**实验内容（1）**

* **观察例程中的构造函数和析构函数的运行顺序；**
* **在main()函数中加入如下代码，观察运行结果：**

**Coordinate y(5);**

**y.InputCoord();**

**y.ShowCoord();**

**y.ShowAvgCoord();**

**1.实验代码**

#include<iostream>

using namespace std;

class Coordinate {

private:

float Coord[100][100]; // 存放输入坐标的数组

int times; // 存放输入坐标数目

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;

}

};

int main()

{

Coordinate y(5);

y.InputCoord();

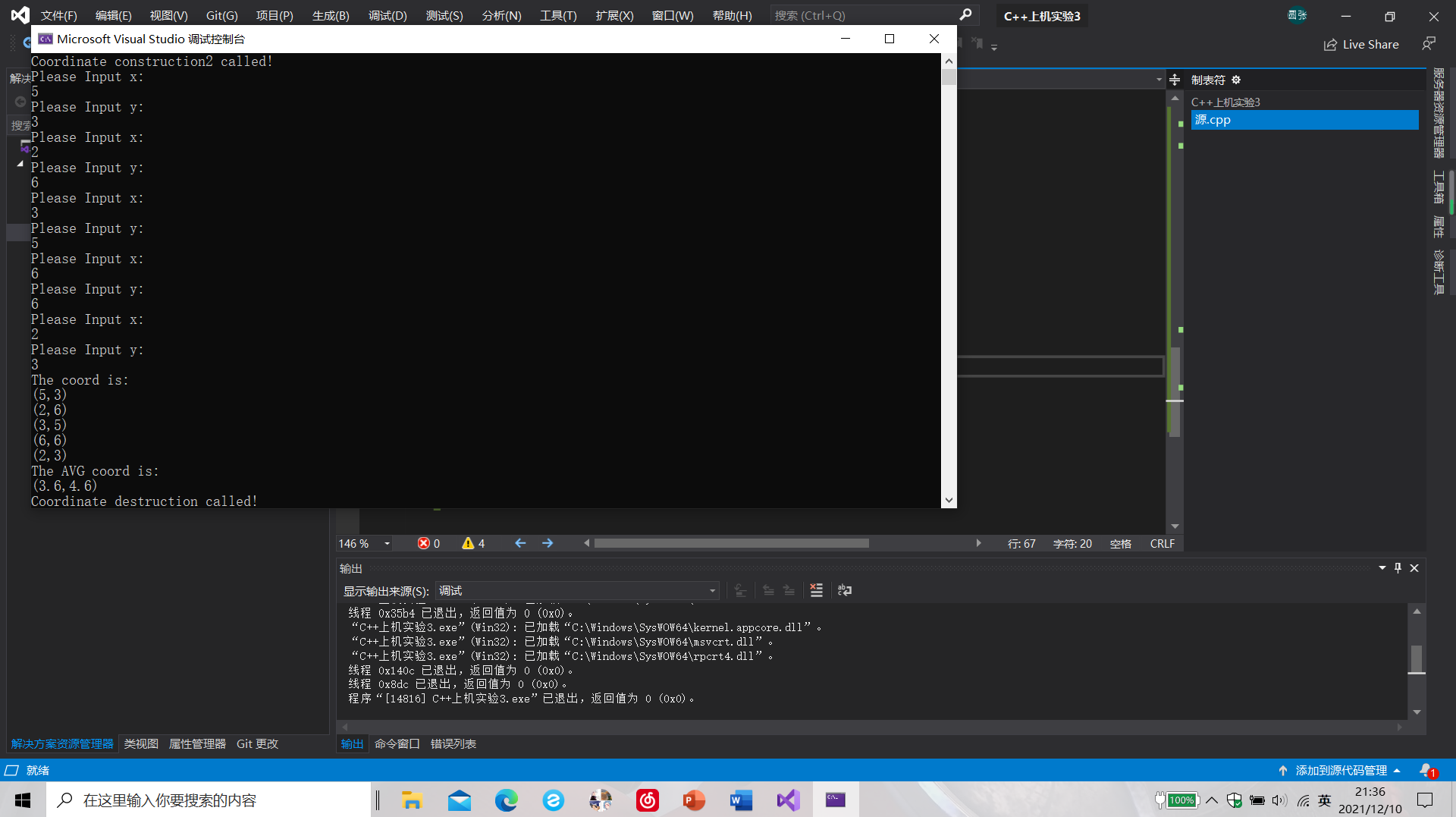
y.ShowCoord();

y.ShowAvgCoord();

return 0;

}

**2.运行结果**



**二.实验内容（2）**

* **创建一个Score类，完成以下功能：**
  + **连续输入多位学生的成绩（成绩=科目A成绩+科目B成绩+科目C成绩）；**
  + **学生数目可以由用户自定义（默认为2个，最多为100个）；**
  + **显示每位同学的每科成绩和平均分；**
  + **显示每门科目的平均成绩；**
  + **对每门成绩进行排序并由高到底显示；**
  + **对整个文件进行打包。**

**程序代码**

#include<iostream>

#include<string>

using namespace std;

class Score {

public:

Score()

{

times = 2;

}

Score(int times1)

{

times = times1;

}

~Score()

{

cout << "结束" << endl;

}

void InputCoord()

{

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的成绩" << endl;

cin >> Coord[i][1];

cout << "请输入科目B的成绩" << endl;

cin >> Coord[i][2];

cout << "请输入科目C的成绩" << endl;

cin >> Coord[i][3];

}

}

void ShowCoord()

{

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

{

cout << "姓名: " << name[i] << " 科目A的成绩:" << Coord[i][1] << " " << " 科目B的成绩:" << Coord[i][2] << " " << " 科目C的成绩:" << Coord[i][3] << endl;

}

}

void ShowAvgCoord1()

{

float avgx = 0;

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

{

avgx = 0;

for (int j = 1; j < 4; j++)

{

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

}

avgx = avgx / 3.0;

cout << "姓名:" << name[i] << " 平均成绩:" << avgx << endl;

}

}

void ShowAvgCoord2()

{

float avgy = 0;

for (int i = 1; i < 4; i++)

{

avgy = 0;

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

{

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

}

avgy = avgy / times;

if (i == 1)

cout << "课程名称：" << "A 平均成绩; " << avgy << endl;

if (i == 2)

cout << "课程名称：" << "B 平均成绩: " << avgy << endl;

if (i == 3)

cout << "课程名称：" << "C 平均成绩: " << avgy << 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++)

{

Coord1[i] = Coord[i][Cid];

}

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

{

name1[i] = name[i];

}

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

{

if (Coord1[i] > Coord1[i - 1])

{

float temp = Coord1[i - 1];

Coord1[i - 1] = Coord1[i];

Coord1[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: " << Coord1[i] << endl;

}

}

private:

float Coord[100][100],Coord1[100]; // 存放输入坐标的数组

int times; // 存放输入学生数目

string name[100],name1[100];

};

int main()

{

Score y;

y.InputCoord();

y.ShowCoord();

y.ShowAvgCoord1();

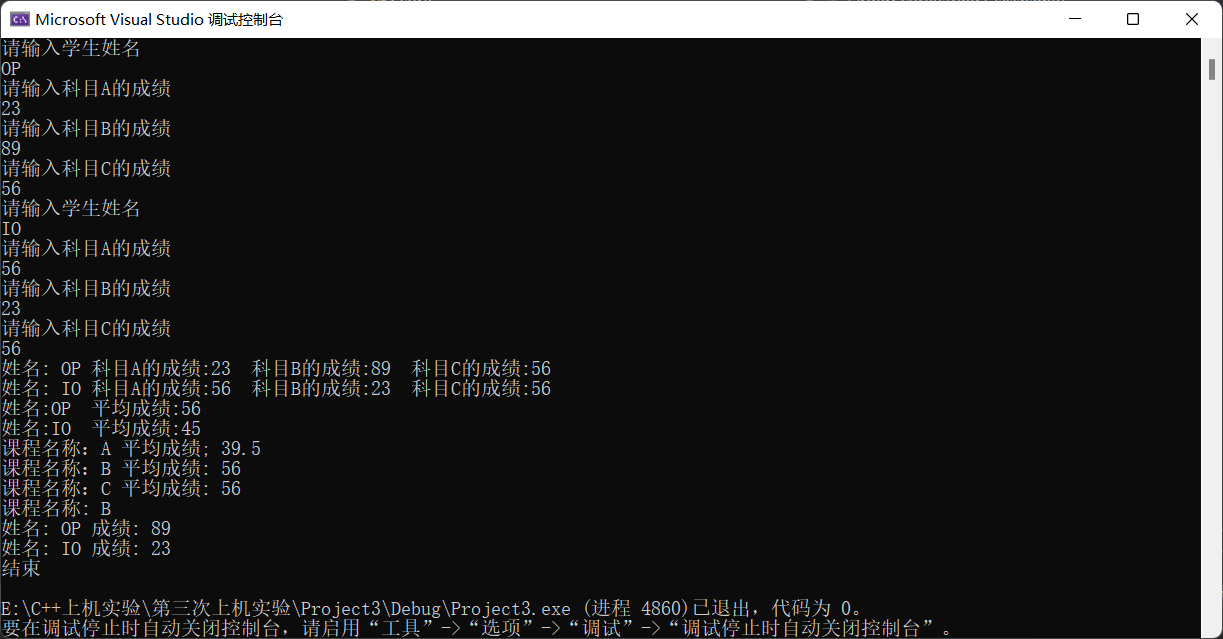
y.ShowAvgCoord2();

y.OrderScore("B");

return 0;

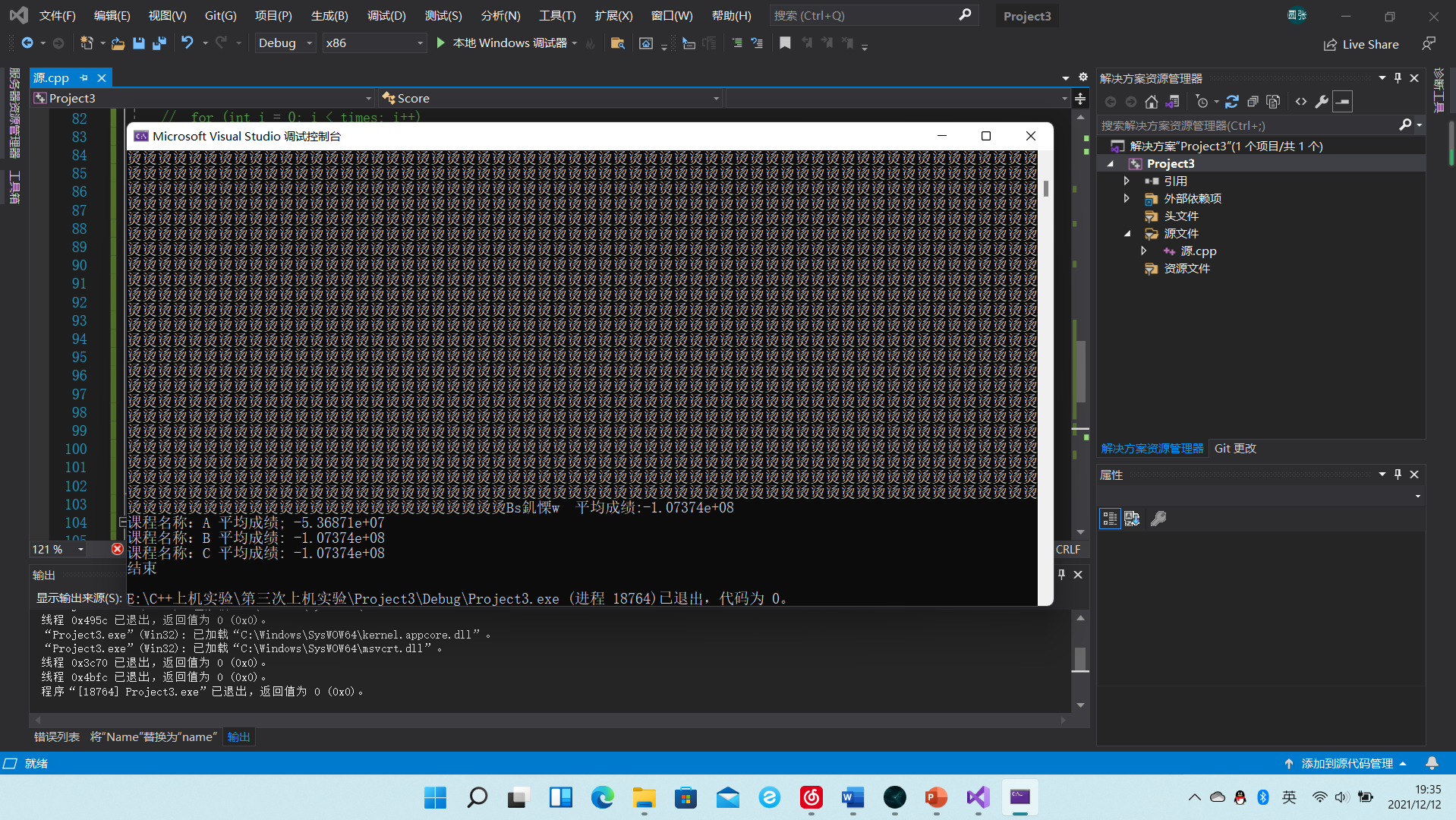
}

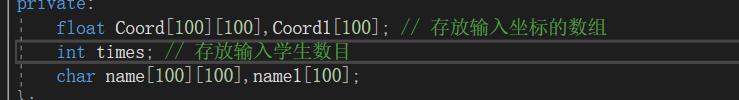
**运行结果**

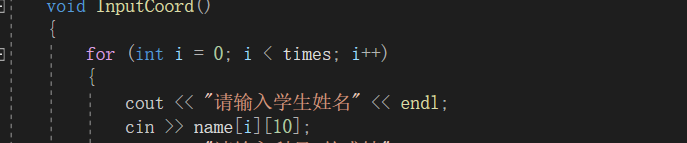


**感悟**

**（1）对char型变量的错误理解。**







**定义了一个char的二维数组，想法上是每一维存储一个名字，但是char型是一个字符一个字符的输入，改用了简单的string型字符串变量解决了问题。**

**++i和i++这两种形式，在不需要得到i改变前的值的情况下，尽量不要采用后置情况，会给编译器增加不必要的负担，需要分出空间来存储这个改变前的值。**