SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
Program Name: B. Tech		**Assignment Type: Lab**	**Academic Year:**2025-2026
Course Coordinator Name	Venkataramana Veeramsetty		
Instructor(s) Name	Dr. V. Venkataramana (Co-Ordinator) Dr. T. Sampath Kumar Dr. Pramoda Patro Dr. Brij Kishor Tiwari Dr.J.Ravichander Dr. Mohammand Ali Shaik Dr. Anirodh Kumar Mr. S.Naresh Kumar Dr. RAJESH VELPULA Mr. Kundhan Kumar Ms. Ch.Rajitha Mr. M Prakash Mr. B.Raju Intern 1 (Dharma teja) Intern 2 (Sai Prasad) Intern 3 (Sowmya) NS_2 (Mounika)		
Course Code	24CS002PC215	**Course Title**	AI Assisted Coding
Year/Sem	II/I	**Regulation**	R24
Date and Day of Assignment	Week1 - Monday	**Time(s)**	
Duration	2 Hours	**Applicable to Batches**	
Assignment Number: 2.1(Present assignment number)/**24**(Total number of assignments)			

Q.No.	Question	*Expected Time to complete*
1	Lab 2: Exploring Additional AI Coding Tools – Gemini (Colab) and Cursor AI **Lab Objectives:** • To explore and evaluate the functionality of Google Gemini for AI-assisted coding within Google Colab. • To understand and use Cursor AI for code generation,	Week1 - Monday

explanation, and refactoring.

- To compare outputs and usability between Gemini, GitHub Copilot, and Cursor AI.
- To perform code optimization and documentation using AI tools.

Lab Outcomes (LOs):

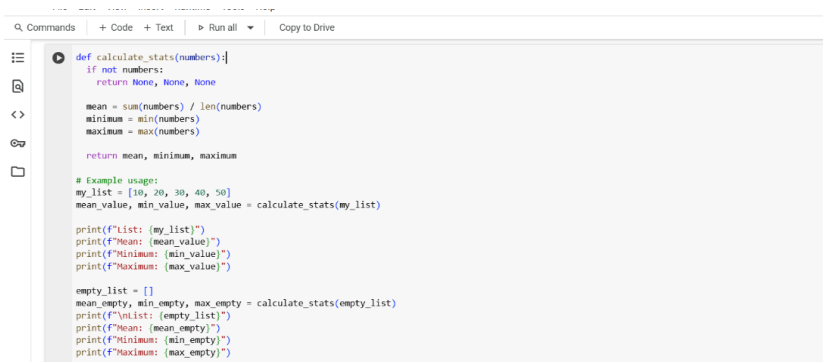
After completing this lab, students will be able to:

- 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.

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.

- **Prompt : 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.

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.

nmands + Code + Text ▷ Run all Copy to Drive

```python
def is_armstrong_number(number):
  # Convert the number to a string to find the number of digits
  num_str = str(number)
  num_digits = len(num_str)
  sum_of_powers = 0

  # Calculate the sum of each digit raised to the power of the number of digits
  for digit in num_str:
    sum_of_powers += int(digit) ** num_digits

  # Check if the sum is equal to the original number
  return sum_of_powers == number

# Example usage:
num1 = 153
num2 = 123
num3 = 9474

print(f"{num1} is an Armstrong number: {is_armstrong_number(num1)}")
print(f"{num2} is an Armstrong number: {is_armstrong_number(num2)}")
print(f"{num3} is an Armstrong number: {is_armstrong_number(num3)}")
```

- **Prompt : write a python function that checks whether a number is an Armstrong number.**

Expected Output #2

- Side-by-side comparison table with observations and screenshots.

```
153 is an Armstrong number: True
123 is an Armstrong number: False
9474 is an Armstrong number: True
```

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.

- **Prompt : write a python function that checks whether a given number is prime or not.**

ands + Code + Text ▷ Run all Copy to Drive

```python
def is_prime(number):
  if number <= 1:
    return False  # Numbers less than or equal to 1 are not prime
  if number <= 3:
    return True   # 2 and 3 are prime numbers
  if number % 2 == 0 or number % 3 == 0:
    return False  # Multiples of 2 or 3 are not prime

  # Check for prime numbers greater than 3
  i = 5
  while i * i <= number:
    if number % i == 0 or number % (i + 2) == 0:
      return False
    i += 6

  return True

# Example usage:
num1 = 11
num2 = 15
num3 = 2

print(f"{num1} is prime: {is_prime(num1)}")
print(f"{num2} is prime: {is_prime(num2)}")
print(f"{num3} is prime: {is_prime(num3)}")
```

Expected Output #3

- Detailed explanation with the code snippet and Gemini's response.

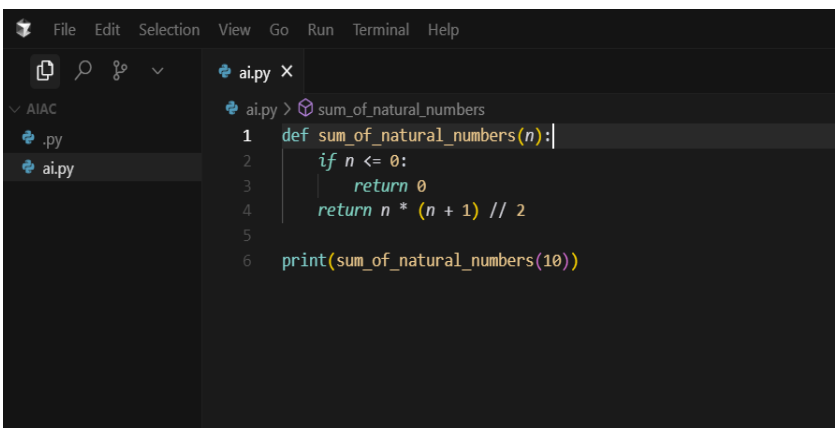
```
print(f"{num3} is prime: {is_prime(num3)}")

11 is prime: True
15 is prime: False
2 is prime: True
```

Task Description #4

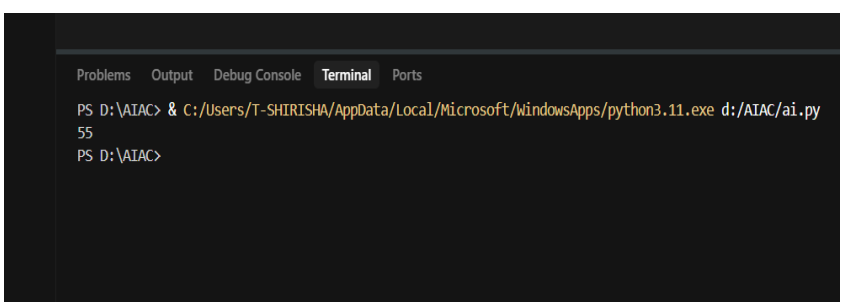
- Install and configure Cursor AI. Use it to generate a Python function (e.g., sum of the first N natural numbers) and test its output.
- Optionally, compare Cursor AI's generated code with Gemini's output.

- **Prompt : write a python function for sum of first N natural numbers.**



Expected Output #4

- Screenshots of Cursor AI setup, prompts used, and generated code with output.



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.

- **Prompt : write a python function to calculate the sum of odd numbers and even numbers in a given tuple.**

```python
def sum_of_odd_and_even_numbers(tuple):
    odd_sum = 0
    even_sum = 0
    for number in tuple:
        if number % 2 == 0:
            even_sum += number
        else:
            odd_sum += number
    return odd_sum, even_sum
print(sum_of_odd_and_even_numbers((1, 2, 3, 4, 5, 6, 7, 8, 9, 10)))
```

Expected Output #5

- Student-written refactored code with explanations and output screenshots.

```
Problems   Output   Debug Console   Terminal   Ports
PS D:\AIAC> & C:/Users/T-SHIRISHA/AppData/Local/Microsoft/WindowsApps/python3.11.exe d:/AIAC/ai.py
(25, 30)
PS D:\AIAC>
```

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**