|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **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 - Thursday | **Time(s)** | |  | | | |
| **Duration** | | | 2 Hours | **Applicableto**  **Batches** | |  | | | |
| **AssignmentNumber:6.4**(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:**  **•** Start a Python class named Student with attributes name, roll\_number, and marks. Prompt GitHub Copilot to complete methods for displaying details and checking if marks are above average.  **Expected Outcome #1:**  **•** Completed class with Copilot-generated methods like display\_details() and is\_passed(), demonstrating use of if-else conditions.  **Task Description #2:**  **•** Write the first two lines of a for loop to iterate through a list of numbers. Use a comment prompt to let Copilot suggest how to calculate and print the square of even numbers only.  **Expected Outcome #2:**  **•** A complete loop generated by Copilot with conditional logic (if number % 2 == 0) and appropriate output.  **Task Description #3:**  **•** Create a class called BankAccount with attributes account\_holder and balance. Use Copilot to complete methods for deposit(), withdraw(), and check for insufficient balance.  **Expected Outcome #3:**  **•** Functional class with complete method definitions using if conditions and self attributes. Code should prevent overdrawing.  **Task Description #4:**  **•** Define a list of student dictionaries with keys name and score. Ask Copilot to write a while loop to print the names of students who scored more than 75.  **Expected Outcome #4:**  **•** A complete while loop generated by Copilot with proper condition checks and formatted output.  **Task Description #5:**  **•** Begin writing a class ShoppingCart with an empty items list. Prompt Copilot to generate methods to add\_item, remove\_item, and use a loop to calculate the total bill using conditional discounts.  **Expected Outcome #5:**  **•** A fully implemented ShoppingCart class with Copilot-generated loops and if-else statements handling item management and discount 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** | | --- | --- | | Class | 1 | | Loop | 1 | | condition | 0.5 | | **Total** | **2.5 Marks** | | | | | | | Week3 - Thursday |  |
|  |  |  | | | | | |  |  |

**AI ASSTING LAB 6.4**

NAME:JARUPULA RAKESH

ROLL.NO:2403a51321

**Task1::**

**Prompt::**Create a Python class named Student with attributes name, roll\_number, and marks.

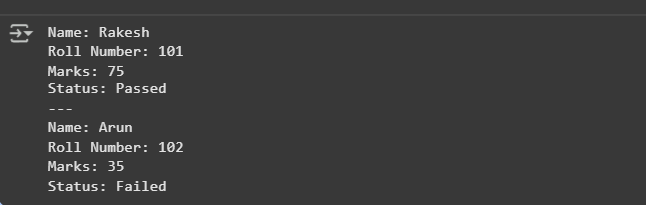
Add a method display\_details that prints the student's details.Add a method is\_passed that checksmarks are above 40 and prints "Passed" or "Failed". Use if-else conditions inside is\_passed

Code:

A screen shot of a computer program

AI-generated content may be incorrect.

Output:



Observation::

The Student class demonstrates object-oriented programming with attributes and methods.

display\_details() shows proper data display, and is\_passed() applies if-else logic to check results.

This program highlights encapsulation and condition-based decision making in Python.

**Task2:**prompt:

Write a Python for loop to iterate through a list of numbers and print the square of even numbers only using if num % 2 == 0.

Code and output::  
A screenshot of a computer program

AI-generated content may be incorrect.

Observation:  
Here are **clear exam-style observations** for the even numbers square program:

1. The program uses a for loop to iterate through the list of numbers.
2. The condition if num % 2 == 0 correctly checks for even numbers.
3. The square of even numbers is calculated using num \*\* 2 and printed.

**Task3::**

**Prompt**

Create a Python class BankAccount with attributes account\_holder and balance.

Add methods deposit and withdraw using if conditions to update balance.

Prevent overdrawing by checking insufficient balance before withdrawal.

**Code:**

**A screen shot of a computer screen

AI-generated content may be incorrect.**

**Output:  
A screenshot of a computer

AI-generated content may be incorrect.**

**Observation:**

The class BankAccount uses attributes account\_holder and balance to store account details.

The deposit() method increases the balance only when the amount is positive.

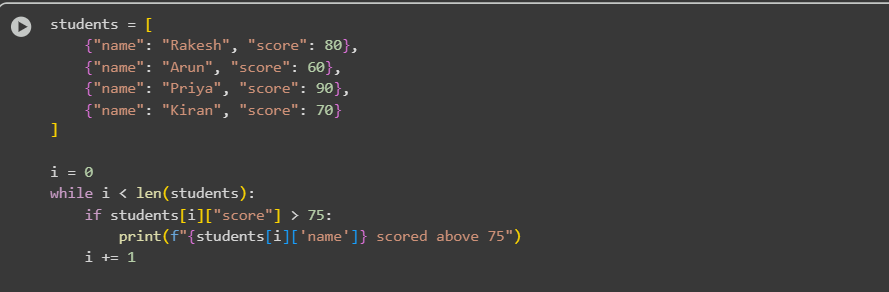
The withdraw() method prevents overdrawing by checking if the withdrawal amount is greater than balance.Proper use of if-else conditions ensures safe banking operations

**Task4:**

**Prompt:**Define a list of student dictionaries with keys name and score. Write a while loop to go through the list.

Print the names of students who scored more than 75**.**

**Code:**

****

**Output**

**A screenshot of a computer

AI-generated content may be incorrect.**

**Observation:**The program stores student details in a list of dictionaries and uses a while loop to process each record. It applies a condition to check if the score is greater than 75 and prints only the names of those students, showing proper use of loops and conditional check

**Task5:  
prompt:**Create a ShoppingCart class with an empty items list. Add methods add\_item() and remove\_item() to manage items. Write calculate\_total() using a loop and if-else to apply discount on the bill.

**Code:**



Output:  
A screenshot of a computer program

AI-generated content may be incorrect.

**Observation:**

The ShoppingCart class manages items using an empty list and provides methods to add or remove products. It calculates the total bill using a loop and applies discounts with if-else conditions when the total exceeds a certain amount. This demonstrates effective use of **OOP, loops, and conditional statements** for real-world scenarios.