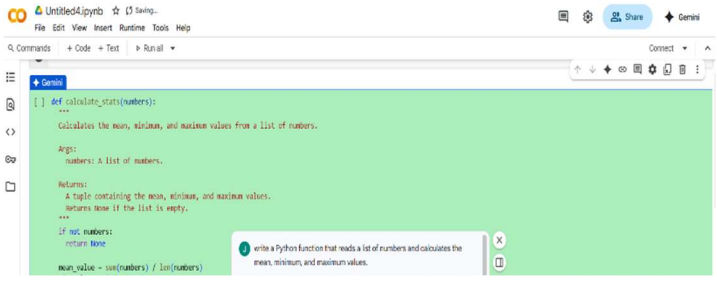
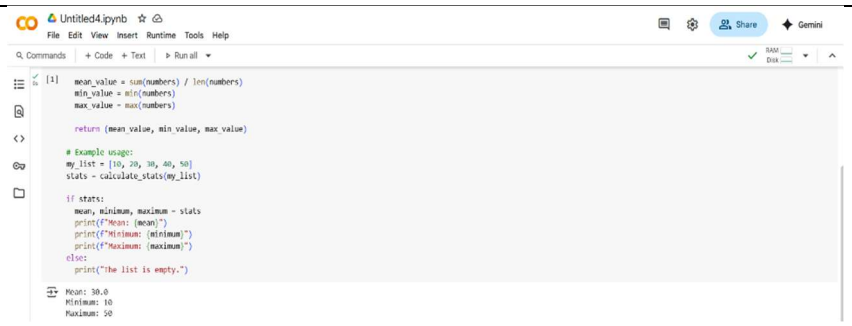
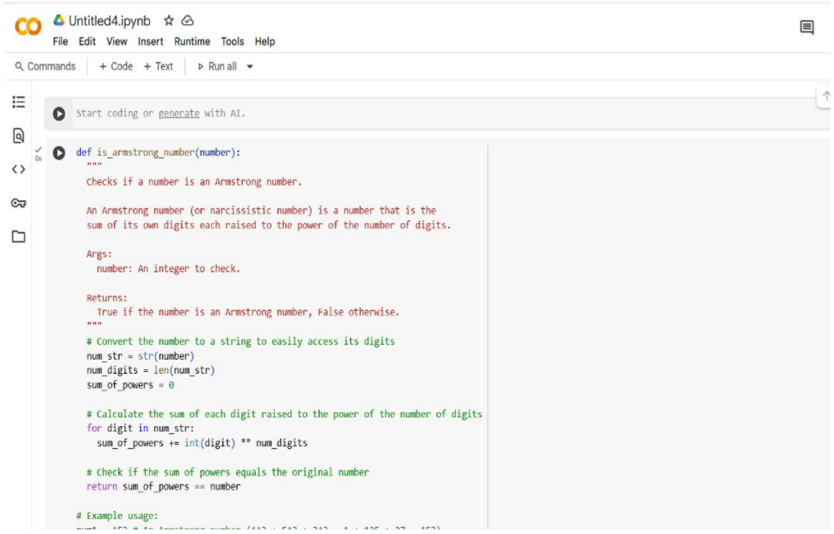
Q.No.	**AI ASSISTED CODING** **NAME:J.KEERTHI PRIYA** **ROLL NO:2403A510G4** **ASSIGNMENT:2.1**	
	Lab Outcomes (LOs): After completing this lab, students will be able to: ● Generate Python code using Google Gemini in Google Colab. ● Analyze the effectiveness of code explanations and suggestions by Gemini. ● Set up and use Cursor AI for AI-powered coding assistance. ● Evaluate and refactor code using Cursor AI features. ● Compare AI tool behavior and code quality across different platforms.	
1	Task Description #1 ● Use Google Gemini in Colab to write a Python function that reads a list of numbers and calculates the mean, minimum, and maximum values. Expected Output #1 ● Functional code with correct output and screenshot. ●  ● **CODE:** ●  ● **OUTPUT:**	

- 

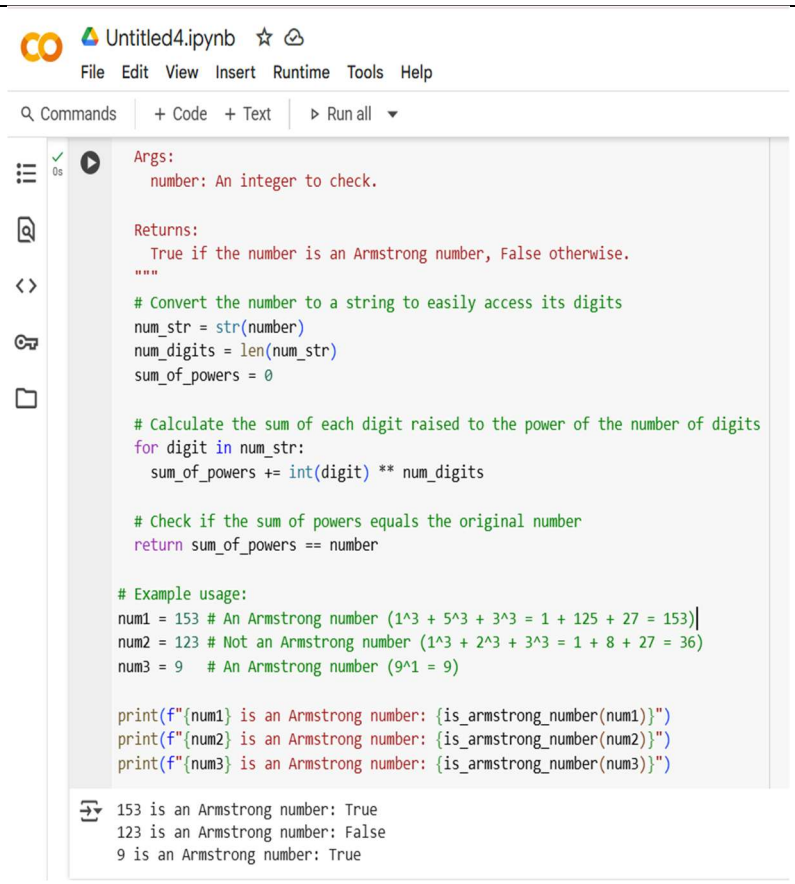
Task Description #2

- Compare Gemini and Copilot outputs for a Python function that checks whether a number is an Armstrong number. Document the steps, prompts, and outputs.

Expected Output #2

- Side-by-side comparison table with observations and screenshots.
- **CODE:**
- 

- **OUTPUT:**
-



- **EXPLANATION:**

- **Test with different inputs:** Try calling the function with a wider range of numbers, including larger numbers, to see how it performs.
- **Find Armstrong numbers within a range:** Write a script or another function that iterates through a range of numbers (e.g., from 1 to 1000) and uses the `is_armstrong_number` function to identify and print all Armstrong numbers within that range.
- **Optimize the function:** For very large numbers, converting to a string and back might not be the most efficient approach. You could explore alternative ways to extract digits and calculate the sum of

function that iterates through a range of numbers (e.g., from 1 to 1000) and uses the `is_armstrong_number` function to identify and print all Armstrong numbers within that range.

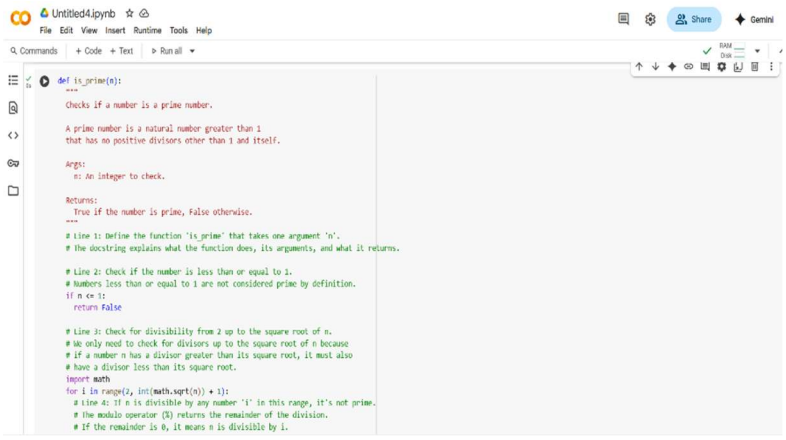
- **Optimize the function:** For very large numbers, converting to a string and back might not be the most efficient approach. You could explore alternative ways to extract digits and calculate the sum of powers using mathematical operations.
- **Explore other types of "narcissistic" numbers:** Research and implement functions to check for other types of numbers with similar properties, such as perfect digital invariants.

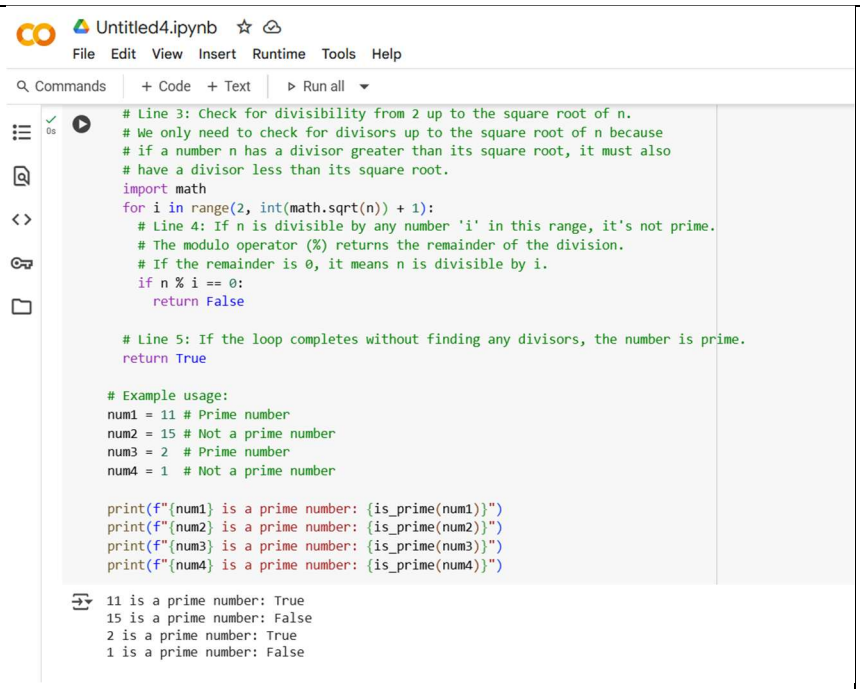
Task Description #3

- Ask Gemini to explain a Python function (e.g., is_prime(n) or is_palindrome(s)) line by line.
- Choose either a prime-checking or palindrome-checking function and document the explanation provided by Gemini.

Expected Output #3

- Detailed explanation with the code snippet and Gemini's response.
- **CODE and LINE BY LINE EXPLANATION:**



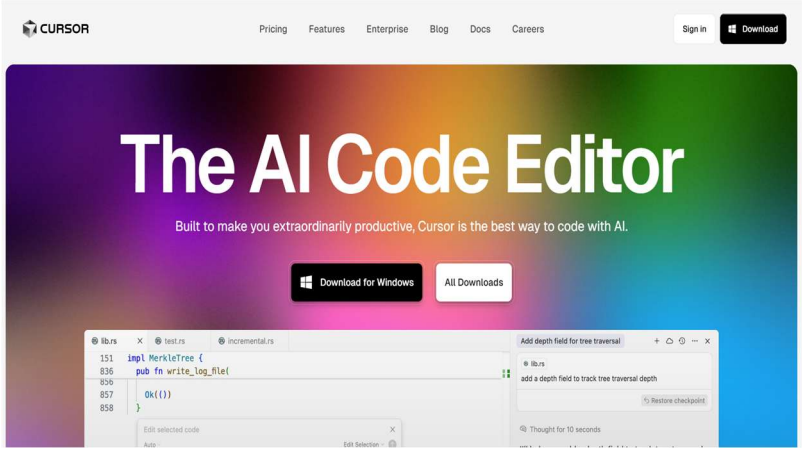


Task Description #4

- Install and configure Cursor AI. Use it to generate a Python function (e.g., sum of the first N natural numbers) and test its output.
- Optionally, compare Cursor AI's generated code with Gemini's output.

Expected Output #4

- Screenshots of Cursor AI setup, prompts used, and generated code with output.
- **INSTALLIZATION OF CURSOR AI:**



- **CURSOR AI's GENERATED CODE :**

```
def sum_of_first_n_naturals(n):
    """
    Calculates the sum of the first N natural numbers.

    Args:
      n: An integer representing the number of natural numbers to sum.

    Returns:
      The sum of the first N natural numbers.
      Returns 0 if n is less than 1.
    """
    if n < 1:
      return 0
    else:
      # The sum of the first N natural numbers can be calculated using the formula: n * (n + 1) / 2
      return n * (n + 1) // 2

# Example usage:
num1 = 5
num2 = 10
num3 = 0
num4 = -3

print(f"The sum of the first {num1} natural numbers is: {sum_of_first_n_naturals(num1)}")
print(f"The sum of the first {num2} natural numbers is: {sum_of_first_n_naturals(num2)}")
print(f"The sum of the first {num3} natural numbers is: {sum_of_first_n_naturals(num3)}")
print(f"The sum of the first {num4} natural numbers is: {sum_of_first_n_naturals(num4)}")
```

- **GEMINI's OUTPUT:**

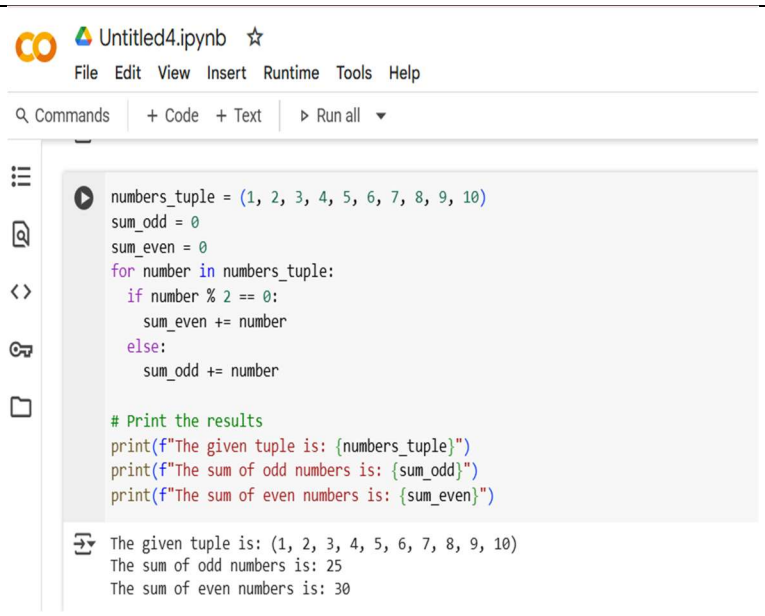
```
The sum of the first 5 natural numbers is: 15
The sum of the first 10 natural numbers is: 55
The sum of the first 0 natural numbers is: 0
The sum of the first -3 natural numbers is: 0
```

Task Description #5

- Students need to write a Python program to calculate the sum of odd numbers and even numbers in a given tuple.
- Refactor the code to improve logic and readability.

Expected Output #5

- Student-written refactored code with explanations and output screenshots.
- CODE WITH OUTPUT :

- 

Note:

- Students must submit a single Word document including:
  - Prompts used for AI tools
  - Copilot/Gemini/Cursor outputs
  - Code explanations
  - Screenshots of outputs and environments

Evaluation Criteria:

Criteria	Max Marks
Successful Use of Gemini in Colab (Task#1 & #2)	1.0
Code Explanation Accuracy (Gemini) (Task#3)	0.5
Cursor AI Setup and Usage (Task#4)	0.5
Refactoring and Improvement Analysis (Task#5)	0.5
Total	**2.5 Marks**