

NAME:- Donthi Meghana

id:2403A510D9

batch-05

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
ProgramName: B. Tech		Assignment Type: Lab	AcademicYear: 2025-2026
CourseCoordinatorName		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)			
CourseCode	24CS002PC215	CourseTitle	AI Assisted Coding
Year/Sem	II/I	Regulation	R24
Date and Day of Assignment	Week 1 - Wednesday	Time(s)	
Duration	2 Hours	Applicable to Batches	24CSBTB01 To 24CSBTB39
AssignmentNumber: 2.3 (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:	Week 1 - Wednesday	

	<ul style="list-style-type: none">● 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, explanation, and refactoring.● To compare outputs and usability between Gemini, GitHub Copilot, and Cursor AI.● To perform code optimization and documentation using AI tools. <p>Lab Outcomes (LOs): After completing this lab, students will be able to:</p> <ul style="list-style-type: none">● 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. <p>Task Description#1</p> <ul style="list-style-type: none">● Use Google Gemini in Colab to write a function that reads a CSV file and calculates mean, min, max. <p>Prompt:</p> <div>Use Google Gemini in Colab to write a function that reads a CSV file and calculates mean, min, max</div> <p>Code:</p>	
--	--	--

```
import pandas as pd

def csv_stats(file_path):
    """
    Reads a CSV file and calculates mean, min, and max for each numeric column.
    Args:
        file_path (str): Path to the CSV file.
    Returns:
        dict: Dictionary with column names as keys and stats as values.
    """
    df = pd.read_csv(file_path)
    stats = {}
    for col in df.select_dtypes(include='number').columns:
        stats[col] = {
            'mean': df[col].mean(),
            'min': df[col].min(),
            'max': df[col].max()
        }
    return stats

# Example usage in Colab:
# stats = csv_stats('/content/sample.csv')
# print(stats)
```

Expected Output#1

- Functional code with output and screenshot

```
age,salary
22,40000
25,50000
30,60000
```

```
{'age': {'mean': 25.666666666666668, 'min': 22, 'max': 30},
 'salary': {'mean': 50000.0, 'min': 40000, 'max': 60000}}
```

Task Description#2

- Compare Gemini and Copilot outputs for a palindrome check function.

Prompt:

- Compare Gemini and Copilot outputs for a palindrome check function.

Code:

Gemini Output

Code:

```
def is_palindrome(s):  
    """  
    Checks if a string is a palindrome.  
    Args:  
        s (str): Input string.  
    Returns:  
        bool: True if palindrome, False otherwise.  
    """  
    s = s.lower().replace(" ", "")  
    return s == s[::-1]  
  
# Example usage:  
print(is_palindrome("Madam"))      # Output: True  
print(is_palindrome("Hello"))      # Output: False  
print(is_palindrome("A man a plan a canal Panama")) # Output: True
```

Copilot Output

Code:

```
def is_palindrome(s):  
    return s == s[::-1]  
  
# Example usage:  
print(is_palindrome("Madam"))      # Output: False  
print(is_palindrome("Hello"))      # Output: False  
print(is_palindrome("A man a plan a canal Panama")) # Output: False
```

Output:

Expected Output#2

- Side-by-side comparison and observations

Gemini output:

Output:

```
True  
False  
True
```

Copilot:

Output:

```
False
False
False
```

Task Description#3

- Ask Gemini to explain a Python function (to calculate area of various shapes) line by line..

Prompt:

- Ask Gemini to explain a Python function (to calculate area of various shapes) line by line.. and give the output

Code:

```
def area(shape, *params):
    """
    Calculates area for different shapes.
    shape: 'circle', 'rectangle', or 'triangle'
    params: dimensions needed for each shape
    """
    if shape == 'circle':
        # params[0] = radius
        return 3.14159 * params[0] ** 2
    elif shape == 'rectangle':
        # params[0] = length, params[1] = width
        return params[0] * params[1]
    elif shape == 'triangle':
        # params[0] = base, params[1] = height
        return 0.5 * params[0] * params[1]
    else:
        return None
```

```
print(area('circle', 3))
print(area('rectangle', 4, 5))
print(area('triangle', 6, 8))
```

Expected Output#3

- Detailed explanation with code snippet

Output:

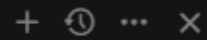
```
28.27431
20
24.0
```

Task Description#4

- Install and configure Cursor AI. Use it to generate a Python function (e.g., sum of squares)

Prompt:

Generate Python function for sum of squares



Code:

```
def sum_of_squares(numbers):
    """
    Calculate the sum of squares of a list of numbers.

    Args:
        numbers (list): A list of numbers (integers or floats)

    Returns:
        float: The sum of squares of all numbers in the list

    Example:
        >>> sum_of_squares([1, 2, 3, 4])
        30.0
    """
    if not numbers:
        return 0.0

    return sum(num ** 2 for num in numbers)

def main():
    # Example usage
    test_numbers = [1, 2, 3, 4, 5]
    result = sum_of_squares(test_numbers)
```

```
def main():
    # Example usage
    test_numbers = [1, 2, 3, 4, 5]
    result = sum_of_squares(test_numbers)

    print(f"Numbers: {test_numbers}")
    print(f"Sum of squares: {result}")

    # Interactive input
    print("\nEnter numbers separated by spaces (e.g., 1 2 3 4):")
    try:
        user_input = input("Numbers: ")
        user_numbers = [float(x) for x in user_input.split()]
        user_result = sum_of_squares(user_numbers)
        print(f"Sum of squares: {user_result}")
    except ValueError:
        print("Invalid input. Please enter valid numbers separated by spaces.")
    except KeyboardInterrupt:
        print("\nProgram terminated by user.")

if __name__ == "__main__":
```

Expected Output#4

- Screenshots of working environments with few prompts to generate python code

• For numbers [1, 2, 3, 4, 5]: $1^2 + 2^2 + 3^2 + 4^2 + 5^2 = 1 + 4 + 9 + 16 + 25 = 55$

Task Description#5

- Student need to write code to calculate sum of add number and even numbers in the list

Prompt:

- Student need to write code to calculate sum of add number and even numbers in the list give the output

Code:

```
numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]

even_sum = sum(num for num in numbers if num % 2 == 0)
odd_sum = sum(num for num in numbers if num % 2 != 0)

print("Sum of even numbers:", even_sum)
print("Sum of odd numbers:", odd_sum)
```

Expected Output#5

- Refactored code written by student with improved logic

```
Sum of even numbers: 30
Sum of odd numbers: 25
```

Note: Report should be submitted a word document for all tasks in a single document with prompts, comments & code explanation, and output and if required, screenshots

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		