

（1）

程序代码：

#include<iostream>

using namespace std;

class Coordinate{ // 定义Coordinate类

public:

Coordinate()

{

times=2;

cout<<"Coordinate construction1 called!"<<endl;

}

Coordinate(int times1)

{

times=times1;

cout<<"Coordinate construction2 called!"<<endl;

}

~Coordinate()

{

cout<<"Coordinate destruction called!"<<endl;

}

void InputCoord()

{

for(int i=0;i<times;i++)

{

cout<<"Please Input x:"<<endl;

cin>>Coord[i][1];

cout<<"Please Input y:"<<endl;

cin>>Coord[i][2];

}

}

void ShowCoord()

{

cout<<"The coord is:"<<endl;

for(int i=0;i<times;i++)

{

cout<<"("<<Coord[i][1]<<","<<Coord[i][2]<<")"<<endl;

}

}

void ShowAvgCoord()

{

float avgx = 0;

float avgy = 0;

for(int i=0;i<times;i++)

{

avgx = avgx + Coord[i][1];

avgy = avgy + Coord[i][2];

}

avgx = avgx/times;

avgy = avgy/times;

cout<<"The AVG coord is:"<<endl;

cout<<"("<<avgx<<","<<avgy<<")"<<endl;

}

private:

float Coord[100][100]; // 存放输入坐标的数组

int times; // 存放输入坐标数目

};

int main()

{

Coordinate x;

x.InputCoord();

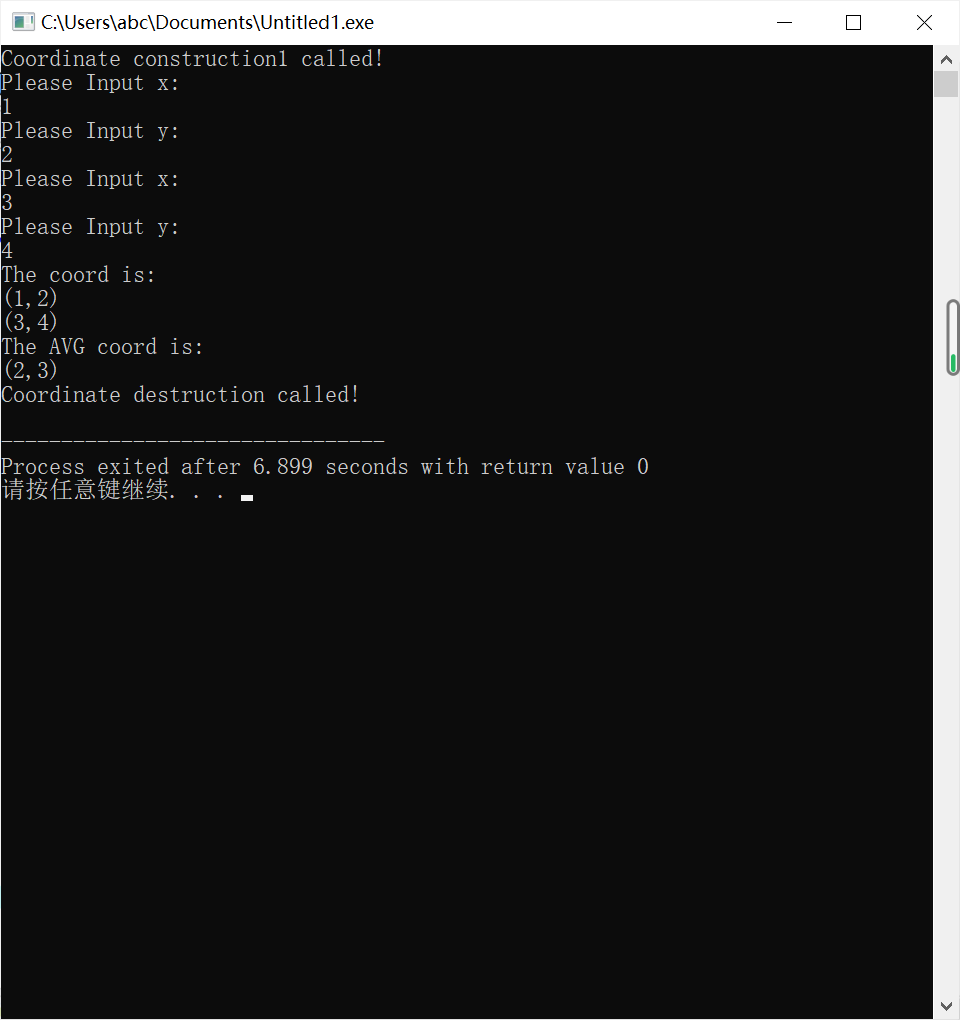
x.ShowCoord();

x.ShowAvgCoord();

return 0;

}

程序结果：



运行顺序：

主函数里Coordinate x调用了Coordinate()这个构造函数，因此在代码运行的时候首先输出Coordinate construction1 called!，根据所学的构造函数和析构函数运行顺序，析构函数 ~Coordinate()最后运行，输出Coordinate destruction called!。

（2）

程序代码：

#include<iostream>

using namespace std;

class Coordinate{ // 定义Coordinate类

public:

Coordinate()

{

times=2;

cout<<"Coordinate construction1 called!"<<endl;

}

Coordinate(int times1)

{

times=times1;

cout<<"Coordinate construction2 called!"<<endl;

}

~Coordinate()

{

cout<<"Coordinate destruction called!"<<endl;

}

void InputCoord()

{

for(int i=0;i<times;i++)

{

cout<<"Please Input x:"<<endl;

cin>>Coord[i][1];

cout<<"Please Input y:"<<endl;

cin>>Coord[i][2];

}

}

void ShowCoord()

{

cout<<"The coord is:"<<endl;

for(int i=0;i<times;i++)

{

cout<<"("<<Coord[i][1]<<","<<Coord[i][2]<<")"<<endl;

}

}

void ShowAvgCoord()

{

float avgx = 0;

float avgy = 0;

for(int i=0;i<times;i++)

{

avgx = avgx + Coord[i][1];

avgy = avgy + Coord[i][2];

}

avgx = avgx/times;

avgy = avgy/times;

cout<<"The AVG coord is:"<<endl;

cout<<"("<<avgx<<","<<avgy<<")"<<endl;

}

private:

float Coord[100][100]; // 存放输入坐标的数组

int times; // 存放输入坐标数目

};

int main()

{

Coordinate x;

x.InputCoord();

x.ShowCoord();

x.ShowAvgCoord();

Coordinate y(5);

y.InputCoord();

y.ShowCoord();

y.ShowAvgCoord();

return 0;

}

程序结果：

Coordinate construction1 called!

Please Input x:

1

Please Input y:

2

Please Input x:

3

Please Input y:

4

The coord is:

(1,2)

(3,4)

The AVG coord is:

(2,3)

Coordinate construction2 called!

Please Input x:

5

Please Input y:

6

Please Input x:

7

Please Input y:

8

Please Input x:

9

Please Input y:

8

Please Input x:

7

Please Input y:

6

Please Input x:

5

Please Input y:

4

The coord is:

(5,6)

(7,8)

(9,8)

(7,6)

(5,4)

The AVG coord is:

(6.6,6.4)

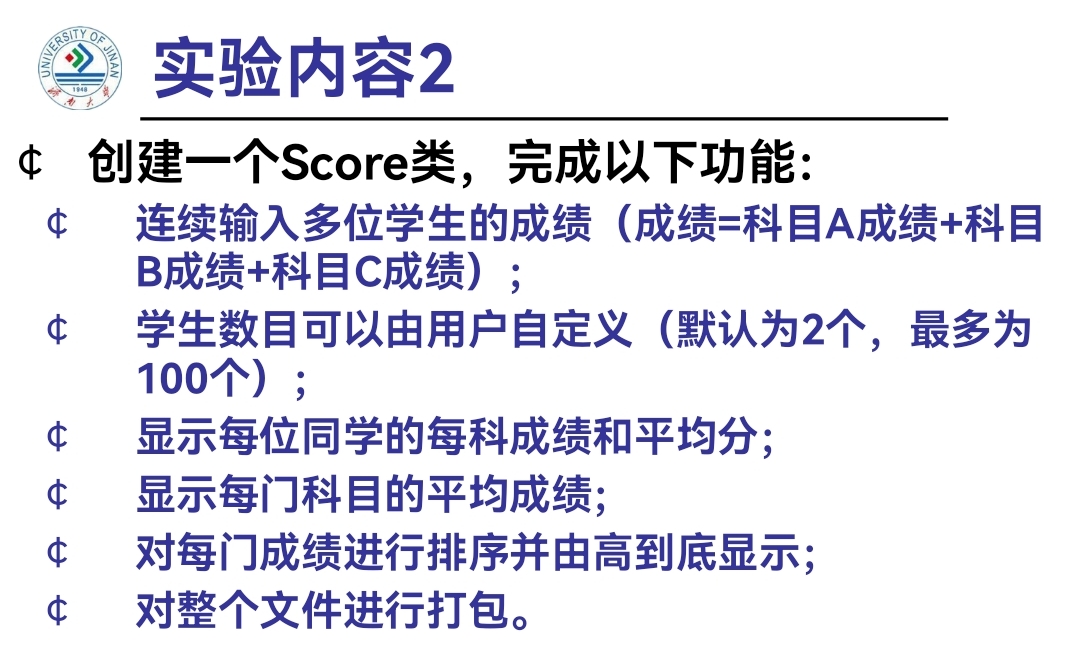
Coordinate destruction called!

Coordinate destruction called!

运行顺序及结果：

在（1）的基础上，又增加了对Coordinate(int times1)这个含有参数的构造函数的调用，因此在（1）的运行结果之后加上了Coordinate construction2 called!

这个调用结果，在调用结束后的结尾部分，对析构函数进行调用，因为析构函数的调用顺序与构造函数的调用顺序相反，因此先执行含有参数构造函数的析构函数，最后执行不含参数构造函数的析构函数。



程序代码：

#include<iostream>

#include<string>

using namespace std;

class Score{

public:

Score()

{ times=2; }

Score(int times1)

{ times=times1; }

void InputNameAndScore()

{

for(int i=0;i<times;i++)

{

cout<<"请输入学生姓名:"<<endl;

cin>>Name[i];

cout<<"请输入科目A成绩:"<<endl;

cin>>SScore[i][1];

cout<<"请输入科目B成绩:"<<endl;

cin>>SScore[i][2];

cout<<"请输入科目C成绩:"<<endl;

cin>>SScore[i][3];

}

}

void ShowNameAndScore()

{

for(int i=0;i<times;i++)

{

cout<<"姓名: "<<Name[i]<<" 科目A成绩: "<<SScore[i][1]<<" 科目B成绩 "<<SScore[i][2]<<" 科目C成绩: "<<SScore[i][3]<<endl;

}

}

void ShowStdentAvgScore(int Sid)

{

float avg = 0;

avg = (SScore[Sid][1]+SScore[Sid][2]+SScore[Sid][3])/3;

cout<<"姓名: "<<Name[Sid]<<" 平均成绩: "<<avg<<endl;

}

void ShowClassAvgScore(string ClassName)

{ int Cid;

float avg = 0;

if(ClassName == "A") Cid = 1;

if(ClassName == "B") Cid = 2;

if(ClassName == "C") Cid = 3;

for(int i=0;i<times;i++)

{ avg = avg + SScore[i][Cid]; }

avg = avg/times;

cout<<"课程名称: "<<ClassName<<"平均成绩: "<<avg<<endl;

}

void OrderScore(string ClassName)

{

int Cid;

if(ClassName == "A") Cid = 1;

if(ClassName == "B") Cid = 2;

if(ClassName == "C") Cid = 3;

for(int i=0;i<times;i++)

{ SScore1[i] = SScore[i][Cid]; }

for(int i=0;i<times;i++)

{ Name1[i] = Name[i]; }

for(int i=1;i<times;i++)

{ if(SScore1[i] > SScore1[i-1])

{ float temp = SScore1[i-1];

SScore1[i-1] = SScore1[i];

SScore1[i] = temp;

string temp1;

temp1 = Name1[i-1];

Name1[i-1] = Name1[i];

Name1[i]= temp1;

}

}

cout<<"课程名称: "<<ClassName<<endl;

for(int i=0;i<times;i++)

{

cout<<"姓名: "<<Name1[i]<<" 成绩: "<<SScore1[i]<<endl;

}

}

private:

float SScore[100][3],SScore1[100];

string Name[100],Name1[100];

int times;

};

int main()

{ Score x;

x.InputNameAndScore();

x.ShowNameAndScore();

x.ShowStdentAvgScore(1);

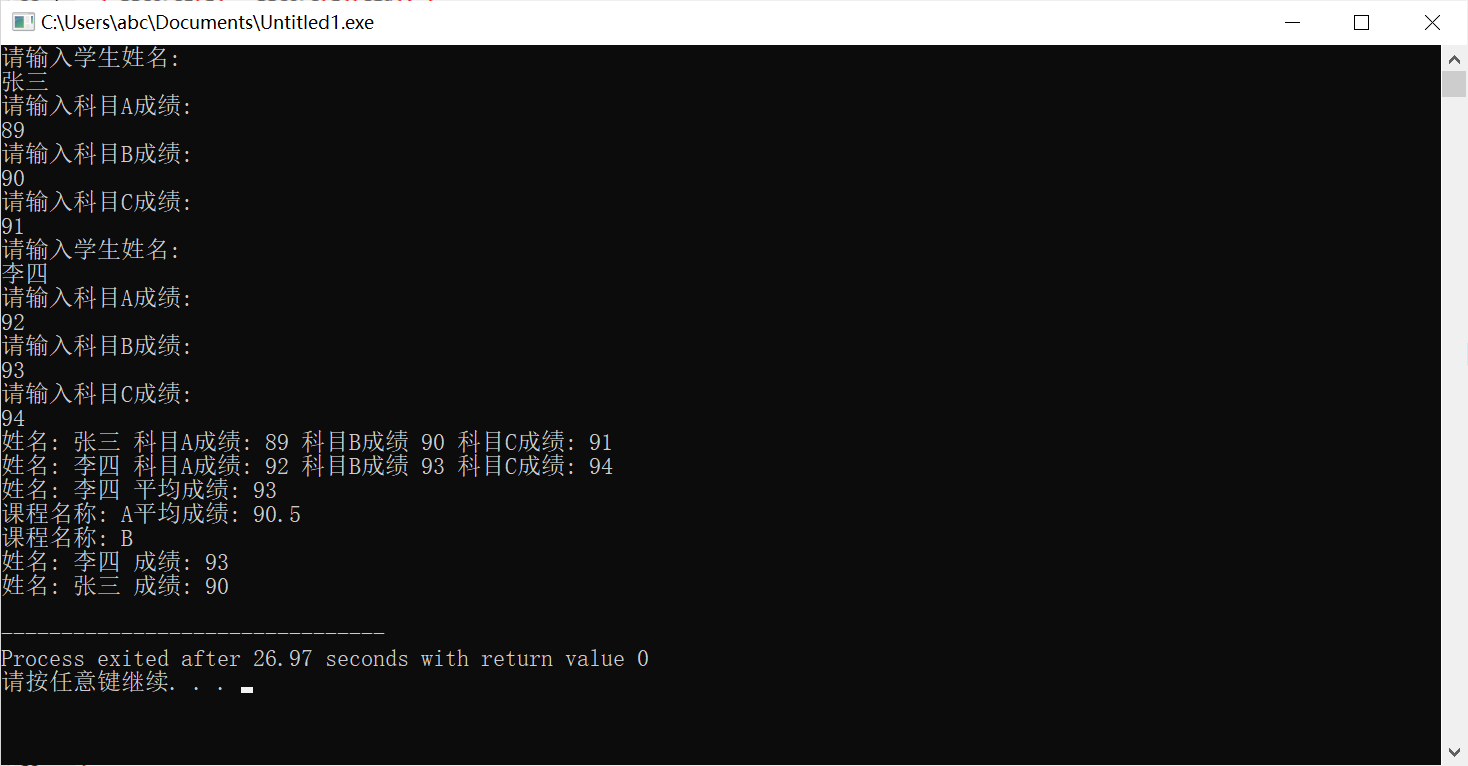
x.ShowClassAvgScore("A");

x.OrderScore("B");

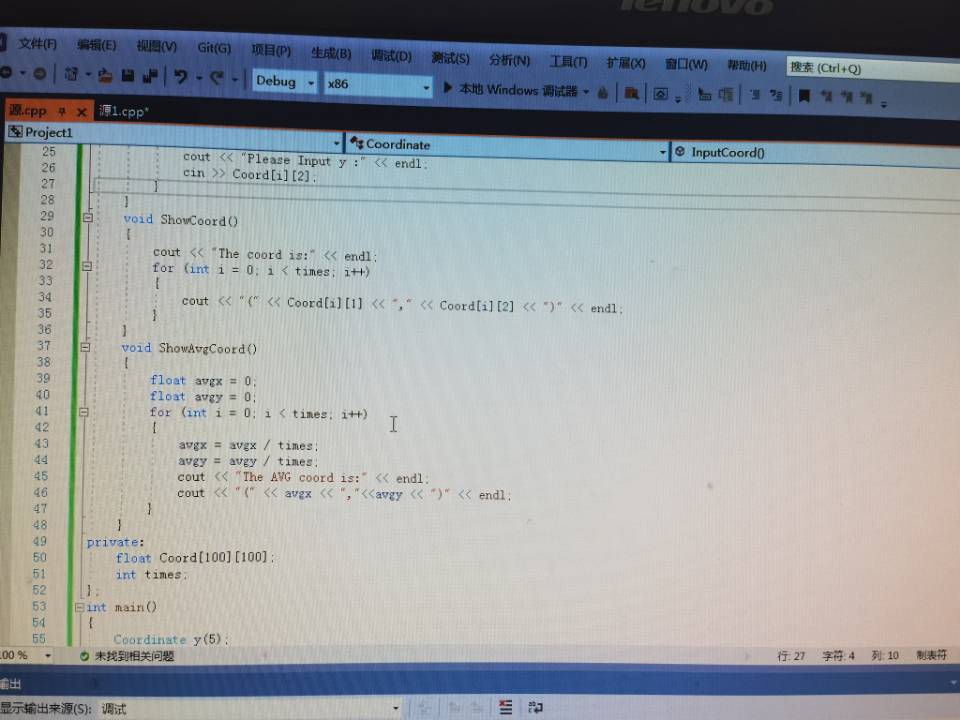
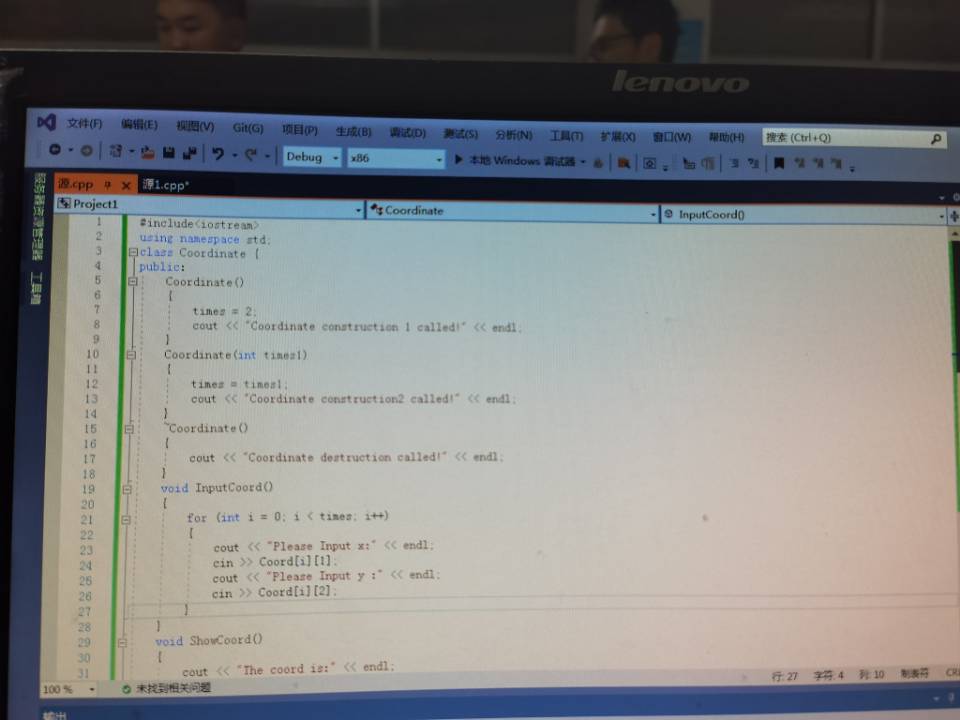
return 0;

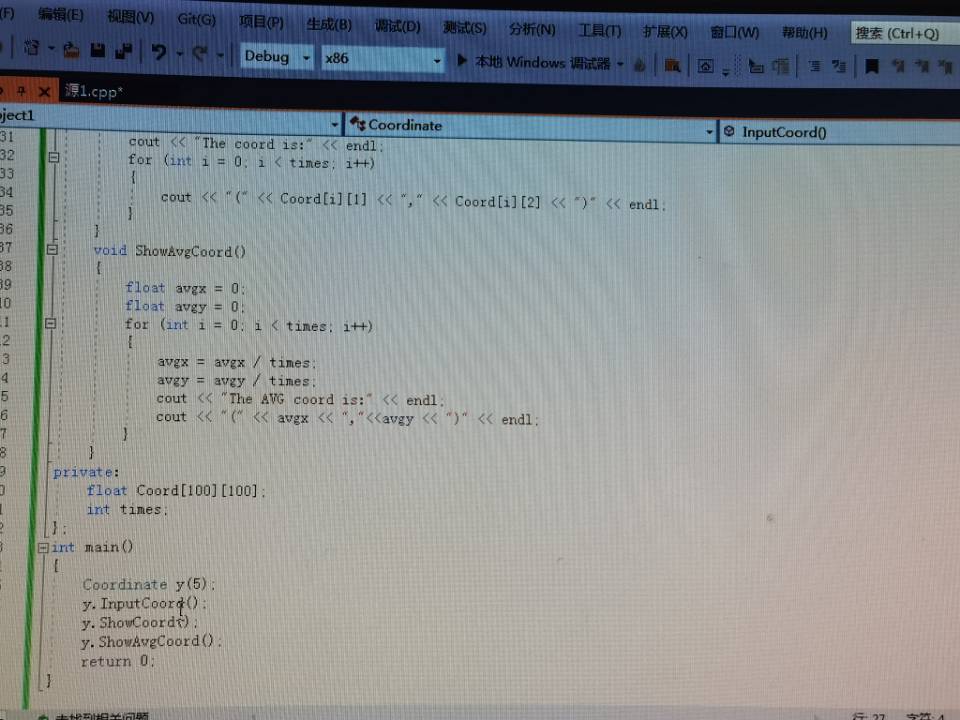
}

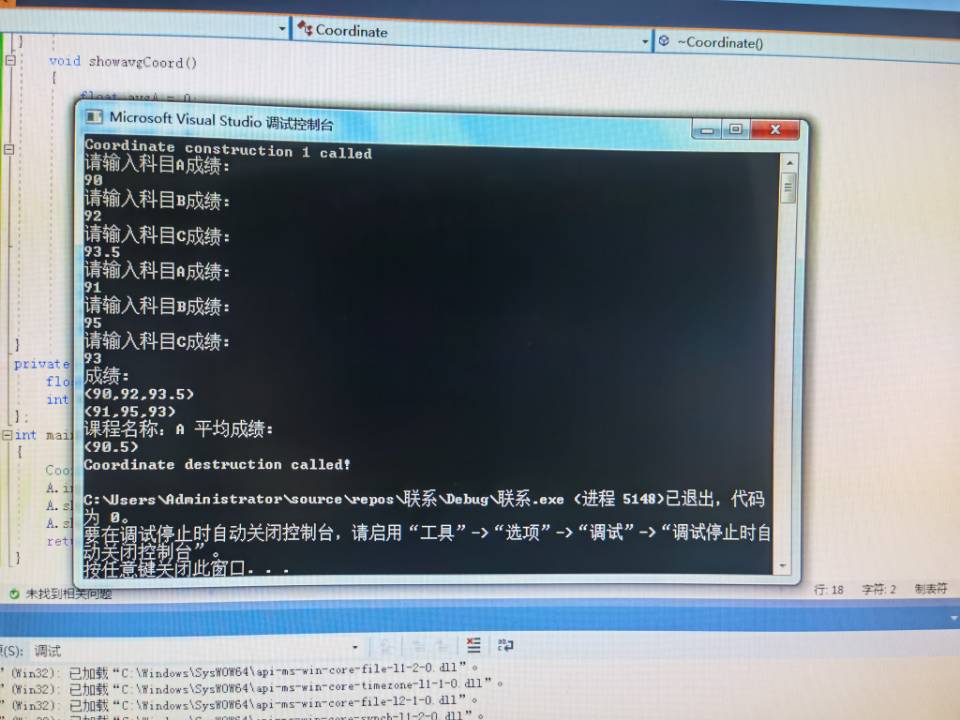
程序结果：



实验2程序代码（有错）







感想心得：

这节科主要学习了构造函数和析构函数的实现方法以及运行顺序，理解了类和对象的概念，掌握了声明类和定义对象的方法，初步掌握了使用类和对象编制的C++程序。通过实验内容1了解到一个类可以有多个构造函数，但是只能有一个析构函数，析构函数的执行顺序与构造函数的执行顺序完全相反，知道这个道理之后不难读懂程序的运行结果；实验内容2实现了连续输入多位学生的成绩，设置了默认值和上限。通过这节课的真实操作，对课本上的之后有了更加深入的了解，使得知识变得容易理解。

Copyright ©2021-2099 HuixinLi. All rights reserved