实践3小结

- 文件流类成员函数open() 与 close()成对使用。
  - 本地文件显示文件扩展名问题！
- ofstream类，输出到文件，基本已熟练掌握
  - 注意保持输出数据之间的可分辨性。
- ifstream类，从文件输入（统一用cin代指输入流类对象）
  - 输入流的操作不安全，稳健性差，而用户输入出错又是不可避免的，对程序员要求高。
  - 出错后，状态字非0，需要人为清零，cin.clear(0)，将流恢复正常，或者需要清空缓冲区。例如：读到与提取数据类型不一致的输入数据，读到文件尾……

例. 用友元函数重载复数的<<, >>运算符后。

```
int main( ) {
    Complex c1,c2;          cin>>c1>>c2;
    cout<<“c1=“<<c1<<endl;
    cout<<“c2=“<<c2<<endl;

    ofstream of(“complex.txt”);
    of<<c1<<’\t’<<c2<<endl;  ……
    return 0;
}
```

重载的<<、>>可以直接被文件流类对象使用，原因后续讲解…

输出set类存储的数据到文件，使用重载的流插入运算符<<是最简便的方式。

重载的流插入运算符<<较简单，将print函数略加改造

```
ostream &operator<<(ostream &os, const set &si)
{
   os<<"集合的元素包括："<<endl;
   for(int i=0; i<si.num; i++)
   {
      os<<si.elements[i]<<"  ";
      if(0==(i+1)%5) os<<endl;
   }
   os <<endl;
   return os;
}
```

写数据到文件也较简单随意，只需要注意数据可分辨

```
ofile<<23<<'\t'<<17<<'\t'<<56<<'\t'……;
ofile<<“23    17    56    ……”;
```

从文件读数据则需要根据使用数据的需求

```
for (int i=0; ;i++){
    ifile>>arr[i];          // arr的类型决定取到什么
    if(ifile.eof()) break;
}
```

从文件输入，本质上跟从键盘输入是一样的。

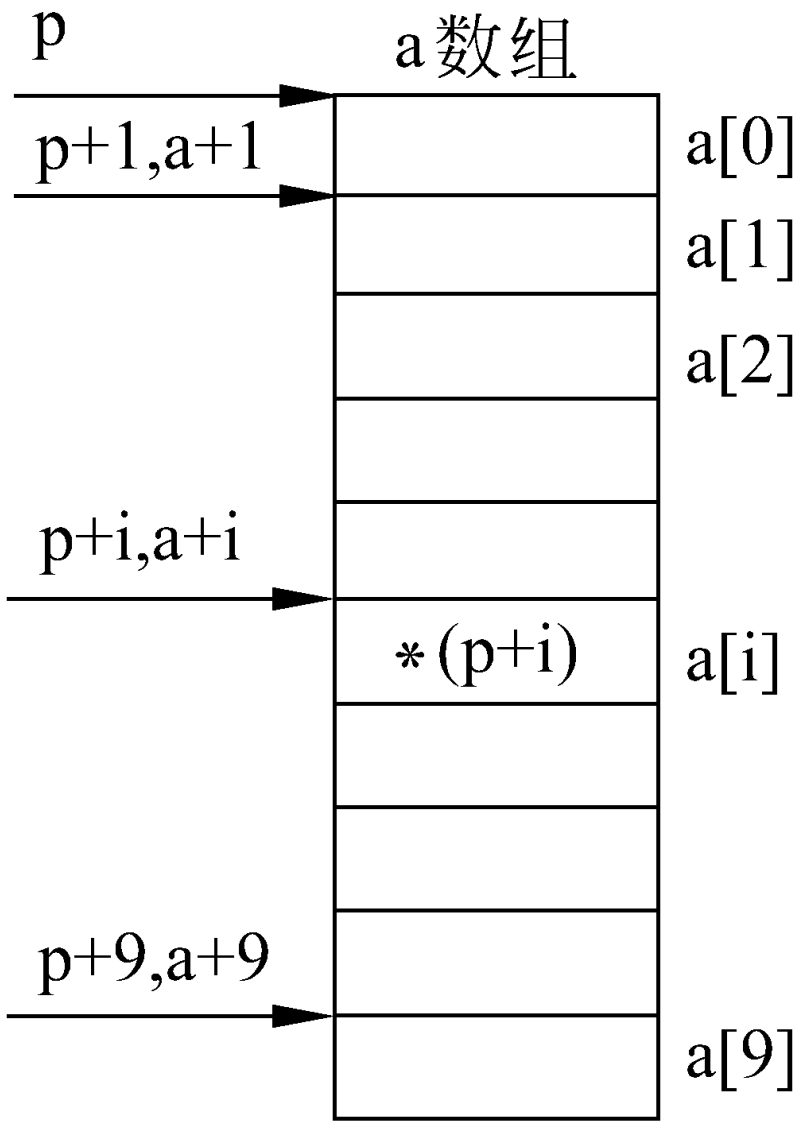
复习 1.8, 2.8

实践3 小结

- 程序稳健性（Robustness，鲁棒性）初探：
  - 上帝视角的程序员 vs.任性的用户——一定是你的打开方式不对！☺☺☺
- MyString类：
  - C的字符数组使用不方便，容易出错，且需要时刻检查是否越界——面向对象的方式使用数据！
    - 操作函数包含在<cstring>头文件中。
  - C++封装了完备的string类并包含在标准库中，越界检查等控制封装在类内，使用者脱离此苦。
    - 提供了<string>头文件供程序员使用。
  - 自定义字符串类(5.6.3节)成为经典面向对象编程练习题。

实践3 小结

- 综合训练，熟练掌握类的声明，定义，使用。
  - 数据(属性)，变量成员的访问和设置。
  - 操作(行为)，函数成员的声明，定义，调用。
  - 运算符重载，直观运算，对程序员友好。
- 引用&，指针*，数组[]的相爱相杀！永远的重难点！
- 防止循环结构“差一”误差。



“差一”误差不可怕，检测到了，找出来改掉就完了。

实践3 小结

- 重申：const引用的一致性需求。
- << 运算符重载，不仅仅是替代了display(), show()函数。
- >>运算符重载，不仅仅替代了input(), set()函数。

```
istream & operator >> (istream &,自定义类 &);
ostream & operator << (ostream &, const自定义类 &);
```

复制构造函数

```
MyString::MyString(const MyString & ms){
    str[n]=ms.str[n];maxsize=ms.maxsize;last=ms.last;
}
```

```
MyString:: MyString(const MyString & ms){
    strcpy(str, ms.str);    //需包含<cstring>
    maxsize=ms.maxsize;
    last=ms.last;
}
```

```
MyString::MyString(const MyString & ms) {
   maxsize=ms.maxsize;
   last=-1;
   do{  last++;   str[last]=ms.str[last];
   }while(ms.str[last] !='\0' && last<maxsize-1);
   str[last] ='\0';
   if(last !=ms.last) cout<<"Copy construction ERROR\n";
}
```

strcat等一系列函数既然是要求自定义成类成员函数，完成作业自然是优先使用自定义函数，而尽量避免调用库函数

重载[]

```
char&MyString::operator[](int m)
{       return str[m];  }

char& MyString::operator[](int i){
        if(i>n) cout<<"error index"<<endl;
        return str[i];
}
```

```
//使用动态内存申请前的一点折衷
char & MyString::operator[](int i){
   if(i>=maxsize-1)
   {  cout<<"[] overflow ERROR\n";
      exit(0);     }
   if(i>=last)     //处理可不同
   {   i=last++;      str[last] ='\0';    }
   return str[i];
}
```

```
//最多存20个字符时
MyString ms0("See");
MyString ms1(ms0); //测试复制构造
cout<<ms0<<'\t'<<ms0.getLast()<<endl;
cout<<ms1<<'\t'<<ms0.getLast()<<endl;
ms1[3]='S';       //测试[] 和 <<
cout<<ms1<<'\t'<<ms1.getLast()<<endl;
ms1[19]='S';
cout<<ms1<<'\t'<<ms1.getLast()<<endl;
ms1[20]='A';
cout<<ms1<<'\t'<<ms1.getLast()<<endl;
```

重载<<和>>

```
ostream &operator<< (ostream &os, const MyString &ms)
{    os<<ms.str;   return os;  }
istream &operator >> (istream &is, MyString &ms)
{    is>>ms.str;    return is; }
```

空格无法读入

有的同学写了循环语句，用get函数来读取，也是可以的，
但当我说只要用getline函数读取时，有同学就有异议了，还要重载>>干嘛，还要定义MyString类干嘛？

```
//测试>>
cout<<"请输入不超过20个字符\n";
cin>>ms0;
cout<<ms0<<ms0.getLast()<<endl;
```

第二版
```
{  is.getline(ms.str, ms.maxsize);  return is; }
```

第三版
```
{  is.getline(ms.str, ms.maxsize);
   ms.last=strlen(ms.str);                return is; }
```

习题5.12， 将习题 5.8 中的字符串处理函数移植到 MyString 类中，请将它们转为成员函数。注意对比成员函数与独立函数在参数传递上有何不同？

字符串函数包括：

（1）char *strcat (char *s, const char *ct); 将串 ct 接到串 s 的后面，形成一个长串。

（2）int strlen(const char * s); 求字符串长度的函数，返回串长（不包括串结束符）。

（3）char * reverse (char *); 反置字符串 s，即可将“break”成为“kaerb”。

（4）char * strchr(const char *cs,char c); 查找字符 c 在串 cs 中第一次出现的位置，返回指向该字符的指针，若没有出现则返回 NULL。

（5）char *strstr (const char *cs1,const char *cs2); 返回串 cs2 作为子串在 cs1 中第一次出现的位置，若没有出现则返回 NULL。

+和关系运算符，我给出的声明是成员函数，是否与此前定义Complex类时讲的矛盾了呢。

进一步体会理解面向对象思想

向一个C风格的字符数组中写入，或者访问字符数组时，时时需要考虑是否越界。
而应用面向对象思想定义了一个MyString类，将对数组的越界检查等控制全部封装在类内，使得MyString类的使用者不需要分心考虑这些问题。

自己定义的MyString类自行设计函数功能，尽量参考string类的成员函数的功能与声明。

```
1  #include<iostream>
2  #include<cstring>
3  const double pi=3.14;
4  using namespace std;
5  class point  {
6  public:
7      point(int=0,int=0);
8  private:
9      double m_x;
10     double m_y;
11 };
12 class material {
13     string name;
```

Compile Log | Debug | Find Results | Close

```
Compilation results...
--------
- Errors: 0
- Warnings: 0
- Output Filename: C:\temp\D1119702_2.exe
- Output Size: 1.30591678619385 MiB
```

“智能”的编译器带来的一些问题

不让程序的解释权过多依赖编译器

实践5：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，在后续实验中#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。