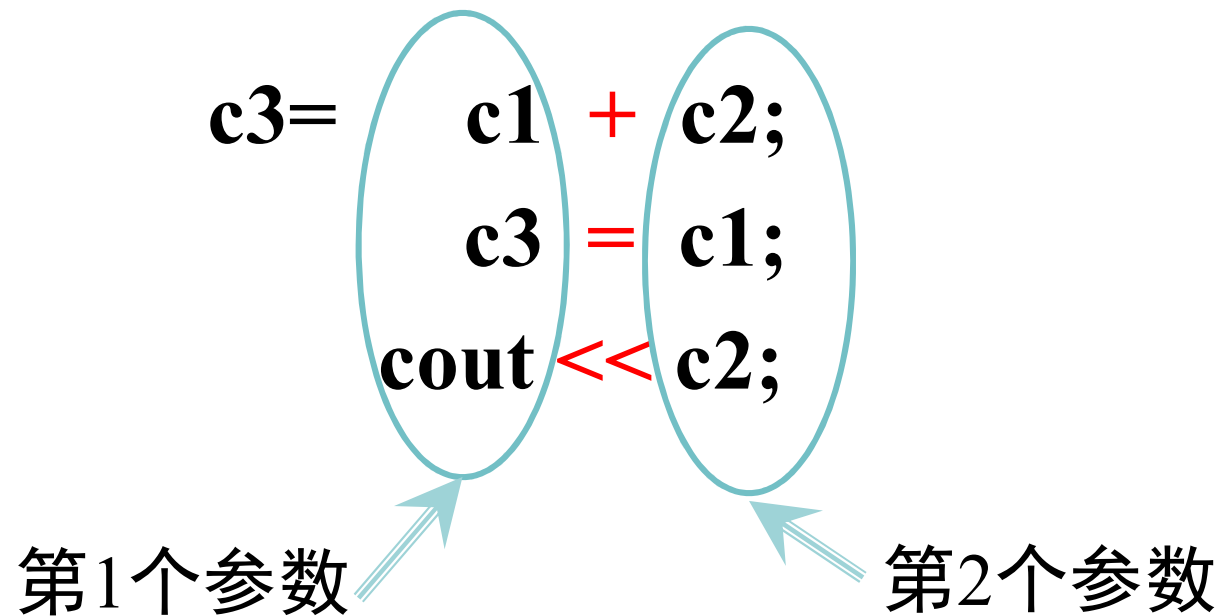


## 实践3 小结

- 比较三个**二元运算符**的重载：



# 实践4：函数模板

- 要求
  - 1.掌握函数模板的定义和使用方法。
- 提交程序
  - 1.习题6.3，实验15，二-1，建立求数组元素最小值和最大值的函数模板。(选做)
  - 2.习题6.4，建立对数组进行顺序，对半查找算法的函数模板。
  - 3.实验16，二-3，从指定位置开始的冒泡排序算法等。
- 注意，随测试完善MyString类。
  - 后续实验中为学期内调试简便，可将MyString的定义和声明放在一个"学号\_MyString.h"文件中，文件拷贝到工程所在目录中，在后续实验中包含#include "学号\_MyString.h"。
  - 没掌握多文件结构的同学，就每次拷贝MySting的代码到一个文件中使用。

# 实践4 小结

- REVIEW:

- char型变量，0不代表任何字符，故用0来作为C风格字符串的终止符。其他类型变量，0是有意义的值。实验四使用set类时已有体验，本次依然有同学使用0作为整型等数组的判断标志。
- 向函数传递数组名时，只传入了一个指针常量，需要另设参数，指定数组长度。

## 实践4 小结

- 循环结构的幽灵——“**差一**”误差。
- 更多的**连接错误**
  - 只要调用前有声明，且调用格式正确，编译就正确。
  - 有声明，有定义，但二者参数列表不一致。
  - 有声明，无定义：没有写定义函数。
- 后续实验中，**仅为学期内调试简便**，myString的**定义和声明**放在一个.h文件中。
  - **存放在工程目录**下，用#include “**学号\_MyString.h**”包含。

## 实践4 小结

- 多文件工程(Project)



初级版：MyString类的声明和定义都在MyString.h中完成。



规范版：MyString类的声明在MyString.h中，定义在MyString.cpp中完成。

后续实验中，仅为学期内调试简便，myString的定义和声明放在一个.h文件中，即初级版方式。头文件存放在工程目录下。

## 实践4 小结

可以发现`#include <iostream>`包含多少次都没有关系，自定义头文件也可以防止重复包含。

多文件的项目，多个cpp文件甚至h文件中，可能包含同一个自定义的h文件，在**每个自定义.h文件**中用**条件编译指令**防止重复定义。

```
#ifndef _MY_STRING_H_
#define _MY_STRING_H_

... 你的声明性质代码 ...

#endif _MY_STRING_H_
```

根据每个文件的功能，  
理顺多文件项目当中各文件的逻辑关系。

## 实践4 小结

- 多个功能模块联调：
  - 每个模块**单独都正确**，放到一起完全无法工作
  - 一致性、完备性
  - “接口”严格按双方协定完成

多功能模块协作时的“接口”

```
int dicing(){return 2;}
```

本次实验测试过程中两种常见错误：

```
char Arr[6]="abcde";      int Arr[6]={3,2,1};  
InsertSort(Arr,6);        InsertSort(Arr,6);  
cout<<Arr<<endl;
```

## 实践4 小结

- 函数完成的功能由函数名一目了然。
- 除了预期的函数功能外，函数不做其他操作。
- 编译时间加长：模板实例化。
- 错误排查：
  - 语法的，编译链接错误，迅速可修正。
  - 逻辑的，计算机世界的好处是只是0和1，坚定信念：只要有错就一定可以找出来，并且一定可以解决的。



进一步理解了字符数组  
与string类对象的差异

```
int a[10]={1, 4, 24, 6, 3, 74, 0};
char b[10]={'a','c','d','e','f','g','a'};
double c[10]={0.3, 1, 5, 18, 1.2, 1.5, 0.1};
string d("qwertyu");
cout<<maxOfArray(a, 7)<<endl;
cout<<maxOfArray(b, 7)<<endl;
cout<<maxOfArray(c, 7)<<endl;
cout<<maxOfArray(d, 7)<<endl;
cout<<minOfArray(a, 7)<<endl;
cout<<minOfArray(b, 7)<<endl;
cout<<minOfArray(c, 7)<<endl;
cout<<minOfArray(d, 7)<<endl;
```

定义模板时的注意点：  
参数化类型的替换不多  
也不少！

```
T maxOfArray(const T *pA, int size)
```

```
{ int i=0;
```

```
{ int i=0;
```

```
int max=*pA;
```

```
do
```

```
{
```

```
if((*pA+1)>max)
```

```
{max=(*pA+1);}
```

```
i++;
```

```
}while(i<size);
```

```
return (T)max;
```

```
}
```

指针使用——永远的重  
点与难点！

- 对MyString类的完善，本次实验常见错误：

- <<重载
- 复制构造函数

```
template <typename T>
```

```
int BinarySearch(const T *pA, T x, int low, int high);
```

传入地址常量

调用复制构造函数


学习断点调试的重要性！

待检验：>>，=，+= 等改变本对象的运算符的重载

另：二分法搜索陷入死循环的测试自行体验。

```
string x="csfw";  
MyString ms="dfb";
```

不推荐的初始化方式！  
匿名调用一次构造函数  
再调用一次重载的=




```
#include"实践课作业\下学期  
\Exp03\Exp03_D2120119_02_MyString.h"
```

路径一改动.....



```
string sa[5]={"上海","北京","沈阳","广州","武汉"};  
string s1=maxOfArray(sa,5);  
cout<<"字典排序最大为: "<<s1<<endl;
```

汉字存储的是机内码



```
string s2=BinarySearch(sa,"上海",0,4);  
cout<<"对半查找结果为: "<<s2<<endl;
```

二分法只可查找已排序数组



*测试程序的改进:*

*for (int i = 0; i < 20; i++) in[i] = rand() % 100;*

*.....*

*@byD2120114, @byD2120215, 25*

*@byD2120216*

```

template<typename T>
void present(const T* pA, T x, int low, int high, string p) {
    cout << p << "型的数组数据为： ";
    for (int i = 0; i <= high; i++) { cout << pA[i] << " "; }
    cout << endl << "查找的数据为： " << x << endl;
    cout << "BinarySearch所查找的下标结果为： " <<
BinarySearch(pA, x, low, high) << endl;
    cout << "SequenceSearch所查找的下标结果为： " <<
SequenceSearch(pA, x, low, high) << endl << endl;
}
    present(mstr, Tofind2, 0, 3, "MyString");

```

但是BinarySearch函数有明显错误，是否有运行测试。

类似的测试  
模板的改进：

```

SortTest(ia, 20, "Int");
SortTest(da, 20, "Double"); @byD2120114

```

```

show(ch, 10, "char");
show(in, 10, "int");

```

```

SortTest(in, 20, "Int");
SortTest(dou, 20, "Double"); byD2120215

```

# 实践5：类模板

- 要求
  - 1.掌握类模板的定义和使用方法。
- 提交程序
  - 完善seqlist类模板：线性表的顺序存储结构的定义已给，要求添加：重载[]，查找，排序等成员函数，具体要求见cpp文件。
  - 为了充分练习对数组的控制，不允许使用动态内存申请。
  - 为了体会各种排序算法的思想，除了对比验证测试结果，在自定义的排序函数体中不允许调用系统自带sort()函数。

后续实验中，为学期内调试简便，myString的定义和声明放在一个“学号\_myString.h”头文件中，存放在工程目录下，即初级版方式。用#include “学号\_myString.h”包含进工程。