**实验五 类与对象**

专业班级：软件工程2401

学 号：8209240125

姓 名：朱 艺 丹

**【实验目的与要求】**

1、掌握声明类的方法，类和类的成员的概念以及定义对象的方法。

2、初步掌握用类和对象编制基于对象的程序。

3、学习检查和调试基于对象的程序。

**【实验内容】**

1、有以下程序：

#include<iostream>

using namespace std；

class Time // 定义Time类

{

public： // 数据成员为公用的

int hour;

int minute;

int sec ;

};

int main()

{

Time tl; //定义t1为Time类对象

cin>>t1.hour; //输入设定的时间

cin>>t1.minute;

cin>>t1.sec;

cout<<t1.hourl<<”：”<<t1.minute<<”：”<<t1.sec<<endl ;

return 0;

}

改写程序，要求：

(1)将数据成员改为私有的；

(2)将输入和输出的功能改为由成员函数实现；

(3)在类体内定义成员函数。

然后编译和运行程序。请分析什么成员应指定为公用的?什么成员应指定为私有的? 什么函数最好放在类中定义? 什么函数最好在类外定义?

2、分别给出如下的3个文件：

(1)含类定义的头文件student.h，

//student.h (这是头文件，在此文件中进行类的声明)

class Student //类声明

{

public: //公用成员函数原型声明

void display();

private:

int num;

char name[20];

char sex;

};

(2)包含成员函数定义的源文件student.cpp

//student.cpp 在此文件中进行函数的定义

#include <iostream>

#include”student.h” //不要漏写此行，否则编译通不过

void Student::display() //在类外定义display类函数

{

cout<<”num：”<<num<<endl;

cout<<”name：”<<name<<endl;

cout<<”sex：”<<sex<<endl;

}

(3)包含主函数的源文件main.cpp。

为了组成一个完整的源程序，应当有包括主函数的源文件：

//main.cpp 主函数模块

#include <iostream> //将类声明头文件包含进来

#include “student.h”

int main()

{

Student stud; //定义对象

Student stud1(007,”tcg”,’m’);

stud.display(); //执行stud对象的display函数

return 0;

}

请完善该程序，在类中增加一个对数据成员赋初值的成员函数set\_value。上机调试并运行。

3、需要求3个长方柱的体积，请编一个基于对象的程序。数据成员包括length(长)、width(宽)、height(高)。要求用成员函数实现以下功能：

(1)由键盘分别输入3个长方柱的长、宽、高；

(2)计算长方柱的体积；

(3)输出3个长方柱的体积。

请编程序，上机调试并运行。

4、建立一个对象数组，内放5个学生的（学号，成绩），设立一个函数max，用指向对象的指针作函数参数，在max函数中找出5个学生的最高成绩者，并输出其学号。

5、设有一描述坐标点的类Point，其私有变量x和y代表一个点的(x,y)坐标值。请编写程序实现以下功能：利用构造函数传递参数，在定义对象时将x、y坐标值初始化为（60,80）；利用公有成员函数void setPoint(int i, int j)将坐标值修改为(60+i,80+j)；利用公有成员函数display()输出修改后的坐标值。主函数中通过定义对象，验证各个函数。

**【算法分析，程序结果】**

**1.** 问题：

（1）分析什么成员应指定为公用的?

**setTime 和 showTime 应指定为公有，因为它们是外部代码与 Time类交互的方式。**

（2）什么成员应指定为私有的?

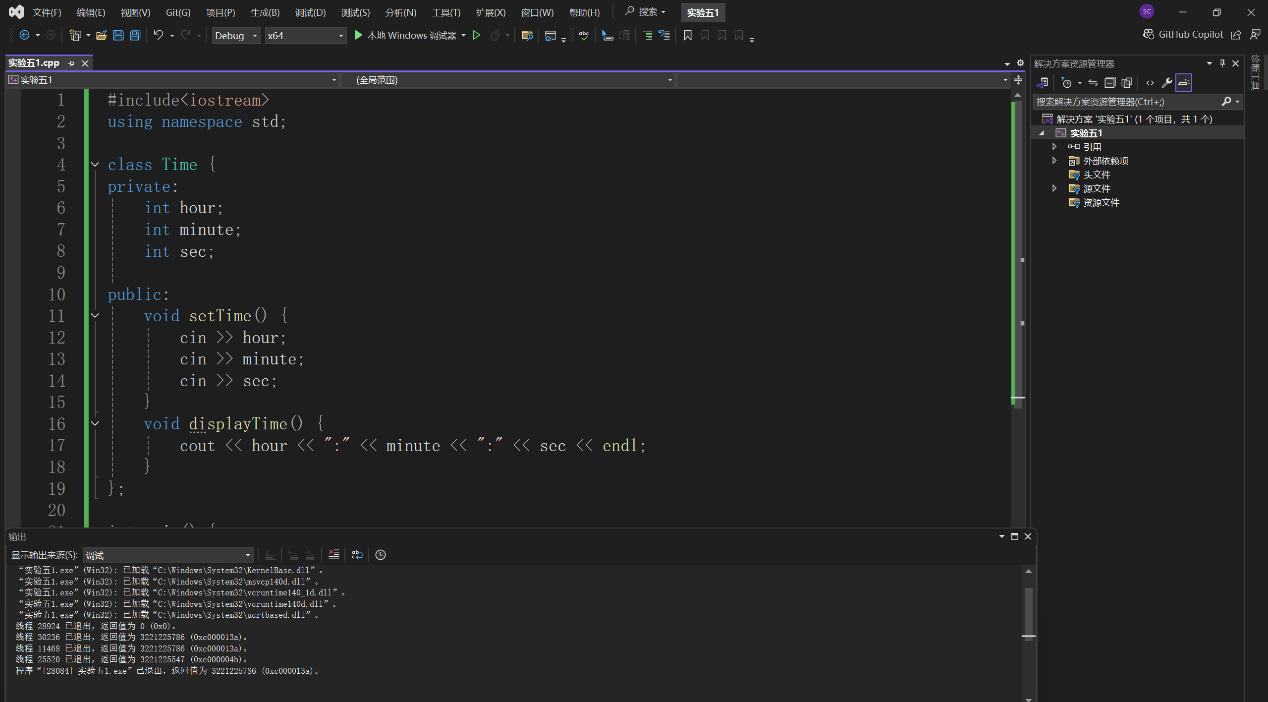
**hour、minute 和 sec 被设置为私有，以确保时间的设置和获取都通过类的方法进行，这样可以在设置时间时进行验证或其他处理。**

（3）什么函数最好放在类中定义?

**setTime 和 showTime 最好在类中定义，因为它们直接操作类的数据成员。**

（4）什么函数最好在类外定义?

**通常是那些不需要直接访问类的数据成员，执行复杂逻辑或者与类的数据成员关系不大的函数。**

#include<iostream>

using namespace std;

class Time {

private:

int hour;

int minute;

int sec;

public:

void setTime() {

cin >> hour;

cin >> minute;

cin >> sec;

}

void displayTime() {

cout << hour << ":" << minute << ":" << sec << endl;

}

};

int main() {

Time t1;

t1.setTime();

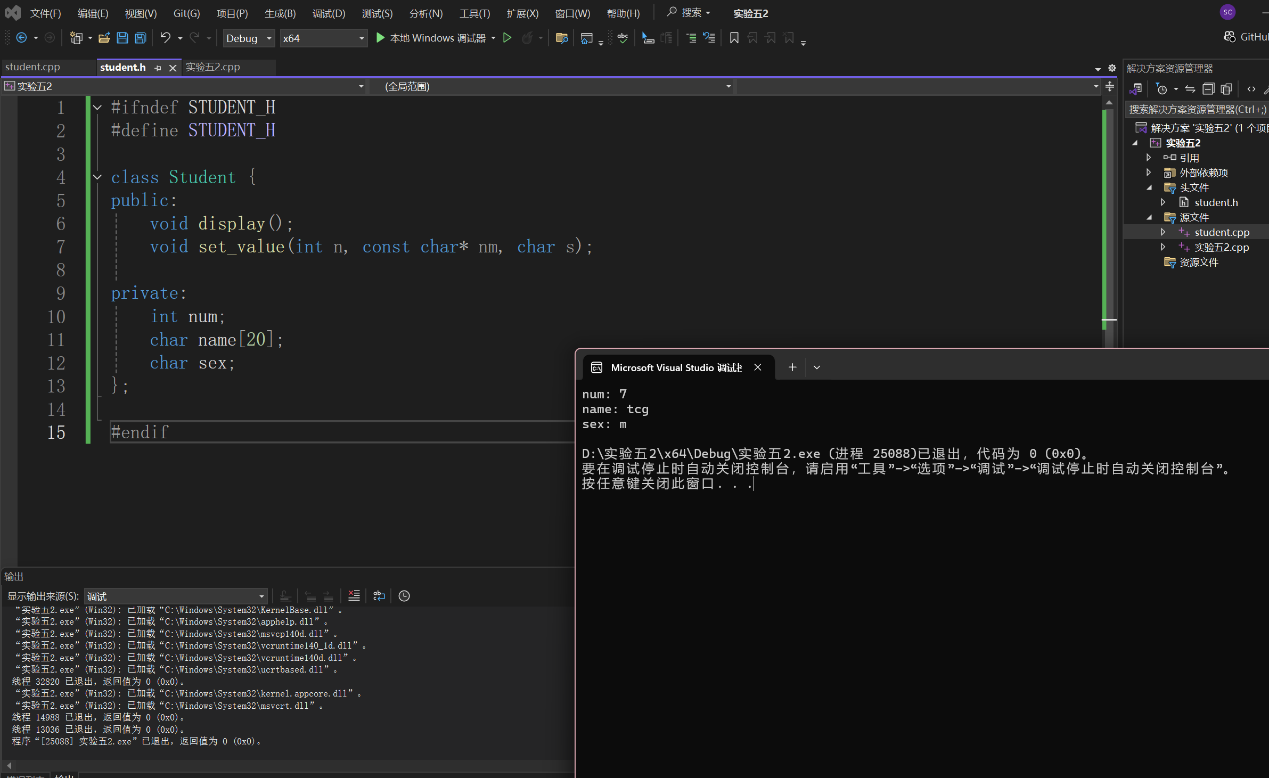
t1.displayTime();

return 0;

}

**2.**

* **student.h**

****

#ifndef STUDENT\_H

#define STUDENT\_H

class Student {

public:

void display();

void set\_value(int n, const char\* nm, char s);

private:

int num;

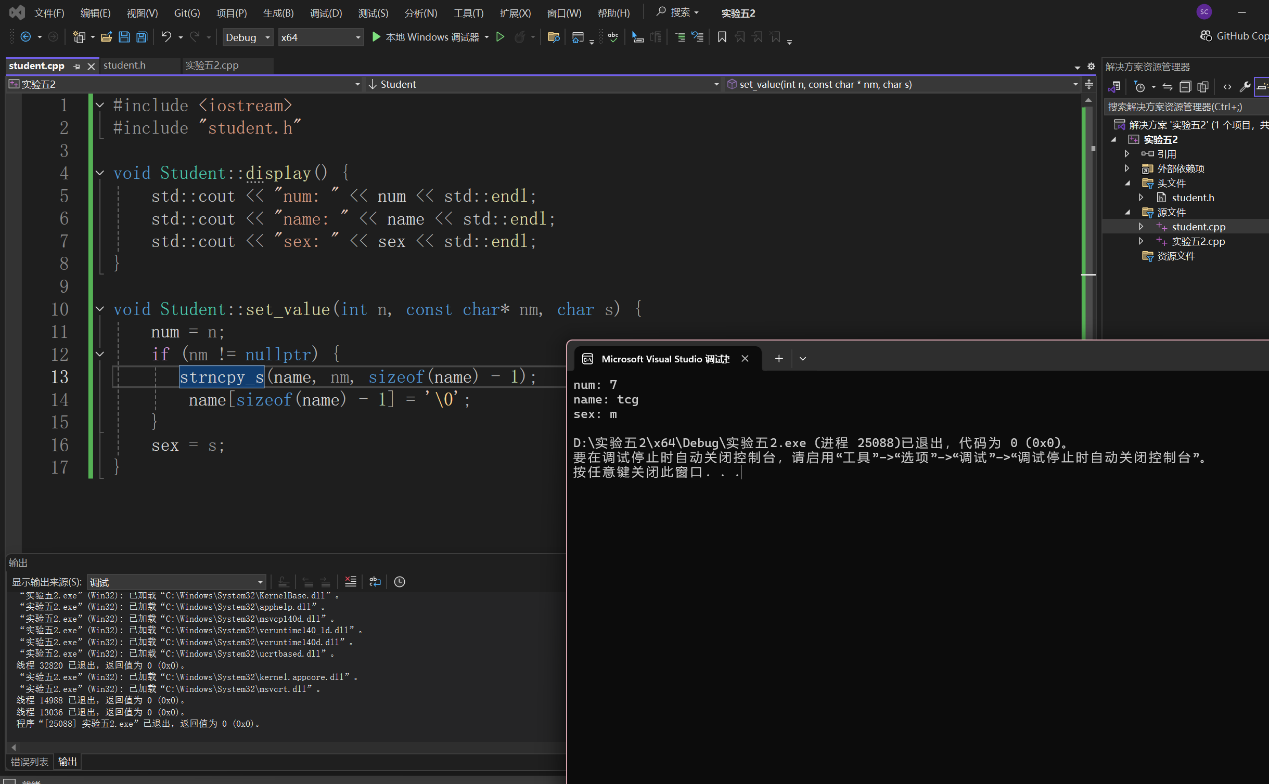
char name[20];

char sex;

};

#endif

* **student.cpp**

****

#include <iostream>

#include "student.h"

void Student::display() {

std::cout << "num: " << num << std::endl;

std::cout << "name: " << name << std::endl;

std::cout << "sex: " << sex << std::endl;

}

void Student::set\_value(int n, const char\* nm, char s) {

num = n;

if (nm != nullptr) {

strncpy\_s(name, nm, sizeof(name) - 1);

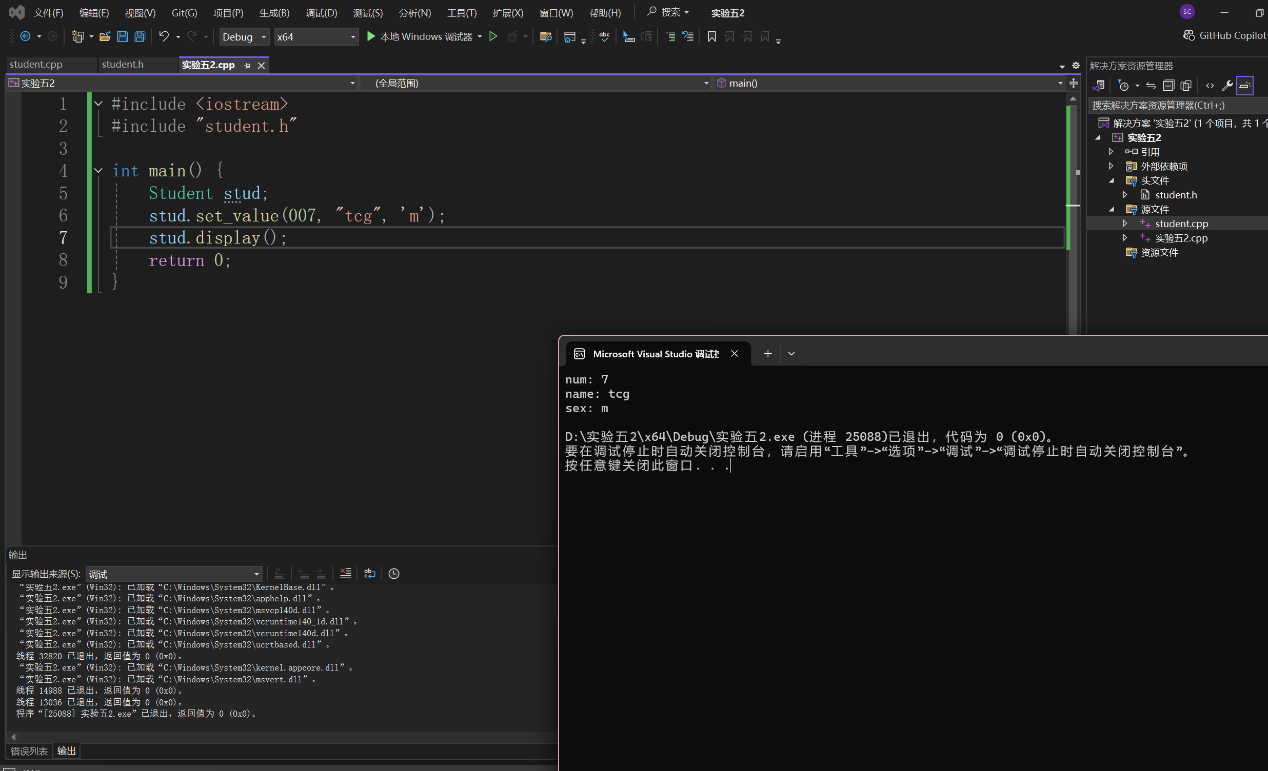
name[sizeof(name) - 1] = '\0';

}

sex = s;

}

* **实验五2.cpp(即main)**

****

#include <iostream>

#include "student.h"

int main() {

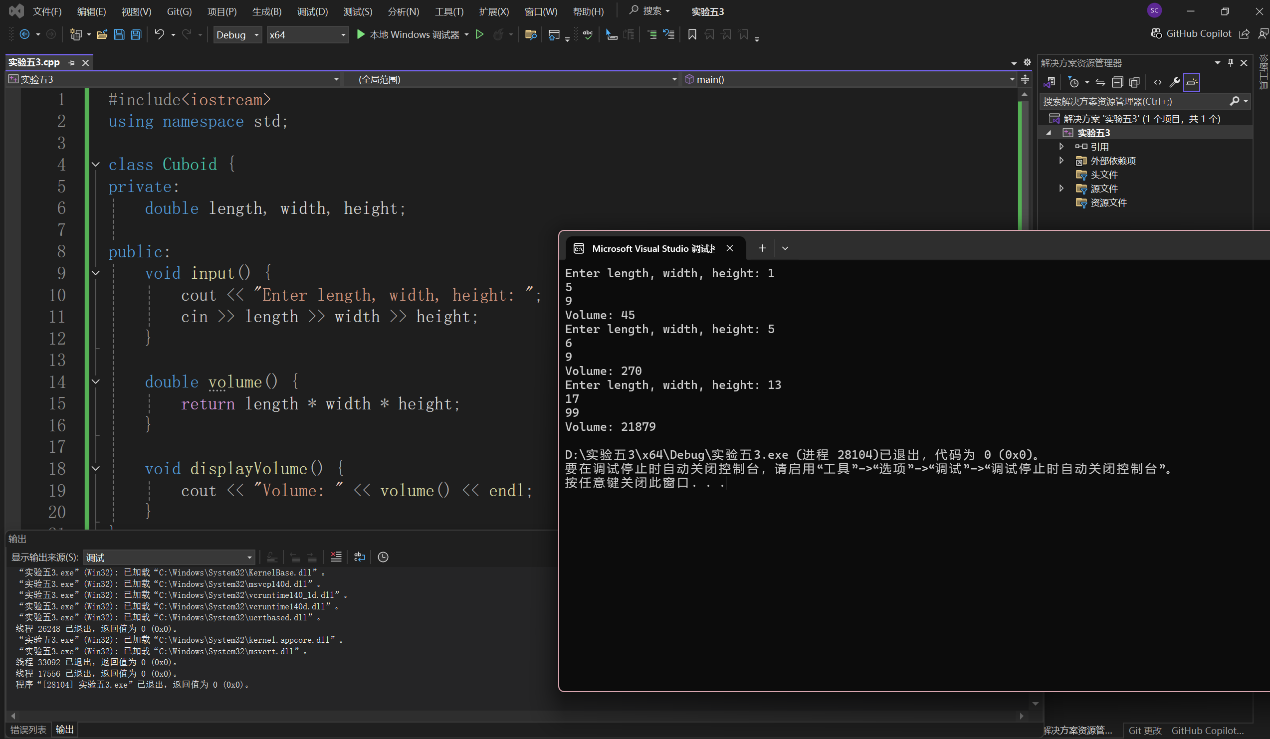
Student stud;

stud.set\_value(007, "tcg", 'm');

stud.display();

return 0;

}

**3.**

#include<iostream>

using namespace std;

class Cuboid {

private:

double length, width, height;

public:

void input() {

cout << "Enter length, width, height: ";

cin >> length >> width >> height;

}

double volume() {

return length \* width \* height;

}

void displayVolume() {

cout << "Volume: " << volume() << endl;

}

};

int main() {

Cuboid cuboids[3];

for (int i = 0; i < 3; ++i) {

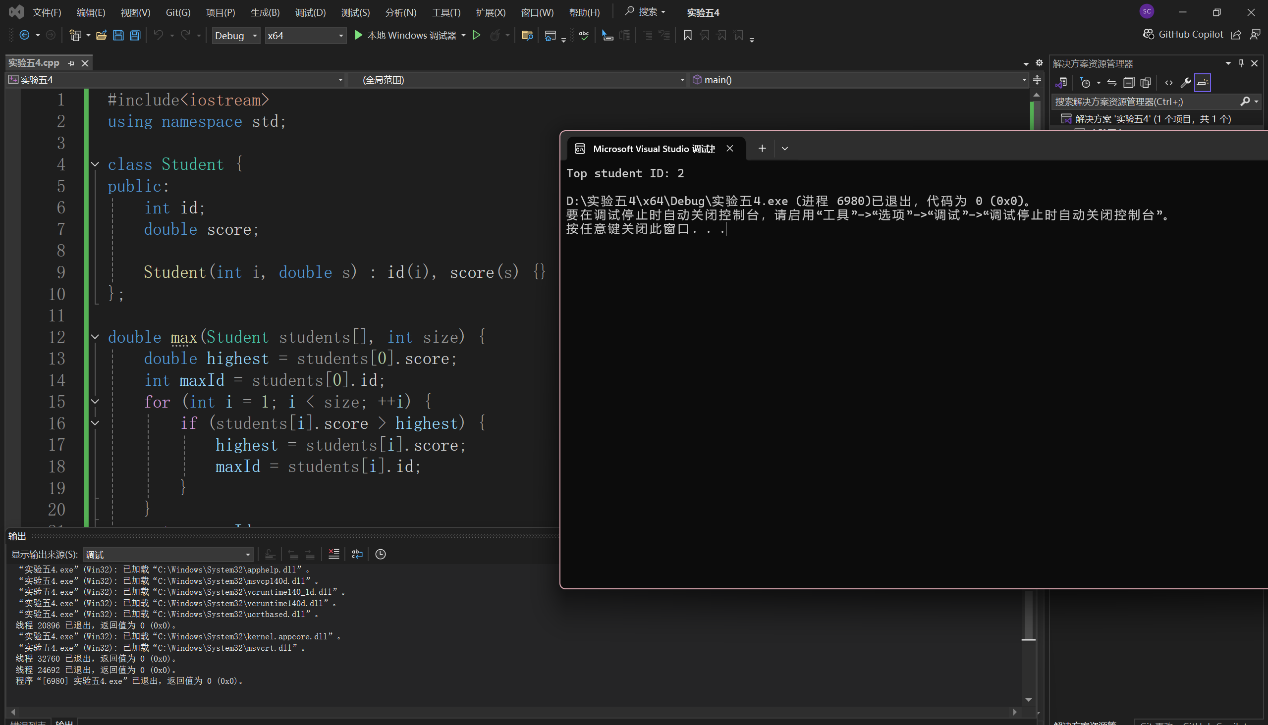
cuboids[i].input();

cuboids[i].displayVolume();

}

return 0;

}

****4.

#include<iostream>

using namespace std;

class Student {

public:

int id;

double score;

Student(int i, double s) : id(i), score(s) {}

};

double max(Student students[], int size) {

double highest = students[0].score;

int maxId = students[0].id;

for (int i = 1; i < size; ++i) {

if (students[i].score > highest) {

highest = students[i].score;

maxId = students[i].id;

}

}

return maxId;

}

int main() {

Student students[5] = {

Student(1, 85.5),

Student(2, 92.0),

Student(3, 78.5),

Student(4, 91.0),

Student(5, 88.0)

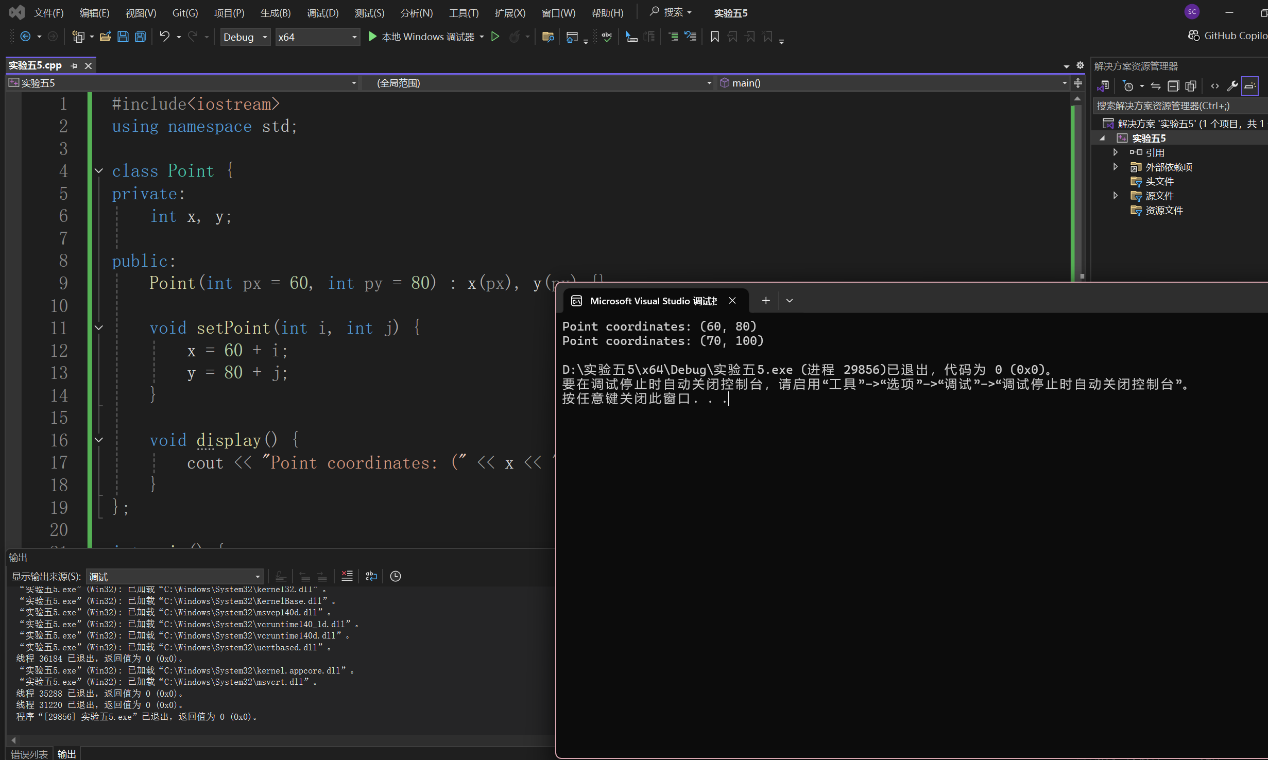
};

int topStudentId = max(students, 5);

cout << "Top student ID: " << topStudentId << endl;

return 0;

}

**5.**

#include<iostream>

using namespace std;

class Point {

private:

int x, y;

public:

Point(int px = 60, int py = 80) : x(px), y(py) {}

void setPoint(int i, int j) {

x = 60 + i;

y = 80 + j;

}

void display() {

cout << "Point coordinates: (" << x << ", " << y << ")" << endl;

}

};

int main() {

Point p;

p.display();

p.setPoint(10, 20);

p.display();

return 0;

}

**【遇到的问题与解决方法】**

1. **没有使用安全函数；解决方法：在处理字符串时，使用安全版本的函数（`strncpy\_s`等）可以减少缓冲区溢出的风险**
2. **没有检查空指针；解决方法：在复制字符串之前检查源字符串是否为`nullptr`，防止潜在的空指针解引用错误。**
3. **续问题1：解决方法：手动添加字符串结束符：在使用`strncpy\_s`后，手动添加字符串结束符`\0`确保字符串正确终止。**

**【体会】**

**编程是一个不断学习和适应新知识的过程。遇到问题时，查阅文档、搜索解决方案和学习最佳实践是非常重要的。在这次实验我学到了代码审查是一个重要的步骤，可以帮助发现和修正错误，通过审查代码，我发现了使用`strncpy`可能带来的问题，认真对待编译器的警告信息，编译器警告提示了`strncpy`的不安全性， 所以我使用了`strncpy\_s`减少缓冲区溢出的风险。**