|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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** | | | Week3 - Wednesday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | |  | | | |
| **AssignmentNumber:6.3**(Present assignment number)/**24**(Total number of assignments) | | | | | | | | | |
|  | | | | | | | | | |
|  | **Q.No.** | **Question** | | | | | | ***ExpectedTime***  ***to complete*** |  |
|  | 1 | Lab 6: AI-Based Code Completion – Classes, Loops, and Conditionals  **Lab Objectives:**   * To explore AI-powered auto-completion features for core Python constructs. * To analyze how AI suggests logic for class definitions, loops, and conditionals. * To evaluate the completeness and correctness of code generated by AI assistants.   **Lab Outcomes (LOs):**  After completing this lab, students will be able to:   * Use AI tools to generate and complete class definitions and methods. * Understand and assess AI-suggested loops for iterative tasks. * Generate conditional statements through prompt-driven suggestions. * Critically evaluate AI-assisted code for correctness and clarity.   **Task Description#1 (Classes)**   * Use AI to complete a Student class with attributes and a method. * Check output * Analyze the code generated by AI tool   **Instructions**:   * **Initialize class with attributes like name, roll no, marks** * **Method to display student details** * **Method to calculate grade based on marks (A:>=90, B: >=75, C: >=60, else Fail)**   Start Writing code and auto complete using any AI tool  **Expected Output#1**   * Class with constructor and display\_details() method   In VS code:    **Output:**    **In Cursor:**    **Output:**    **Compare:** **Both files implement a Student class with:**   * **Constructor (\_\_init\_\_): Takes name, roll number, and marks as parameters** * **display\_details() method: Prints student information in a formatted way** * **calculate\_grade() method: Returns letter grades based on marks:** * **A: ≥90** * **B: ≥75** * **C: ≥60** * **Fail: <60**   **Main Program:**  **Both files have the same main program flow:**   1. **Get user input for student details** 2. **Create a Student object** 3. **Display student details** 4. **Calculate and display the grade**   **Task Description#2 (Loops)**   * Prompt AI to complete a function that prints the first 10 multiples of a number using a loop. * Analyze the generated code * Ask AI to generate code using other controlled looping   Write code using **For** Loop, later complete code using **While** Loop  **Expected Output#2**   * Correct loop-based implementation   In VS Code:      Output:    In Cursor:      Output:    Compare:  **Code Structure:**   * **task\_2c.py (Command-line version):** * Direct execution of code in the main body * No function definition * **task\_2v.py (Version with function):** * Code is wrapped in a function called print\_multiples\_while(number) * Function is called after getting user input   **2. Reusability:**   * **task\_2c.py:** Less reusable - the logic is hardcoded in the main execution flow * **task\_2v.py:** More reusable - the function can be imported and called multiple times with different numbers   **3. Maintainability:**   * **task\_2c.py:** Simpler structure, easier to read for beginners * **task\_2v.py:** Better separation of concerns, easier to modify the logic without affecting the input/output   **4. Code Organization:**   * **task\_2c.py:** Linear execution flow * **task\_2v.py:** Modular approach with clear function definition and call   **Task Description#3 (Conditional Statements)**   * Ask AI to write nested if-elif-else conditionals to classify age groups. * Analyze the generated code * Ask AI to generate code using other conditional statements   **Expected Output#3**   * Age classification function with appropriate conditions and with explanation   **In VS Code:**    **Output:**    **In Cursor:**    **Output:**    **Compare:  Age Group Boundaries:**   * **task\_3c.py: Adult age range is 20-59 (Senior starts at 60)** * **task\_3v.py: Adult age range is 20-64 (Senior starts at 65)**   **2. Input Validation:**   * **task\_3c.py: No input validation - assumes valid input** * **task\_3v.py: Includes input validation with try-catch for invalid input**   **3. Age Validation:**   * **task\_3c.py: No negative age check** * **task\_3v.py: Checks for negative ages and returns "Invalid age"**   **4. Main Program Structure:**   * **task\_3c.py: Direct execution without if \_\_name\_\_ == "\_\_main\_\_": guard** * **task\_3v.py: Uses proper main guard for modular execution**   **5. User Interface:**   * **task\_3c.py: Simple prompt "Enter age: "** * **task\_3v.py: More user-friendly prompt "Enter your age: "**   **6. Output Format:**   * **task\_3c.py: "Age group: {group}"** * **task\_3v.py: "You are classified as: {group}"**   **Task Description#4 (For and While loops)**   * Generate a sum\_to\_n() function to calculate sum of first n numbers * Analyze the generated code * Get suggestions from AI with other controlled looping   **Expected Output#4**   * Python code with explanation   In VS Code:    **Output:**    **In Cursor:**    **Output:**    **Compare:**  **sum\_to\_n\_for(n) function:**   * **Uses a for loop to calculate the sum of numbers from 1 to n** * **Returns the total sum**   **2. Main program:**   * **Input validation loop that ensures a positive integer is entered** * **Error handling for invalid input (non-integer values)** * **Calls the function and displays the result**   **3. Input validation features:**   * **Checks if input is a positive integer** * **Handles ValueError exceptions for non-integer input** * **Provides user-friendly error messages** * **Continues prompting until valid input is received**   **4. Output format:**   * **Both display: "Sum from 1 to {n} is {result}"**   **Task Description#5 (Class)**   * Use AI to build a BankAccount class with deposit, withdraw, and balance methods. * Analyze the generated code * Add comments and explain code   **Instructions**   * **Initialize BankAccount class with attributes like name, balance** * **Method to deposit amount** * **Method to withdraw amount** * **Method to check balance**   **Expected Output#5**   * Python code with explanation   **In VS Code:**    **Output:**    **In Cursor:**    **Output:**    **Compare:**  **Currency Formatting:**   * **task\_5v.py: No currency symbols (e.g., "Deposited 100. New balance: 100")** * **task\_5c.py: Includes dollar signs (e.g., "Deposited $100. New balance: $100")**   **2. Withdrawal Logic:**   * **task\_5v.py: Single condition if 0 < amount <= self.balance: (more concise)** * **task\_5c.py: Nested conditions with separate checks for positive amount and sufficient funds**   **3. Error Messages:**   * **task\_5v.py: "Insufficient funds or invalid amount." (combined message)** * **task\_5c.py: Separate messages: "Insufficient funds." and "Withdrawal amount must be positive."**   **4. Balance Display:**   * **task\_5v.py: "{name}'s balance: {balance}"** * **task\_5c.py: "{name}, your current balance is: ${balance}" (more personal, includes currency)**   **5. Input Prompts:**   * **task\_5v.py: "Enter account holder's name: "** * **task\_5c.py: "Enter your name: " (more personal)**   **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** | | --- | --- | | Class | 1.0 | | Loops | 1.0 | | Conditional Statements | 0.5 | | **Total** | **2.5 Marks** | | | | | | | Week3 - Wednesday |  |