NAME:sirivella sanjana **id:2403A510D4** **batch-05**

SCHOOLOFCOMPUTERSCIENCEANDARTIFICIAL INTELLIGENCE		**DEPARTMENTOFCOMPUTERSCIENCE ENGINEERING**	
ProgramName:B.Tech		**AssignmentType:Lab**	**AcademicYear:**2025-2026
CourseCoordinatorName	VenkataramanaVeeramsetty		
Instructor(s)Name	Dr.V.Venkataramana(Co-ordinator) Dr.T.SampathKumar Dr.PramodaPatro Dr.BrijKishor Tiwari Dr.J.Ravichander Dr.MohammandAliShaik Dr.AnirodhKumar Mr.S.Naresh Kumar Dr.RAJESHVELPULA Mr.KundhanKumar Ms.Ch.Rajitha Mr.MPrakash Mr.B.Raju Intern1(Dharmateja) Intern2(Sai Prasad) Intern3(Sowmya) NS_2(Mounika)		
CourseCode	24CS002PC215	**CourseTitle**	AIAssistedCoding
Year/Sem	II/I	**Regulation**	R24
DateandDay of Assignment	Week1 - Wednesday	**Time(s)**	
Duration	2 Hours	**Applicableto Batches**	24CSBTB01To 24CSBTB39
AssignmentNumber:2.3(Presentassignmentnumber)/**24**(Totalnumberofassignments)			

Q.No.	**Question**	***ExpectedTime to complete***
1	Lab2:Exploring AdditionalAICodingTools–Gemini(Colab)and CursorAI **Lab Objectives:**	Week1 - Wednesday

- ToexploreandevaluatethefunctionalityofGoogleGeminiforAI-assistedcoding within Google Colab.
- Tounderstand anduseCursorAIforcodegeneration,explanation,and refactoring.
- TocompareoutputsandusabilitybetweenGemini,GitHubCopilot,andCursorAI.
- Toperformcodeoptimization anddocumentationusingAItools.

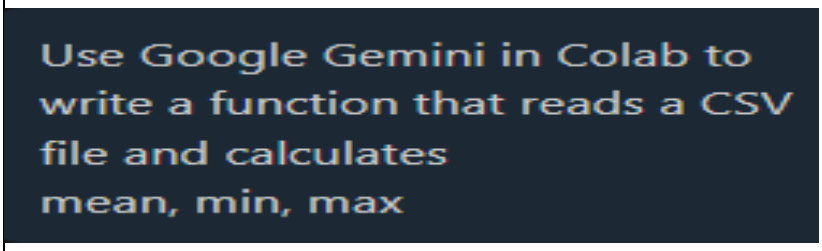
LabOutcomes (LOs):
Aftercompleting thislab,studentswill beableto:

- GeneratePython codeusingGoogleGeminiin GoogleColab.
- Analyzetheeffectivenessofcodeexplanationsand suggestionsbyGemini.
- SetupanduseCursor AIfor AI-powered codingassistance.
- EvaluateandrefactorcodeusingCursorAIfeatures.
- CompareAItoolbehaviorand codequality acrossdifferentplatforms.

TaskDescription#1

- UseGoogleGeminiinColabtowriteafunctionthatreadsaCSVfileandcalculates mean, min, max.

Prompt:



Code:

```python
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)
```

ExpectedOutput#1

- Functionalcodewithoutputand 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}}
```

TaskDescription#2

- CompareGeminiand Copilotoutputsforapalindromecheck 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:

ExpectedOutput#2

- Side-by-sidecomparisonandobservations

Gemini output:

Output:

```
True
False
True
```

Copilot:

Output:

```
False
False
False
```

TaskDescription#3

- AskGeminitoexplainaPythonfunction(tocalculateareaofvariousshapes)lineby 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))
```

ExpectedOutput#3

- Detailedexplanationwith code snippet

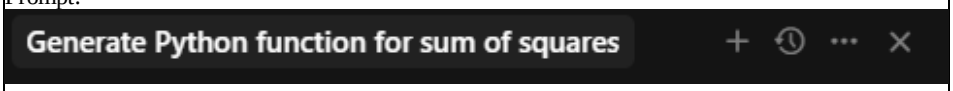
Output:

```
28.27431
20
24.0
```

TaskDescription#4

- InstallandconfigureCursorAI.UseittogenerateaPythonfunction(e.g.,sumof squares)

Prompt:



Code:

```python
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)
```

```python
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__":
```

ExpectedOutput#4

- Screenshotsofworking environmentswith fewpromptstogeneratepython code

```
For numbers [1, 2, 3, 4, 5]: 1² + 2² + 3² + 4² + 5² = 1 + 4 + 9 + 16 + 25 = 55
```

TaskDescription#5

- Studentneedtowritecodetocalculatesumofaddnumberandevennumbersinthe 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)
```

ExpectedOutput#5

- Refactoredcodewrittenbystudentwithimprovedlogic

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

Note:Reportshouldbesubmittedaworddocumentforalltasksinasingledocumentwith prompts, comments & code explanation, and output and if required, screenshots

Evaluation Criteria:

Criteria	Max Marks
SuccessfulUseofGeminiinColab(Task#1&#2)	1.0
CodeExplanationAccuracy(Gemini) (Task#3)	0.5

	CursorAISetupandUsage(Task#4)	0.5		
	RefactoringandImprovementAnalysis(Task#5)	0.5		
	Total	**2.5 Marks**		