实验总结（六）

实验一：

#include<iostream>

//#include<cmath>

#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::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 << " ";

cout << endl;

}

int main()

{

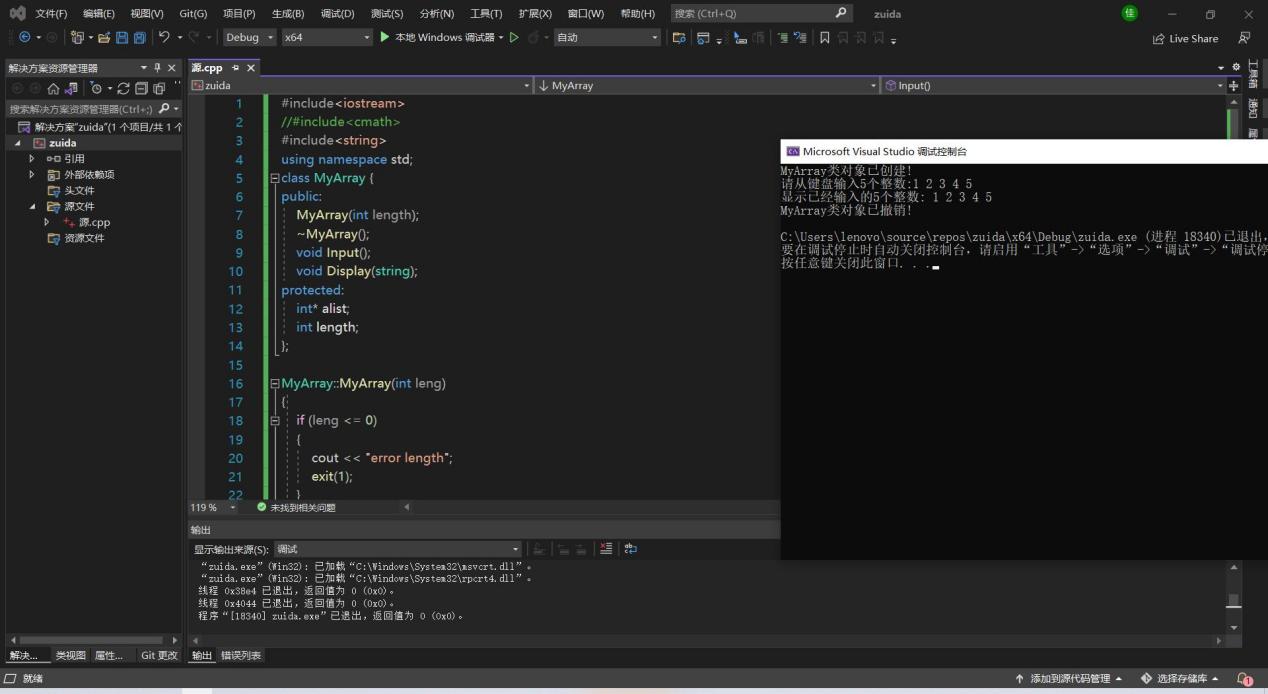
MyArray a(5);

a.Input();

a.Display("显示已经输入的");

return 0;

}

图片：  


总结一：本次实验让我明白了派生类的构造函数与析构函数的执行顺序：先构造函数再析构函数，由于即使编译构造函数后未定义析构函数，系统也会自动定义一个析构函数，因此平时我们基本无法很深入地了解析构函数，但本次实验通过自己定义一个析构函数，让我能够实实在在地看到构造函数与析构函数的执行顺序，让我更加深刻地了解析构函数。

实验二：

代码：

#include<iostream>

#include<string>

#include<cmath>

using namespace std;

class myarray {

protected:

int a[5];

int\* p=a;

public:

myarray();

~myarray();

void setmyarray();

void show();

};

myarray::myarray()

{

cout << "MyArray 类对象已创建！" << endl;

}

myarray::~myarray()

{

cout << "MyArray 类对象已撤销！" << endl;

}

void myarray::setmyarray()

{

int i;

cout << "请从键盘输入5个数：";

for (i = 0; i < 5; i++)

{

cin >> a[i];

}

}

void myarray::show()

{

int q;

cout << "显示排序以前的5个整数：";

for (q = 0; q < 5; q++)

{

cout << a[q] << " ";

}

cout << endl;

}

class sortarray:public myarray {

private:

int b[5];

int\* t=b;

public:

sortarray();

~sortarray();

void sort();

void show();

};

sortarray::sortarray()

{

cout << "SortArray 类对象已创建！" << endl;

}

sortarray::~sortarray()

{

cout << "SortArray 类对象已撤销！" << endl;

}

void sortarray::sort()

{

int n,m,w;

for (n = 0; n < 5; n++, p++)

{

b[n] = \*p;

}

for (n=0; n < 5; n++)

{

for(w=0;w<4-n;w++)

if (b[w] > b[w + 1])

{

m = b[w];

b[w] = b[w + 1];

b[w + 1] = m;

}

}

}

void sortarray::show()

{

int i;

cout << "显示排序之后的5个整数：";

for (i=0;i<5 ; i++,t++)

cout << \*t << " ";

cout << endl;

}

int main()

{

sortarray aa;

aa.setmyarray();

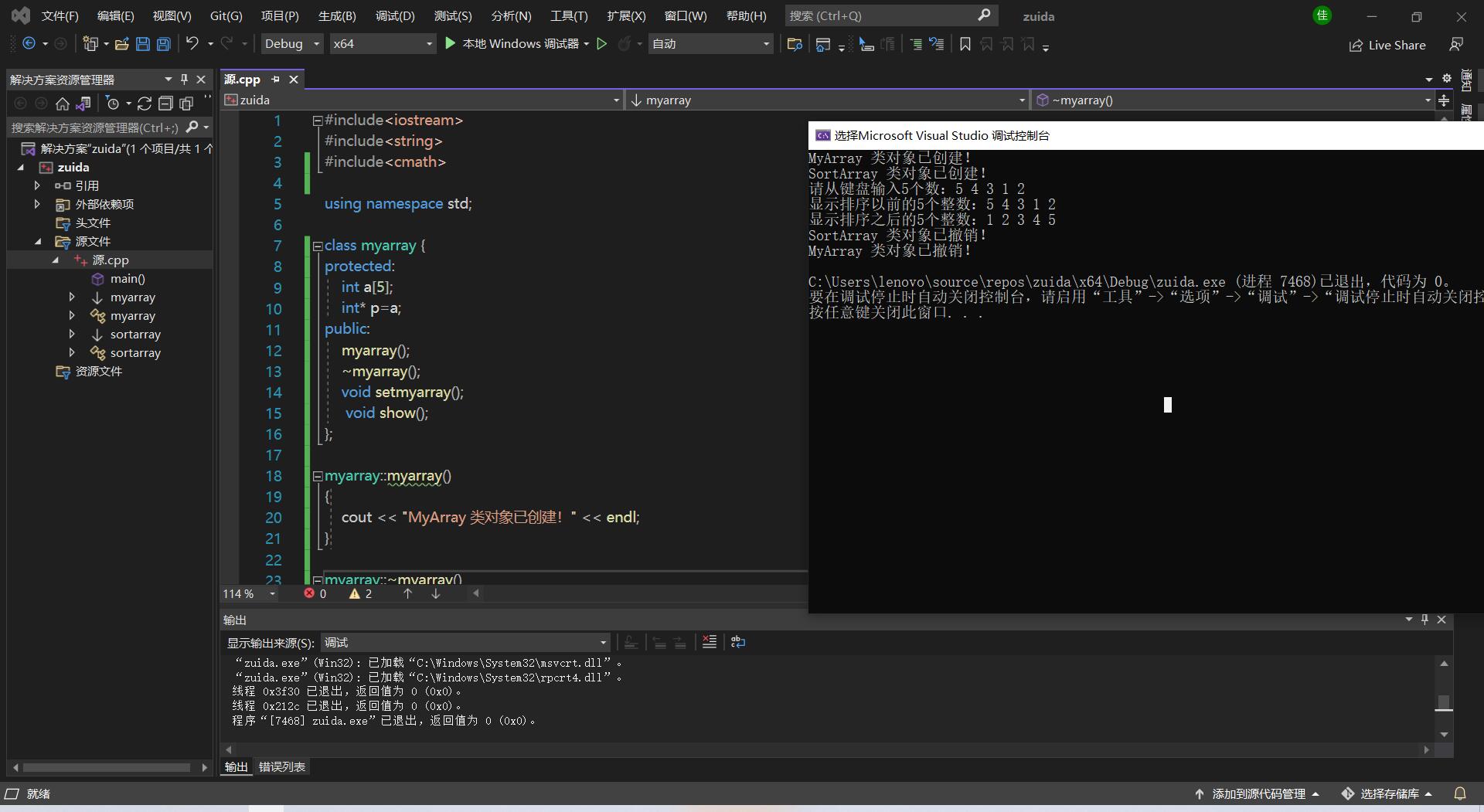
aa.sort();

aa.myarray::show();

aa.show();

return 0;

}



总结二：

该实验不仅考察了对于C++类与派生类的运用，而且还考察了关于如何利用冒泡法进行排序，派生类公有继承基类为优解，并且基类中的指针变量最好声明为保护类型，以便基类进行访问，若声明成私有类，那么派生类无法进行访问，从而无法实现程序中指针指向的参数从基类传递到派生类中。通过本次实验，使我更加深刻地了解并体会到合理设置基类与派生类之间的关系的重要性。