Object Oriented Programming Using Cpp

Topic:

Class And Object

Chapter: 6 Class and Object

6.1 Introduction

In C language, one of the major disadvantages was that the data was not secure. This was because, structure in C programming language, the members were accessed directly by the variables of that structure i.e. By default the structure members are public in nature which makes them easily accessible in the program. This produced serious drawbacks in terms of security of data.

To overcome the above said problem, C++ introduced a new version of structure which enhanced the security features. Initially it was known as "Structure in C++" but later it was called as "class". Class provided the much needed security to the data using access specifier (which will be discussed later).

6.2 Class in C++

Before discussing class, we need to understand the basic differences between structure in C and structure in C++.

In C, structure can be defined as below:

Syntax:

```
struct      <structure_name>
{
        ……..……;
        ……..……;
        ……..……;
};
```


Now, this structure has following limitations:

- It can contain only data members. Functions are not allowed inside structure.
- The structure members are public in nature which makes them easily accessible throughout the program.
- Use of struct keyword is compulsory while declaring the structure variables.

On the other hand, C++ provided a new version of structure by removing the above mentioned limitations. Technically it is known as class.

6.2.1 Class and Object

Object: An object is a real world entity which has some features and behaviours. An object has its own existence.

Class: A class is a collection of objects which possess similar features and behaviours.

Ideally, class does not exist on its own. It exists due to the objects.

To understand this, consider the following examples.

1. Class: **student**

Object: **Rahul, Mukesh, Priya etc…**

Features:

- **Name**
- **Branch**
- **Roll**
- **Semester**
- **Contact Number**

Behaviours:

- **Academic performance**
- **Reading capacity**
- **Understanding level**

2. Class: **bike**

Object: **pulsar, shine, activa etc.**

Features:

- **Company name**
- **Model name and number**
- **Color**

Behaviours:

- **Speed**
- **Average**
- **Mileage**
- **Brake system**
- **Sound system**
- **Lamp**

If we analyse the above two examples then we find that both class have some objects. Objects of student class possess similar properties whereas objects of bike class possess similar properties.So, it is very important that the objects of similar features must belong to the same class.

Syntax for class definition

```
class       <class_name>
{
    Access specifier:
        Data member 1;
        Data member 2;
        .....;
        Data member n;
    Access specifier:
        Member function()
        {
        }
};
```

Object Declaration: An object is a real world entity which has its own existence. We know that a class is a collection of some data members and some functions. So, Objects are actually the class variables. Once an object is created, memory allocation is done. In this fashion, we can declare number of objects and each object will have the separate copies of data members of the class. Memory allocation is done for each object.

An object can be declared using following syntaxes:

Syntax 1:

```
class       <class_name>
{
    Access specifier:
        Data member 1;
```

```
                Data member 2;
                ……;
                Data member n;
        Access specifier:
                Member function()
                {

                }
    }c1,c2;
```

Syntax 2:

```
    class       <class_name>
    {
        Access specifier:
                Data member 1;
                Data member 2;
                ……;
                Data member n;
        Access specifier:
                Member function()
                {

                }
    };
    void main()
    {
        <class_name>        object_name;
    }
```

In the 1st syntax, the declared objects are global in nature and they can be used throughout the program. On the other hand, in 2nd syntax, the declared objects are local in nature as they are declared inside main function. Hence, they can't be used out of main function. A user can opt any of the two syntaxes based on the requirement and choice.

6.2.2 Access Specifiers

C++ provides security to the data using a mechanism known as access specifiers. Following three types of access specifier is available in C++.

- **Public:** Members defined under public keyword are public in nature which means that they can be accessed directly using an object of a class with the help of dot (.) operator.

Program 23: Write a program in C++ to enter the details of a student and display them using class and object. The details should be public in nature.

Code:

```
#include<iostream.h>
#include<conio.h>
class     STUDENT
{
        public:                    // Public specifier
          char    name[10];
          char    dept[5];
        int       roll;
        int       sem;
          float   CGPA;
};
void    main()
{
        STUDENT    s1;
        cout<<"Enter the student details";
        cin>>s1.name>>s1.dept>>s1.roll>>s1.sem>>s1.CGPA;
        cout<<"The student details are as below"<<endl;
        cout<<"Name: "<<s1.name<<endl;
        cout<<"Dept: "<<s1.dept<<endl;
        cout<<"Roll No: "<<s1.roll<<endl;
        cout<<"Semester: "<<s1.sem<<endl;
```

```
        cout<<"CGPA: "<<s1.CGPA<<endl;
        getch();
    }
```

O/P:

Enter the student details
Rahul
CSE
01
04
8.56 // Press enter here
The student details are as below:
Name: Rahul
Dept: CSE
Roll No: 01
Semester: 04
CGPA: 8.56

- **Private:** Members defined under private keyword are secured enough that even the object of that class not access them directly. In order to use the private members of a class, a member function is required. That member function must be public in nature so that they can be accessed by object with the help of dot (.) operator. This public member function can access the private members of the class.

Program 24: Write a program in C++ to enter the details of a student and display them using class and object. The details should be private in nature and use public member function to access the private members.

Code:

```
#include<iostream.h>
#include<conio.h>
class    STUDENT
{
    private:                    // Private specifier
        char    name[10];
        char    dept[5];
        int  roll;
        int  sem;
        floatCGPA;
    public:
        void  input()
        {
            cout<<"Enter the student details"
```

```
            cin>>name>>dept>>roll>>sem>>CGPA;
        }
        void    display()
        {
            cout<<"The student details are as below:"
            cout<<"Name: "<<name<<endl;
            cout<<"Dept: "<<dept<<endl;
            cout<<"Roll No: "<<roll<<endl;
            cout<<"Semester: "<<sem<<endl;
            cout<<"CGPA: "<<CGPA<<endl;
        }
    };
void    main()
{
    STUDENT    s1;
    s1.input();          // Object accesses public function
    s1.display();        // Object accesses public function
    getch();
}
```

O/P:

Enter the student details
Rahul
CSE
01
04
8.56 // Press enter here
The student details are as below:
Name: Rahul
Dept: CSE
Roll No: 01
Semester: 04
CGPA: 8.56

Note:

1. The default access specifier for a class is private. Hence, writing private is not compulsory.
2. A class member can never be accessed directly. It must be accessed using object of that class with the help of dot (.) operator. Again, an object can also access only the public members of a class
3. The 3rd access specifier is "protected" which will be discussed in Inheritance.

6.2.3 Defining Member Functions of a class

A function which is a member of a class is known as member function. It may be defined in following two ways:

1. Inside the class: A member function may be defined inside the class. Now, the member function may be public or private. The most general practice is to make member function as public.

Syntax:

```
class       <class_name>
{
        Access specifier:
                ……..;
                ……..;
        Access specifier:
                Return_type   function_name()
                {

                }
};
```

A member function is made public so that an object can access this member function. This public member function can access the private members of a class.

Program 25: Write a program in C++ to enter the details of a student and display them using class and object. The details should be private in nature and use public member function to access the private members.

Code:

```
#include<iostream.h>
#include<conio.h>
class     STUDENT
{
                                        // Private specifier by default
```

```
        char    name[10];
        char    dept[5];
    int         roll;
    int         sem;
        float   CGPA;
public:
    void    input()
    {
            cout<<"Enter the student details"
            cin>>name>>dept>>roll>>sem>>CGPA:
    }
    void        display()
    {
            cout<<"The student details are as below:"
            cout<<"Name: "<<name<<endl;
            cout<<"Dept: "<<dept<<endl;
            cout<<"Roll No: "<<roll<<endl;
            cout<<"Semester: "<<sem<<endl;
            cout<<"CGPA: "<<CGPA<<endl;
    }
};
void    main()
{
    STUDENT     s1;
    s1.input();         // Object accesses public function
    s1.display();       // Object accesses public function
    getch();
}
```


O/P:

Enter the student details
Rahul
CSE
01
04
8.56 // Press enter here
The student details are as below:
Name: Rahul
Dept: CSE
Roll No: 01
Semester: 04
CGPA: 8.56

2. Outside the class: Before defining the member function outside of the class, we must know that why is it required?

Actually, the member function defined inside the class behaves as inline function. Due to this some restrictions are applicable on the member functions like definition should be small, it should not contain large number of variables, arrays can't be used as inline function does not support loop statements.

The above mentioned restrictions produced a requirement of defining member functions outside the class so that it can be used efficiently. Now, a member function can be defined outside using following syntax:

Syntax

```
class       <class_name>
{
      Access specifier:
            .....;
            .....;
      Access specifier:
            Return_type   function_name();
};
Return_type       class_name : : function_name()
{


}
```


Here, we see that the function prototype is mentioned in the class definition and the member function is defined outside the class using class name and scope resolution operator. The reason for this is, a program may contain multiple classes and it may be possible that different classes have member function with same name. There is absolutely no problem in this because the scope of the member function is limited to that class only where it is defined. So, when a member function is defined inside the system knows that function belongs to which class but when a function is defined outside the class, this information is known to the system using the corresponding class name and scope resolution operator.

Program 26: Write a program in C++ to enter the details of a student and display them using class and object. The details should be private in nature and use public member function to access the private members. The member functions must be defined outside the class.

Code:

```
#include<iostream.h>
#include<conio.h>
class    STUDENT
{
                                        // Private specifier by default
          char    name[10];
          char    dept[5];
        int       roll;
        int       sem;
          float   CGPA;
    public:
          void    input();
          void    display();
};
void   STUDENT: :input()
{
    cout<<"Enter the student details"
    cin>>name>>dept>>roll>>sem>>CGPA;
}
void   STUDENT: : display()
{
    cout<<"The student details are as below:"
    cout<<"Name: "<<name<<endl;
    cout<<"Dept: "<<dept<<endl;
    cout<<"Roll No: "<<roll<<endl;
    cout<<"Semester: "<<sem<<endl;
    cout<<"CGPA: "<<CGPA<<endl;
}
void   main()
{
    STUDENT     s1;
    s1.input();          // Object accesses public function
    s1.display();        // Object accesses public function
    getch();
}
```

O/P:

Enter the student details
Rahul
CSE
01
04
8.56 // Press enter here
The student details are as below:
Name: Rahul
Dept: CSE
Roll No: 01
Semester: 04
CGPA: 8.56

6.2.4 Characteristics of member functions

A member function has following properties:

- A member function of a class can be invoked only by using an object of that class with the help of dot (.) operator.
- Same function can be used in any number of classes without any problem. This is because the scope of the member function is limited to the class where it is defined.
- One member function can invoke another member function of the same class without using object.

6.3 Data Hiding

Declaring class members under private keyword provides some security to the data. This is known as data hiding. Generally, data members are declared as private however sometimes member functions can also be declared as private. In this case, the objects can't access these member functions also. To access them, the objects must access a public member function, which in turn can access the private member function.

Program 27: Write a program in C++ to enter the details of a student and display them using class and object. The details should be private in nature and use private member function for input purpose.

Code:

```
#include<iostream.h>
```

```
#include<conio.h>
class     STUDENT
{
                                                  // Private specifier by default
                  char    name[10];
                  char    dept[5];
               int        roll;
               int        sem;
                  float   CGPA;
                  void    input();                // Private member function
        public:
               void    display();                 // Public member function
};
void   STUDENT: :input()
{
        cout<<"Enter the student details"
        cin>>name>>dept>>roll>>sem>>CGPA:
}
void   STUDENT: : display()
{
        input();          // Call to private member function
        cout<<"The student details are as below:"
        cout<<"Name: "<<name<<endl;
        cout<<"Dept: "<<dept<<endl;
        cout<<"Roll No: "<<roll<<endl;
        cout<<"Semester: "<<sem<<endl;
        cout<<"CGPA: "<<CGPA<<endl;
}
void   main()
{
        STUDENT      s1;
        s1.display();          // Object accesses public function
        getch();
}
```

O/P:

Enter the student details
Rahul CSE 01 04 8.56 // Press enter
The student details are as below:
Name: Rahul
Dept: CSE
Roll No: 01
Semester: 04

CGPA: 8.56

6.4 Static Members of a class

A class may contain following type of static members:

- Static data members
- Static member functions

6.4.1 Static data member

When a data member of a class is declared with **static** keyword then it is known as static data member. It can be done using following syntax:

Syntax:

static data_type variable;

6.4.1.1 Features of static data member

- Static data member is associated with the class and not with any object i.e. It is declared only once and then it is shared among the objects of the class. Here, we must remember that objects of a class have a separate copy of non-static data members whereas they share the same copy of static data member.
- The initialization of static data member is extremely important else it will contain garbage value. A static data member is initialized using following syntax:
  - **Data_type class_name : : variable_name=value;**
  - **Data_type class_name : : variable_name;**

The 1st syntax assigns the value to the static data member whereas in 2nd syntax no value is provided by the user. In this case, system assigns the default value to the static data member as per the data type.

- Static data member is initialized only once. After that it always retains the last changed value. It means that any change made to static data member using an object is also reflected to other objects. **This point is very important**.
- Static data member can be accessed using normal member function as well as static member function.

Program 28: Write a C++ program to illustrate the concept of static data member.

Code:

```
#include<iostream.h>
#include<conio.h>
class NUM
{
        int   n1;
        static int n2;
        public:
                void  input()
                {
                        cout<<"Enter the value for n1";
                        cin>>n1;
                }
                void  calculate()
                {
                        ++n1;
                        ++n2;
                        cout<<"n1="<<n1<<endl<<"n2="<<n2<<endl;
                }
};
int     NUM : : n2=10;
void  main()
{
        NUM    obj1, obj2, obj3;
        obj1.input();
        obj1.calculate();
        obj2.input();
        obj2.calculate();
        obj3.input();
        obj3.calculate();
        getch();
}
```

O/P:

Enter the value for n1

5

n1=6 // Non-static data member for obj1

n2=11 // static data member for obj1

Enter the value for n1

20

n1=21 // **Non-static data member for obj2**

n2=12 // **static data member for obj2**

Enter the value for n1

30

n1=31 // **Non-static data member for obj3**

n2=13 // **static data member for obj3**

Explanation: Here we can see that each object has a separate copy of non-static data member n1 due to which every time it asks for input. On the other hand, objects share the same static data member due to which for n2 we get 11 for obj1, 12 for obj2 and 13 for obj3.

6.4.2 Static Member Function

When a member function is declared with static keyword then it is known as static member function. A static member function can be declared using following syntax:

Syntax:

```
static return_type function_name()
{

}
```


6.4.2.1 Features of static member function

- A static member function may be public or private. Most of the time it is public. When it is public, it can be invoked using its class name without using object. When it is private, the function is invoked inside a static public member function.
- A static member function can access only static members of a class.
- It is declared only once and the same copy of static function is shared by the objects of that class.

Program 28: Write a C++ program to illustrate the concept of static data member.

Code:

```
#include<iostream.h>

#include<conio.h>

class NUM
{
        int   n1;
        static int   n2;
        public:
                void   input()
                {
                        cout<<"Enter the value for n1";
                        cin>>n1;
                }
                static  void      show()
                {
                        cout<<"n2="<<n2<<endl;
                }
                void   display()
                {
                        cout<<"n1="<<n1<<endl;
                }
};
int     NUM : : n2=10;
void  main()
{
        NUM    obj1;
        obj1.input();
        obj1.display();
        NUM : : show();
        getch();
}
```

O/P:

Enter the value for n1

5

n1=5 **// Non-static data member for obj1**

n2=10 // static data member for obj1

Note: Only one data member can be made static whereas we can make any number of static member functions.

6.4.3 Static Object

When an object is made static, then all the data members for this object is initialized by a default value as per the data type . However, only one object can be made static.

Program 29: Write a C++ program to illustrate the concept of static object.

Code:

```
#include<iostream.h>
#include<conio.h>
class       NUM
{
      int     n1,n2,n3;
      public:
            void fun()
            {
                  n1=n1+5;
                  n2=n2+10;
                  cout<<"n1="<<n1<<"\t"<<"n2="<<n2;
            }
};
void main()
{
      static NUM   ob1;          // Static Object
      ob1.fun();
      getch();
}
```

O/P:

n1=5 n2=10

6.5 Array of Objects

We know that an array is a collection of similar data type elements. Now, an array can be of any data type including class type. Hence, it is possible to create an array of class type where all the elements of the array will be objects.

Program 30: Write a C++ program to illustrate the concept of array of objects

Code:

```
#include<iostream.h>
#include<conio.h>
class     STUDENT
{
                                        // Private specifier by default
          char    name[10];
          char    dept[5];
        int       roll;
        int       sem;
          float   CGPA;
public:
          void    input();
          void    display();
};
void    STUDENT: :input()
{
        cout<<"Enter the student details"
        cin>>name>>dept>>roll>>sem>>CGPA;
}
void    STUDENT: : display()
{
        cout<<"The student details are as below:"
        cout<<"Name: "<<name<<endl;
        cout<<"Dept: "<<dept<<endl;
        cout<<"Roll No: "<<roll<<endl;
        cout<<"Semester: "<<sem<<endl;
        cout<<"CGPA: "<<CGPA<<endl;
}
```

```
void  main()
      {
            STUDENT     stu[5];          // Array of objects
            for(inti=0;i<5;i++)
                  stu[i].input();
            for(i=0;i<5;i++)
                  stu[i].display();
            getch();
      }
```

O/P:

```
Enter the student details
Rahul     CSE     01     04     8.56
Sanjay    CSE     02     04     7.98
Rakesh    ETC     01     04     8.00
Rajeev    ETC     02     04     8.13
Priya     IT      01     04     7.67    // Press enter
The student details are as below:
Name: Rahul
Dept: CSE
Roll No: 01
Semester: 04
CGPA: 8.56
The student details are as below:
Name: Sanjay
Dept: CSE
Roll No: 02
Semester: 04
CGPA: 7.98
The student details are as below:
Name: Rakesh
Dept: ETC
Roll No: 01
Semester: 04
CGPA: 8.00
The student details are as below:
Name: Rajeev
Dept: ETC
Roll No: 02
Semester: 04
CGPA: 8.13
The student details are as below:
Name: Priya
Dept: IT
Roll No: 01
Semester: 04
CGPA: 7.67
```