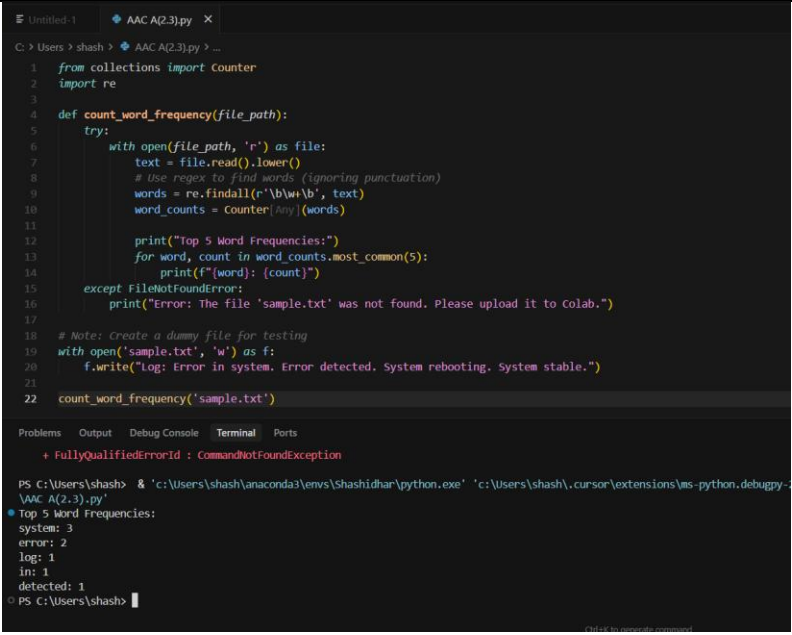
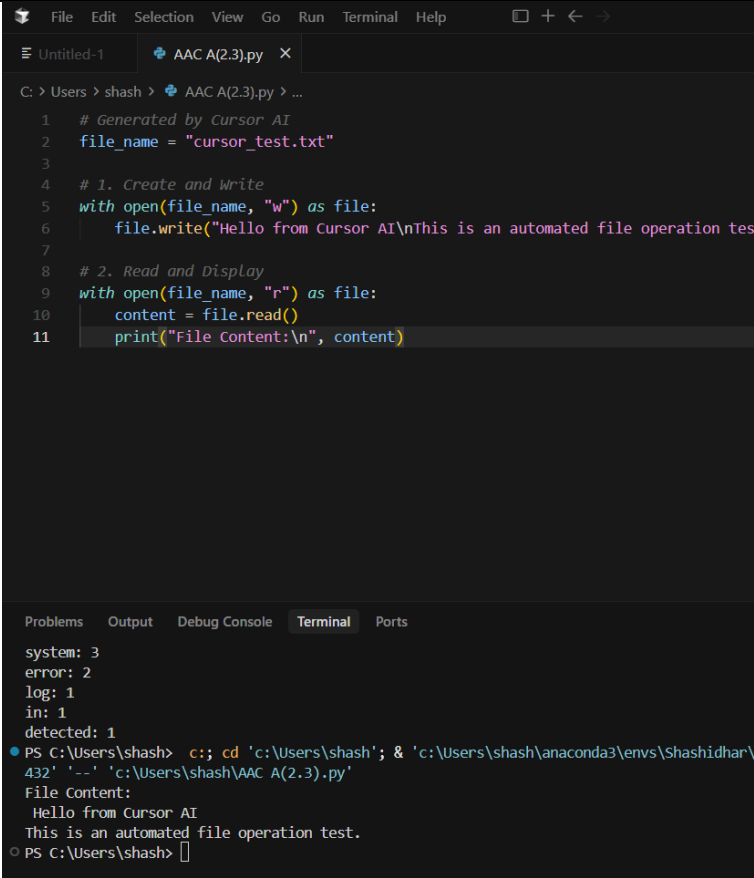


NAME:CH.Kruthankiran
H.NO:2303A51404
BATCH:26

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		Dr. Rishabh Mittal	
Instructor(s) Name		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			
CourseCode	23CS002PC304	Course Title	AI Assisted Coding
Year/Sem	III/II	Regulation	R23
Date and Day of Assignment	Week1 – Wednesday	Time(s)	23CSBTB01 To 23CSBTB52
Duration	2 Hours	Applicable to Batches	All batches
Assignment Number:1.3(Present assignment number)/24(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	Week1 - Monday	

	<p>Lab Objectives:</p> <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 1: Word Frequency from Text File</p> <ul style="list-style-type: none"> ❖ 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: <ul style="list-style-type: none"> ➤ Working code ➤ Explanation ➤ Screenshot 	

	 <pre> 1 from collections import Counter 2 import re 3 4 def count_word_frequency(file_path): 5 try: 6 with open(file_path, 'r') as file: 7 text = file.read().lower() 8 # Use regex to find words (ignoring punctuation) 9 words = re.findall(r'\b\w+\b', text) 10 word_counts = Counter(words) 11 12 print("Top 5 Word Frequencies:") 13 for word, count in word_counts.most_common(5): 14 print(f"{word}: {count}") 15 except FileNotFoundError: 16 print("Error: The file 'sample.txt' was not found. Please upload it to Colab.") 17 18 # Note: Create a dummy file for testing 19 with open('sample.txt', 'w') as f: 20 f.write("Log: Error in system. Error detected. System rebooting. System stable.") 21 22 count_word_frequency('sample.txt') </pre> <p>Problems Output Debug Console Terminal Ports</p> <p>+ FullyQualifiedErrorId : CommandNotFoundException</p> <pre> PS C:\Users\shash> & 'c:\Users\shash\anaconda3\envs\shashidhar\python.exe' 'c:\Users\shash\cursor\extensions\ms-python.debugpy-> \AAC A(2.3).py' Top 5 Word Frequencies: system: 3 error: 2 log: 1 in: 1 detected: 1 PS C:\Users\shash> </pre>	
	<p>Task 2: File Operations Using Cursor AI</p> <ul style="list-style-type: none"> ❖ Scenario: You are automating basic file operations. ❖ Task: Use Cursor AI to generate a program that: <ul style="list-style-type: none"> ➤ Creates a text file ➤ Writes sample text ➤ Reads and displays the content ❖ Expected Output: <ul style="list-style-type: none"> ➤ Functional code ➤ Cursor AI screenshots 	

	 <p>The screenshot shows a code editor with a file named 'AAC A(2.3).py'. The code is a Python script that creates a file 'cursor_test.txt', writes 'Hello from Cursor AI\nThis is an automated file operation test' to it, and then reads the file back to display its content. The terminal output shows the command execution and the resulting file content.</p> <pre> 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) 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> </pre>	
	<p>Task 3: CSV Data Analysis</p> <ul style="list-style-type: none"> ❖ 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: <ul style="list-style-type: none"> ➤ Correct output ➤ Screenshot 	

```
AAC_A(2.3).py 1 AAC_A(2.3).ipynb
C: > Users > shash > Downloads > AAC_A(2.3).ipynb > import pandas as pd
+ Code + Markdown | ▶ Run All | Clear All Outputs | Outline ...

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

```
File Edit Selection View Go Run Terminal Help
AAC A(2.3).py X
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
import pandas as pd
ModuleNotFoundError: No module named 'pandas'
PS C:\Users\shash> c:; cd 'c:\Users\shash'; & 'c:\Users\shash\anaconda3\envs\Shashid
'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\Shashid
'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.