SOS IN COMPUTER SCIENCE & APPLICATION
JIWAJI UNIVERSITY

Class : MBA (E-Commerce) II Semester

Subject : OOPS using C++

Paper Code: (201)

Topic: Friend Function In C++

Introduction To Friend Function

- In some real-time applications, sometimes we might want to access data outside the bundled unit. For example, an outsider class might want to access private and protected data of a C++ class. C++ provides a facility for accessing private and protected data by means of a special feature called “friend” function or class which we will discuss here in this tutorial.
- A friend function in C++ is a function that is preceded by the keyword “friend”. When the function is declared as a friend, then it can access the private and protected data members of the class.

Declaration of friend function in C++

```
class class_name
{
    friend data_type function_name(argument/s);
        // syntax of friend function.
};
```

In the above declaration, the friend function is preceded by the keyword friend. The function can be defined anywhere in the program like a normal C++ function. The function definition does not use either the keyword **friend or scope resolution operator**.

Characteristics of a Friend function:

- The function is not in the scope of the class to which it has been declared as a friend.
- It cannot be called using the object as it is not in the scope of that class.
- It can be invoked like a normal function without using the object.
- It cannot access the member names directly and has to use an object name and dot membership operator with the member name.
- It can be declared either in the private or the public part.
- Friend function has to be declared in all classes whose private or protected members needs to be access but all the classes will share the same definition.

Program using Friend Function

```
#include <iostream>
using namespace std;
class B;          // forward declarartion.
class A
{
    int x;
    public:
    void setdata(int i)
    {
        x=i;
    }
    friend void min(A,B);         // friend function.
};
  class B
{
    int y;
    public:
    void setdata(int i)
    {
        y=i;
    }
    friend void min(A,B);                  // friend function
};
```

Program(cont.)-

```
. void min(A a, B b)
{
    if(a.x<=b.y)
    std::cout << a.x << std::endl;
    else
    std::cout << b.y << std::endl;
   }
   int main()
  {
    A a;
    B b;
    a.setdata(10);
    b.setdata(20);
    min(a,b);
     return 0;
 }
```

Output:

10

Explanation & Friend Class

- In the above example, min() function is friendly to two classes, i.e., the min() function can access the private members of both the classes A and B.
- **C++ Friend class**
- A friend class can access both private and protected members of the class in which it has been declared as friend.
- **Let's see a simple example of a friend class.**

Program using Friend Class

```
#include <iostream>

using namespace std;

class A
{
    int x =5;
    friend class B;          // friend class.
};
class B
{
  public:
    void display(A &a)
    {
        cout<<"value of x is : "<<a.x;
    }
};
```

.

Cont.

- **int** main()
- {
- A a;
- B b;
- b.display(a);
- **return** 0;
- }
- **Output:**
- value of x is : 5

Explanation Of code

- In the example given in previous slide, class B is declared as a friend inside the class A. Therefore, B is a friend of class A. Class B can access the private members of class A.

*****