第六次上机实验报告

**一、实验目的**

掌握派生类的声明方法和派生类构造函数的定义方法

掌握不同方式下，构造函数与析构函数的执行顺序与构造规则 **二、实验内容**

写出程序运行的结果。

声明一个SortArray继承类MyArray，在该类中定义一个函数，具有将输入的整数从小到大进行排序的功能

**三、程序代码及运行结果**

#include<iostream>

#include<string>

using namespace std;

class MyArray {

public:

MyArray(int length);

~MyArray();

void Input();

void Display(string);

protected:

int\* a;

int length;

};

MyArray::MyArray(int leng)

{

if (leng <= 0)

{

cout << "error length";

exit(1);

}

length = leng;

a = new int[length];

if (a == NULL)

{

cout << "assign failure";

exit(1);

}

cout << "MyArray类对象已创建!" << endl;

}

MyArray::~MyArray()

{

delete[] a;

cout << "MyArray类对象已撤销!" << endl;

}

void MyArray::Display(string str)

{

int i;

int\* p = a;

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 = a;

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

cin >> \*p;

}

class SortArray : public MyArray {

public:

void Sort();

SortArray(int leng) :MyArray(leng)

{

cout << "SortArray类对象已创建!" << endl;

}

~SortArray();

};

SortArray::~SortArray()

{

cout << "SortArray类对象已撤销!" << endl;

}

void SortArray::Sort()

{

int i, j, temp;

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

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

{

if (a[j] > a[j + 1])

{

temp = a[j];

a[j] = a[j + 1];

a[j + 1] = temp;

}

}

}

int main()

{

SortArray s(5);

s.Input();

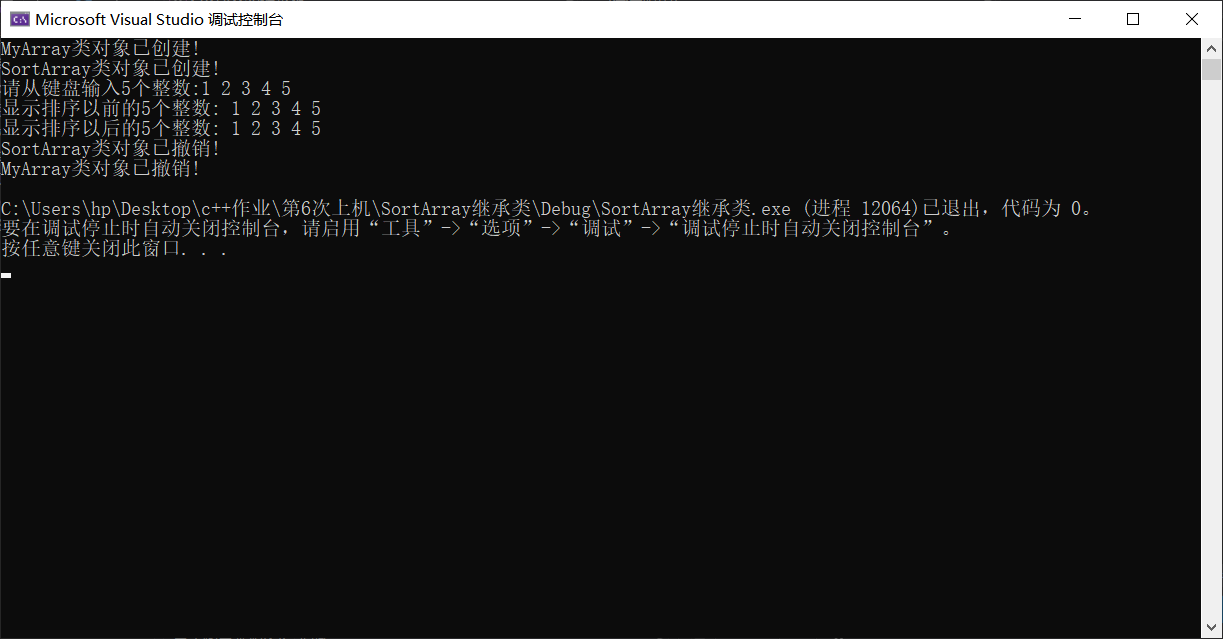
s.Display("显示排序以前的");

s.Sort();

s.Display("显示排序以后的");

return 0;

}



**四、感想心得**

基类中没有定义构造函数，或者定义了没有参数的构造函数：  
在派生类构造函数中可不写调用基类构造函数的语句，调用派生类构造函数时系统会自动调用基类的默认构造函数。  
基类或子对象类型的声明中定义了带参数的构造函数：  
必须显示地定义派生类的构造函数，并在派生类构造函数中写出基类或子对象类型的构造函数及参数表。  
基类中既定义无参数的构造函数，又重载了有参数的构造函数：  
派生类构造函数中可以写明调用带参数的基类构造函数，也可以不写调用基类的构造函数。

构造和析构函数的执行顺序：先执行基类的构造函数，接着执行派生类的构造函数；析构函数的执行顺序刚好与构造函数相反。