Classes, Public and Private access modifiers in C++ | C++ Tutorials for Beginners #21

In this tutorial, we will discuss classes, public and private access modifiers in C++

Why use classes instead of structures

Classes and structures are somewhat the same but still, they have some differences. For example, we cannot hide data in structures which means that everything is public and can be accessed easily which is a major drawback of the structure because structures cannot be used where data security is a major concern. Another drawback of structures is that we cannot add functions in it.

Classes in C++

Classes are user-defined data-types and are a template for creating objects. Classes consist of variables and functions which are also called class members.

Public Access Modifier in C++

All the variables and functions declared under public access modifier will be available for everyone. They can be accessed both inside and outside the class. Dot (.) operator is used in the program to access public data members directly.

Private Access Modifier in C++

All the variables and functions declared under a private access modifier can only be used inside the class. They are not permissible to be used by any object or function outside the class.

An example program to demonstrate classes, public and private access modifiers are shown in Code Snippet 1.

```
class Employee
{
    private:
        int a, b, c;
    public:
        int d, e;
        void setData(int a1, int b1, int c1); // Declaration
        void getData(){
            cout<<"The value of a is "<<a<<endl;
            cout<<"The value of b is "<<b<<endl;
            cout<<"The value of c is "<<c<<endl;
            cout<<"The value of d is "<<d<<endl;
            cout<<"The value of e is "<<e<<endl;
        }
};

void Employee :: setData(int a1, int b1, int c1){
    a = a1;
    b = b1;
    c = c1;
}
```

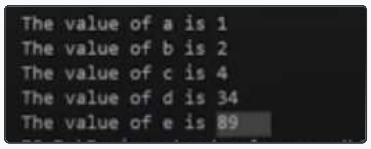
Code Snippet 1: Class Program

As shown in Code Snippet 1, 1st we created an “employee” class, 2nd three integer variables “int a”, “int b”, and “int c” were declared under the private access modifier, 3rd two integer variables “int d” and “int e” was declared under the public access modifiers, 4th “setData” function was declared, 5th “getData” function was defined and values of all the variables are printed. 6th “setData” function was defined outside the “employee” class by using a scope resolution operator; “setData” function is used to assign values to the private member of the class. An example to create the object of the class and use its class members is shown in Code Snippet 2.

```
int main(){
    Employee harry;
    harry.d = 34;
    harry.e = 89;
    harry.setData(1,2,4);
    harry.getData();
    return 0;
}
```

Code Snippet 2: Creating Object Example

As shown in Code Snippet 2, 1st we created an object “harry” of the class “employee”; 2nd we assigned values to “int d” and “int e” which are public class members. If we try to assign values to the private class member’s compiler will throw an error. 3rd we passed the values to the function “setData” and at the end, we called “getData” function which will print the values of all the variables. The output for the following program is shown in figure 1.



```
The value of a is 1
The value of b is 2
The value of c is 4
The value of d is 34
The value of e is 89
```

Figure 1: Class Program Output

As shown in figure 1, all the values of our data members are printed.

Code as described/written in the video

```
#include<iostream>
using namespace std;

class Employee
{
    private:
        int a, b, c;
    public:
        int d, e;
        void setData(int a1, int b1, int c1); // Declaration
        void getData(){
            cout<<"The value of a is "<<a<<endl;
            cout<<"The value of b is "<<b<<endl;
            cout<<"The value of c is "<<c<<endl;
            cout<<"The value of d is "<<d<<endl;
            cout<<"The value of e is "<<e<<endl;
        }
};

void Employee :: setData(int a1, int b1, int c1){
    a = a1;
    b = b1;
    c = c1;
}
```

← Previous

Next →