**OBJECT-ORIENTED PROGRAMMING**

|  |  |
| --- | --- |
| Lab 11-b | |
| **Topic** | multilevel Inheritance |
| **Objective** | demonstrates the concept of inheritance and its types. |

**Task 1: Banking System with Multilevel Inheritance - Public Inheritance**

Create a C++ program to model a simple banking system using multilevel inheritance. Design three classes: **Account**, **SavingsAccount**, and **CurrentAccount**.

* **Base Class: Account**
  + Private Member Variables:
    - **accountNumber** (string): to store the account number.
    - **balance** (double): to store the account balance.
  + Public Member Functions:
    - Default constructor: Initializes accountNumber as "Undefined" and balance as 0.0.
    - Parameterized constructor: Accepts parameters for accountNumber and balance and initializes the **Account**.
    - **void displayAccountInfo()**: Displays the account number and balance.
* **Derived Class 1: SavingsAccount**
  + Inherits publicly from **Account**.
  + Additional Private Member Variable:
    - **interestRate** (double): to store the interest rate.
  + Additional Public Member Functions:
    - Default constructor: Calls the base class default constructor and sets the interestRate to 0.0.
    - Parameterized constructor: Accepts parameters for accountNumber, balance, and interestRate, and initializes the **SavingsAccount**.
    - **void displaySavingsAccountInfo()**: Extends the **displayAccountInfo()** to include the interest rate.
* **Derived Class 2: CurrentAccount**
  + Inherits publicly from **SavingsAccount**.
  + Additional Private Member Variable:
    - **overdraftLimit** (double): to store the overdraft limit.
  + Additional Public Member Functions:
    - Default constructor: Calls the base class default constructor and sets the overdraftLimit to 0.0.
    - Parameterized constructor: Accepts parameters for accountNumber, balance, interestRate, and overdraftLimit, and initializes the **CurrentAccount**.
    - **void displayCurrentAccountInfo()**: Extends the **displaySavingsAccountInfo()** to include the overdraft limit.

**Task 2: School System with Multilevel Inheritance - Private Inheritance**

Create a C++ program to model a school system using multilevel inheritance. Design three classes: **Person**, **Student**, and **Teacher**.

* **Base Class: Person**
  + Private Member Variables:
    - **name** (string): to store the name of the person.
  + Public Member Functions:
    - Default constructor: Initializes name as "Unknown".
    - Parameterized constructor: Accepts a parameter for the name and initializes the **Person**.
    - **void displayName()**: Displays the name of the person.
* **Derived Class 1: Student**
  + Inherits privately from **Person**.
  + Additional Private Member Variable:
    - **studentID** (int): to store the student ID.
  + Additional Public Member Functions:
    - Default constructor: Calls the base class default constructor and sets the studentID to 0.
    - Parameterized constructor: Accepts parameters for name and studentID, and initializes the **Student**.
    - **void displayStudentInfo()**: Extends the **displayName()** to include the student ID.
* **Derived Class 2: Teacher**
  + Inherits privately from **Student**.
  + Additional Private Member Variable:
    - **subject** (string): to store the subject taught by the teacher.
  + Additional Public Member Functions:
    - Default constructor: Calls the base class default constructor and sets the subject as "Unknown".
    - Parameterized constructor: Accepts parameters for name, studentID, and subject, and initializes the **Teacher**.
    - **void displayTeacherInfo()**: Extends the **displayStudentInfo()** to include the subject.

**Task 3: Employee Hierarchy with Multilevel Inheritance - Protected Inheritance**

Create a C++ program to model a hierarchy of employees in a company using multilevel inheritance. Design three classes: **Person**, **Employee**, and **Manager**.

* **Base Class: Person**
  + Private Member Variables:
    - **name** (string): to store the name of the person.
  + Public Member Functions:
    - Default constructor: Initializes name as "Unknown".
    - Parameterized constructor: Accepts a parameter for the name and initializes the **Person**.
    - **void displayName()**: Displays the name of the person.
* **Derived Class 1: Employee**
  + Inherits protectedly from **Person**.
  + Additional Private Member Variable:
    - **employeeID** (int): to store the employee ID.
  + Additional Public Member Functions:
    - Default constructor: Calls the base class default constructor and sets the employeeID to 0.
    - Parameterized constructor: Accepts parameters for name and employeeID, and initializes the **Employee**.
    - **void displayEmployeeInfo()**: Extends the **displayName()** to include the employee ID.
* **Derived Class 2: Manager**
  + Inherits protectedly from **Employee**.
  + Additional Private Member Variable:
    - **department** (string): to store the department of the manager.
  + Additional Public Member Functions:
    - Default constructor: Calls the base class default constructor and sets the department as "Undefined".
    - Parameterized constructor: Accepts parameters for name, employeeID, and department, and initializes the **Manager**.
    - **void displayManagerInfo()**: Extends the **displayEmployeeInfo()** to include the department.

**Task Requirements for All:**

* Use public, private, and protected access specifiers appropriately in the base and derived classes.
* Create objects of each class and demonstrate how access specifiers affect member accessibility.
* Observe and document any compile-time or runtime errors related to access specifiers.
* Discuss and explain how the choice of access specifiers affects the design and functionality of the classes.