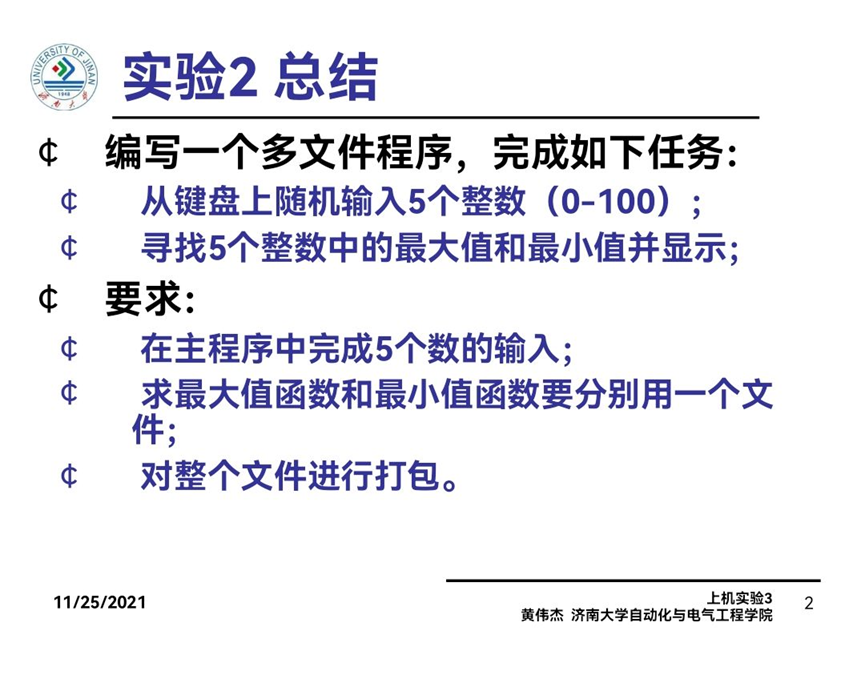
Copyright ©2021-2099 SuDequan. All rights reserved

**实验要求：**



**实验程序：**

**main.cpp**

#include <iostream>

#include "big.h"

#include "short.h"

using namespace std;

int main()

{

int m[10],i,m1,m2;

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

{

cin>>m[i];

if(m[i]>100||m[i]<0)

{

i--;

cout<<"输入错误"<<endl;

}

}

m1=max(m);

m2=min(m);

cout<<"数组最大值是： "<<m1<<" 数组最小值是： "<<m2<<endl;

system("pause");

}

**Short.cpp**

#include <iostream>

using namespace std;

int min(int a[5])

{

int i,m=a[0],x;

for(i=1;i<5;i++)

{

if(m>a[i])

{

x=m;

m=a[i];

a[i]=x;

}

}

return m;

}

**Big.cpp**

#include <iostream>

using namespace std;

int max(int a[5])

{

int i,m=a[0],x;

for(i=1;i<5;i++)

{

if(m<a[i])

{

x=m;

m=a[i];

a[i]=x;

}

}

return m;

}

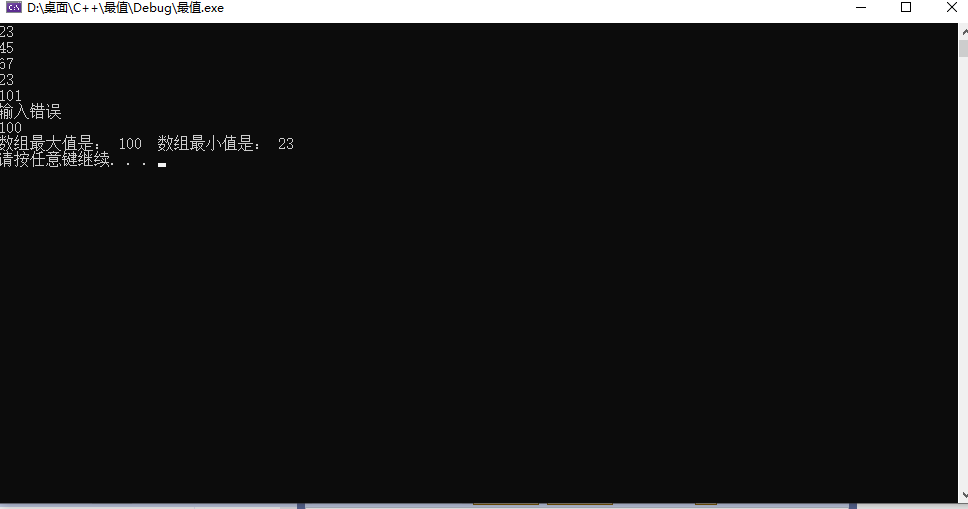
**Short.h**

int min(int a[5]);

**Big.h**

int max(int a[5]);

**实验程序输出结果：**



实验总结：

第一次编译出现错误，经查询资料后得到错误原因：函数被使用前需先声明。

多文件编程时需得在cpp文件的同名头文件中声明函数，且需得与cpp文件中的函数一致。

使用别的文件的函数时需先包含相应头文件，再用。

认识到了多文件编程的好处：

1、方便代码复用: 模块化的代码可以很方便的迁移到其他项目中去，改写模块比重写模块更快;

2、方便分工合作: 各个功能模块分成多个文件同时编辑，可以有效的提高团队开发的分工协作效率;

3、方便后续维护: 项目源码交接时，不管是客户还是同事，对于划分明确的文件是不会排斥的;

4、保证了库支持: 个人函数库都是由c文件产生库，采用“库文件+h文件”的形式，可以实现快速编译和保护源码的效果。