**上机六—继承下的构造函数与析构函数**

**一程序代码**

1. #include<iostream>

#include<string>

using namespace std;

class MyArray {

protected:

int\* alist;

int length;

public:

MyArray(int leng);

~MyArray();

void input();

void Display(string);

};

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::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;

}

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 << " ";

}

int main()

{

MyArray x(5);

x.input();

x.Display("显示已经输入");

return 0;

}

2. #include<iostream>

#include<string>

using namespace std;

class MyArray {

protected:

int\* alist;

int length;

public:

MyArray(int leng);

~MyArray();

void input();

void Display(string);

};

MyArray::MyArray(int leng)

{

if (leng <= 0)

{

cout << "error length";

exit(1);

}

alist = new int[length];

length = leng;

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::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;

}

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 << " ";

}

class ShortMarry :public MyArray {

public:

void paixv();

ShortMarry(int leng) :MyArray(leng)

{

cout << "调用派生类构造" << endl;

}

~ShortMarry()

{

cout << "调用派生类析构" << endl;

}

};

void ShortMarry::paixv()

{

int i,j, temp = 0;

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

{

for (j = 0; j < length; j++)

{

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

{

temp = alist[j + 1];

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

alist[j] = temp;

}

}

}

}

int main()

{

ShortMarry x(5);

x.input();

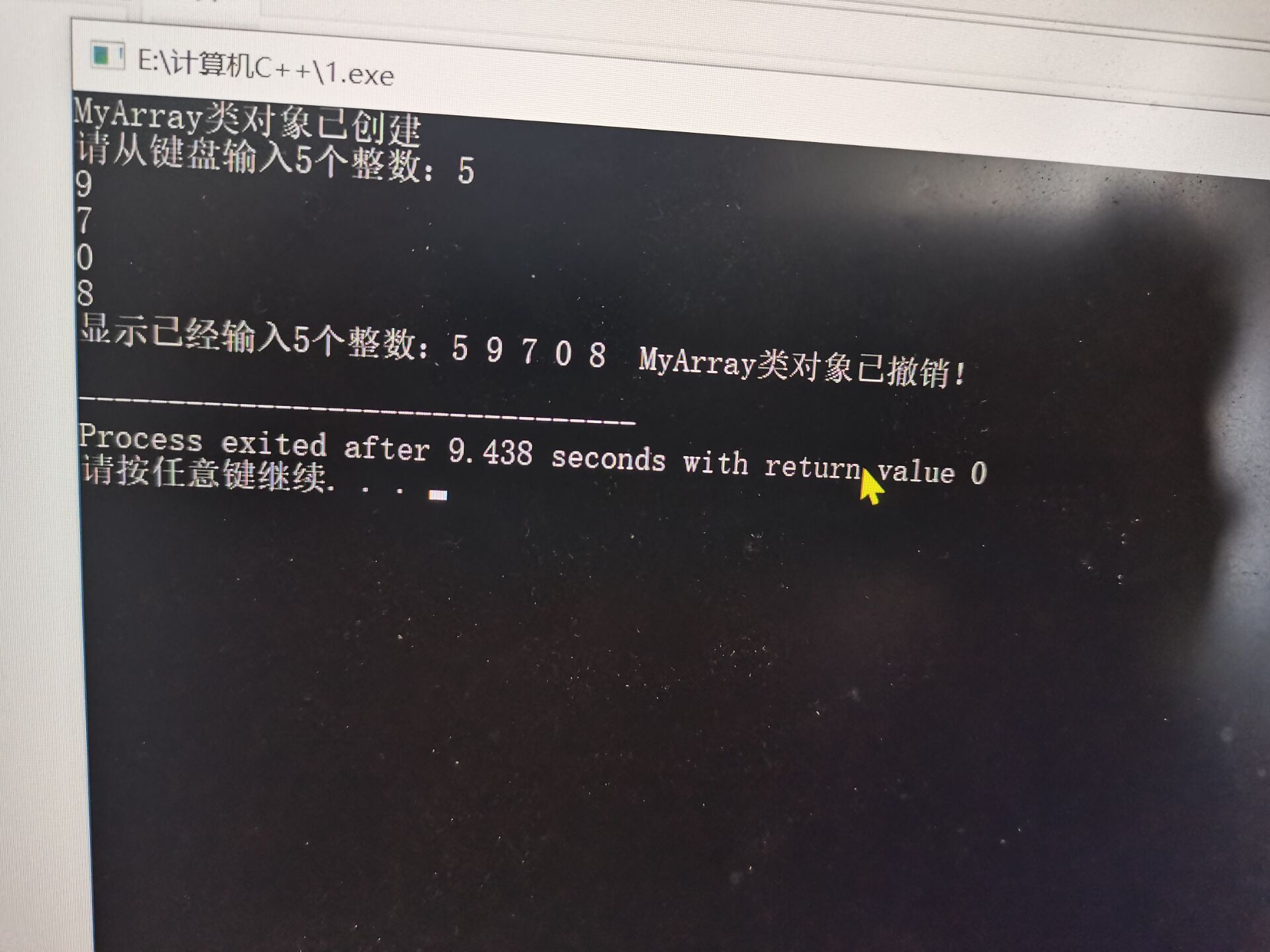
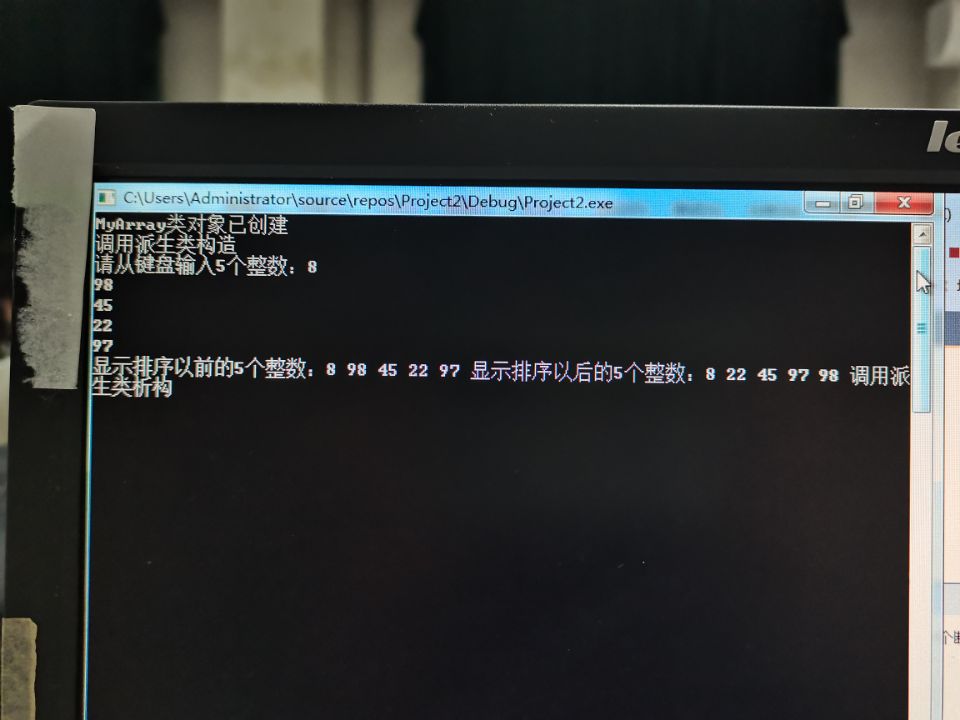
x.Display("显示排序以前的");

x.paixv();

x.Display("显示排序以后的");

return 0;

}



**二感想心得**

1.当创建派生类对象时，先执行基类构造函数，再执行派生类构造函数；析构时相反。

2.什么时候执行析构函数？

当对象调用结束时

3.冒泡排序法的应用有些许不熟，课下仍应多巩固这一知识点。

**Copyright 2021-2099 XiaowenRen. All rights reserved**