**Task: Implement Inheritance and Virtual Functions**

**Objective**: Create a C++ program that demonstrates the concepts of inheritance, virtual functions, and polymorphism. This will help students understand how to define and use base and derived classes, and how virtual functions enable dynamic binding.

**Requirements:**

1. **Class Definitions**:
2. Create a base class named Base with the following features:
3. **Public Member Functions**:
4. Implement a virtual function show() that outputs the message: "Base class show function".
5. Provide a virtual destructor to ensure proper cleanup of derived class objects.
6. Create a derived class named Derived that publicly inherits from Base:
7. **Override Member Function**:
8. Override the show() function to output the message: "Derived class show function".
9. **Main Function**:
10. In the main() function, perform the following tasks:
11. Create a pointer of type Base and assign it a new instance of Derived using dynamic allocation.
12. Call the show() method using the base class pointer to demonstrate polymorphism. This should invoke the show() function of the Derived class.
13. Deallocate the memory used for the Derived object using the base class pointer.

**Example Interaction:**

When the program is executed, the expected output should be:

Derived class show function

**Additional Notes:**

1. Emphasize the importance of using virtual functions for achieving polymorphism in C++.
2. Discuss the necessity of providing a virtual destructor in the base class to ensure proper cleanup of derived objects.
3. Encourage students to consider the implications of dynamic memory allocation and the importance of memory management in C++.

**Task: Demonstrate Method Overriding in Inheritance**

**Objective**: Create a C++ program that showcases the concepts of inheritance and method overriding. This will help students understand how derived classes can provide specific implementations of methods defined in base classes.

**Requirements:**

1. **Class Definitions**:
2. Create a base class named Base with the following features:
3. **Public Member Functions**:
4. Implement a public method show() that outputs the message: "Base class show function".
5. Ensure that the show() method is declared as a virtual function to allow overriding in derived classes.
6. Create a derived class named Derived that publicly inherits from Base:
7. **Override Member Function**:
8. Override the show() method to output the message: "Derived class show function".
9. **Main Function**:
10. In the main() function, perform the following tasks:
11. Create an instance of the Derived class.
12. Call the show() method on the Derived instance to demonstrate that it calls the overridden function in the Derived class.
13. Call the show() method from the Base class on the same instance to demonstrate how to access the base class's method explicitly.

**Example Interaction:**

When the program is executed, the expected output should be:

Derived class show function

Base class show function

**Additional Notes:**

1. Emphasize the significance of method overriding in achieving polymorphism in C++.
2. Discuss the use of the scope resolution operator (::) to access the base class's method explicitly when there is a derived class method with the same name.
3. Encourage students to think about when and why they would want to override methods in a derived class.\