Object-Oriented Programming Lab

Q1: Multiple Inheritance (20 minutes)

Problem Statement:
Design a program to demonstrate multiple inheritance. You will model a scenario where a person is also an employee and may take on a managerial role.

Class Design:

- - Class Person
  - Data Members: string name, int age
  - Member Functions: getPersonData(), displayPersonData()
- - Class Employee
  - Data Members: int employeeID, string designation
  - Member Functions: getEmployeeData(), displayEmployeeData()
- - Class Manager (inherits from both Person and Employee)
  - Data Members: string department
  - Member Functions: getManagerData(), displayManagerData()

Requirements:
- Input details for a manager using functions of all three classes.
- Display all data using a single function call on a Manager object.

Q2: Polymorphism Using Virtual Functions (30 minutes)

Problem Statement:
Create a program to demonstrate runtime polymorphism through virtual functions. You will work with geometric shapes and calculate their areas using dynamic dispatch.

Class Design:

- - Base Class Shape
  - Virtual Function: float area() = 0; (pure virtual)
- - Derived Class Circle
  - Data Member: float radius
  - Override area() to return $\pi \times \text{radius}^2$
- - Derived Class Rectangle
  - Data Members: float length, float width
  - Override area() to return length × width

Requirements:

- Use a Shape* pointer to point to both Circle and Rectangle objects.
- Demonstrate dynamic binding by calling the area() function using the base class pointer.

Sample Input/Output:
Shape* s;
Circle c(5.0);
s = &c;
s->area(); // Should print area of circle

Rectangle r(4.0, 6.0);
s = &r;
s->area(); // Should print area of rectangle

Q3: Virtual Inheritance to Resolve Diamond Problem (30 minutes)

Problem Statement:
Demonstrate the concept of virtual inheritance to avoid the diamond problem.

Class Hierarchy:

- - Class Student (virtual base class)
  - Data Members: int rollNo, string name
  - Functions: getStudentData(), displayStudentData()
- - Class Sports (virtual base class)
  - Data Member: int sportsScore
  - Functions: getSportsScore(), displaySportsScore()
- - Class Result (derived from both Student and Sports)
  - Data Member: int academicScore
  - Function: displayResult() which shows all details

Requirements:
- Prevent ambiguity in accessing rollNo and name.
- Use the Result class to accept and display all data.

Q4: Template Classes and Functions (40 minutes)

Problem Statement:
Create a C++ program to demonstrate generic programming using both function templates and class templates.

Part A: Template Function swapValues()
- Define a template function swapValues() to swap two variables of any type.
- Test it with int, float, and char.

Part B: Template Class Calculator<T>

- Create a class template that works with any numeric type T.
- Member Functions:
  - T add(T, T)
  - T subtract(T, T)
  - T multiply(T, T)
  - T divide(T, T) (Handle divide-by-zero)

Example Usage:
Calculator<int> calc1;
calc1.add(4, 2); // Output: 6

Calculator<float> calc2;
calc2.divide(5.5, 2.0); // Output: 2.75