实验总结（上机实验六）

自2001 杨洺宇

实验一：

代码：

#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()

{

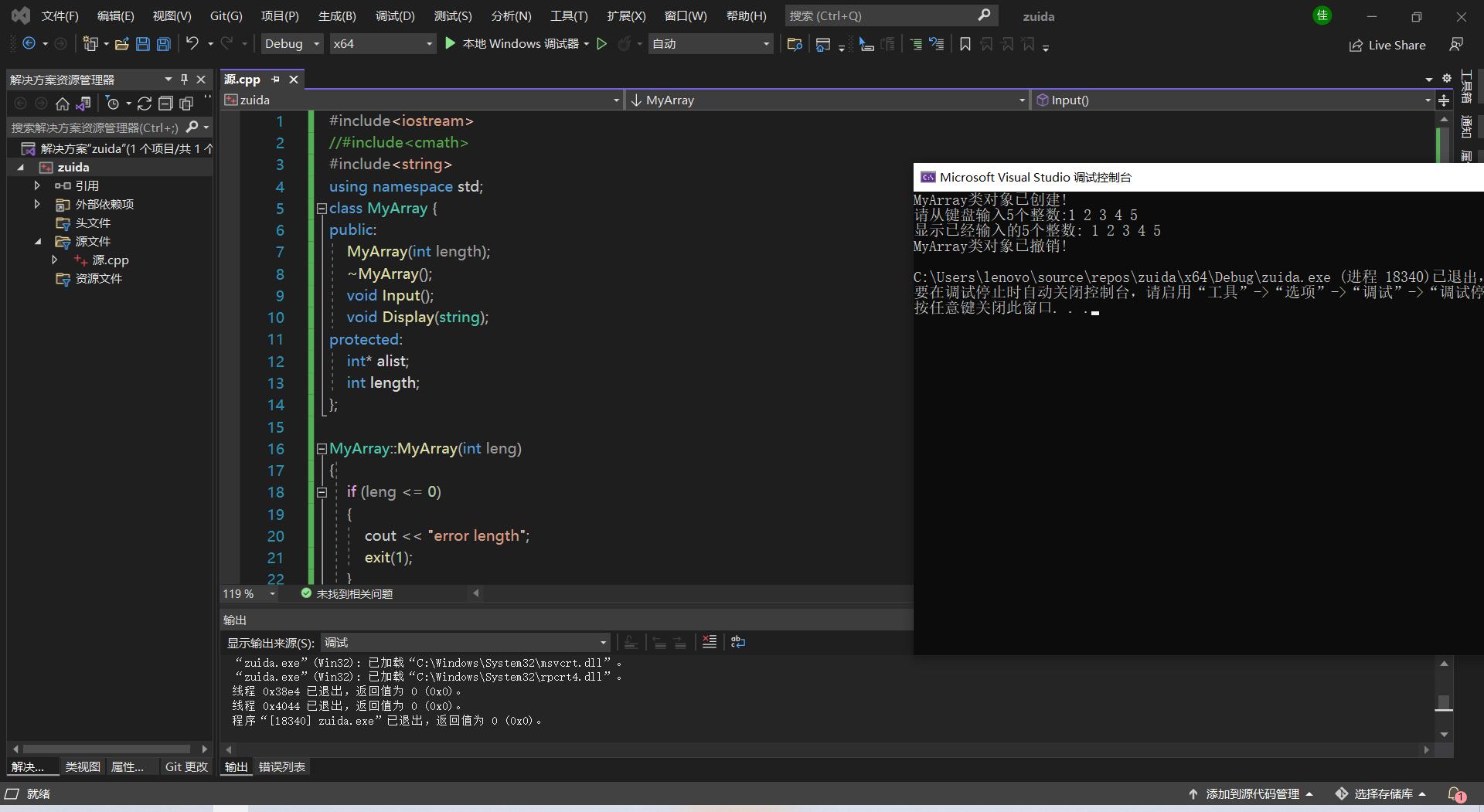
MyArray a(5);

a.Input();

a.Display("显示已经输入的");

return 0;

}

图片：  


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();

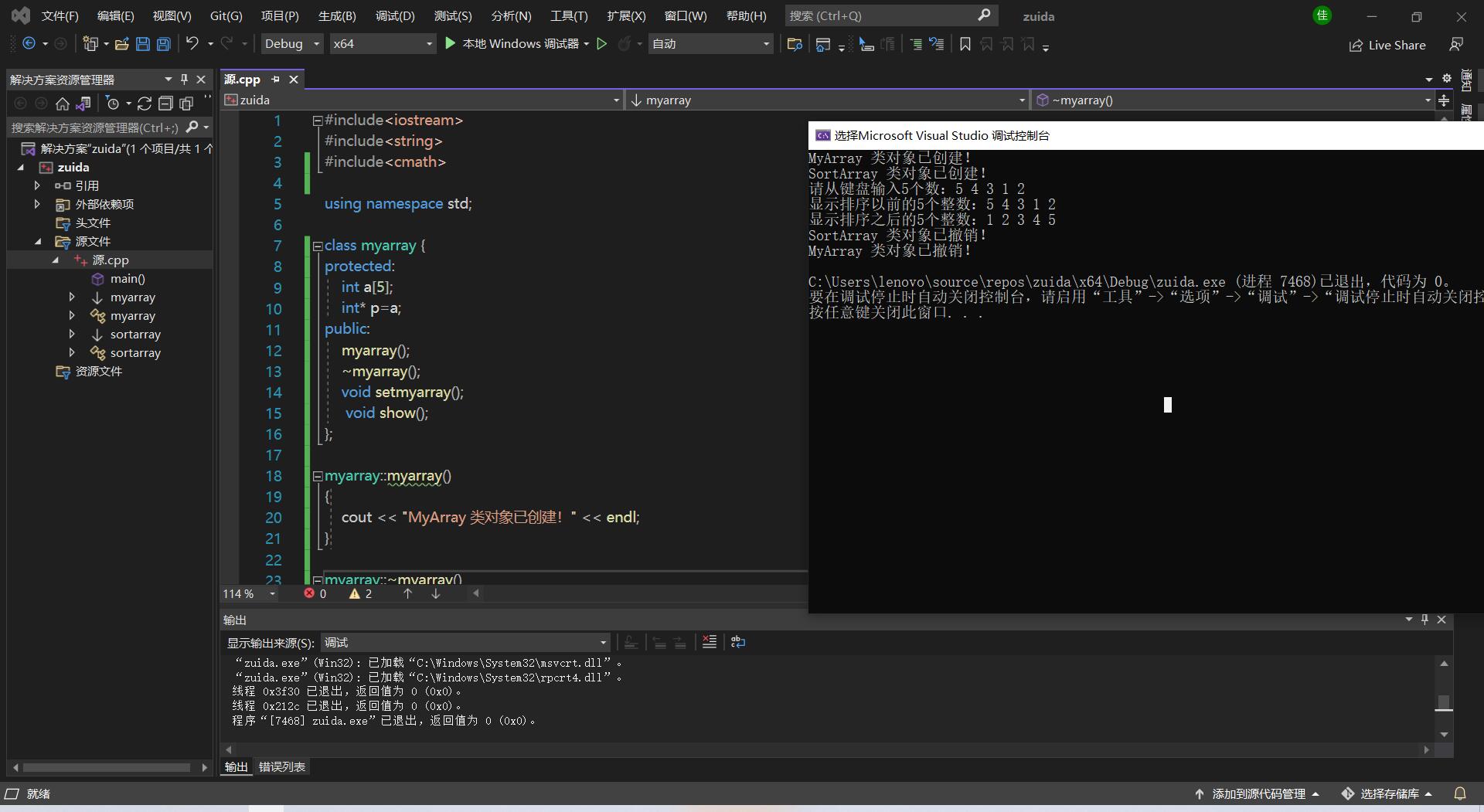
aa.myarray::show();

aa.show();

return 0;

}

图片：



总结二：

该实验1）可以将派生类构造函数定义在类的外部，而在类体内只写该函数的声明。（在类中声明派生类构造函数时，不包括基类构造函数名及其参数表，只在类外定义构造函数时才将它列出）

（2）若基类使用默认构造函数或不带参数的构造函数，则在派生类中定义构造函数时，可略去“:基类构造函数名（参数表）”，此时若派生类不需要构造函数，则可不定义派生类构造函数。

（3）当基类构造函数不带参数时，派生类不一定需要定义构造函数，然而当基类的构造函数哪怕只带有一个参数，它所有的派生类都必须定义构造函数，甚至所定义的派生类构造函数的函数体可能为空，仅仅起参数的传递作用。