Friend Functions & Friend Classes

Friend Functions/Classes

`Friends` allow functions/classes access to private data of other classes.

Friend Functions/Classes

Friend functions

A **'friend'** function has access to all 'private' members of the class for which it is a 'friend'.

To declare a 'friend' function, include its prototype within the class, preceding it with the C++ keyword 'friend'.

Friend Functions

- Consider the following class:

```
class X{
private:
   int a, b;
public:
   void MemberFunction();
   …
}
```

Friend Functions

- Global function:

```
void DoSomething(X obj){
    obj.a = 3; //Error
    obj.b = 4; //Error
}
```

Friend Functions

►In order to access the member variables of the class, function definition must be made a friend function:

```
class X{
private:
    int a, b;
public:
        …
     friend void DoSomething(X obj);
}
```

►Now the function **`DoSomething`** can access data members of class X

Friend Functions

- Prototypes of friend functions appear in the class definition
- But friend functions are NOT member functions

Friend Functions

- Friend functions can be placed anywhere in the class without any effect
- Access specifiers don't affect friend functions or classes

```
class X{
        ...
private:
        friend void DoSomething(X);
public:
        friend void DoAnything(X);
         ...
};
```

Friend Functions

- While the definition of the friend function is:

```
void DoSomething(X obj){
   obj.a = 3;      // No Error
   obj.b = 4;      // No Error
   …
}
```

- **`friend`** keyword is not given in definition

Friend Functions

- If keyword **`friend`** is used in the function definition, it's a syntax error

```
//Error…

friend void DoSomething(X obj){
    …
}
```

Friend Classes

- Entire classes can be friends
  - Similar to function being friend to class
  - Example:
    class F is friend of class C
    - All class F member functions are friends of C
    - NOT reciprocated
    - Friendship granted, not taken
- Syntax: friend class F
  - Goes inside class definition of "authorizing" class

Friend Classes

- When a class is made a friend class, all the member functions of that class becomes friend functions.

Friend Classes

```
class B;
class A {
// class B is a friend class of class A
friend class B;
… .. …
} ;
class B {
… .. …
};
```

Friend Classes

- Similarly, one class can also be made friend of another class:

```
class X{
   friend class Y;
   …
};
```

- Member functions of class Y can access private data members of class X

Friend Classes

- Example:

```
class X{
friend class Y;
private:
     int x_var1, x_var2;
   ...
};
```

Friend Classes

```
class Y{
private:
      int y_var1, y_var2;
      X objX;
public:
      void setX(){
            objX.x_var1 = 1;
      }
};
```

Friend Classes

```
int main(){
    Y objY;
    objY.setX();
    return 0;
}
```