Friend Functions

As we have seen in the previous sections, private and protected data or function members are normally only accessible by the code which is part of same class. However, situations may arise in which it is desirable to allow the explicit access to private members of class to other functions.

If we want to declare an external function as friend of a class, thus allowing this function to have access to the private and protected members of this class, we do it by declaring a prototype of this external function within the class, and preceding it with the keyword friend. This is illustrated in the following code fragment:

```
#include <iostream>
using namespace std;

class Rectangle
{
    private :
        int length;
        int width;
    public:

        void setData(int len, int wid)
        {
            length = len;
            width = wid;
        }

        int getArea()
        {
            return length * width ;
        }

        friend double getCost(Rectangle); //friend of class Rectangle
};

//friend function getCost can access private member of class
double getCost (Rectangle rect)
{
    double cost;
    cost = rect.length * rect.width * 5;
    return cost;
}
```

```
int main ()
{
    Rectangle floor;
    floor.setData(20,3);
    cout << "Expense " << getCost(floor) << endl;
    return 0;
}
```

Output :
Expense 300

The getCost function is a friend of Rectangle. From within that function we have been able to access the members length and width, which are private members.

Friend Classes

One class member function can access the private and protected members of other class. We do it by declaring a class as friend of other class. This is illustrated in the following code fragment:

```
#include <iostream>
using namespace std;

class CostCalculator;

class Rectangle
{
    private :
        int length;
        int width;
    public:
        void setData(int len, int wid)
        {
            length = len;
            width = wid;
        }

        int getArea()
        {
            return length * width ;
        }

        friend class CostCalculator; //friend of class Rectangle
};
```

```
//friend class costCalculator can access private member of class
Rectangle

class CostCalculator
{
    public :
        double getCost (Rectangle rect)
        {
            double cost;
            cost = rect.length * rect.width * 5;
            return cost;
        }
};

int main ()
{
    Rectangle floor;
    floor.setData(20,3);
    CostCalculator calc;
    cout << "Expense " << calc.getCost(floor) << endl;
    return 0;
}
```

Output :
Expense 300

Note : An empty declaration of class costCalculator at top is necessary.