1. 程序代码

第一个程序

#include <iostream>

using namespace std;

class Coordinate { // 定义Coordinate类

public:

Coordinate()

{

times = 2;

cout << "Coordinate construction1 called!" << endl;// 设置默认的输入坐标数目

}

Coordinate(int times1)

{

times = times1;

cout << "Coordinate construction2 called!" << endl; // 设置输入坐标数目

}

~Coordinate()

{ // 析构函数

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

}

void InputCoord()

{ // 输入坐标

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()

{ // 显示已经输入的坐标

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

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

{

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

}

}

void ShowAvgCoord()

{ // 显示输入坐标的均值

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; // 定义对象

x.InputCoord();

x.ShowCoord();

x.ShowAvgCoord(); // 执行显示和坐标均值运算

Coordinate y(5);

y.InputCoord();

y.ShowCoord();

y.ShowAvgCoord();

system("pause");

return 0;

}

第二个程序

#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");

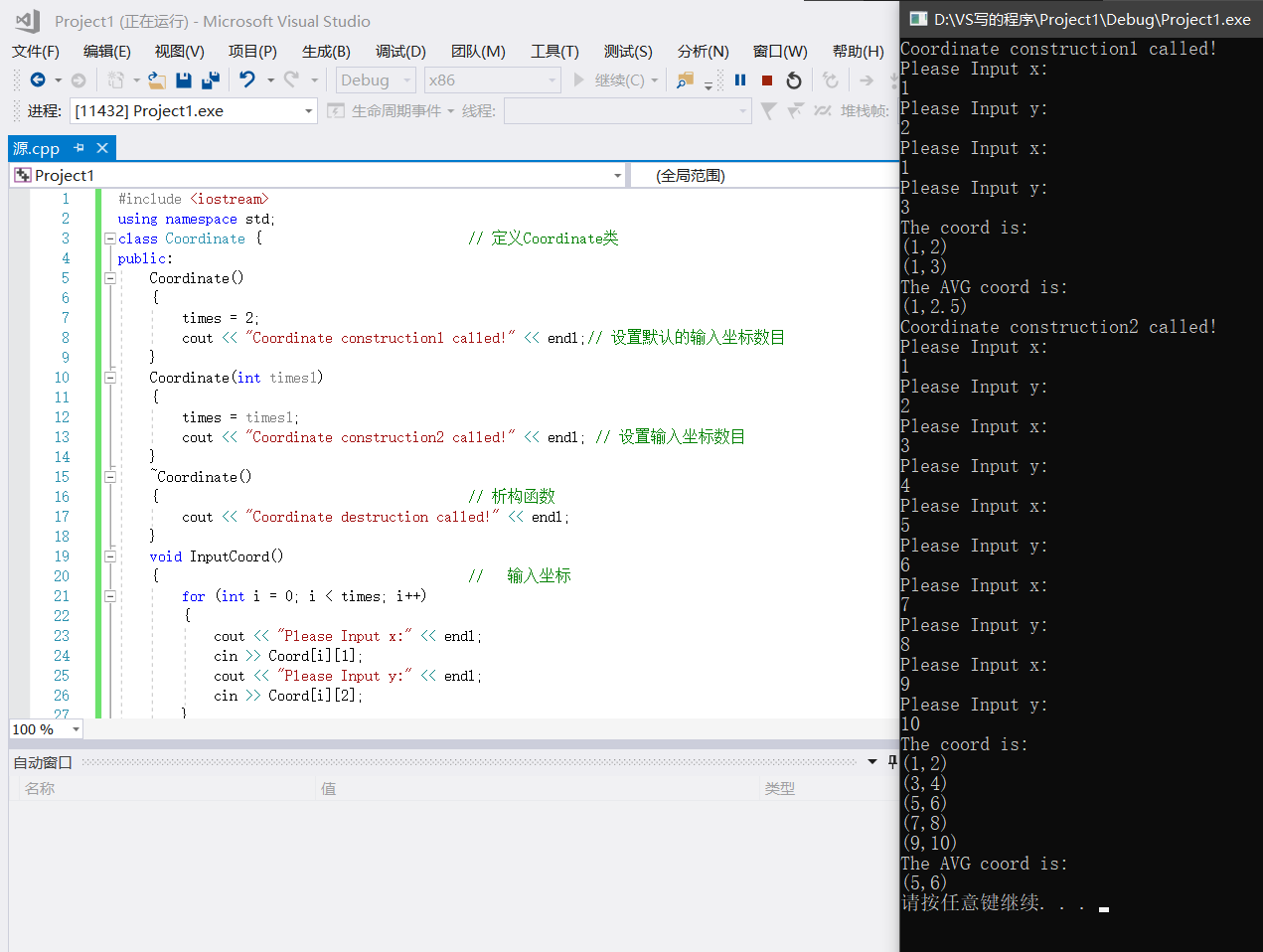
system("pause");

return 0;

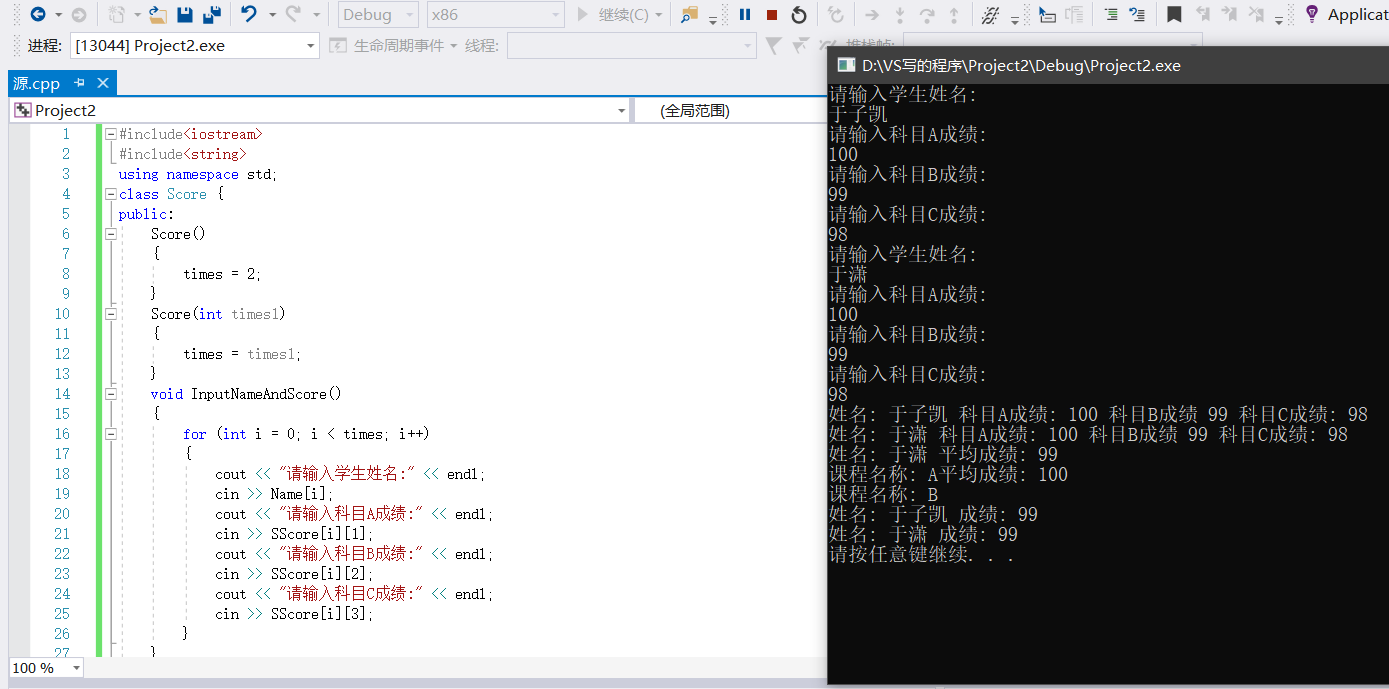
}

1. 运行结果

第一个程序



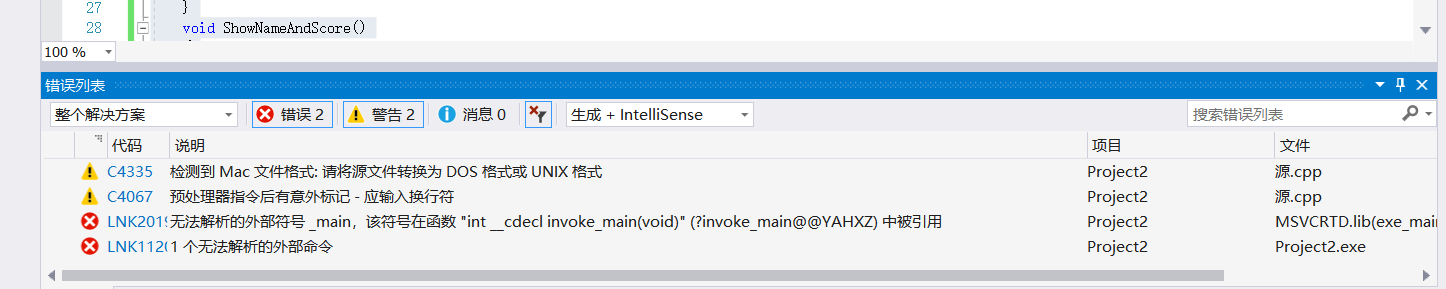
第二个程序



1. 心得和感想

第一个程序

因为上机写过了，没保留代码，只有运行结果，所以又做了一遍。运行过程中发现如下问题



通过自己查阅CSDN发现是因为自己写的字形属于MAC而不是windows又复制粘贴到文档，改成了windows的字形，解决问题。

第二个程序

成员变量的初始化顺序与声明次序有关，与初始化列表顺序无关。内置类型的成员变量在初始化列表里初始化和在构造函数体里初始化的效率是一致的。类类型的成员变量在初始化列表里初始化，才是真正的初始化，在构造函数体里仅仅是赋值。若在构造函数体里对类类型的成员变量赋值，实际上构造函数先是在执行初始化列表时先调用该类类型的默认构造函数进行初始化后，再在构造函数体里再次赋值。