**Name:JAITARAM RUTHVIK REDDY** **ROLL NO:2403A510B5**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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** | | | Week1 - Thursday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | | 24CSBTB01 To 24CSBTB39 | | | |
| **AssignmentNumber:2.4**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | Lab 2: Exploring Additional AI Coding Tools – 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 Description #1**  **•** Open Google Colab and use Google Gemini to generate Python code that performs sorting of a list using both the bubble sort algorithm and Python’s built-in sort() function. Compare the two implementations. **Expected Output #1**  **•** Two sorting implementations: Bubble sort (manual logic) and Built-in sort() **Prompt:** give me a python code in a manner sorting of a list using both the bubble sort algorithm and Python’s built-in sort() function. Compare the two implementations using dynamic input.    **Observation:** 🡪This code provides a simple comparison between a basic sorting algorithm (bubble sort) and a highly optimized one (Python's built-in sort, which is Timsort).  🡪Bubble Sort: This algorithm has a time complexity of O(n^2) in the worst and average cases. This means that as the size of the list (n) increases, the time it takes to sort the list grows quadratically. It's simple to understand but very inefficient for larger lists.  🡪Python's Built-in Sort (sorted()): Python's sorting algorithm (Timsort) has a time complexity of O(n log n) in the average and worst cases. This is much more efficient than O(n^2). As the list size increases, the time taken grows much slower compared to bubble sort.  **Task Description #2**  **•** In Colab, use Google Gemini to generate a Python function that takes a string and returns:  The number of vowels, The number of consonants, The number of digits in the string  **Expected Output #2-**  **•** Complete function that Iterates through characters of a string and Counts vowels, consonants, and digits  **Prompt:**  Write a Python function that returns the number of vowels, consonants, and digits in a given string.  **Observation:**  **🡪**This code defines a function count\_chars that takes a string as input. It then iterates through each character in the string and categorizes it as a vowel, consonant, digit, or other character based on predefined sets of characters. Finally, it returns a dictionary containing the total counts for each category. The example usage demonstrates how to call the function and prints the results**.**  **Task Description #3**  **•** Install and set up Cursor AI. Use it to generate a Python program that performs file handling:  Create a text file  Write sample text  Read and display the content  **Expected Output #3**  **•** Functional code that creates a .txt file, writes content to it, and reads it back.  • Screenshot of Cursor AI interface showing: Prompt used,  Generated code, Output of file operations  **Prompt:** Write a Python program that creates a .txt file, writes sample text into it, then reads and displays the file content.    **Observation:**  🡪This Python code demonstrates basic file handling operations. It first defines a filename and some sample text. Then, it attempts to create a file with that name in write mode ("w"), writing the sample text into it. After successfully writing, it closes the file and then opens the same file again in read mode ("r") to read its entire content. Finally, it prints a confirmation message and the content that was read from the file. The try...except blocks are used to handle potential input/output errors during the file operations.  Task Description #4  • Ask Google Gemini to generate a Python program that implements a simple calculator using functions (add, subtract, multiply, divide). Then, ask Gemini to explain how the code works.  **Expected Output #4**  **•** Complete calculator code with user input and operation selection.  • Line-by-line explanation or markdown-style explanation provided by Gemini.  • Screenshot of both the code and explanation in Colab.  **Prompt:** Generate a Python program for a simple calculator using functions for add, subtract, multiply, and divide. Include user input for numbers and operation selection.    **Observation:**  🡪This Python program creates a basic calculator. It defines four functions (add, subtract, multiply, divide) to perform the core arithmetic operations. The program then presents a menu of options to the user and takes their choice of operation. It prompts the user to enter two numbers, handles potential errors if the input is not a number, and then calls the appropriate function based on the user's selection to perform the calculation and display the result. The divide function also includes a check to prevent division by zero. The while True loop allows the user to keep making calculations until they choose to exit (although an exit option isn't explicitly implemented in this version).  **Task Description #5** • Use Cursor AI to create a Python program that checks if a given year is a leap year or not. Try different prompt styles and see how Cursor modifies its code suggestions.  **Expected Output #5** • A functional program to check leap year with sample input/output • At least two versions of the code (from different prompts) • A short comparison of which version is better and why **Prompt:** Write a Python program to check if a given year is a leap year.    **Observation:**  🡪This Python program defines a function is\_leap\_year that takes a year as input and determines if it's a leap year based on the standard rules: a year is a leap year if it's divisible by 4, unless it's divisible by 100 but not by 400. The program prompts the user to enter a year, handles invalid input by asking for a whole number, and then calls the is\_leap\_year function to check the year. Finally, it prints a message indicating whether the entered year is a leap year or not.  **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** | | --- | --- | | Two sorting implementations: Bubble sort (manual logic) and Built-in sort() (Task#1) | 0.5 | | Counts vowels, consonants, and digits(Task#2) | 0.5 | | Functional code that creates a .txt file, writes content to it, and reads it back- Use cursor (Task#3) | 0.5 | | Complete calculator code with user input and operation selection. (Task#4) | 0.5 | | A functional program to check leap year with sample input/output-use Cursor (Task#5) | 0.5 | | **Total** | **2.5 Marks** | | | | | | | Week1 - Thursday |  |