Protected Access Modifier in C++ | C++ Tutorials for Beginners #39

In this tutorial, we will discuss protected access modifiers in C++

Protected Access Modifiers in C++

Protected access modifiers are similar to the private access modifiers but protected access modifiers can be accessed in the derived class whereas private access modifiers cannot be accessed in the derived class. A table is shown below which shows the behavior of access modifiers when they are derived “public”, “private”, and “protected”.

	Public Derivation	Private Derivation	Protected Derivation

Private members	Not Inherited	Not Inherited	Not Inherited
Protected members	Protected	Private	Protected
Public members	Public	Private	Protected

As shown in the table,

1. If the class is inherited in public mode then its private members cannot be inherited in child class.
2. If the class is inherited in public mode then its protected members are protected and can be accessed in child class.
3. If the class is inherited in public mode then its public members are public and can be accessed inside child class and outside the class.
4. If the class is inherited in private mode then its private members cannot be inherited in child class.
5. If the class is inherited in private mode then its protected members are private and cannot be accessed in child class.
6. If the class is inherited in private mode then its public members are private and cannot be accessed in child class.
7. If the class is inherited in protected mode then its private members cannot be inherited in child class.
8. If the class is inherited in protected mode then its protected members are protected and can be accessed in child class.
9. If the class is inherited in protected mode then its public members are protected and can be accessed in child class.

An example program to demonstrate the concept of protected access modifiers is shown below.

```
#include<iostream>
```

```
using namespace std;

class Base{
    protected:
        int a;
    private:
        int b;

};

class Derived: protected Base{

};

int main(){
    Base b;
    Derived d;
    // cout<<d.a; // Will not work since a is protected in both base as well as derived clas
    return 0;
}
```

Code Snippet 1: Protected Access Modifier Example Program

As shown in a code snippet 1,

- 1st we created a “Base” class which consists of protected data member integer “a” and private data member integer “b”.

- 2nd we created a “Derived” class which is inheriting the “Base” class in protected mode.
- 3rd the object “b” of the data type “Base” is created.
- 4th the object “d” of the data type “Derived” is created.
- 5th if we try to print the value of the data member “a” by using the object “d”; the program will throw an error because the data member “a” is protected and the derived class is inherited in the protected mode. So the data member “a” can only be accessed in the “derived” but not outside the class.

Previous Next

CodeWithHarry