|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 – Monday** | **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: Statistical Summary for Survey Data**   * **Scenario:** You are a **data analyst intern** working with survey responses stored as numerical lists. * **Task:** Use **Google Gemini in Colab** to generate a Python function that reads a list of numbers and calculates the **mean, minimum, and maximum** values. * **Expected Output:**   + Correct Python function   + Output shown in Colab   + Screenshot of Gemini prompt and result   Output :  Prompt :  Generate a Python function that takes a list of numerical survey responses as input and calculates the mean, minimum, and maximum values. Also include a sample list and display the output    Code :  Output :  Sample Survey Responses: [4, 5, 2, 8, 7, 3, 9, 6, 1, 10]  Mean: 5.50  Minimum: 1  Maximum: 10    **Explanation:**  The function analyze\_survey\_responses() computes the mean, minimum, and maximum values from a list of numerical survey responses. It uses Python’s built-in sum() and len() functions to calculate the mean, and min() and max() to find the smallest and largest values. An empty-list check is included to avoid errors. The results are returned as a tuple and printed in a formatted manner**.**  **Task 2: Armstrong Number – AI Comparison**   * **Scenario:** You are evaluating AI tools for numeric validation logic. * **Task:** Generate an **Armstrong number checker** using **Gemini** and **GitHub Copilot**. Compare their outputs, logic style, and clarity. * **Expected Output:**   + Side-by-side comparison table   + Screenshots of prompts and generated code   Output :  Prompt :  Write a Python program to check whether a given number is an Armstrong number. Include user input and display the result.  Using Google Gemini :  Code :    Output for the gemini code :  Enter a number: 153  153 is an Armstrong number.    Using GitHub Copilot:  Code :    Output :    comparison table:   | **Aspect** | **Gemini** | **GitHub Copilot** | | --- | --- | --- | | Output | Correctly identifies Armstrong numbers | Correctly identifies Armstrong numbers | | Input Handling | Validates input using isdigit() | Assumes valid integer input | | Logic Style | Step-by-step procedural logic | Functional and compact logic | | Use of Functions | No separate function used | Uses a reusable function | | Handling Negative Numbers | Explicitly checks and restricts | Handles negatives using abs() | | Code Clarity | Very clear and beginner-friendly | Concise but slightly advanced | | Readability | Easy to follow with comments | Shorter but less descriptive |   **Task 3: Leap Year Validation Using Cursor AI**   * **Scenario:** You are validating a calendar module for a backend system. * **Task:** Use **Cursor AI** to generate a Python program that checks whether a given year is a leap year. Use **at least two different prompts** and observe changes in code. * **Expected Output:**   + Two versions of code   + Sample inputs/outputs   + Brief comparison   Output :  Prompt 1 :Write a simple Python program to check whether a given year is a leap year.    Output :    **Task 4: Student Logic + AI Refactoring (Odd/Even Sum)**   * **Scenario:** Company policy requires developers to write logic before using AI. * **Task:** Write a Python program that calculates the **sum of odd and even numbers in a tuple**, then refactor it using any AI tool. * **Expected Output:**   + Original code   + Refactored code   + Explanation of improvements   Output :      **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 |  |