**C++ Classes and Objects**

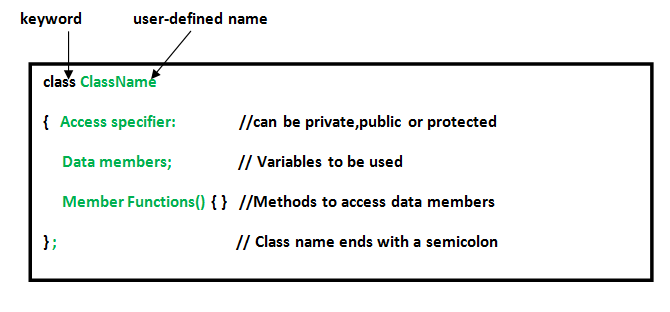
**Class in C++** is the building block that leads to Object-Oriented programming. It is a user-defined data type, which holds its own data members and member functions, which can be accessed and used by creating an instance of that class. A C++ class is like a blueprint for an object. For Example: Consider the Class of **Cars**. There may be many cars with different names and brands but all of them will share some common properties like all of them will have *4 wheels*, *Speed Limit*, *Mileage range,* etc. So here, Car is the class, and wheels, speed limits, and mileage are their properties.

* A Class is a user-defined data type that has data members and member functions.
* Data members are the data variables and member functions are the functions used to manipulate these variables together, these data members and member functions define the properties and behavior of the objects in a Class.
* In the above example of class *Car*, the data member will be *speed limit*, *mileage,* etc, and member functions can be *applying brakes*, *increasing speed,* etc.

An **Object** is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.

**Defining Class and Declaring Objects**

A class is defined in C++ using the 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.



## ****Declaring 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, you need to create objects.

### ****Syntax****

ClassName ObjectName;

**Accessing data members and member functions**: The data members and member functions of the class can be accessed using the dot(‘.’) operator with the object. For example, if the name of the object is *obj* and you want to access the member function with the name *printName()* then you will have to write *obj.printName()*.

## ****Accessing Data Members****

The public data members are also accessed in the same way given however the private data members are not allowed to be accessed directly by the object. Accessing a data member depends solely on the access control of that data member. This access control is given by  There are three access modifiers: **public, private, and protected**.

## What are Classes in C++?

A class is a template or a blueprint that binds the properties and functions of an entity. You can put all the entities or objects having similar attributes under a single roof, known as a class. Classes further implement the core concepts like encapsulation, data hiding, and abstraction. In C++, a class acts as a data type that can have multiple objects or instances of the class type.

### Description of the Syntax:

* class: This is the keyword used to declare a class that is followed by the name of the class.
* class\_name: This is the name of the class which is specified along with the keyword class.
* access\_specifier: It provides the access specifier before declaring the members of the class. These specifiers control the access of the class members within the class. The specifiers can be public, protected, and private.
* data\_member: These are the variables of the class to store the data values.
* member\_function: These are the functions declared inside the class.

## What are Objects in C++?

Objects in C++ are analogous to real-world entities. There are objects everywhere around you, like trees, birds, chairs, tables, dogs, cars, and the list can go on. There are some properties and functions associated with these objects. Similarly, C++ also includes the concept of objects. When you define a class, it contains all the information about the objects of the class type. Once it defines the class, it can create similar objects sharing that information, with the class name being the type specifier.

Significance of Class and Object in C++

The concept of class and object in C++ makes it possible to incorporate real-life analogy to programming. It gives the data the highest priority using classes. The following features prove the significance of class and object in C++:

* Data hiding: A class prevents the access of the data from the outside world using access specifiers. It can set permissions to restrict the access of the data.
* Code Reusability:  You can reduce code redundancy by using reusable code with the help of inheritance. Other classes can inherit similar functionalities and properties, which makes the code clean.
* Data binding: The data elements and their associated functionalities are bound under one hood, providing more security to the data.
* Flexibility: You can use a class in many forms using the concept of polymorphism. This makes a program flexible and increases its extensibility.

## Member Functions in Classes

The member functions are like the conventional functions. It defines these methods inside a class and has direct access to all the data members of its class. When you define a member function, it only creates and shares one instance of that function by all the instances of that class.

## Method Definition Outside and Inside of a Class

The following two ways can define a method or member functions of a class:

1. Inside class definition
2. Outside class definition

The function body remains the same in both approaches to define a member function. The difference lies only in the function's header. Now, have a deeper understanding of these approaches.

### Inside Class Definition

This approach of defining a member function is generally preferred for small functions. It defines a member function inside a class in the same familiar way as it defines a conventional function. It specifies the return type of the function, followed by the function name, and it provides arguments in the function header. Then it provides the function body to define the complete function. The member functions that are defined inside a class are automatically inline

C++ is an object-oriented language that is used to model real-world entities into programs. All object-oriented programming languages achieve this task using classes and objects. Classes act as a blueprint to create objects with similar properties. The concept of classes and objects in C++ is the fundamental idea around which the object-oriented approach revolves around. It enhances the program’s efficiency by reducing code redundancy and debugging time.

Now, you will understand the concept of the class and object in C++ with the help of a real-life example. Suppose you have a small library. In a library, all books have some common properties like book\_name, author\_name, and genre. Now imagine you want to create a catalog of all the books in your collection. Instead of creating separate classes for every book you own, you can create a Book class that serves as a template for all the books in your library.

## What are Classes in C++?

A class is a template or a blueprint that binds the properties and functions of an entity. You can put all the entities or objects having similar attributes under a single roof, known as a class. Classes further implement the core concepts like encapsulation, data hiding, and abstraction. In C++, a class acts as a data type that can have multiple objects or instances of the class type.

Consider an example of a railway station having several trains. A train has some characteristics like train\_no, destination, train\_type, arrival\_time, and departure\_time. And its associated operations are arrival and departure. You can define a class can for a train as follows:

class train

{

    // characteristics

    int train\_no;

    char destination;

    char train\_type;

    int arrival\_time;

    int departure\_time;

    // functions

    int arrival(delayed\_time)

    {

        arrival\_time += delayed\_time;

        return arr\_time;

    }

    int departure(delayed\_time)

    {

        departure\_time += delayed\_time;

        return departure\_time;

    }

}

The above class declaration contains properties of the class, train\_no, destination, train\_type, arrival\_time, and departure\_time as the data members. You can define the operations, arrival, and departure as the member functions of the class.

### Syntax to Declare a Class in C++:

class class\_name

{

    // class definition

    access\_specifier:         // public, protected, or private

    data\_member1;   // data members

    data\_member2;

    func1(){}       // member functions

    func2(){}

};

#### Want a Top Software Development Job? Start Here!

Full Stack Developer - MERN Stack[EXPLORE PROGRAM](https://www.simplilearn.com/full-stack-developer-course-mern-certification-training?source=GhPreviewCTABanner)

### Description of the Syntax:

* class: This is the keyword used to declare a class that is followed by the name of the class.
* class\_name: This is the name of the class which is specified along with the keyword class.
* access\_specifier: It provides the access specifier before declaring the members of the class. These specifiers control the access of the class members within the class. The specifiers can be public, protected, and private.
* data\_member: These are the variables of the class to store the data values.
* member\_function: These are the functions declared inside the class.

## What are Objects in C++?

Objects in C++ are analogous to real-world entities. There are objects everywhere around you, like trees, birds, chairs, tables, dogs, cars, and the list can go on. There are some properties and functions associated with these objects. Similarly, C++ also includes the concept of objects. When you define a class, it contains all the information about the objects of the class type. Once it defines the class, it can create similar objects sharing that information, with the class name being the type specifier.

Consider the example of a railway station discussed in the previous section. After defining the class train, you can create similar objects for this class. For example, train\_A and train\_B. You can create the objects for the class defined above in the following way:

    train train\_A, train\_B;

The syntax to create objects in C++:

    class\_name object\_name;

The object object\_name once created, can be used to access the data members and member functions of the class class\_name using the dot operator in the following way:

    obj.data\_member = 10;   // accessing data member

    obj.func();             // accessing member function

## Significance of Class and Object in C++

The concept of class and object in C++ makes it possible to incorporate real-life analogy to programming. It gives the data the highest priority using classes. The following features prove the significance of class and object in C++:

* Data hiding: A class prevents the access of the data from the outside world using access specifiers. It can set permissions to restrict the access of the data.
* Code Reusability:  You can reduce code redundancy by using reusable code with the help of inheritance. Other classes can inherit similar functionalities and properties, which makes the code clean.
* Data binding: The data elements and their associated functionalities are bound under one hood, providing more security to the data.
* Flexibility: You can use a class in many forms using the concept of polymorphism. This makes a program flexible and increases its extensibility.

## Array of Objects

In C++, you can declare an array of any data type. This also includes the class type. The Array of objects is an array containing the elements of the class type. The definition of an array of objects is similar to the usual array definition with the data type replaced with the class name. The following syntax is used to define an array of objects:

class\_name obj[20];

The array obj contains 20 elements of the type class\_name. These elements are, obj[1], obj[2], obj[3], …… obj[20]. An array of objects is preferred when there is a requirement for numerous objects of the same class. Instead of creating obj1, obj2, obj3,....obj20, you can simply declare obj[20].

## Objects as Function Arguments

In C++, you can pass the objects of a class as arguments in the same way you pass a variable to a function as arguments. An object can be passed as an argument to a member function, a friend function, and a non-member function. The private and public members of the object are accessible to the member function and the friend function. However, the non-member function is only allowed to access the public members of the object.