|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 - Wednesday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | | 24CSBTB01 To 24CSBTB39 | | | |
| **AssignmentNumber:2.3**(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**   * Use Google Gemini in Colab to write a function that reads a CSV file and calculates mean, min, max.   **Expected Output#1**   * Functional code with output and screenshot     **Task Description#2**   * Compare Gemini and Copilot outputs for a palindrome check function.   **Expected Output#2**   * Side-by-side comparison and observations       **Observations**   * **First program is more comprehensive: It handles punctuation and other non-alphanumeric characters, making it suitable for full sentences like "A man, a plan, a canal: Panama".** * **second program is better for quick user interaction: It’s simpler and works well for basic words or phrases typed by the user.** * **Function name conflict: Both use is\_palindrome, which would cause the second definition to overwrite the first if run in the same script. You should rename one of them or separate them into different files or scopes.**   **Task Description#3**   * Ask Gemini to explain a Python function (to calculate area of various shapes) line by line..   **Expected Output#3**   * Detailed explanation with code snippet * **Here is a line-by-line explanation of the calculate\_area function:** * **[ ]** * **import math** * **This line imports the math module, which provides access to mathematical functions, including math.pi which is used for calculating the area of a circle.** * **[ ]** * **def calculate\_area(shape, \*\*kwargs):** * **This line defines a function named calculate\_area that takes two arguments:** * **shape: A string that specifies the type of shape (e.g., 'square', 'circle').** * **\*\*kwargs: This is a special syntax that allows the function to accept an arbitrary number of keyword arguments. These arguments will be used to provide the dimensions of the shape (like side, radius, length, width, base, height).** * **[ ]** * **"""   Calculates the area of different shapes.    Args:     shape: A string indicating the shape ('square', 'circle', 'rectangle', or 'triangle').     \*\*kwargs: Keyword arguments for the shape's dimensions (e.g., side, radius, length, width, base, height).    Returns:     The area of the shape, or None if the shape is not supported or dimensions are missing.   """** * **This is a docstring, which explains what the function does, its arguments (Args), and what it returns (Returns). It's good practice to include docstrings to make your code understandable.** * **[ ]** * **if shape == 'square':** * **This line starts a conditional block. It checks if the value of the shape argument is exactly the string 'square'.** * **[ ]** * **side = kwargs.get('side')** * **If the shape is 'square', this line tries to get the value associated with the keyword argument side from the kwargs dictionary. The .get() method is used here. If 'side' is not found in kwargs, it returns None by default.** * **[ ]** * **if side is not None:** * **This line checks if the side variable has a value (i.e., it's not None). This means the user provided the 'side' argument when calling the function.** * **[ ]** * **return side\*\*2** * **If side is provided, this line calculates the area of the square by squaring the side value and returns the result.** * **[ ]** * **else:       print("Error: Side is required for a square.")       return None** * **If side was not provided (i.e., it's None), this block is executed. It prints an error message indicating that the 'side' is required for a square and then returns None.** * **[ ]** * **elif shape == 'circle':** * **This line is the start of another conditional block. It's checked if the first if condition (shape == 'square') was false. This line checks if the shape is 'circle'.** * **[ ]** * **radius = kwargs.get('radius')** * **If the shape is 'circle', this line gets the value for the 'radius' keyword argument from kwargs.** * **[ ]** * **if radius is not None:** * **This line checks if the radius variable has a value.** * **[ ]** * **return math.pi \* radius\*\*2** * **If radius is provided, this line calculates the area of the circle using the formula π \* radius² and returns the result. math.pi provides the value of Pi.** * **else:       print("Error: Radius is required for a circle.")       return None** * **If radius was not provided, this block prints an error and returns None.** * **elif shape == 'rectangle':** * **This line checks if the shape is 'rectangle'.** * **length = kwargs.get('length')** * **If the shape is 'rectangle', this line gets the value for the 'length' keyword argument from kwargs** * **width = kwargs.get('width')** * **This line gets the value for the 'width' keyword argument from kwargs.** * **if length is not None and width is not None:** * **This line checks if both length and width variables have values** * **return length \* width** * **If both length and width are provided, this line calculates the area of the rectangle and returns the result** * **else:       print("Error: Length and width are required for a rectangle.")       return None** * **If either length or width (or both) were not provided, this block prints an error and returns None.** * **elif shape == 'triangle':** * **This line checks if the shape is 'triangle'.** * **base = kwargs.get('base')** * **If the shape is 'triangle', this line gets the value for the 'base' keyword argument from kwargs.** * **height = kwargs.get('height')** * **This line gets the value for the 'height' keyword argument from kwargs** * **if base is not None and height is not None:** * **This line checks if both base and height variables have values.** * **return 0.5 \* base \* height** * **If both base and height are provided, this line calculates the area of the triangle using the formula 0.5 \* base \* height and returns the resu** * **If either base or height (or both) were not provided, this block prints an error and returns None.**       **Task Description#4**   * Install and configure Cursor AI. Use it to generate a Python function (e.g., sum of squares).   **Expected Output#4**   * Screenshots of working environments with few prompts to generate python code     **Task Description#5**   * Student need to write code to calculate sum of add number and even numbers in the list   **Expected Output#5**   * Refactored code written by student with improved logic     **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** | | --- | --- | | Successful Use of Gemini in Colab (Task#1 & #2) | 1.0 | | Code Explanation Accuracy (Gemini) (Task#3) | 0.5 | | Cursor AI Setup and Usage (Task#4) | 0.5 | | Refactoring and Improvement Analysis (Task#5) | 0.5 | | **Total** | **2.5 Marks** | | | | | | | Week1 - Wednesday |  |