

Lab Assignment - 3.5

Name : P. Gouri Prasad Varma

[Hall.t.No.](#) : 2303A51035

Batch No. : 01

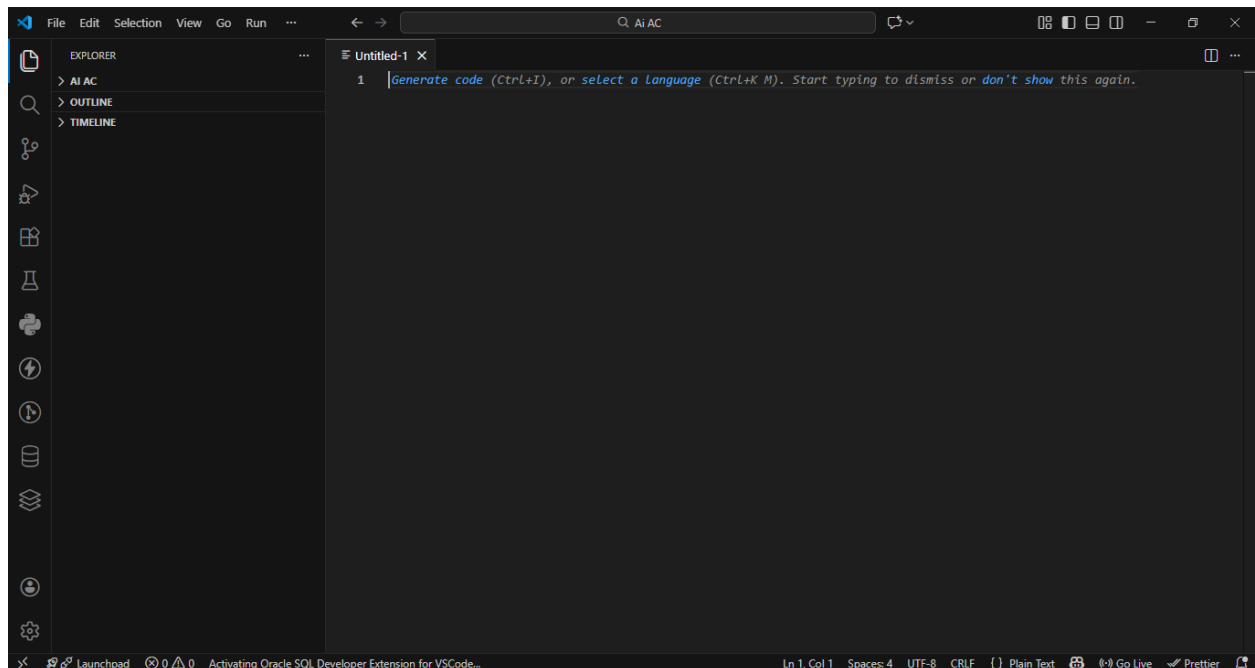
Question 1: Zero-Shot Prompting (Leap Year Check)

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

Task:

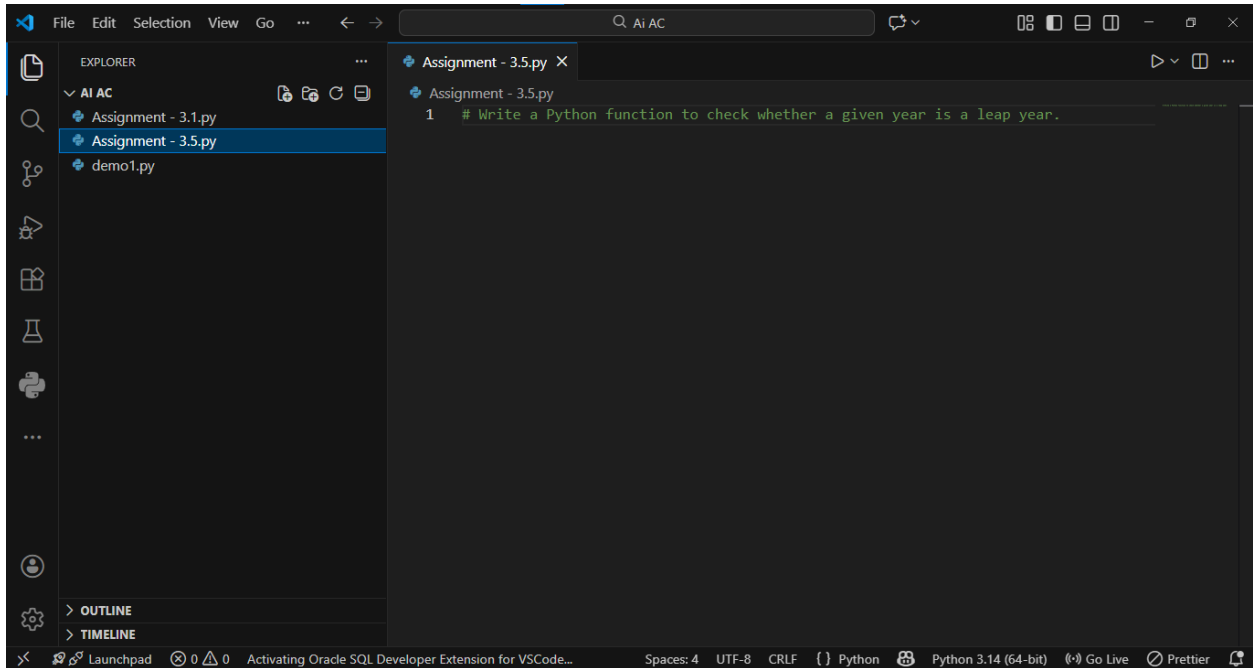
- Record the AI-generated code.
- Test with years like 1900, 2000, 2024.
- Identify logical flaws or missing conditions.

Copilot VSCode Workspace :

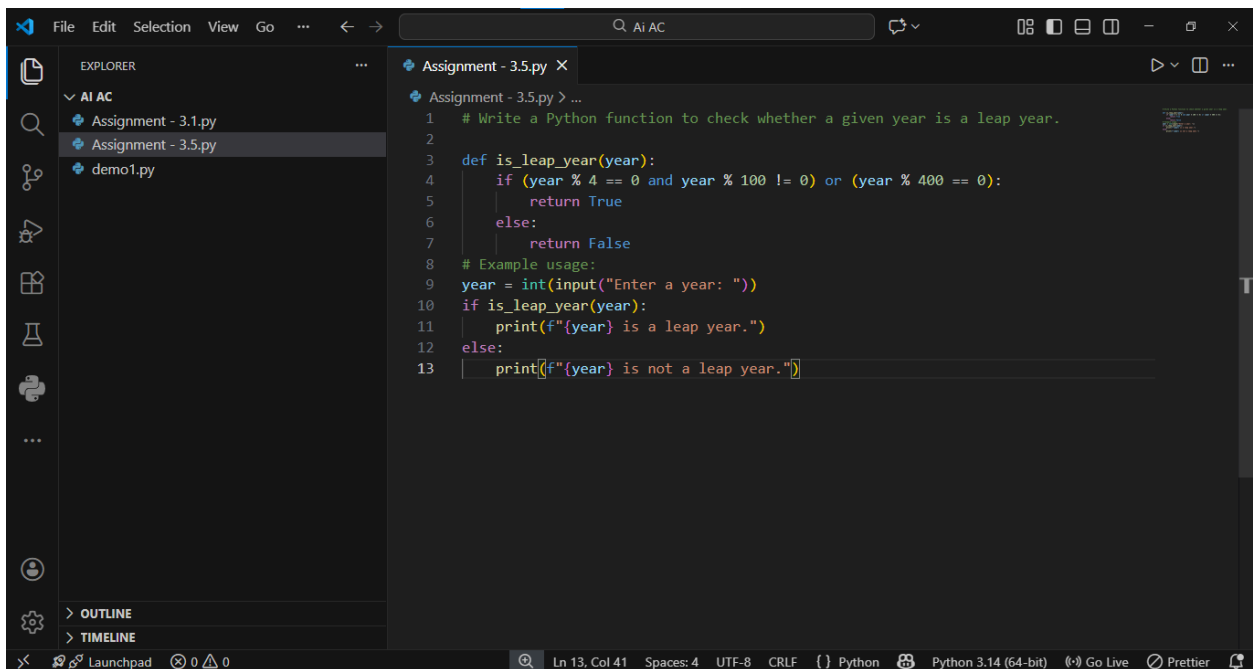


Prompt Used (Zero-Shot):

Write a Python function to check whether a given year is a leap year.



Copilot Suggested Code :



Input / Output test cases screenshot :

```
● PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
1900 is not a leap year.
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
● 2000 is a leap year.
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
Enter a year: 2024
● 2024 is a leap year.
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
Enter a year: 2023
● 2023 is not a leap year.
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
● Enter a year: 2100
2100 is not a leap year.
```

Observation & Logical Flaws :

- Fails for **century years**
- Missing condition:

Years divisible by **100** are **not leap years** unless divisible by **400**.

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 Used (One-Shot):

Write a Python function to find GCD of two numbers.

Example: Input: 12, 18 → Output: 6

```
14
15 # Write a Python function to find GCD of two numbers.
16 # Example: Input: 12, 18 -> Output: 6
```

Copilot Suggested Code :

```
14
15 # Write a Python function to find GCD of two numbers.
16 # Example: Input: 12, 18 -> Output: 6
17
18 def gcd(a, b):
19     while b:
20         a, b = b, a % b
21     return a
22
23 # Example usage:
24 num1 = int(input("Enter first number: "))
25 num2 = int(input("Enter second number: "))
26 print(f"The GCD of {num1} and {num2} is {gcd(num1, num2)}")
```

Input / Output test cases screenshot :

```
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
Enter first number: 12
Enter second number: 18
The GCD of 12 and 18 is 6
PS C:\Btech\3 - 2\Ai AC> |
```

Comparison with Zero-Shot :

Aspect	Zero-Shot	One-Shot
Clarity	Medium	High
Algorithm	Trial-based	Euclidean
Efficiency	Low	High

Efficiency Analysis :

- Time Complexity: $O(\log \min(a, b))$
- Very efficient for large numbers

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 \rightarrow Output: 12
- Input: 5, 10 \rightarrow Output: 10
- Input: 7, 3 \rightarrow Output: 21

Task:

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

Prompt Used (Few-Shot Prompt):

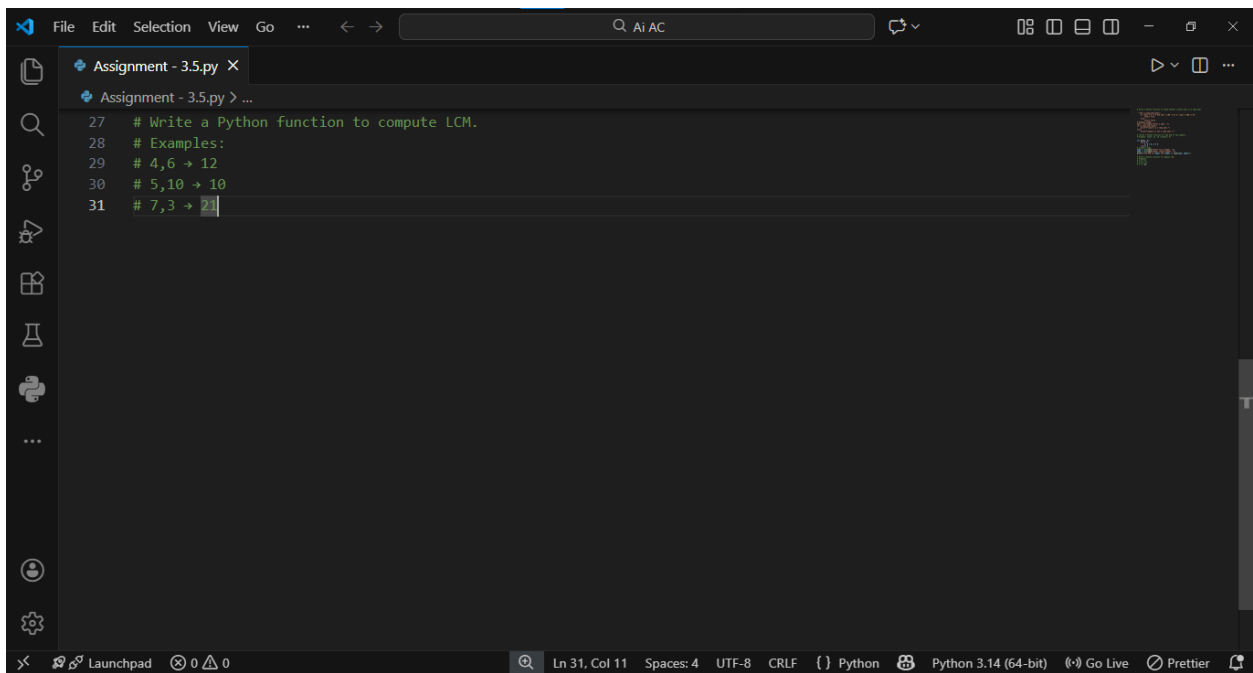
Write a Python function to compute LCM.

Examples:

4,6 \rightarrow 12

5,10 \rightarrow 10

7,3 \rightarrow 21

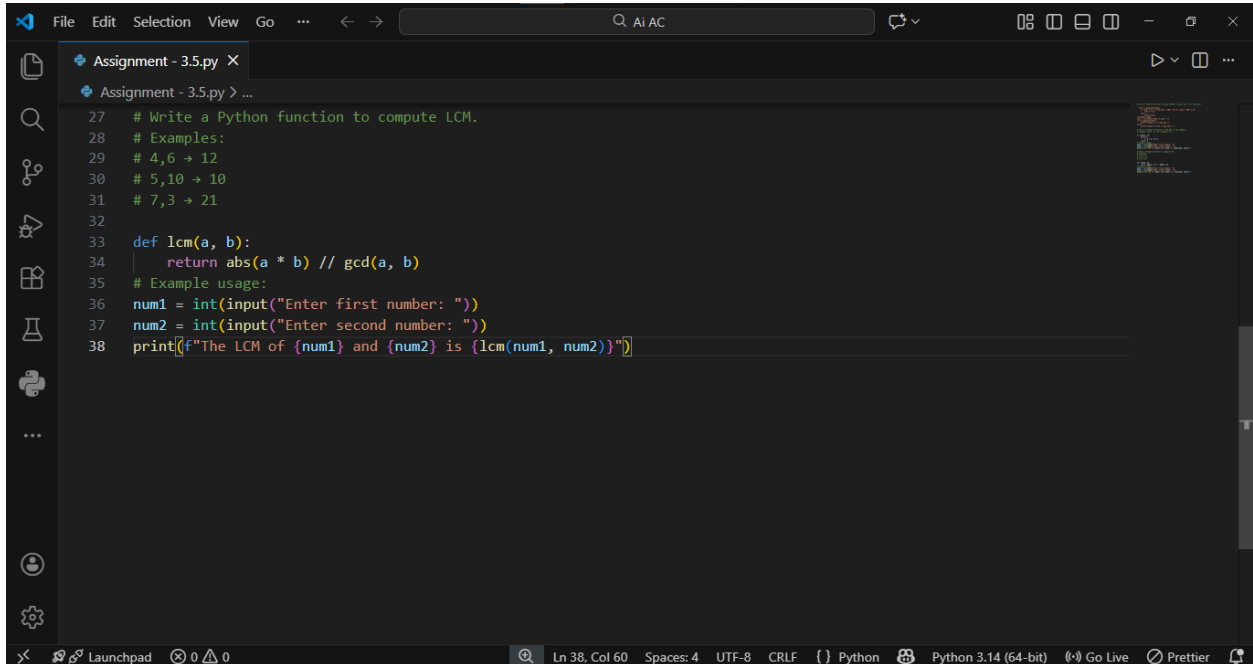


The screenshot shows a code editor window with a dark theme. The title bar at the top reads "Assignment - 3.5.py". The editor contains the following code:

```
27 # Write a Python function to compute LCM.  
28 # Examples:  
29 # 4,6  $\rightarrow$  12  
30 # 5,10  $\rightarrow$  10  
31 # 7,3  $\rightarrow$  21
```

The status bar at the bottom indicates the current position is "Ln 31, Col 11", the file encoding is "UTF-8", the line ending is "CRLF", the language is "Python", and the interpreter is "Python 3.14 (64-bit)". Other icons in the status bar include "Launchpad", "0 0", "Go Live", and "Prettier".

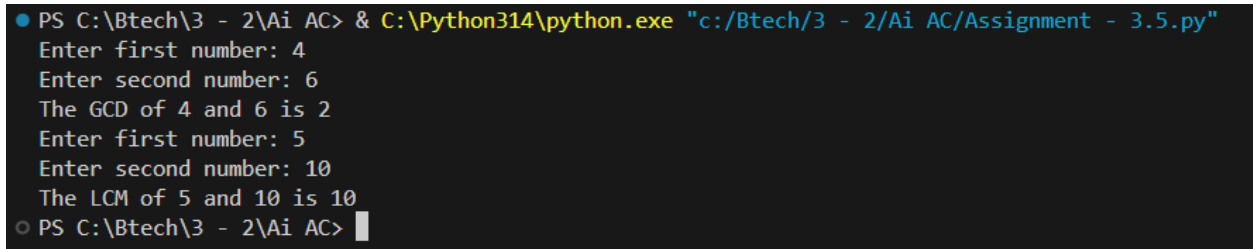
Copilot Suggested Code :



The screenshot shows a VS Code editor window with a file named 'Assignment - 3.5.py'. The code is a Python script that defines a function 'lcm(a, b)' which returns the Least Common Multiple of two numbers. It includes comments for examples and a usage example. The code is as follows:

```
27 # Write a Python function to compute LCM.
28 # Examples:
29 # 4,6 → 12
30 # 5,10 → 10
31 # 7,3 → 21
32
33 def lcm(a, b):
34     return abs(a * b) // gcd(a, b)
35 # Example usage:
36 num1 = int(input("Enter first number: "))
37 num2 = int(input("Enter second number: "))
38 print(f"The LCM of {num1} and {num2} is {lcm(num1, num2)}")
```

Input / Output test cases screenshot :



The screenshot shows a terminal window with the following output:

```
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
Enter first number: 4
Enter second number: 6
The GCD of 4 and 6 is 2
Enter first number: 5
Enter second number: 10
The LCM of 5 and 10 is 10
PS C:\Btech\3 - 2\Ai AC>
```

How Examples Guided the Model :

- Correctly inferred $LCM = (a \times b) / GCD$
- Used efficient GCD internally

Edge Case Testing :

```
print(lcm(0, 5))    # Output: 0
```

```
print(lcm(1, 1))    # Output: 1
```

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

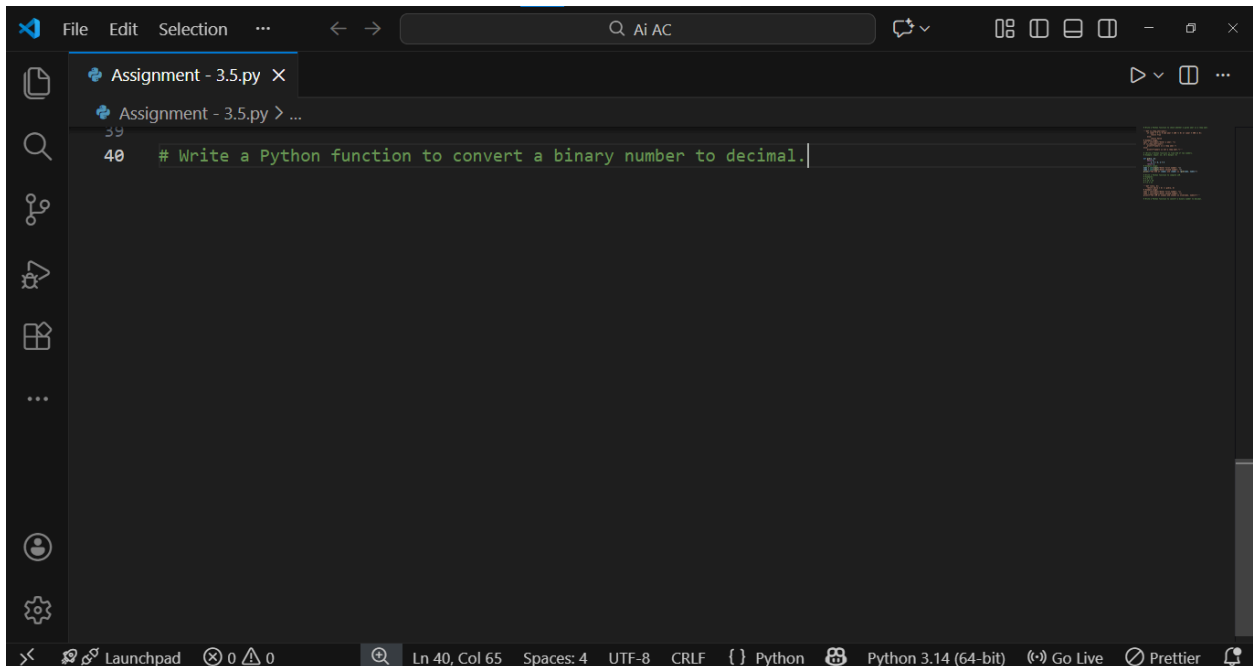
Write a zero-shot prompt to generate a Python function that converts a binary number to decimal.

Task:

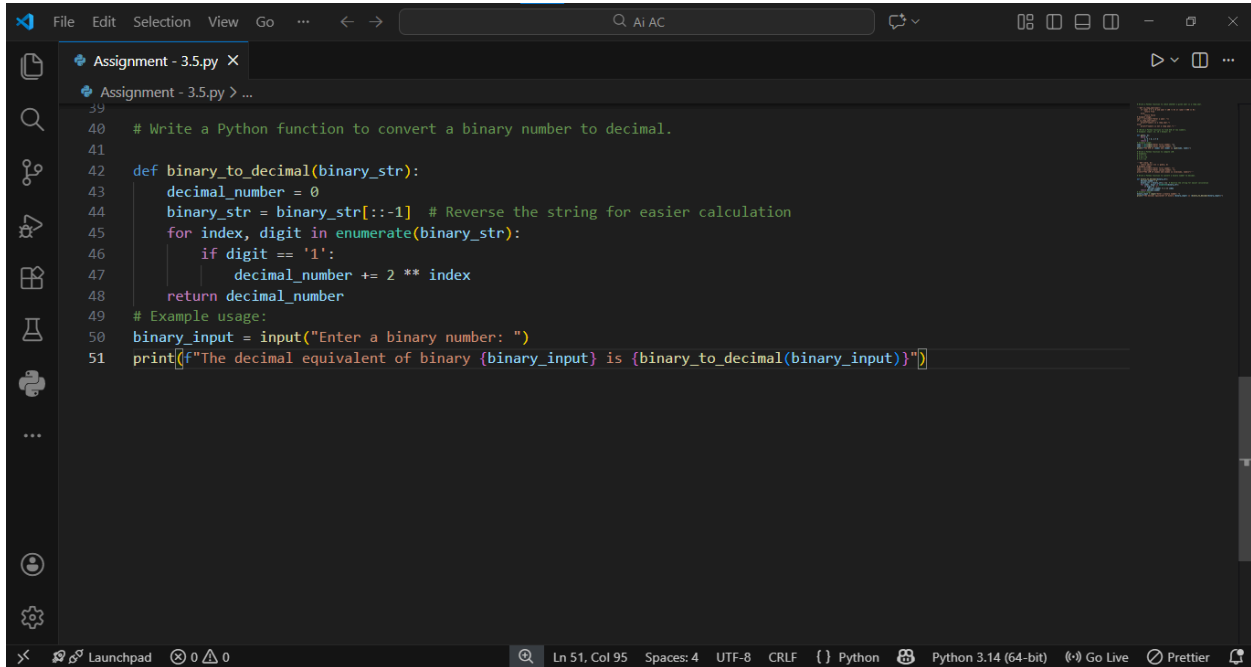
- Test with valid and invalid binary inputs.
- Identify missing validation logic.

Prompt Used (Zero-Shot):

Write a Python function to convert a binary number to decimal.



Copilot Suggested Code :

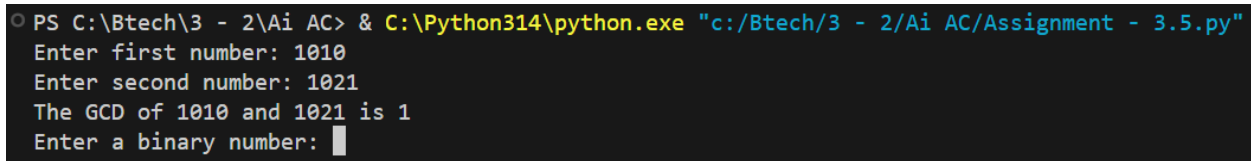


The screenshot shows a code editor with a dark theme. The file name is 'Assignment - 3.5.py'. The code is as follows:

```
39
40 # Write a Python function to convert a binary number to decimal.
41
42 def binary_to_decimal(binary_str):
43     decimal_number = 0
44     binary_str = binary_str[::-1] # Reverse the string for easier calculation
45     for index, digit in enumerate(binary_str):
46         if digit == '1':
47             decimal_number += 2 ** index
48     return decimal_number
49
50 # Example usage:
51 binary_input = input("Enter a binary number: ")
52 print(f"The decimal equivalent of binary {binary_input} is {binary_to_decimal(binary_input)}")
```

The status bar at the bottom indicates: Ln 51, Col 95, Spaces: 4, UTF-8, CRLF, Python, Python 3.14 (64-bit), Go Live, Prettier.

Input / Output test cases screenshot :



The screenshot shows a terminal window with the following output:

```
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
Enter first number: 1010
Enter second number: 1021
The GCD of 1010 and 1021 is 1
Enter a binary number: 
```

Testing :

`print(binary_to_decimal("1010"))` # Output: 10

`print(binary_to_decimal("1021"))` # Error

Missing Validation :

- No check for invalid binary digits

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

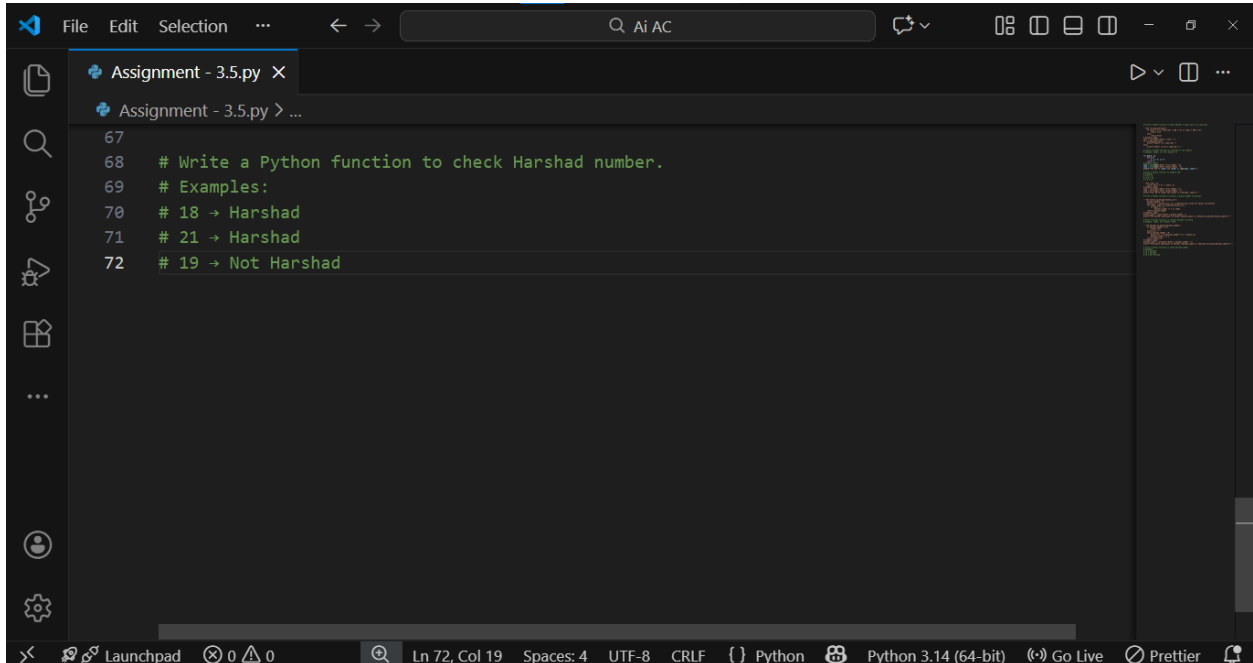
Task:

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

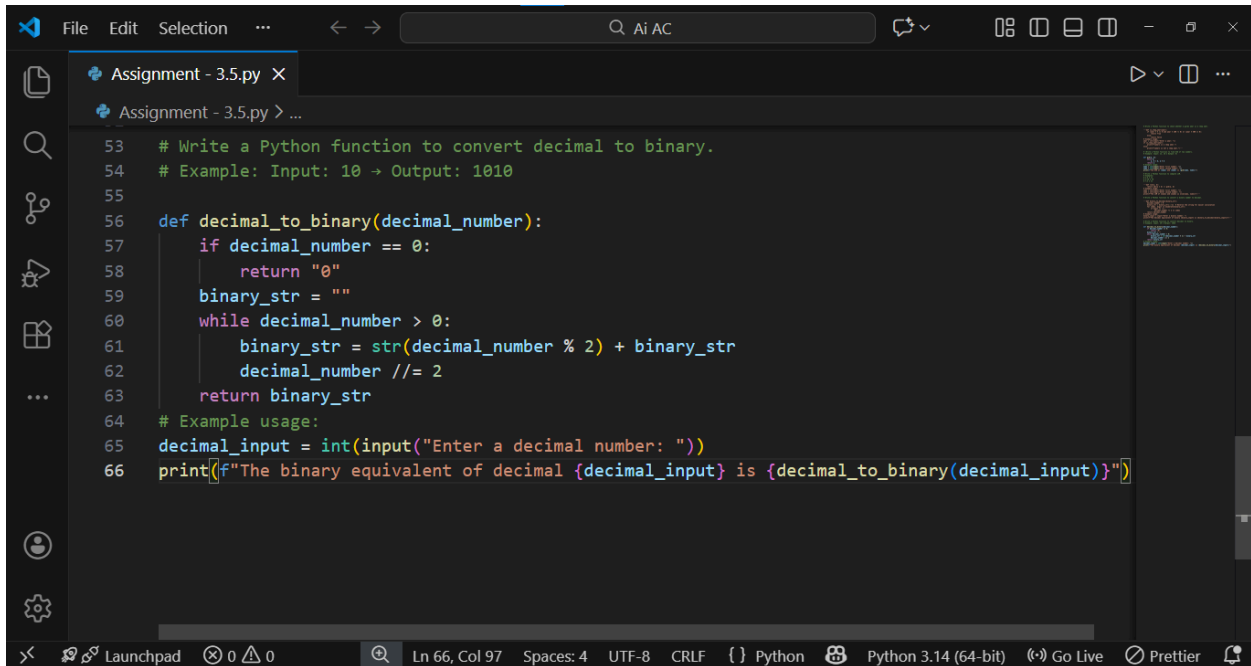
Prompt Used (One-Shot):

Write a Python function to convert decimal to binary.

Example: Input: 10 → Output: 1010



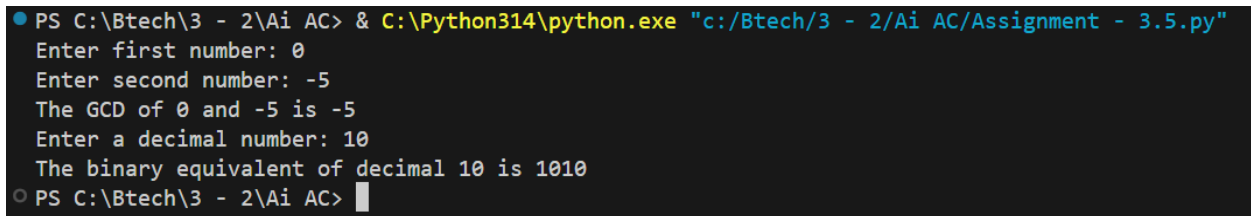
Copilot Suggested Code :



The screenshot shows a VS Code editor window with a file named 'Assignment - 3.5.py'. The code is a Python function 'decimal_to_binary' that converts a decimal number to a binary string. It includes comments and an example usage section. The status bar at the bottom indicates the file is at line 66, column 97, with 4 spaces, UTF-8 encoding, and CRLF line endings. The Python version is 3.14 (64-bit).

```
53 # Write a Python function to convert decimal to binary.
54 # Example: Input: 10 → Output: 1010
55
56 def decimal_to_binary(decimal_number):
57     if decimal_number == 0:
58         return "0"
59     binary_str = ""
60     while decimal_number > 0:
61         binary_str = str(decimal_number % 2) + binary_str
62         decimal_number //= 2
63     return binary_str
64
65 # Example usage:
66 decimal_input = int(input("Enter a decimal number: "))
67 print(f"The binary equivalent of decimal {decimal_input} is {decimal_to_binary(decimal_input)}")
```

Input / Output test cases screenshot :



The screenshot shows a terminal window with the following output:

```
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
Enter first number: 0
Enter second number: -5
The GCD of 0 and -5 is -5
Enter a decimal number: 10
The binary equivalent of decimal 10 is 1010
PS C:\Btech\3 - 2\Ai AC>
```

Edge Case Analysis :

`print(decimal_to_binary(0))` # Output: 0

`print(decimal_to_binary(-5))` # Output: b101
(incorrect)

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 Used (Few-Shot Prompt):

Write a Python function to check Harshad number.

Examples:

18 → Harshad

21 → Harshad

19 → Not Harshad

```
67
68 # Write a Python function to check Harshad number.
69 # Examples:
70 # 18 → Harshad
71 # 21 → Harshad
72 # 19 → Not Harshad
```

Ln 72, Col 19

Copilot Suggested Code :

```
67
68 # Write a Python function to check Harshad number.
69 # Examples:
70 # 18 → Harshad
71 # 21 → Harshad
72 # 19 → Not Harshad
73
74 def is_harshad_number(number):
75     digit_sum = sum(int(digit) for digit in str(number))
76     return number % digit_sum == 0
77 # Example usage:
78 num = int(input("Enter a number: "))
79 if is_harshad_number(num):
80     print(f"{num} is a Harshad number.")
81 else:
82     print(f"{num} is not a Harshad number.")
```

Ln 82, Col 45

Input / Output test cases screenshot :

```
PS C:\Btech\3 - 2\Ai AC> & C:\Python314\python.exe "c:/Btech/3 - 2/Ai AC/Assignment - 3.5.py"
Enter first number: 18
Enter second number: 19
The GCD of 18 and 19 is 1
Enter a number: 21
21 is a Harshad number.
PS C:\Btech\3 - 2\Ai AC> █
```

Testing :

```
print(is_harshad(18)) # True
```

```
print(is_harshad(21)) # True
```

```
print(is_harshad(19)) # False
```

Boundary Conditions :

- Fails for $n = 0$ (division by zero)

Robust Version :

```
def is_harshad(n):
    if n <= 0:
        return False

    digit_sum = sum(int(d) for d in str(n))
    return n % digit_sum == 0
```