**第六次上机实验报告**

#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()

{

SortArray s(5);

s.Input();

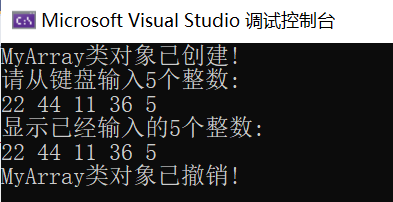
s.Display("显示排序以前的");

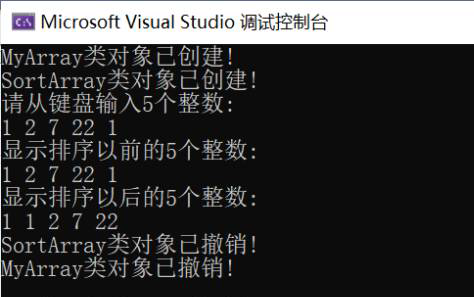
s.Sort();

s.Display("显示排序以后的");

return 0;

}





第六次上机实验感想：

首先按照老师给出的程序进行程序运行，并进行调试。

默认的继承方式是private多重继承的构造函数的执行顺序与单继承构造函数的继承的执行顺序相同，也是遵循先执行基类的构造函数，再执行对象成员的构造函数，最后执行派生类构造函数的原则。析构函数的执行顺序则与构造函数的执行顺序相反。

掌握了派生类的声明方法和派生类构造函数的定义方法，掌握了不同方式下，构造函数与析构函数的执行顺序与构造规则。其声明的一般形式为：

Class 派生类名：继承方式1 基类名1，.....，继承方式n 基类名n{

派生类新增的数据成员和成员函数

}；

声明一个SortArray继承类MyArray，在该定义中定义一个函数，具有将输入的整数从小到大进行排序的功能。

Copyright ©2021-2099 FeifeiLv. All rights reserved