

LAB 9: TO ILLUSTRATE THE CONCEPT OF PUBLIC, PROTECTED AND PRIVATE KEYWORD IN INHERITANCE.

OBJECTIVE:

1. Understand the concept of inheritance in object-oriented programming.
2. Demonstrate the usage of the public, protected, and private access modifiers in class inheritance.
3. Differentiate between the visibility of inherited members based on access modifiers.

THEORY:

In the world of object-oriented programming, the concept of inheritance stands as a cornerstone for building efficient and organized software systems. It allows us to create new classes (derived or child classes) based on existing ones (base or parent classes), thereby inheriting their attributes and behaviors. This mechanism promotes code reusability and maintains a hierarchical structure that facilitates the organization of complex software projects.

The primary objective of this lab is to delve into the intricacies of these access modifiers and understand their implications when used in the context of inheritance. In this report, we will comprehensively explore the definitions, characteristics, and practical applications of public, protected, and private access modifiers in inheritance scenarios.

Access Modifiers in Brief:

Public: The public access modifier allows class members to be accessible from any part of the program, regardless of whether the code is within the class itself or outside of it. Public members are, in essence, fully visible and can be accessed without any restrictions.

Protected: Members declared as protected are accessible within the class itself, as well as in any derived (child) classes. This means that while they are not visible to external code, they can be inherited and accessed by subclasses.

Private: The private access modifier restricts members to be accessible only within the class where they are declared. These members are entirely hidden from external code and cannot be accessed by derived classes.

REVIEWING PRIVATE, PROTECTED AND PUBLIC BASE CLASS ACCESS SPECIFIER.

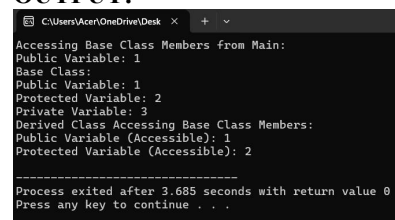
Base class access specifier	Private members	Protected members	Public members
Private	Never inherited they remain private to base class.	Become private members of the derived class.	Become private members of the derived class.
Protected	Never inherited they remain private to base class.	Become protected members of the derived class.	Become protected members of the derived class.
Public	Never inherited they remain private to base class.	Become protected members of the derived class.	Become public members of the derived class.

Fig: Access Specifier

//SOURCE CODE:

```
#include <iostream>
using namespace std;
class Base {
public:
    int publicVar;
    Base() : publicVar(1), protectedVar(2), privateVar(3) {}
    void display() {
        cout << "Base Class:" << endl;
        cout << "Public Variable: " << publicVar << endl;
        cout << "Protected Variable: " << protectedVar << endl;
        cout << "Private Variable: " << privateVar << endl;
    }
protected:
    int protectedVar;
private:
    int privateVar;
};
class Derived : public Base {
public:
    void accessBaseMembers() {
        cout << "Derived Class Accessing Base Class Members:" <<endl;
        cout << "Public Variable (Accessible): " << publicVar <<endl;
        cout << "Protected Variable (Accessible): " << protectedVar <<endl;
        // Private variable is not accessible in derived class
        // cout << "Private Variable (Not Accessible): " << privateVar <<endl;
    }
};
int main() {
    Base baseObj;
    Derived derivedObj;
    cout << "Accessing Base Class Members from Main:" <<endl;
    cout << "Public Variable: " << baseObj.publicVar <<endl;
    // Protected and private variables are not accessible in main
    // cout << "Protected Variable: " << baseObj.protectedVar << endl;
    // cout << "Private Variable: " << baseObj.privateVar << endl;
    baseObj.display();
    derivedObj.accessBaseMembers();
    return 0;
}
```

OUTPUT:



```
C:\Users\Acer\OneDrive\Desktop
Accessing Base Class Members from Main:
Public Variable: 1
Base Class:
Public Variable: 1
Protected Variable: 2
Private Variable: 3
Derived Class Accessing Base Class Members:
Public Variable (Accessible): 1
Protected Variable (Accessible): 2
-----
Process exited after 3.685 seconds with return value 0
Press any key to continue . . .
```

CONCLUSION:

In this C++ program, we've demonstrated the core concepts of access specifiers (public, protected, and private) within an inheritance context. public members are accessible from any part of the program. We showcased this with the publicVar from the Base class. protected members are accessible within the class itself and in derived classes. Understanding these access modifiers is essential for controlling member visibility, ensuring data encapsulation, and designing robust object-oriented programs.

