

NAME: SIDDHARTHA REDDY KOTHA || HALL TICKET NUMBER:-2303A51491 || BATCH:25

SCHOOL OF COMPUTER SCIENCE AND ARTIFICIAL INTELLIGENCE		DEPARTMENT OF COMPUTER SCIENCE ENGINEERING	
<b>Program Name:</b> B. Tech		<b>Assignment Type:</b> Lab	
<b>Course Coordinator Name</b>		Dr. Rishabh Mittal	
<b>Instructor(s) Name</b>		Mr. S Naresh Kumar Ms. B. Swathi Dr. Sasanko Shekhar Gantayat Mr. Md Sallauddin Dr. Mathivanan Mr. Y Srikanth Ms. N Shilpa Dr. Rishabh Mittal (Coordinator) Dr. R. Prashant Kumar Mr. Ankushavali MD Mr. B Viswanath Ms. Sujitha Reddy Ms. A. Anitha Ms. M.Madhuri Ms. Katherashala Swetha Ms. Velpula sumalatha Mr. Bingi Raju	
<b>CourseCode</b>	23CS002PC304	<b>Course Title</b>	AI Assisted Coding
<b>Year/Sem</b>	III/II	<b>Regulation</b>	R23
<b>Date and Day of Assignment</b>	Week1 – Wednesday	<b>Time(s)</b>	23CSBTB01 To 23CSBTB52
<b>Duration</b>	2 Hours	<b>Applicable to Batches</b>	All batches
<b>Assignment Number:</b> 1.3(Present assignment number)/ <b>24</b> (Total number of assignments)			

Q.No.	Question	Expected Time to complete
1	Lab 2: Exploring Additional AI Coding Tools beyond Copilot – Gemini (Colab) and Cursor AI  <b>Lab Objectives:</b>	Week1 - Monday

- ❖ 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.

#### **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 1: Word Frequency from Text File**

##### **❖ Scenario:**

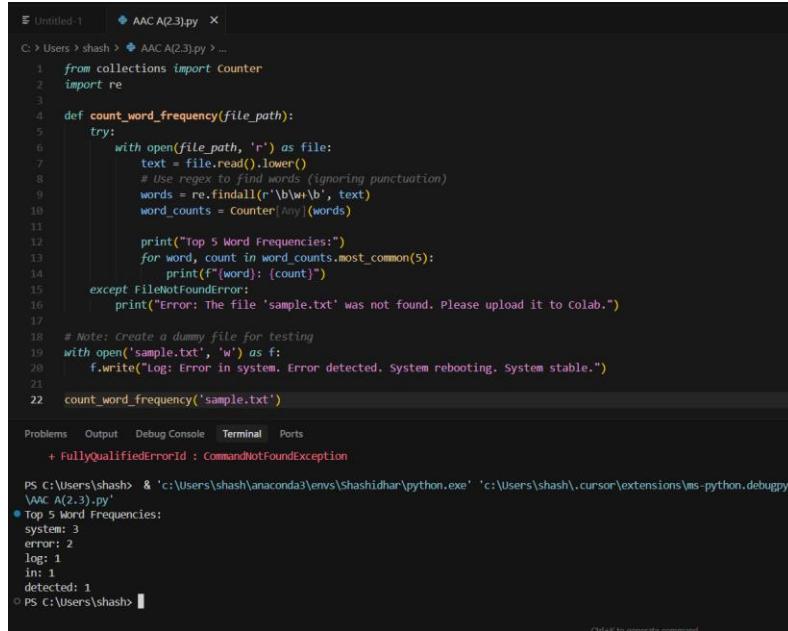
You are analyzing log files for keyword frequency.

##### **❖ Task:**

Use Gemini to generate Python code that reads a text file and counts word frequency, then explains the code.

##### **❖ Expected Output:**

- Working code
- Explanation
- Screenshot



The screenshot shows a Google Colab notebook interface. The code cell contains Python code to count word frequencies in a text file. The output cell shows the code running and producing the following output:

```

PS C:\Users\shash> & 'c:\Users\shash\anaconda3\envs\Shashidhar\python.exe' 'c:\Users\shash\cursor\extensions\ms-python.debugpy-2.0.0\AAC A(2.3).py'
● Top 5 Word Frequencies:
system: 3
error: 2
log: 1
ini: 1
detected: 1

```

## Task 2: File Operations Using Cursor AI

❖ **Scenario:**

You are automating basic file operations.

❖ **Task:**

Use Cursor AI to generate a program that:

- Creates a text file
- Writes sample text
- Reads and displays the content

❖ **Expected Output:**

- Functional code
- Cursor AI screenshots

The screenshot shows a code editor with a dark theme. A file named 'AAC A(2.3).py' is open, containing the following Python code:

```
C: > Users > shash > AAC A(2.3).py > ...
1 # Generated by Cursor AI
2 file_name = "cursor_test.txt"
3
4 # 1. Create and Write
5 with open(file_name, "w") as file:
6     file.write("Hello from Cursor AI\nThis is an automated file operation test")
7
8 # 2. Read and Display
9 with open(file_name, "r") as file:
10     content = file.read()
11     print("File Content:\n", content)
```

Below the code editor is a terminal window showing the execution of the script and its output:

```
Problems   Output   Debug Console   Terminal   Ports
system: 3
error: 2
log: 1
in: 1
detected: 1
● PS C:\Users\shash> c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhar\432' '--' 'c:\Users\shash\AAC A(2.3).py'
File Content:
Hello from Cursor AI
This is an automated file operation test.
○ PS C:\Users\shash> []
```

## Task 3: CSV Data Analysis

❖ **Scenario:**

You are processing structured data from a CSV file.

❖ **Task:**

Use Gemini in Colab to read a CSV file and calculate mean, min, and max.

❖ **Expected Output:**

- Correct output
- Screenshot

The screenshot shows a Jupyter Notebook interface with two tabs: 'AAC\_A(2.3).py' and 'AAC\_A(2\_3).ipynb'. The current tab is 'AAC\_A(2\_3).ipynb'. The code cell contains the following Python script:

```
import pandas as pd
import io

# Step 1: Create a sample CSV dataset (or Load your own)
csv_data = """City,Temperature,Humidity
New York,22,60
London,15,80
Tokyo,18,70
Sydney,25,55
Paris,14,75
"""

# Step 2: Read the CSV data
df = pd.read_csv(io.StringIO(csv_data))

# Step 3: Calculate Mean, Min, and Max for the 'Temperature' column
temp_mean = df['Temperature'].mean()
temp_min = df['Temperature'].min()
temp_max = df['Temperature'].max()

# Step 4: Display the results
print("--- CSV Data Analysis Results ---")
print(df)
print("-" * 33)
print(f"Mean Temperature: {temp_mean:.2f}")
print(f"Min Temperature: {temp_min}")
print(f"Max Temperature: {temp_max}")

... --- CSV Data Analysis Results ---
   City  Temperature  Humidity
0  New York        22       60
1    London        15       80
2     Tokyo        18       70
3   Sydney        25       55
4    Paris        14       75
-----
Mean Temperature: 18.80
Min Temperature: 14
Max Temperature: 25
```

**Task 4: Sorting Lists – Manual vs Built-in**

❖ **Scenario:**

You are reviewing algorithm choices for efficiency.

❖ **Task:**

Use **Gemini** to generate:

- Bubble sort
- Python's built-in sort()
- Compare both implementations.

❖ **Expected Output:**

- Two versions of code
- Short comparison

The screenshot shows a code editor window with a Python file named `AAC A(2.3).py`. The code implements a bubble sort algorithm and includes an example usage at the bottom.

```
C:\> Users > shash > AAC A(2.3).py > ...
1 def bubble_sort(arr):
2     n = len(arr)
3     # Outer loop to traverse through all array elements
4     for i in range(n):
5         # Last i elements are already in place, so we ignore them
6         for j in range(0, n - i - 1):
7             # Swap if the element found is greater than the next
8             if arr[j] > arr[j + 1]:
9                 arr[j], arr[j + 1] = arr[j + 1], arr[j]
10    return arr
11
12 # Example usage
13 data = [64, 34, 25, 12, 22, 11, 90]
14 print(f"Manual Bubble Sort: {bubble_sort(data.copy())}")

Problems Output Debug Console Terminal Ports
● ModuleNotFoundError: No module named 'pandas'
PS C:\Users\shash> c;; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhara\Scripts\python.exe' 'c:\Users\shash\AAC A(2.3).py'
Traceback (most recent call last):
  File "c:\Users\shash\AAC A(2.3).py", line 1, in <module>
    import pandas as pd
● ModuleNotFoundError: No module named 'pandas'
PS C:\Users\shash> c;; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashidhara\Scripts\python.exe' 'c:\Users\shash\AAC A(2.3).py'
Manual Bubble Sort: [11, 12, 22, 25, 34, 64, 90]
PS C:\Users\shash> []
```

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