CSDN 学院

C++语言基础

迁者 - 贺利坚

http://blog.csdn.net/sxhelijian/

http://edu.csdn.net

本节主题：

范型编程简介

曾经的查找

```
int main( )
{
   int d[10]={2,7,4,8,12,1,3,5,9,11},i,key,index=-1;
   cout<<"Input a key you want to search:\n";
   cin>>key;
   for(i=0; i<10; i++)
      if(key == d[i])
      {
         index = i;
         break;
      }
   if(index >= 0)   cout<<"It is in "<<index<<"\n";
   else   cout<<"Not found.\n";
   return 0;
}
```

```
int main()
{
   int d[10] = {1,3,9,12,32,41,45,62,75,77};
   int low, high,mid,key,index=-1;
   cin>>key;
   low=0,high=9;
   while(low<=high)
   {
      mid=(low+high)/2;
      if(d[mid]==key)
      {
         index=mid;
         break;
      }
      else if(d[mid]>key)
         high=mid-1;
      else
         low=mid+1;
   }
   if(index >= 0)
      cout<<"在："<<index<<".\n";
   else
      cout<<"没有找到.\n";
}
```

可以这样做

```
#include <iostream>
#include <algorithm>
using namespace std;
#define SIZE 100
int iarray[SIZE];
int main()
{
   iarray[20] = 50;
   int* ip = find(iarray, iarray + SIZE, 50);
   if (ip == iarray + SIZE)
      cout << "50 not found in array" << endl;
   else
      cout << *ip << " found in array" << endl;
   return 0;
}
```

```
#include <iostream>
#include <algorithm>
#include <vector>
using namespace std;
vector<int> intVector(100);
int main()
{
   intVector[20] = 50;
   vector<int>::iterator intIter =
      find(intVector.begin(), intVector.end(), 50);
   if (intIter != intVector.end())
      cout << "Vector contains value " << *intIter << endl;
   else
      cout << "Vector does not contain 50" << endl;
   return 0;
}
```

范型编程，编写不依赖数据类型的程序

```
find(iarray, iarray + SIZE, 50);
find(intVector.begin(), intVector.end(), 50);
```

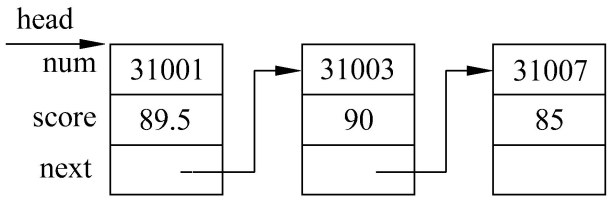
- 泛型编程（generic programming，GP），一种全新的程序设计思想
- 一般讲，任何算法都是作用在一种特定的数据结构上的，在设计一种算法的时候，总是先要考虑其应用的数据结构。
- 泛型设计的根本思想就是想把算法和其作用的数据结构分离
- 泛型设计的理想状态是算法是通用的、泛型的，可以作用于数组，联表，树，图等各种数据结构之上。
- GP抽象度更高
- 基于GP设计的组件之间偶合度底，没有继承关系，组件间的交互性和扩展性都非常高。
- 泛型技术的实现方法：模板，多态等。

- 高级语言：抽象数据类型
- 模块化程序设计：数据参数化
- 面向对象程序设计：数据及其操作的封装
- 泛型编程：数据类型的参数化

用模板实现的泛型

```
class Student  //结点类
{
public:
   Student(int n,double s);
   ~Student();
   Student *next;  //指向下一个结点
   int num;
   double score;
};
class MyList  //链表类
{
public:
   MyList();
   ~MyList();
private:
   Student *head;
};
```

```
template <class T>
class Node
{
public:
   Node *next;
   T data;
};
class MyList  //链表类
{
public:
   MyList();
   ~MyList();
private:
   Node *head;
};
```



CSDN 学院

THANKS

本课程由 迁者-贺利坚 提供

CSDN网站：www.csdn.net
企业服务：http://ems.csdn.net/
人才服务：http://job.csdn.net/
CTO俱乐部：http://cto.csdn.net/
高校俱乐部：http://student.csdn.net/
程序员杂志：http://programmer.csdn.net/

CODE平台：https://code.csdn.net/
项目外包：http://www.csto.com/
CSDN博客：http://blog.csdn.net/
CSDN论坛：http://bbs.csdn.net/
CSDN下载：http://download.csdn.net/