Friend class and function in C++

Friend Class A friend class can access private and protected members of other class in which it is declared as friend. It is sometimes useful to allow a particular class to access private members of other class. For example a LinkedList class may be allowed to access private members of Node.

```
class Node
{
private:
  int key;
  Node *next;
  /* Other members of Node Class */

  friend class LinkedList; // Now class  LinkedList can
                           // access private members of Node
};
```

Friend Function Like friend class, a friend function can be given special grant to access private and protected members. A friend function can be:

a) A method of another class

b) A global function

```
class Node
{
private:
  int key;
  Node *next;

  /* Other members of Node Class */
  friend int LinkedList::search(); // Only search() of linkedList
                                   // can access internal members
};
```

Following are some important points about friend functions and classes:

1) Friends should be used only for limited purpose. too many functions or external classes are declared as friends of a class with protected or private data, it lessens the value of encapsulation of separate classes in object-oriented programming.

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

3) Friendship is not inherited (See this for more details)

4) The concept of friends is not there in Java.

A simple and complete C++ program to demonstrate friend Class

```
#include <iostream>
class A {
private:
    int a;
public:
    A() { a=0; }
    friend class B;     // Friend Class
};

class B {
private:
    int b;
public:
    void showA(A& x) {
        // Since B is friend of A, it can access
        // private members of A
        std::cout << "A::a=" << x.a;
    }
};

int main() {
   A a;
   B b;
   b.showA(a);
   return 0;
}
```

Output:

```
A::a=0
```

A simple and complete C++ program to demonstrate friend function of another class

```
#include <iostream>

class B;

class A
{
public:
    void showB(B& );
};

class B
{
private:
    int b;
public:
    B()  {  b = 0; }
    friend void A::showB(B& x); // Friend function
};

void A::showB(B &x)
{
    // Since show() is friend of B, it can
    // access private members of B
    std::cout << "B::b = " << x.b;
}

int main()
{
    A a;
    B x;
    a.showB(x);
    return 0;
}
```

Output:

```
B::b = 0
```

A simple and complete C++ program to demonstrate global friend

```
#include <iostream>

class A
{
    int a;
public:
    A() {a = 0;}
    friend void showA(A&); // global friend function
};

void showA(A& x) {
    // Since showA() is a friend, it can access
    // private members of A
    std::cout << "A::a=" << x.a;
}

int main()
{
    A a;
    showA(a);
    return 0;
}
```

Output:

```
A::a = 0
```

References:

http://en.wikipedia.org/wiki/Friend_class

http://en.wikipedia.org/wiki/Friend_function

http://www.cprogramming.com/tutorial/friends.html

http://www.parashift.com/c++-faq/friends-and-encap.html

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.

Recommended Posts:

Can we access private data members of a class without using a member or a friend function?

C++ Program to swap two members using Friend Function

C++ interview questions on virtual function and abstract class

How to convert a class to another class type in C++?

std::any Class in C++

std:: valarray class in C++

std::string class in C++

Array class in C++

Structure vs class in C++

Difference between namespace and class

Virtual base class in C++

String class in Java | Set 1

C++ String Class and its Applications | Set 2

C++ string class and its applications

Scanner Class in Java

Article Tags : C++ School Programming cpp-friend

Practice Tags : CPP

8

To-do Done

3

Based on **54** vote(s)

Feedback/ Suggest Improvement Add Notes Improve Article

Please write to us at contribute@geeksforgeeks.org to report any issue with the above content.

Previous

When are static objects destroyed?

Next

Templates in C++

Most popular in C++

- Difference between C and C++
- Dividing a Large file into Separate Modules in C/C++, Java and Python
- Memory leak in C++ and How to avoid it?
- GDB (Step by Step Introduction)
- Types of Operator Overloading in C++

Most visited in School Programming

- Structuring Python Programs
- Python program to add two numbers
- How to assign values to variables in Python and other languages

Python program to check whether a number is Prime or not

Logic Gates in Python