## 程序代码

**实验一：**

#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()

{

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

}// 析构函数

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];

}

}// 输入坐标

void ShowCoord()

{

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

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

{

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

}

}// 显示已经输入的坐标

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;

}

// 显示输入坐标的均值}

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 y(5);

y.InputCoord();

y.ShowCoord();

y.ShowAvgCoord();

return 0;

}

**实验二：**

#include <iostream>

#include<string>

using namespace std;

class Score {

public:

Score()

{

times = 2;

}

Score(int times1)

{

times = times1;

}

~Score()

{

cout<< "Thanks for using!" << endl;

}

void InputScore()

{

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 >> score[i][0];

cout << "请输入科目B成绩：" << endl;

cin >> score[i][1];

cout << "请输入科目C成绩：" << endl;

cin >> score[i][2];

}

}

void ShowStu()

{

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

{

cout <<"姓名："<< name[i];

cout << " 科目A成绩：" << score[i][0] ;

cout << " 科目B成绩：" << score[i][1] ;

cout << " 科目C成绩：" << score[i][2] << endl;

}

}

void ShowStuAVG()

{

float avg;

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

{

cout << "姓名：" << name[i] ;

avg = 1.0\*(score[i][0] + score[i][1] + score[i][2]) / 3;

cout << " 平均成绩：" << avg << endl;

}

}

void ShowSubAVG()

{

float avgA=0,avgB=0,avgC=0;

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

{

avgA += 1.0\*score[i][0];

}

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

{

avgB +=1.0\*score[i][1];

}

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

{

avgC += 1.0\*score[i][2];

}

avgA = avgA / times;

avgB = avgB /times;

avgC = avgC /times;

cout << "课程名称A:" <<" 平均成绩：" << avgA;

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

cout << " 课程名称C:" << " 平均成绩：" << avgC<<endl ;

}

void order()

{

int i = 0, j = 0;

float avgA[100], avgB[100], avgC[100], temp;

string name1[100], name2[100], name3[100], tempname;

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

{

avgA[i] = 1.0\*score[i][0];

avgB[i] = 1.0\*score[i][1];

avgC[i] = 1.0\*score[i][2];

name1[i] = name[i];

name2[i] = name[i];

name3[i] = name[i];

}

cout << "课程名称：A" << endl;

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

{

for (j = 0; j < times - i - 1; j++)

{

if (avgA[j] < avgA[j + 1])

{

temp = avgA[j];

avgA[j] = avgA[j + 1];

avgA[j + 1] = temp;

tempname = name1[j];

name1[j] = name1[j + 1];

name1[j + 1] = tempname;

}

}

}

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

{

cout << "姓名：" << name1[i] << " 成绩：" << avgA[i] << endl;

}

cout << "课程名称：B" << endl;

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

{

for (j = 0; j < times - i - 1; j++)

{

if (avgB[j] < avgB[j + 1])

{

temp = avgB[j];

avgB[j] = avgB[j + 1];

avgB[j + 1] = temp;

tempname = name2[j];

name2[j] = name2[j + 1];

name2[j + 1] = tempname;

}

}

}

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

{

cout << "姓名：" << name2[i] << " 成绩：" << avgB[i] << endl;

}

cout << "课程名称：C" << endl;

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

{

for (j = 0; j < times - i - 1; j++)

{

if (avgC[j] < avgC[j + 1])

{

temp = avgC[j];

avgC[j] = avgC[j + 1];

avgC[j + 1] = temp;

tempname = name3[j];

name3[j] = name3[j + 1];

name3[j + 1] = tempname;

}

}

}

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

{

cout << "姓名：" << name3[i] << " 成绩：" << avgC[i] << endl;

}

}

private:

int score[100][3],times;

string name[100];

};

int main()

{

Score x;

x.InputScore();

x.ShowStu();

x.ShowStuAVG();

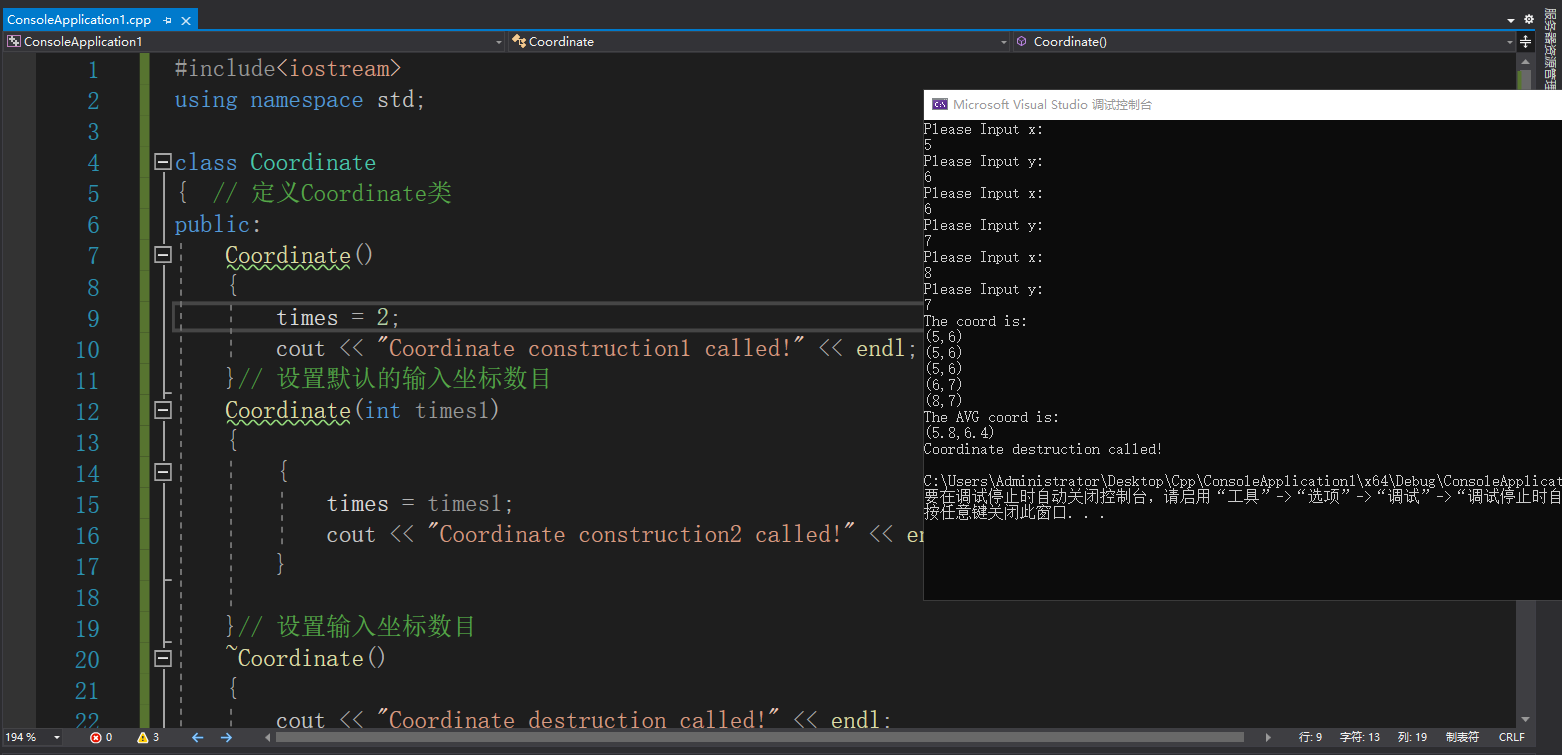
x.ShowSubAVG();

x.order();

}

## 程序结果

## 实验三：1



## 程序心得

1. 构造函数是一种特殊的成员函数，它主要用于为对象分配空间，进行初始化。构造函数的名字必须与类名相同，而不能由用户随意命名，不能有返回值且由程序在建立对象时自动执行。在实验内容2中，用到了构造函数的重载。尽管在一个类中可以包含多个构造函数，但对每一个对象而言，建立对象时只执行其中的一个构造函数。在主函数中建立对象时，要注意是否给出参数，及给出的参数对应的是哪一构造函数。
2. 构造函数在建立对象时执行，而析构函数执行的是与构造函数相反的操作，通常用于执行清理任务，如释放分配给对象的内存等。析构函数名与与类名相同，但前面必须加一个（~）。

3、C++程序设计的好处在于能够把程序分为几个部分，通过连续的调用节省大量的内存与时间，大大提高效率。我的代码还有很大的优化空间。

Copyright ©2021-2099 ZiyiLiu. All rights reserved