f"{num} is not a Harshad number.")
```

The terminal output shows the execution of the program for three test cases:

```
PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter a number: 18
18 is a Harshad number.
PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter a number: 21
21 is a Harshad number.
PS D:\AI assisted coding> & C:/Users/vaish/AppData/Local/Programs/Python/Python314/python.exe "d:/AI assisted coding/assignment_3.5.py"
Enter a number: 19
19 is not a Harshad number.
PS D:\AI assisted coding>
```

The right sidebar shows a chat window with a few-shot prompt for the Harshad number check, including the function signature, docstring, and example usage.