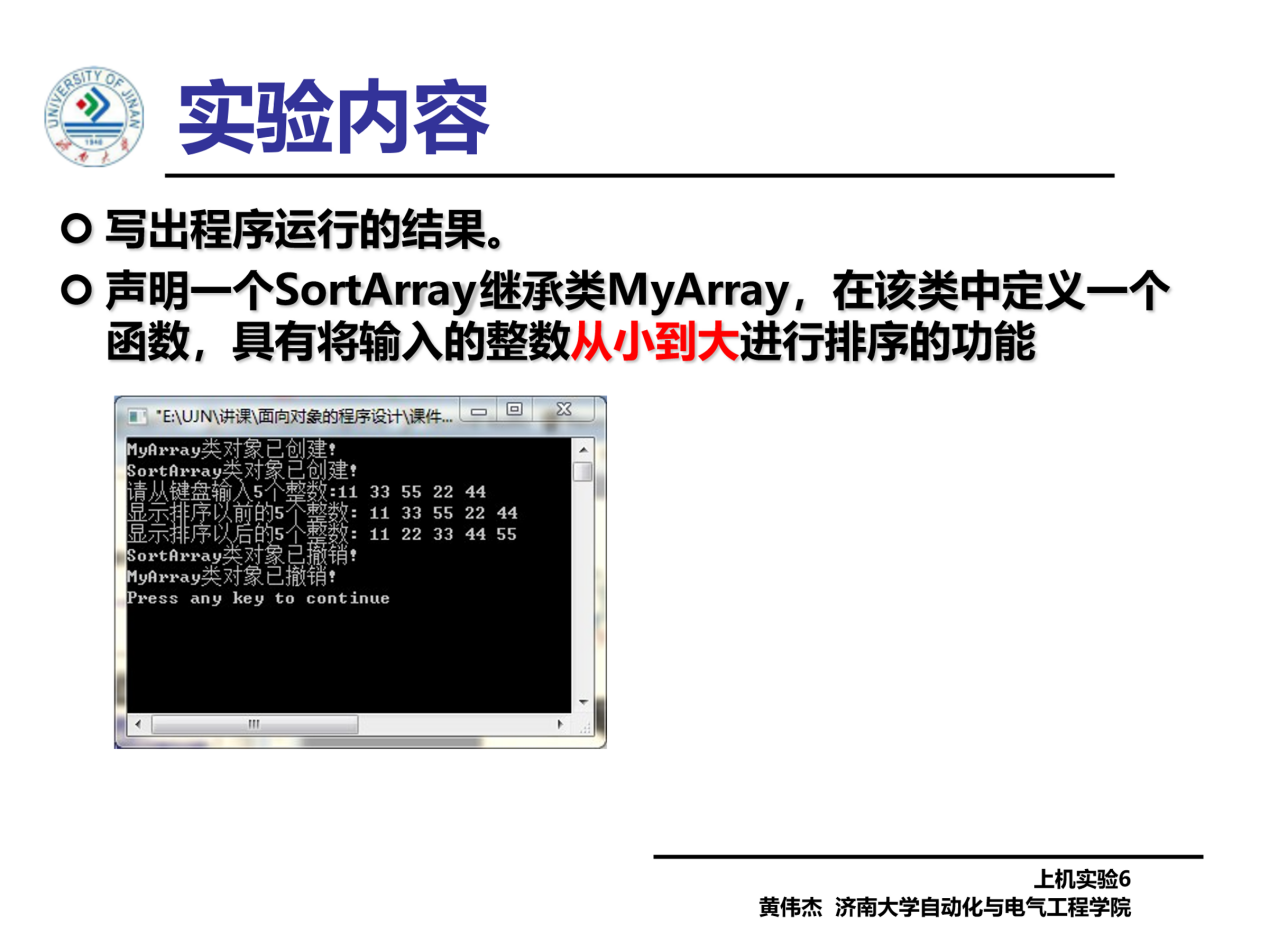
Copyright ©2021-2099wuxinmiao All rights reserved

**实验要求：**



**程序：**

**#include<iostream>**

**#include<string>**

**using namespace std;**

**class MyArray {**

**public:**

**MyArray(int length);**

**~MyArray();**

**void Input();**

**void Display(string);**

**protected:**

**int \*alist;**

**int length;**

**};**

**MyArray::MyArray(int leng)**

**{**

**if (leng <= 0)**

**{**

**cout << "error length";**

**exit(1);**

**}**

**length = leng;**

**alist = new int [length];**

**if (alist == NULL)**

**{**

**cout << "assign failure";**

**exit(1);**

**}**

**cout << "MyArray类对象已创建!" << endl;**

**}**

**MyArray::~MyArray()**

**{**

**delete []alist;**

**cout << "MyArray类对象已撤销!" << endl;**

**}**

**void MyArray::Display(string str)**

**{**

**int i;**

**int \*p = alist;**

**cout << str << length << "个整数: ";**

**for (i = 0; i < length; i++, p++)**

**cout << \*p << " ";**

**cout << endl;**

**}**

**void MyArray::Input()**

**{**

**cout << "请从键盘输入" << length << "个整数:";**

**int i;**

**int \*p = alist;**

**for (i = 0; i < length; i++, p++)**

**cin >> \*p;**

**}**

**class SortArray :public MyArray {**

**public:**

**void px();**

**SortArray(int leng);**

**~SortArray();**

**};**

**SortArray::SortArray(int leng):MyArray(leng)**

**{**

**cout << "SortArray类对象已创建" << endl;**

**}**

**SortArray::~SortArray()**

**{**

**cout << "SortArray类对象已撤销" << endl;**

**}**

**void SortArray::px()**

**{**

**int \*p = alist;**

**for (int i = 0; i < length - 1; i++)**

**for (int j = i + 1; j < length; j++)**

**if (alist[i] < alist[j])**

**{**

**int t = p[i];**

**p[i] = p[j];**

**p[j] = t;**

**}**

**}**

**int main()**

**{**

**SortArray a(5);**

**a.Input();**

**a.Display("显示排序以前的");**

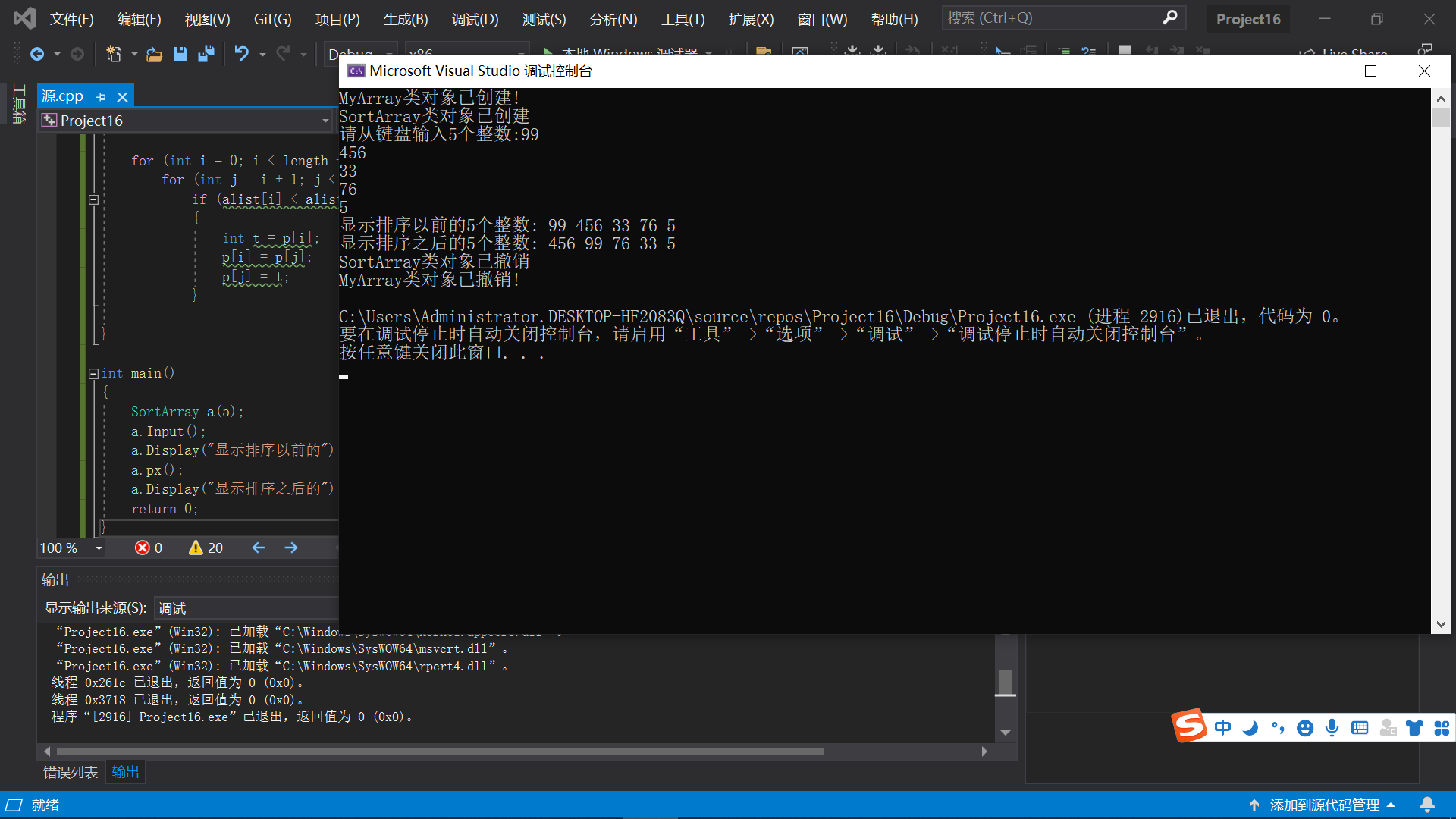
**a.px();**

**a.Display("显示排序之后的");**

**return 0;**

**}**

**实验程序输出结果：**



**实验总结：**

依旧是编译出现多次错误，经查询资料后得到错误原因：Vs控制台程序的入口不同，控制台为main,windows为winmain。

多文件编程时需得在cpp文件的同名头文件中声明函数，且需得与cpp文件中的函数一致。

使用别的文件的函数时需先包含相应头文件，再用。

**认识到了继承的好处：**

（1）代码重用，减少创建类的成本，每个子类都拥有父类的方法和属性；

（2）子类与父类基本相似，但又与父类有所区别；

（3）提高代码的可扩展性。