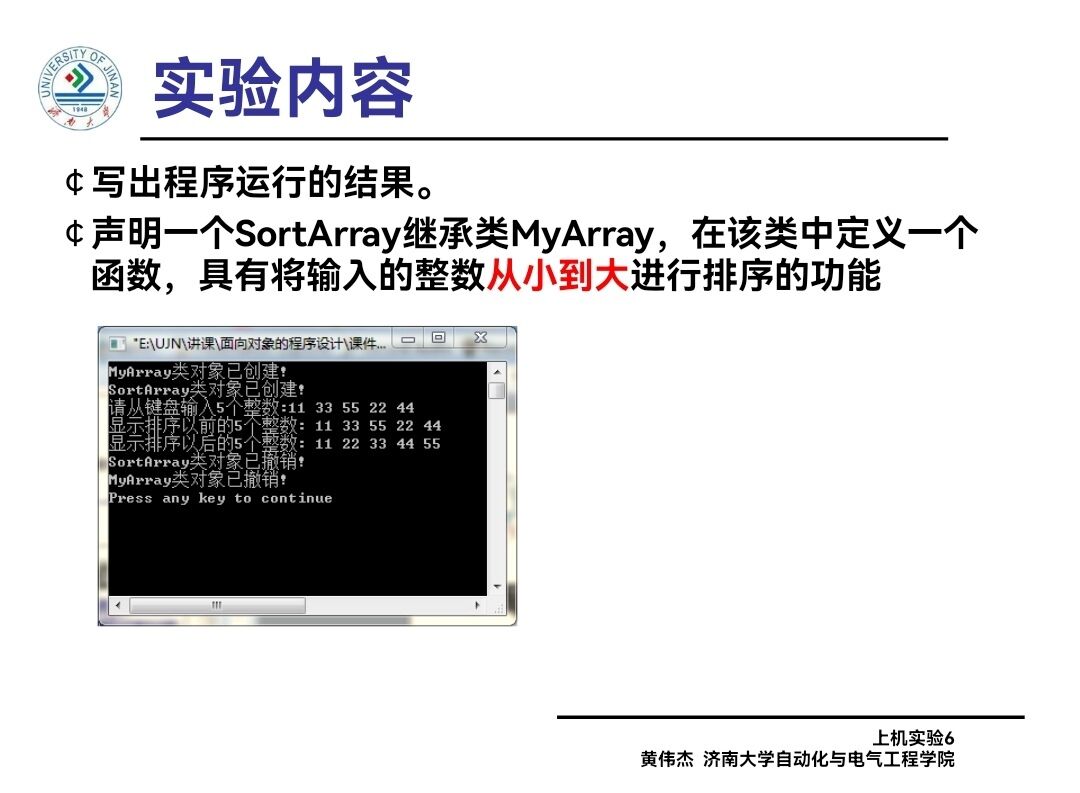
# 上机实验6



# 程序代码1

#include<iostream>

#include<iostream>

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 (length <= 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类对象被撤销。" << 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 << "请从键盘里输入" << length << "个整数:";

int i;

int\* p = alist;

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

cin >> \*p;

}

int main()

{

myarray a(5);

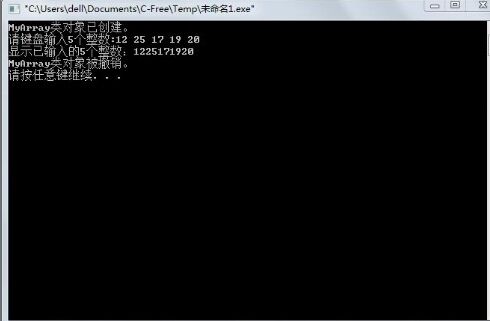
a.input();

a.display("显示已经输入的");

return 0;

}

# 程序结果1



# 程序代码2

#include<iostream>

#include<iostream>

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 (length <= 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类对象被撤销。" << 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 << "请从键盘里输入" << 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类对象被撤销。" << 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");

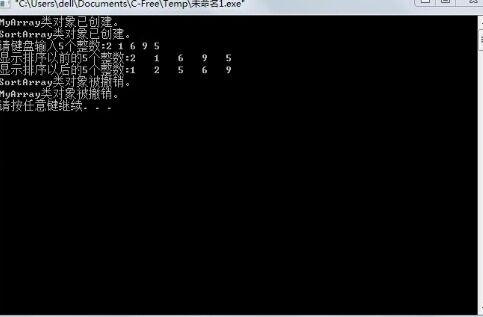
s.sort();

s.display("显示排序以后的");

return 0;

}

# 程序结果2



# 感想心得

调用函数要注意其访问权限，还要注意调用派生类的构造函数时的参数列表表达形式。这次实验掌握了派生类声明方法和派生类构造函数的定义方法，注意构造函数和派生类调用基类的构造函数赋值，在后面对新增的参数初始化，通过对象赋值后在对其输出。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：先基类构造函数，后派生类构造函数；先派生类析构，后基类析构函数。

Copyright 2021-2099 Hubing. All rights reserved