#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 << "请从键盘上输入" << length << "个整数:";

int i;

int\* p = alist;

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

cin >> \*p;

}

class SortArray: public MyArray {

public:

void 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;i++)

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

{

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

{

temp = alist[j];

alist[j] = alist[j + 1];

alist[j + 1] = temp;

}

}

}

int main()

{

SortArray s(5);

s.input();

s.Display("显示排序以前的");

s.Sort();

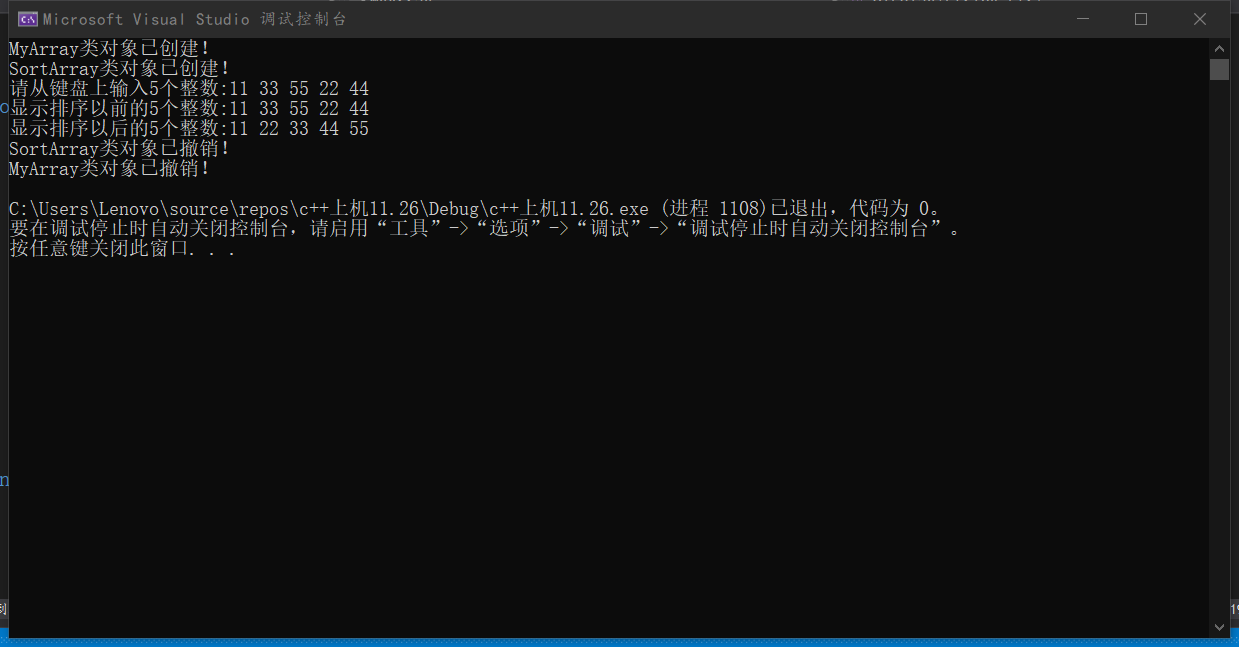
s.Display("显示排序以后的");

return 0;

}

代码

结果



感悟：在进行c++函数编写的时候，不要搞混了构造函数和析构函数的执行顺序。

先执行基类的构造函数，再执行派生类的构造函数。

先执行派生类的析构函数，再执行基类的析构函数。

如果派生类中有成员对象时，先执行基类的构造函数，再执行成员对象的构造函数，最后执行派生类的构造函数。

Copyright ©2021-2099 luanxiang wang.All rights reserved