**KARABI MANDAL ROLL:-2303A51620 BATCH: 25 ASSIGNMENT:- 2.3**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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  **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, 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   **PROMPT**: Generate Python code that reads a text file, counts word frequency, prints results in descending order, and explain the code simply.  **Explaination:** The code reads the text file, cleans it, and picks out all the words.  It counts how many times each word appears using a smart dictionary.  Finally, it sorts the words by frequency and prints them from most common to least.    **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 screenshot   **PROMPT**: Generate a program that creates a text file, writes sample text into it, then reads the file and displays its content.  **EXPLAINATION**: The code first creates a new text file and writes some sample lines into it.  Then it opens the same file again, reads everything inside, and prints the content to the screen.  Error handling is added so the program shows a friendly message if something goes wrong, like the file not being found.    **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   **PROMPT:** Write Colab-ready Python code that loads a CSV file using pandas, calculates mean, minimum, and maximum for each numeric column, prints the results clearly, and then explain the code simply.  **EXPLAINATION:** The code loads a CSV file into a pandas DataFrame and picks only the numeric columns.  It calculates the mean, minimum, and maximum for each numeric column.  Finally, it prints these statistics neatly for easy understanding.      **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   **PROMPT:** Write Python code for bubble sort and also using Python’s built-in sort(), then explain in simple words which one is faster and why.  **Explaination**:  The code defines Bubble Sort (manual swapping) and compares it with Python’s built-in sort(), which uses a highly optimized algorithm.  It generates a large list of random numbers, runs both sorting methods, and measures how long each one takes.  The results show that Python’s built-in sort() is dramatically faster because Bubble Sort is O(n²), while Python uses the much more efficient Timesort.    **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.** | | | | | | Week1 - Monday |  |