**National University of Computer and Emerging Sciences**



**Lab Manual 10**

***for***

**Object Oriented Programming (OOP)**

| **Course Instructor** | **Ms. Hina Iqbal** |
| --- | --- |
| **Lab Instructor(s)** | **Amina Qaiser** |
| **Section** | **B** |
| **Semester** | **Fall 2024** |

**Department of Computer Science**

**FAST-NU, Lahore, Pakistan**

**Lab 10**

**Objective:**

This lab task is designed to help you understand:

1. The order of constructor and destructor calls in inheritance.
2. How to use function overriding in derived classes.
3. The purpose and usage of initializer lists in constructors.
4. The concept and use cases of private constructors.

**Task Description:**

You are required to create a **School Management System** that demonstrates the following concepts:

1. **Constructor and Destructor Call Sequence in Inheritance**:
   * Create a base class and a derived class to observe the order in which constructors and destructors are called.
2. **Function Overriding**:
   * Override a function from the base class in the derived class to show polymorphic behavior.
3. **Initializer List**:
   * Use an initializer list in constructors for initializing constant and reference members.
4. **Private Constructor**:
   * Use a private constructor in a helper class that restricts the creation of objects directly.

**Class Structure:**

1. **Person (Base Class)**:
   * Data members: name (string), age (int).
   * Constructor: Initializes name and age using an initializer list.
   * Destructor: Outputs a message indicating destruction of the Person object.
   * Virtual function displayInfo() to display name and age (to be overridden in the derived class).
2. **Student (Derived Class)**:
   * Inherits from Person.
   * Data members: studentID (string).
   * Constructor: Uses initializer list to initialize name, age, and studentID.
   * Destructor: Outputs a message indicating destruction of the Student object.
   * Overrides displayInfo() to display name, age, and studentID.
3. **School**:
   * Demonstrates a **private constructor**.
   * Contains static functions that provide controlled access to the object creation (factory method pattern).
   * Data member: schoolName (string, constant).

**Steps to Complete the Task:**

1. **Define the Person Class**:
   * Use an initializer list to initialize name and age.
   * Implement a virtual function displayInfo() that displays name and age.
   * Implement a destructor that displays a message when a Person object is destroyed.
2. **Define the Student Class (Inheritance and Overriding)**:
   * Inherit from Person.
   * Use an initializer list in the constructor to initialize inherited and new data members.
   * Override displayInfo() to include studentID.
   * Implement a destructor to display a message indicating a Student object’s destruction.
3. **Define the School Class (Private Constructor)**:
   * Make the constructor private and include a static member function to create and return a School object.
   * Use const string for schoolName and initialize it with an initializer list.
4. **Main Program**:
   * Create instances of Person and Student to observe the constructor and destructor call sequence.
   * Call displayInfo() on Person and Student to show function overriding.
   * Use the static function in School to create a School instance and display its schoolName.