在成员函数中使用 STL 的 find_if 函数

STL 的 find_if 函数功能很强大，可以使用输入的函数替代等于操作符执行查找功能。

比如查找一个数组中的奇数，可以用如下代码完成（具体参考这里：http://www.cplusplus.com/reference/algorithm/find_if/）：

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

bool IsOdd (int i) {
  return ((i%2)==1);
}

int main () {
  vector<int> myvector;
  vector<int>::iterator it;
  myvector.push_back(10);
  myvector.push_back(25);
  myvector.push_back(40);
  myvector.push_back(55);
  it = find_if (myvector.begin(), myvector.end(), IsOdd);
  cout << "The first odd value is " << *it << endl;
  return 0;
}
```

运行结果：

```
The first odd value is 25
```

如果把上述代码加入到类里面，写成类的成员函数，又是什么效果呢？

比如如下类代码：

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

class CTest
{
public:
	bool IsOdd (int i) {
		return ((i%2)==1);
	}

	int test () {
		vector<int> myvector;
		vector<int>::iterator it;
		myvector.push_back(10);
		myvector.push_back(25);
		myvector.push_back(40);
		myvector.push_back(55);
```

```
                it = find_if (myvector.begin(), myvector.end(), IsOdd);
                cout << "The first odd value is " << *it << endl;
                return 0;
        }
};
int main()
{
        CTest t1;
        t1.test();
        return 0;
}
```

会出现类似下面的错误：

```
 error C3867: 'CTest::IsOdd': function call missing argument list; use '&CTest::IsOdd' to create a pointer
to member
```

解决办法：

```
it = find_if (myvector.begin(), myvector.end(), IsOdd);
```

改为：

```
it = find_if(myvector.begin(), myvector.end(),std::bind1st(std::mem_fun(&CTest::IsOdd),this));
```

代码如下：

```
#include <iostream>
#include <functional>
#include <algorithm>
#include <vector>
using namespace std;

class CTest
{
public:
        bool  IsOdd ( int i)
        {
                return ((i%2)==1);
        }
        void close()
        {
                vector<int> myvector;
                vector<int>::iterator it;

                myvector.push_back(10);
                myvector.push_back(25);
                myvector.push_back(40);
                myvector.push_back(55);

                it = find_if(myvector.begin(),
myvector.end(),std::bind1st(std::mem_fun(&CTest::IsOdd),this));
                cout << "The first odd value is " << *it << endl;

                cout<<"All odd value :"<<endl;
                for(;it != myvector.end();++it)
                {
```

```
                        cout<<*it<<endl;
                }
        }
};

int main () {
        CTest st1;
        st1.close();
        return 0;
}
```