

**Introduction to Class and Object**

A class is user defined data type which consists of two sections, a private and a protected section that holds data and a public section that holds the interface operations. A class definition is a process of naming a class and data variables, and methods or interface operations of the class. A class is defined in C++ using keyword class followed by the name of class. The body of class is defined inside the curly brackets and terminated by a semicolon at the end.

class ABC

           {

                       private:

                                   data\_type members

                                   implementation operations

                       public:

                                   data\_type members

                                        implementation operations

                       protected:

                                   data\_type operations

                                   implementation operations

           };

class user\_defined \_name var\_1,var\_2,…….,var\_n;

#include<iostream.h>

#include<conio.h>

class emp

{

public:

int empid;

char name[15];

class data

{

public:

int day,month,year;

}doj;

}e;

void main()

{

clrscr();

cout<<"enter employee id"<<endl;

cin>>e.empid;

cout<<"enter name"<<endl;

cin>>e.name;

cout<<"enter date of joining"<<endl;

cin>>e.doj.day;

cout<<"enter month of joining"<<endl;

cin>>e.doj.month;

cout<<"enter year of joining"<<endl;

cin>>e.doj.year;

cout<<"emp.id:"<<e.empid<<endl;

cout<<"emp name:"<<e.name<<endl;

cout<<"date of joing:"<<e.doj.day<<"-"<<e.doj.month<<"-"<<e.doj.year<<endl;

getch();

}

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. For example: in real life, a car is an object. The car has attributes, such as weight and color, and methods, such as drive and brake. Attributes and methods are basically variables and functions that belongs to the class.

**Creating Objects** can be done as following:

**ABC ob1;**

**ABC ob2;**

Both of the objects ob1 and ob2 will have their own copy of data members.

**Here is the following Example:**

class Car {

  public:

   string brand;

   string model;

    int year;

};

int main() {

  // Create an object of Car

 Car carObj1;

 carObj1.brand = "BMW";

  carObj1.model = "X5";

 carObj1.year = 1999;

  // Create another object of Car

 Car carObj2;

 carObj2.brand = "Ford";

 carObj2.model = "Mustang";

 carObj2.year = 1969;

  // Print attribute values

 cout << carObj1.brand << " " << carObj1.model << " " << carObj1.year << "\n";

 cout << carObj2.brand << " " << carObj2.model << " " << carObj2.year << "\n";

  return 0;

}

**Accessing the Class Members**

Accessing a data member depends on the access control of that data member. If its public, then the data member can be easily accessed using the direct member access (.) operator with the object of that class.

If, the data member is defined as private or protected, then we cannot access the data variables directly. Then we will have to create special public member functions to access, use or initialize the private and protected data members. These member functions are also called Accessors and Mutator methods or getter and setter functions.

 Here is the example how to access class members.

class MyClass

{

 int myNum;

 string myString;

};

int main()

{

 MyClass myObj;

  myObj.myNum = 15;                        //Accessing    Data   members

  myObj.myString = "Sometext";

 cout<<myObj.myNum << "\n";

 cout<<myObj.myString;

  return 0;

}

**Accessing Public Data Members**

Following is an example to show you how to initialize and use the public data members using the dot (.) operator and the respective object of class.

class Student

{

 public:

 int rollno;

 string name;

};

int main()

{

   Student A;

   Student B;

    // setting values for A object

   A.rollno=2021;

   A.name="Karan";

   // setting values for B object

   B.rollno=20212121;

B.name="Arjun";

cout <<"Name and Roll no of A is: "<< A.name << "-" << A.rollno;

 cout <<"Name and Roll no of B is: "<< B.name << "-" << B.rollno;

}

OUTPUT:

Name and Roll no of A is: Karan-2021

Name and Roll no of B is: Arjun-20212121

**Accessing Private Data Members**

To access, use and initialize the private data member you need to create getter and setter functions, to get and set the value of the data member.

The setter function will set the value passed as argument to the private data member, and the getter function will return the value of the private data member to be used. Both getter and setter function must be defined public.

class Student

{

   private:   // private data member

   int rollno;

    public:

   // public function to get value of rollno - getter

   int getRollno()

   {

       return rollno;

   }

   // public function to set value for rollno - setter

   void setRollno(int i)

   {

       rollno=i;

   }

};

int main()

{

   Student A;

   A.rollono=1; //Compile time error

   cout<< A.rollno; //Compile time error

   A.setRollno(1); //Rollno initialized to 1

   cout<< A.getRollno(); //Output will be 1

}

So this is how we access and use the private data members of any class .

**Summary**

* In this lecture we have discussed about Class and object.
* We have discussed how to create objects of class.
* We have discussed how to access class members.

**Frequently Asked Questions**

Some of the most frequently asked questions are:

**Q1** What is a class?

A class is user defined data type which consists of two sections, a private and a protected section that holds data and a public section that holds the interface operations. A class definition is a process of naming a class and data variables, and methods or interface operations of the class. A class is defined in C++ using keyword class followed by the name of class. The body of class is defined inside the curly brackets and terminated by a semicolon at the end.

**Q2** What is an object?

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. For example: in real life, a car is an object.

**Q3** How to create an object?

Objects are created from classes. Class objects are declared in a similar way as variables are declared. The class name must start, followed by the object name.

**class name object name;**