**实验五 类与对象**

**【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?

·公有：setTime()、displayTime()等构造函数

私有：hour,min,sec等私密数据

类内定义的函数：简单的，仅涉及内部数据操作的成员函数，或者访问器函数应当放在类内。

类外定义的函数： 函数实现较复杂，包含较多逻辑或与外部环境交互的函数应当放在类外定义，以提高代码的组织性和可维护性

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；

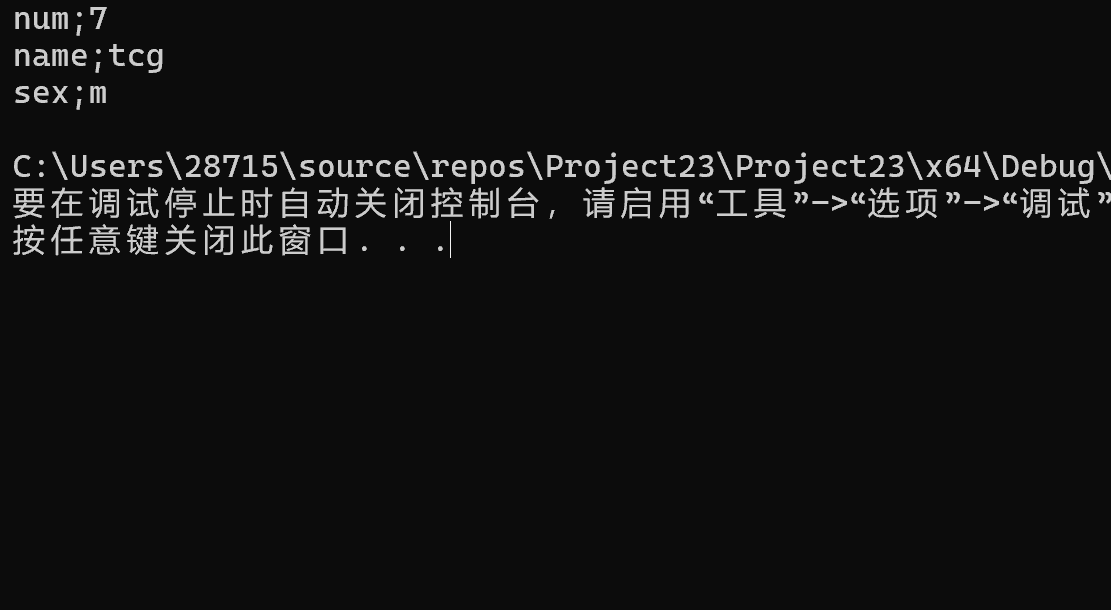
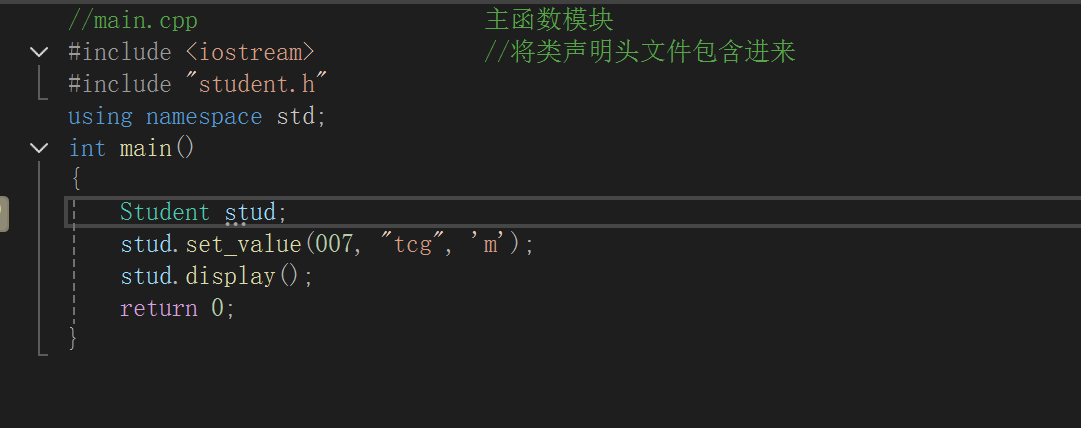
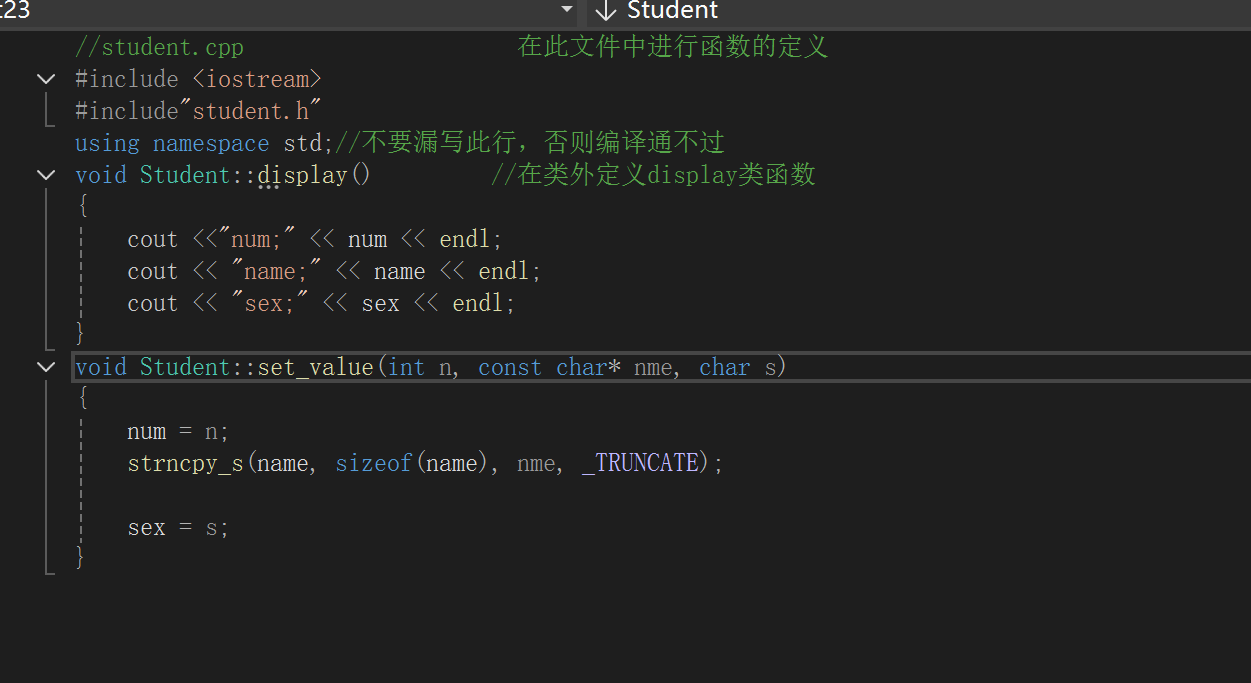
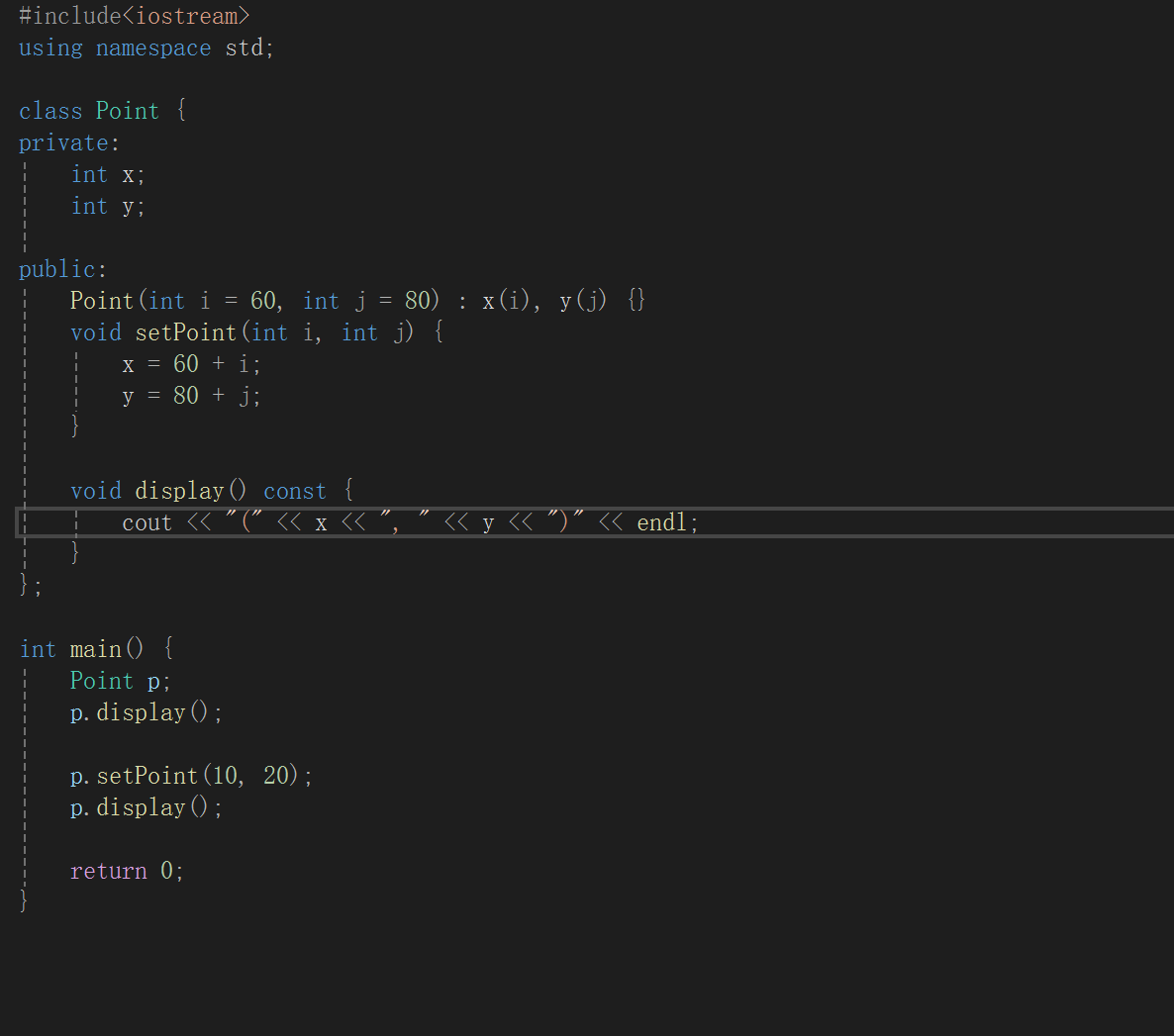
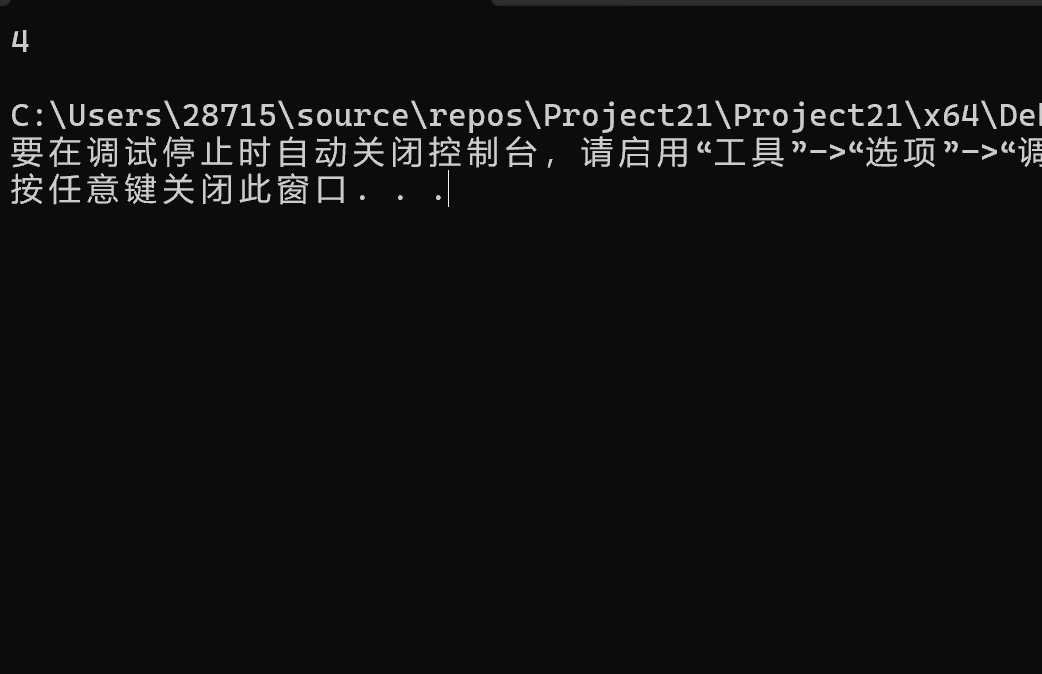
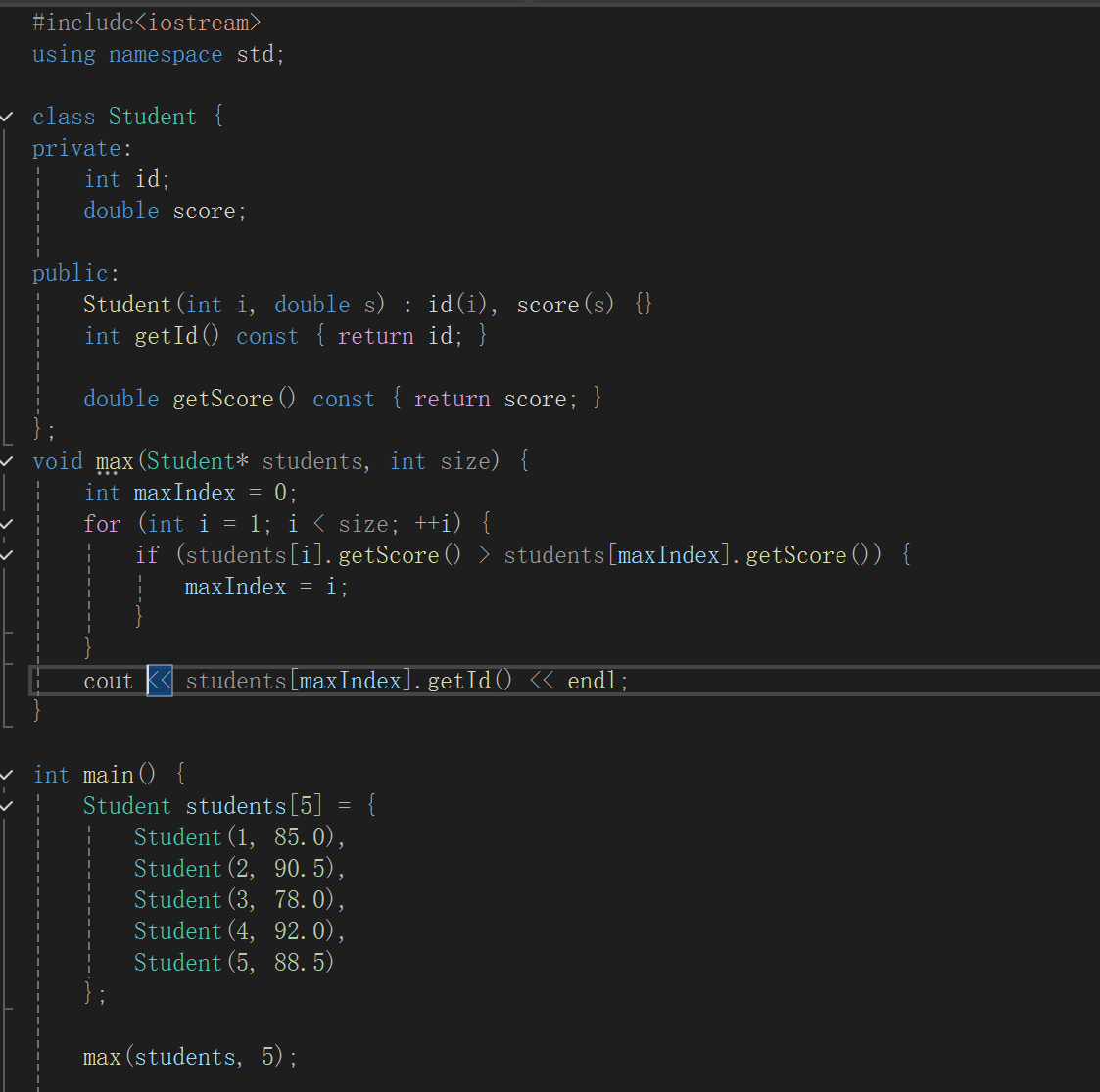
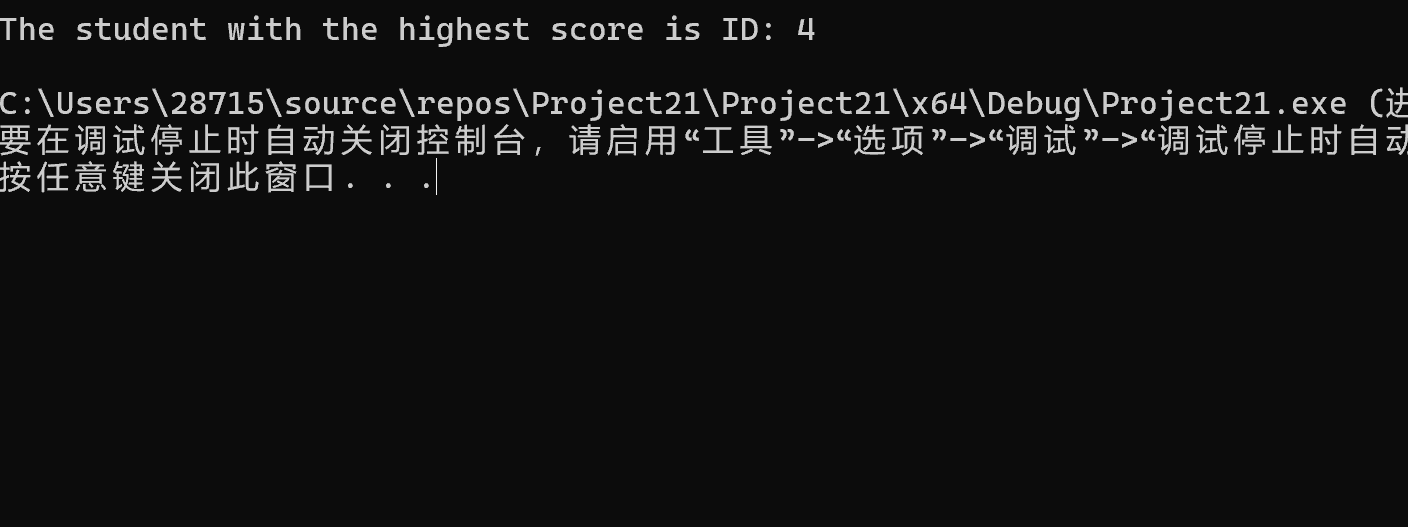
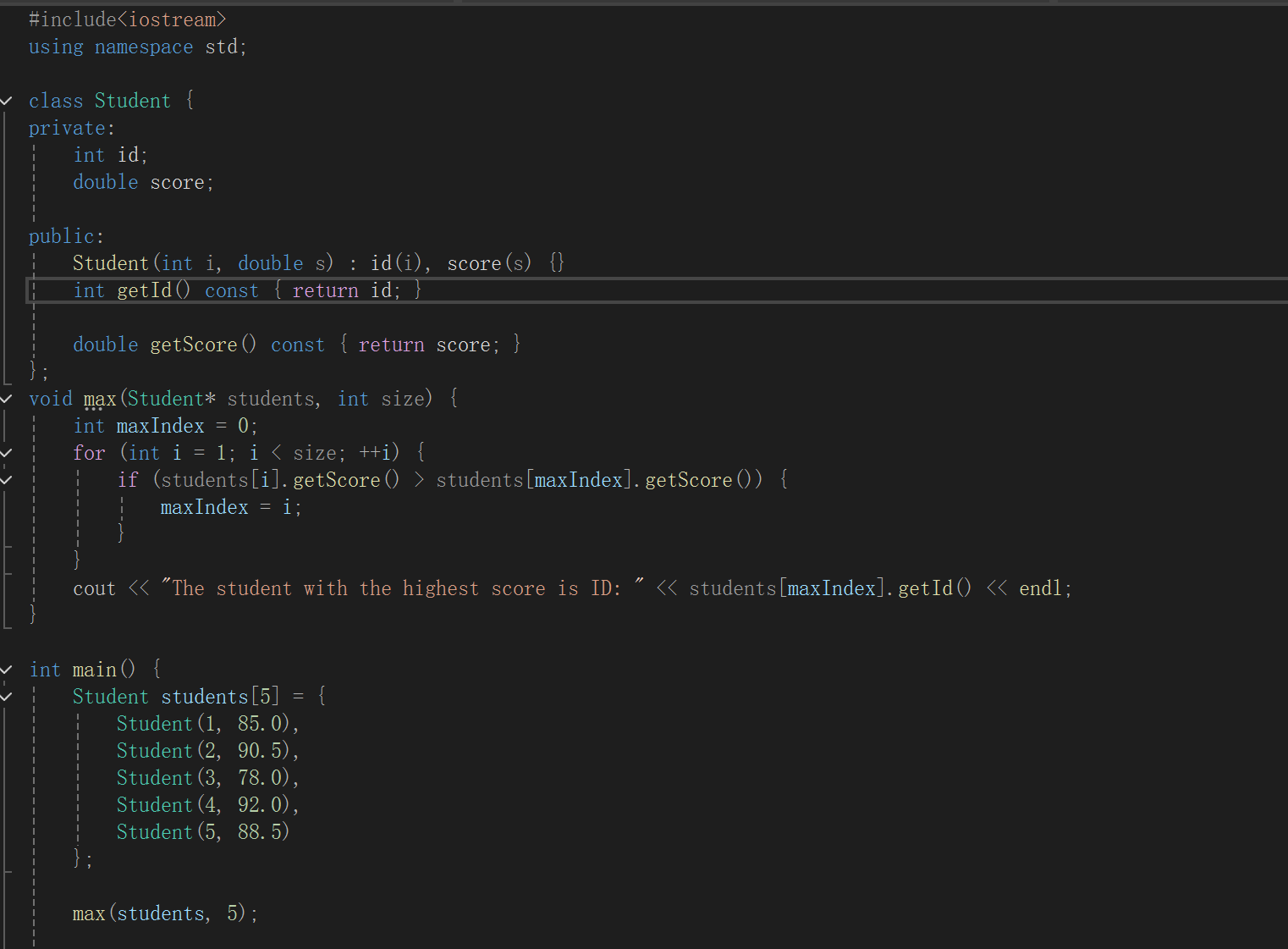
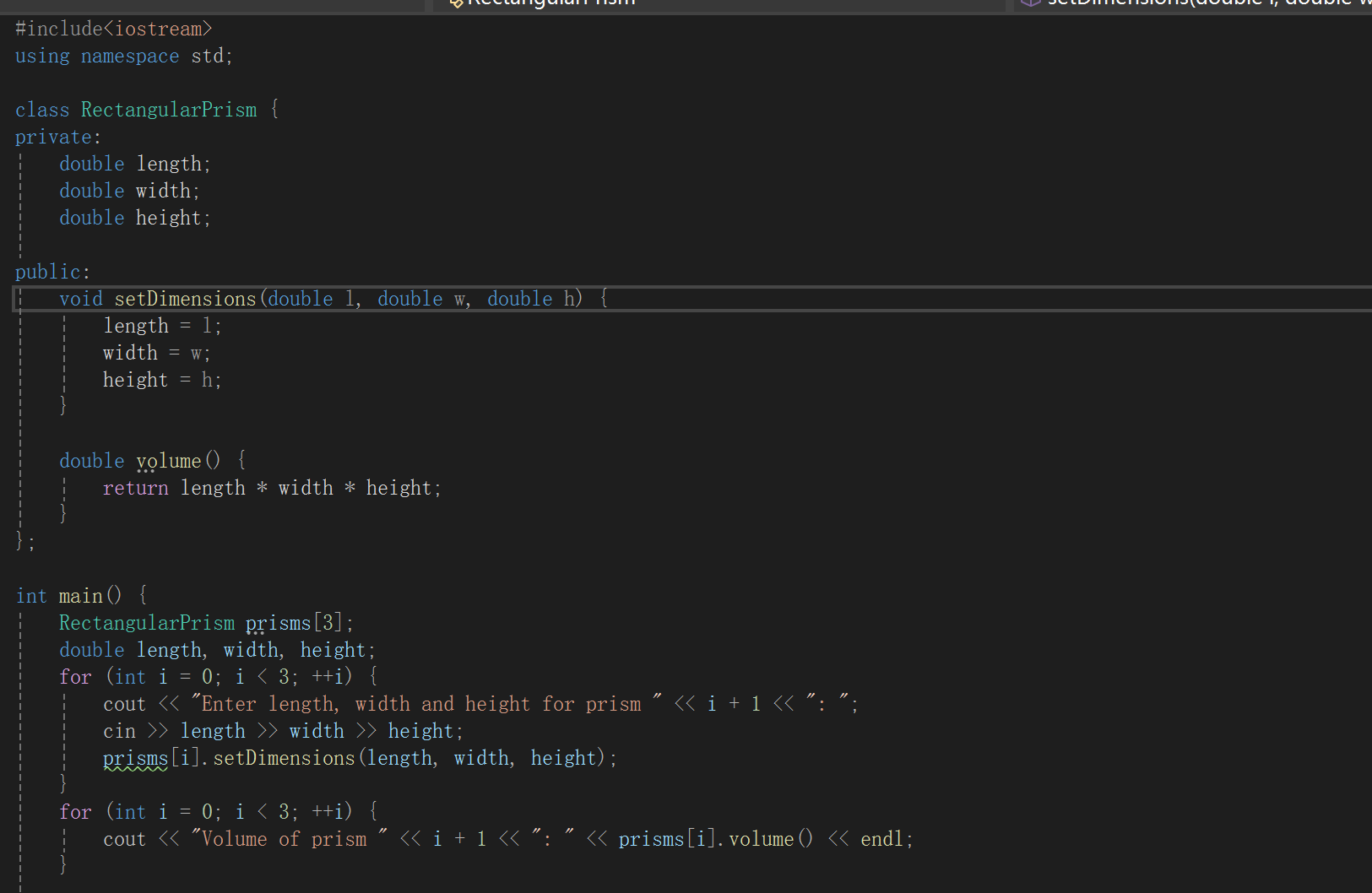
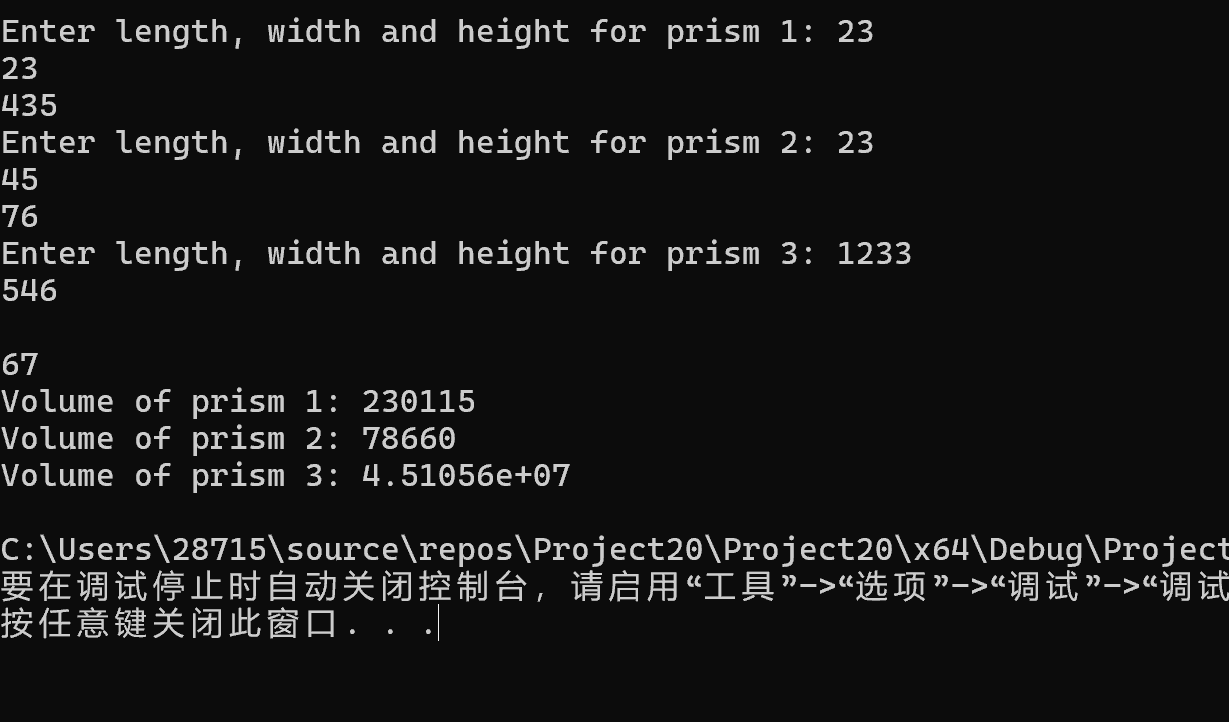
(3)输出3个长方柱的体积。

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】



【遇到的问题和解决办法】

首先，头文件的路径和引号格式有可能会引起错误，特别是在包含 "student.h" 时。此外，字符串操作函数 strncpy 在 student.cpp 中使用时，未包含 #include <cstring>，这也导致了编译失败。解决办法是在 student.cpp 文件中添加了正确的头文件引用。最后，也是卡的最久的一个部分，就是在student.cpp中对于strncpy函数的使用。因为图省事懒得改，我曾试图用#define \_CRT\_SECURE\_NO\_WARNINGS来取消对安全威胁的警告，虽然没有出什么问题，但也打消了我偷懒的想法，也让我意识到程序安全的重要性。最终还是查资料用常规方法解决了。

【体会】

通过解决这些问题，我深刻体会到在编写和调试 C++ 程序时，细节至关重要。编译错误往往源于一些看似微小的细节问题，例如头文件路径、函数声明和字符串操作的使用。如果忽视了这些问题，可能会浪费大量时间在调试上。特别是在处理文件包含和库函数时，正确引入头文件和库是成功编译的基础。此外，编写清晰、可维护的代码，确保每个函数的作用明确，能帮助在遇到问题时更容易定位错误。通过这次练习，我更加理解了 C++ 中封装和模块化的概念，如何通过类来组织程序结构，合理使用构造函数、成员函数等特性。