

## AIAC Assignment 3.5

Name : K.Nikshitha

Ht.no : 2303A51692

Batch :24

### 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.

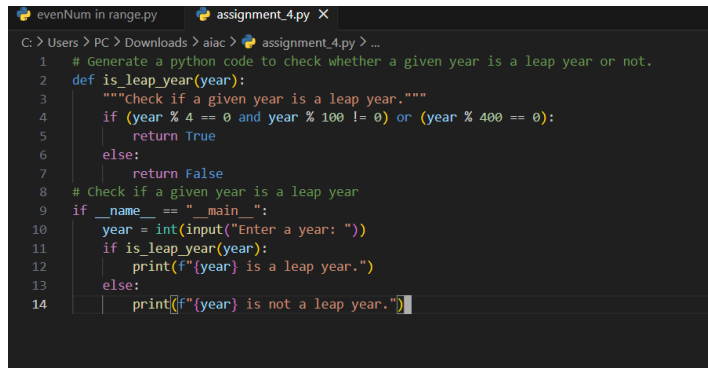
#### Task:

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

**Prompt :** # Generate a python code to check whether a given year is a leap year or not.

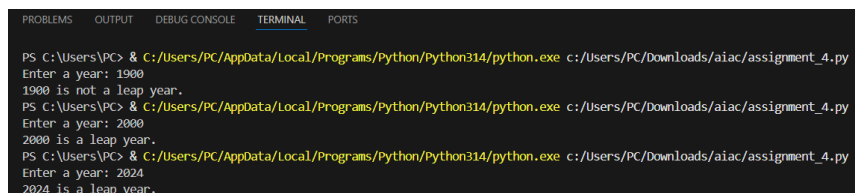
#### Code:

```
def is_leap_year(year):
    """Check if a given year is a leap year."""
    if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
        return True
    else:
        return False
# Check if a given year is a leap year
if __name__ == "__main__":
    year = int(input("Enter a year: "))
    if is_leap_year(year):
        print(f"{year} is a leap year.")
    else:
        print(f"{year} is not a leap year.")
```

A screenshot of a code editor window titled 'assignment\_4.py'. The code is a Python function 'is\_leap\_year' that checks if a year is a leap year based on the rules: divisible by 4 but not 100, or divisible by 400. The code also includes a main block that prompts the user for a year and prints the result. The code is as follows:

```
1 # Generate a python code to check whether a given year is a leap year or not.
2 def is_leap_year(year):
3     """Check if a given year is a leap year."""
4     if (year % 4 == 0 and year % 100 != 0) or (year % 400 == 0):
5         return True
6     else:
7         return False
8 # Check if a given year is a leap year
9 if __name__ == "__main__":
10     year = int(input("Enter a year: "))
11     if is_leap_year(year):
12         print(f"{year} is a leap year.")
13     else:
14         print(f"{year} is not a leap year.")
```

### OUTPUT :

A screenshot of a terminal window showing the execution of the Python code. The terminal displays the command to run the script and the resulting output for three different years: 1900, 2000, and 2024. The output is as follows:

```
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a year: 1900
1900 is not a leap year.
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a year: 2000
2000 is a leap year.
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a year: 2024
2024 is a leap year.
```

### 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 : (One short)

"""

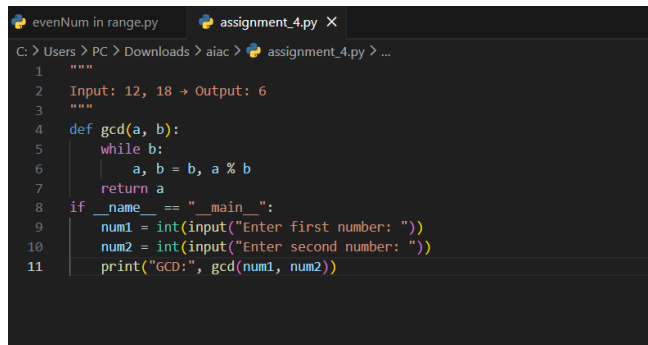
Input: 12, 18 → Output: 6

"""

#### Code:

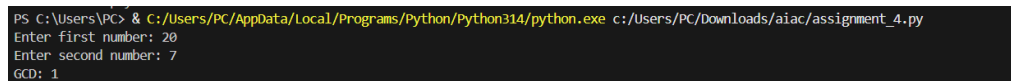
```
def gcd(a, b):
    while b:
        a, b = b, a % b
    return a

if __name__ == "__main__":
    num1 = int(input("Enter first number: "))
    num2 = int(input("Enter second number: "))
    print("GCD:", gcd(num1, num2))
```

A screenshot of a code editor with two tabs: 'evenNum in range.py' and 'assignment\_4.py'. The 'assignment\_4.py' tab is active, showing the Python code for calculating the GCD of two numbers. The code includes a docstring with an example input and output, a function definition for gcd, and a main block that takes user input and prints the result. The code is as follows:

```
1 """
2 Input: 12, 18 → Output: 6
3 """
4 def gcd(a, b):
5     while b:
6         a, b = b, a % b
7     return a
8 if __name__ == "__main__":
9     num1 = int(input("Enter first number: "))
10    num2 = int(input("Enter second number: "))
11    print("GCD:", gcd(num1, num2))
```

#### OUTPUT:

A screenshot of a terminal window showing the execution of the Python program. The prompt is 'PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiaac/assignment\_4.py'. The user enters '20' for the first number and '7' for the second number. The output is 'GCD: 1'.

```
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiaac/assignment_4.py
Enter first number: 20
Enter second number: 7
GCD: 1
```

### 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:** """This program calculates the Least Common Multiple (LCM) of two given numbers.

Input: 4, 6 → Output: 12

Input: 5, 10 → Output: 10

Input: 7, 3 → Output: 21

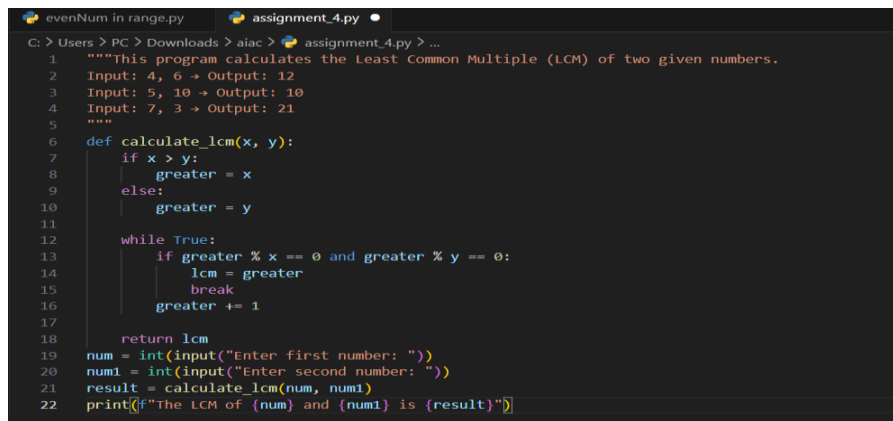
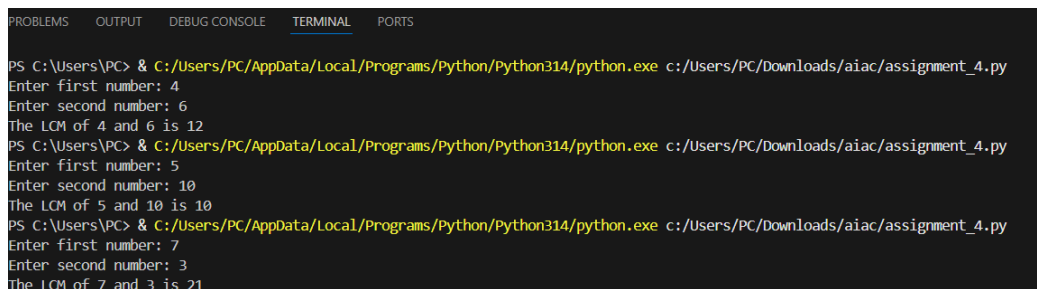
"""

**Code:**

```
def calculate_lcm(x, y):
    if x > y:
        greater = x
    else:
        greater = y

    while True:
        if greater % x == 0 and greater % y == 0:
            lcm = greater
            break
        greater += 1
    return lcm

num = int(input("Enter first number: "))
num1 = int(input("Enter second number: "))
result = calculate_lcm(num, num1)
print(f"The LCM of {num} and {num1} is {result}")
```

A screenshot of a code editor with a dark background. The editor shows a Python file named 'assignment\_4.py'. The code is a function 'calculate\_lcm(x, y)' that finds the Least Common Multiple (LCM) of two numbers. It uses a 'while' loop to increment a 'greater' variable until it is divisible by both 'x' and 'y'. The function then returns the LCM. Below the function, there are three test cases: (4, 6) resulting in 12, (5, 10) resulting in 10, and (7, 3) resulting in 21. The code is syntax-highlighted with colors like blue for keywords and green for comments.**OUTPUT :**A screenshot of a terminal window with a dark background. The terminal shows the execution of the Python program. It prompts the user to enter two numbers and displays the LCM for three different pairs: (4, 6) gives 12, (5, 10) gives 10, and (7, 3) gives 21. The terminal text is as follows:  
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aia/assignment\_4.py  
Enter first number: 4  
Enter second number: 6  
The LCM of 4 and 6 is 12  
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aia/assignment\_4.py  
Enter first number: 5  
Enter second number: 10  
The LCM of 5 and 10 is 10  
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aia/assignment\_4.py  
Enter first number: 7  
Enter second number: 3  
The LCM of 7 and 3 is 21**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.

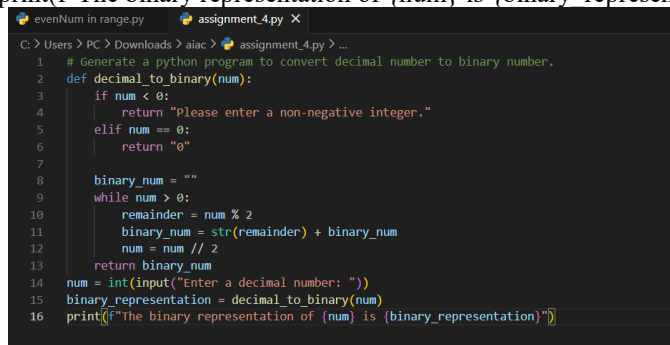
**Prompt :** # Generate a python program to convert decimal number to binary number.

**Code :**

```
def decimal_to_binary(num):
    if num < 0:
        return "Please enter a non-negative integer."
    elif num == 0:
        return "0"

    binary_num = ""
    while num > 0:
        remainder = num % 2
        binary_num = str(remainder) + binary_num
        num = num // 2
    return binary_num

num = int(input("Enter a decimal number: "))
binary_representation = decimal_to_binary(num)
print(f"The binary representation of {num} is {binary_representation}")
```



```
evenNum in range.py  assignment_4.py X
C:\Users\PC> Downloads > aiac > assignment_4.py > _
1 # generate a python program to convert decimal number to binary number.
2 def decimal_to_binary(num):
3     if num < 0:
4         return "Please enter a non-negative integer."
5     elif num == 0:
6         return "0"
7
8     binary_num = ""
9     while num > 0:
10        remainder = num % 2
11        binary_num = str(remainder) + binary_num
12        num = num // 2
13    return binary_num
14 num = int(input("Enter a decimal number: "))
15 binary_representation = decimal_to_binary(num)
16 print(f"The binary representation of {num} is {binary_representation}")
```

**OUTPUT :**

```
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a decimal number: 36
The binary representation of 36 is 100100
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a decimal number: 2
The binary representation of 2 is 10
```

**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 :** """

Input: 10 → Output: 1010  
"""

**Code:**

```
def decimal_to_binary(n):
    if n == 0:
        return "0"
    binary = ""
    while n > 0:
```

```

        binary = str(n % 2) + binary
        n = n // 2
    return binary
num = int(input("Enter a decimal number: "))
print(f"The binary representation of {num} is {decimal_to_binary(num)}")

```

```

evenNum in range.py  assignment_4.py X
C:\Users\PC> Downloads > aiac > assignment_4.py > ...
1  """
2  Input: 10 → Output: 1010
3  """
4  def decimal_to_binary(n):
5      if n == 0:
6          return "0"
7      binary = ""
8      while n > 0:
9          binary = str(n % 2) + binary
10         n = n // 2
11     return binary
12 num = int(input("Enter a decimal number: "))
13 print(f"The binary representation of {num} is {decimal_to_binary(num)}")

```

**OUTPUT :**

```

PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a decimal number: 10
The binary representation of 10 is 1010

```

### 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 :**

```

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

```

**Code :**

```

def is_harshad_number(n):
    digit_sum = sum(int(digit) for digit in str(n))
    if n % digit_sum == 0:
        return f"{n} is a Harshad Number"
    else:
        return f"{n} is Not a Harshad Number"
num = int(input("Enter a number: "))
print(is_harshad_number(num))

```

```
evenNum in range.py  assignment_4.py X
C: > Users > PC > Downloads > aiac > assignment_4.py > ...
1  """
2  Input: 18 -> Output: Harshad Number
3  Input: 21 -> Output: Harshad Number
4  Input: 19 -> Output: Not a Harshad Number
5
6  """
7  def is_harshad_number(n):
8      digit_sum = sum(int(digit) for digit in str(n))
9      if n % digit_sum == 0:
10         return f"{n} is a Harshad Number"
11     else:
12         return f"{n} is Not a Harshad Number"
13  num = int(input("Enter a number: "))
14  print(is_harshad_number(num))
```

## OUTPUT :

```
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a number: 18
18 is a Harshad Number
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a number: 21
21 is a Harshad Number
PS C:\Users\PC> & C:/Users/PC/AppData/Local/Programs/Python/Python314/python.exe c:/Users/PC/Downloads/aiac/assignment_4.py
Enter a number: 19
19 is Not a Harshad Number
```