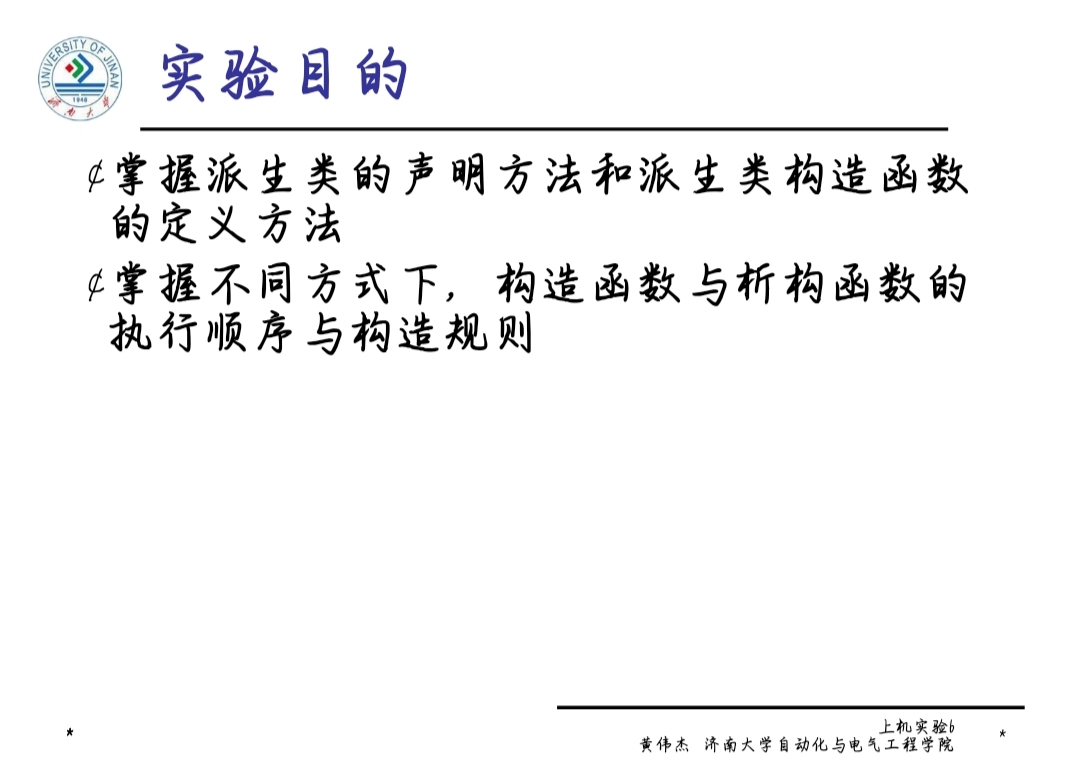
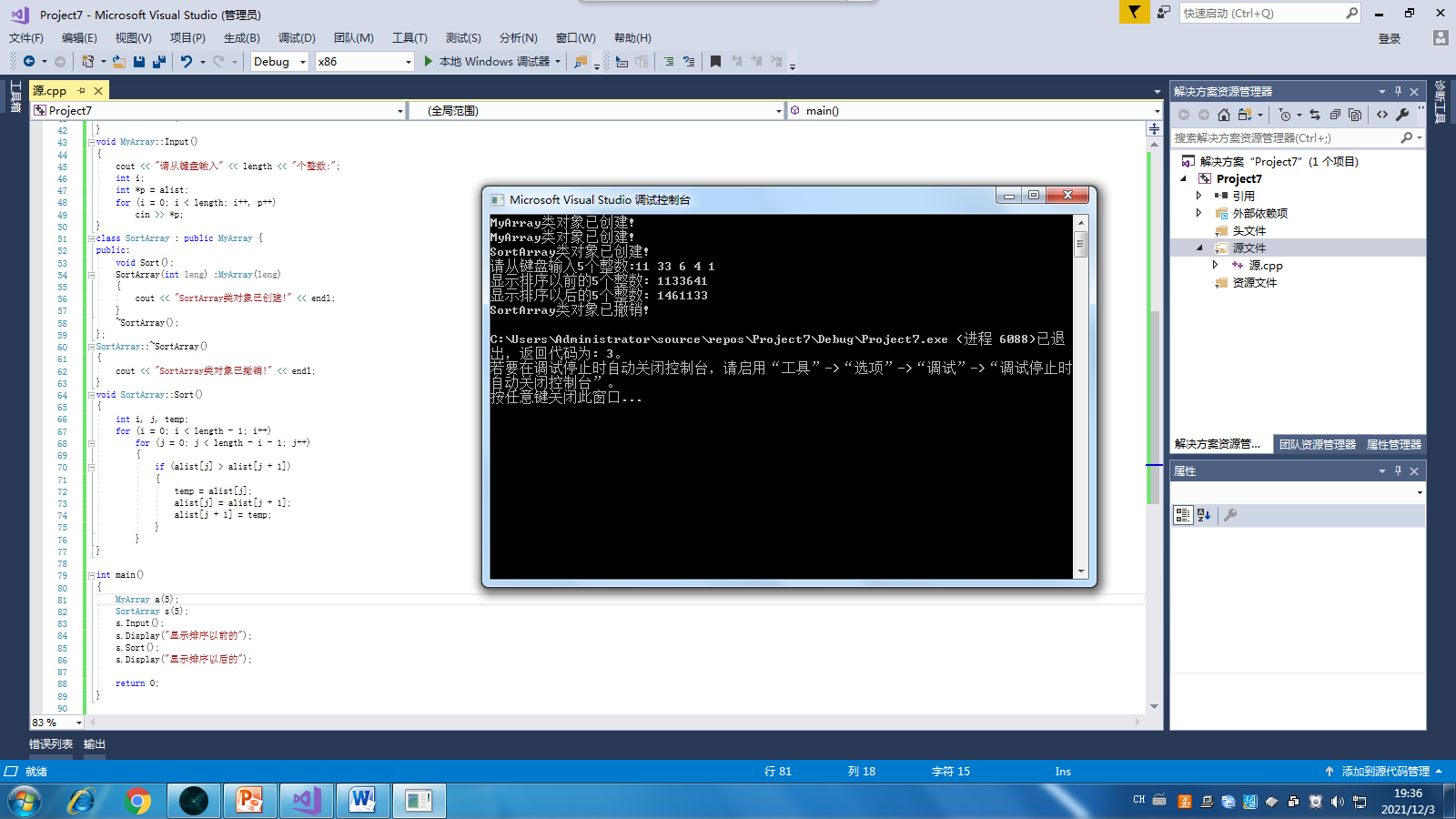
**实验六 11.26**

****

**一.操作**



#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);

}

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

{

temp = alist[j];

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

alist[j + 1] = temp;

}

}

}

int main()

{

MyArray a(5);

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;

}

**二.问题**

1.派生类构造函数调用原则是：先基类，后子对象，再派生类。  
 派生类析构函数调用原则是：先派生类，后子对象，再基类。过程刚好与派生 类的构造函数相反。

1. 基类的成员函数可以被继承，可以通过派生类的对象访问，但这仅仅指的是普通的成员函数，类的构造函数不能被继承。因为即使继承了，它的名字和派生类的名字也不一样，不能成为派生类的构造函数，更不能成为普通的成员函数。

**三、感受心得**

派生类的构造函数和析构函数的执行顺序和基类的相类似，只不过是先基类，后子对象，再派生类。他们的声明和定义的结构是一样的。

对我来说本次上机实验难度比较大，需要更好的去学习课本的内容，看着容易但是实际操作却不是那么简单（头秃），需要更多的实际操作练习。

Copyright©2021-2099 MengyuanWang.All rights reserved.