CLASS

C++ CLASSES AND OBJECTS

Introduction of C++ Class

A class is a blueprint for the object.
We can think of a class as a sketch (prototype) of a house. It contains all the details about the floors, doors, windows, etc. Based on these descriptions we build the house. House is the object.

Create a Class

A class is defined in C++ using keyword class followed by the name of the class.

The body of the class is defined inside the curly brackets and terminated by a semicolon at the end.

```
class className {
  // some data
  // some functions
};
```

For example,

```
class Room {
   public:
      double length;
      double breadth;
      double height;

      double calculateArea(){
         return length * breadth;
      }
```

```
    double calculateVolume(){

      return length * breadth * height;

    }


};
```

Here, we defined a class named Room.

The variables length, breadth, and height declared inside the class are known as data members. And, the functions calculateArea() and calculateVolume() are known as member functions of a class.

C++ Objects

When a class is defined, only the specification for the object is defined; no memory or storage is allocated.

To use the data and access functions defined in the class, we need to create objects.

Syntax to Define Object in C++

```
className objectVariableName;
```

We can create objects of Room class (defined in the above example) as follows:

```
// sample function

void sampleFunction() {

  // create objects

  Room room1, room2;

}


int main(){

  // create objects

  Room room3, room4;
}
```

Here, two objects room1 and room2 of the Room class are created in sampleFunction(). Similarly, the objects room3 and room4 are created in main().

As we can see, we can create objects of a class in any function of the program. We can also create objects of a class within the class itself, or in other classes.

Also, we can create as many objects as we want from a single class.

C++ Access Data Members and Member Functions

We can access the data members and member functions of a class by using a . (dot) operator. **For example**,

```
room2.calculateArea();
```

This will call the calculateArea() function inside the Room class for object room2.

Similarly, the data members can be accessed as:

```
room1.length = 5.5;
```

In this case, it initializes the length variable of room1 to 5.5.

Example 1: Object and Class in C++ Programming

```
// Program to illustrate the working of
// objects and class in C++ Programming

#include <iostream>
using namespace std;

// create a class
class Room {

   public:
    double length;
    double breadth;
    double height;

    double calculateArea() {
        return length * breadth;
    }

    double calculateVolume() {
        return length * breadth * height;
    }
};

int main() {

    // create object of Room class
    Room room1;

    // assign values to data members
    room1.length = 42.5;
    room1.breadth = 30.8;
    room1.height = 19.2;

    // calculate and display the area and volume of the room
    cout << "Area of Room =  " << room1.calculateArea() << endl;
    cout << "Volume of Room =  " << room1.calculateVolume() << endl;


    return 0;

}
```

Run Code

Output

```
Area of Room =  1309

Volume of Room =  25132.8
```

In this program, we have used the Room class and its object room1 to calculate the area and volume of a room.

In main(), we assigned the values of length, breadth, and height with the code:

```
room1.length = 42.5;

room1.breadth = 30.8;

room1.height = 19.2;
```

We then called the functions calculateArea() and calculateVolume() to perform the necessary calculations.

Note the use of the keyword public in the program. This means the members are public and can be accessed anywhere from the program.

As per our needs, we can also create private members using the private keyword. The private members of a class can only be accessed from within the class. **For example**,

```
class Test {


private:

    int a;

    void function1() { }
public:
   int b;
   void function2() { }
}
```

Here, a and function1() are private. Thus they cannot be accessed from outside the class.

On the other hand, b and function2() are accessible from everywhere in the program.

To learn more about public and private keywords, please visit our C++ Class Access Modifiers tutorial.

Example 2: Using public and private in C++ Class

```
// Program to illustrate the working of
// public and private in C++ Class

#include <iostream>
using namespace std;

class Room {

  private:
   double length;
   double breadth;
   double height;

  public:
   // function to initialize private variables
   void initData(double len, double brth, double hgt) {
       length = len;
       breadth = brth;
       height = hgt;
   }

   double calculateArea() {
       return length * breadth;
   }

   double calculateVolume() {
       return length * breadth * height;
   }
};

int main() {

   // create object of Room class
   Room room1;

   // pass the values of private variables as arguments
   room1.initData(42.5, 30.8, 19.2);

   cout << "Area of Room =  " << room1.calculateArea() << endl;
   cout << "Volume of Room =  " << room1.calculateVolume() << endl;


   return 0;

}
```

Run Code

Output

```
Area of Room =  1309

Volume of Room =  25132.8
```

The above example is nearly identical to the first example, except that the class variables are now private.

Since the variables are now private, we cannot access them directly from main(). Hence, using the following code would be invalid:

```
// invalid code

obj.length = 42.5;

obj.breadth = 30.8;

obj.height = 19.2;
```

Instead, we use the public function initData() to initialize the private variables via the function parameters double len, double brth, and double hgt.