Object Oriented Programming

4 Friend Classes and Functions

-

Problematic (1/2)

How can a **function f()** and/or **a class B** access the **private member data** of **a class A**?

- ➢ **Solution:**

✓ Make class member data public.

X Disadvantage: we lose their protections.

✓ Addition of access functions to private members.

X Disadvantage: penalizing execution time.

✓ **Friend Functions**

Problematic (2/2)

It is possible to declare that one or more functions (external to the class) are "friends"; then the class authorizes them to access private data in the same way as any member function.

- ✓ Advantage: to allow access control at the level of the concerned class .

Friend Class (1/2)

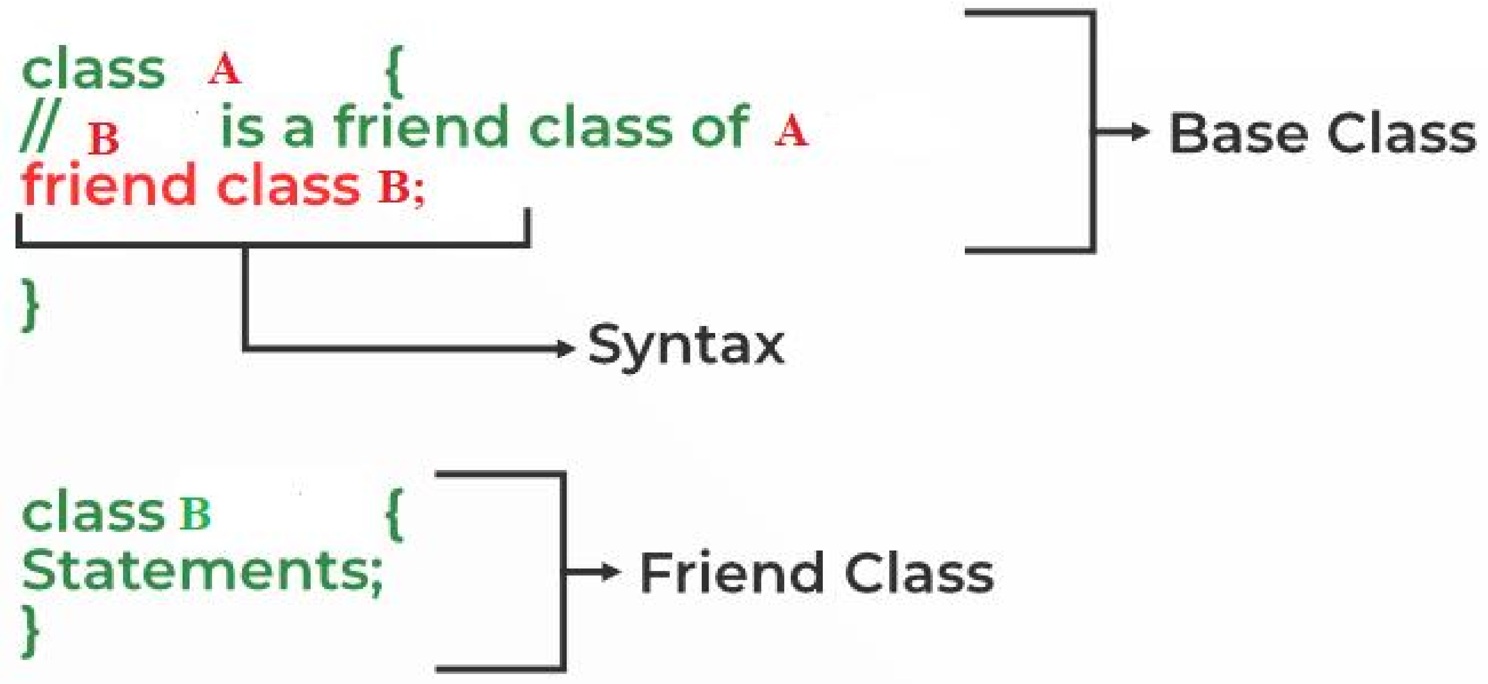
- A **friend class** can access private and protected members of other classes in which it is declared as a friend.

- It is sometimes useful to allow a particular class to access private and protected members of other classes.

- Syntax:

```
friend class class_name;    // declared in the base class
```

Friend Class (2/2)



```
class  A        {
//  B      is a friend class of  A
friend class B;

}

class B         {
Statements;
}
```

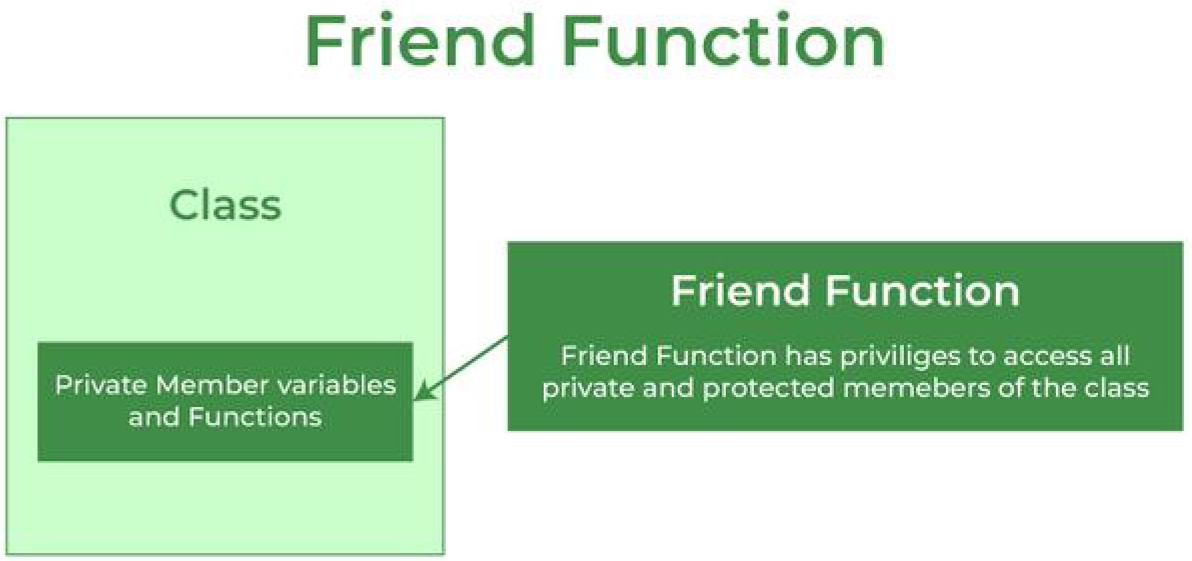
Friend Function (1/2)

- Like a friend class, a friend function can be granted special access to private and protected members of a class.

- They are not the member functions of the class but can access and manipulate the private and protected members of that class for they are declared as friends.

- A friend function can be:
    - ✓ A global function.
    - ✓ A member function of another class.

Friend Function (2/2)


Friend Function
Class
Private Member variables and Functions
Friend Function
Friend Function has priviliges to access all private and protected memebers of the class

- Syntax:

```
friend return_type function_name (arguments);     // for a global function
            or
friend return_type class_name::function_name (arguments);     // for a member function of another class
```

Friendship situations

- ✓ Global function as friend function
- ✓ Member function of another class as friend function
- ✓ A Function Friendly to Multiple Classes
- ✓ All member functions of a class as friend functions of another class (Friend classes (Go to Slide 4)).

Global Function as Friend Function

We can declare any global function as a friend function. The following example demonstrates how to declare a global function as a friend function in C++:

```
class point
{ private :
 int x,y ;
 public :
 point (int abs= 0, int ord = 0)
 { x= abs ;
 y= ord ;
 }

 friend int coincide (point p, point
q) ; // A global function as Friend
function declaration } ;
```

```
int coincide (point p, point q)
{ if ((p.x == q.x) && (p.y == q.y) )
return (1);
 else
 return (0);
}
int main()
{ point a(1,0), b (1), c ;
 if ( coincide (a,b) )
 cout<< "A coincide with B”<<endl;
 else
 cout<<”A et B are diffrent”<<endl; }
```

Member Function of Another Class as Friend Function (1/2)

We can also declare a member function of another class as a friend function in C++:

➢ *Syntax:* Consider two classes A and B:

int f(char, A) a member function of B and f must be able to access the private members of A; it will be declared a friend within class A: friend int B :: f(char, A);

Member Function of Another Class as Friend Function (2/2)

```
class A
{ private :
  ......
 public :
 .....
 friend int B :: f (char, A) ;
 ...
 ...
} ;
```

```
class B
private :
 .......
 public :
 .......
 int f (char, A) ;
 ...
 ...
} ;
int B :: f (char, A) ;
{ // here we have access to the
private members of any object of
type A
} ;
```

A Function Friendly to Multiple Classes

In C++, the same function (global or member) can be declared friends in different classes.

```
class A;
class B {
    int x;

public:
    void set_data(int a)
    {
      x = a;
    }

    friend void max(B,
A);
};
```

```
class A {
    int y;

public:
    void set_data(int
a)
    {
      y = a;
    }

    friend void max
(B, A);
};
```

```
void max(B b1, A a2)
{
    if (b1.x > a2.y)
        cout << b1.x;
    else
        cout << a2.y; }
int main()
{
    A a;
    B b;
    b.set_data(20);
    a.set_data(35);
    max(b, a);
    return 0; }
```

Advantages of Friend Functions

- ✓ A friend function is able to access members without the need of inheriting the class.
- ✓ The friend function acts as a bridge between two classes by accessing their private data.
- ✓ It can be declared either in the public or private or protected part of the class.

Disadvantages of Friend Functions

- X Friend functions have access to private members of a class from outside the class which violates the law of data hiding.
- X Friend functions cannot do any run-time polymorphism in their members.

Some Remarks Concerning Friend Classes and Functions

- Friends should be used ***only for limited purposes***. Too many functions or external classes are declared as friends of a class with protected or private data access lessens the value of encapsulation of separate classes in object-oriented programming.

- Friendship is ***not mutual***. If class A is a friend of B, then B doesn't become a friend of A automatically.

- Friendship is not inherited.