**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;

}

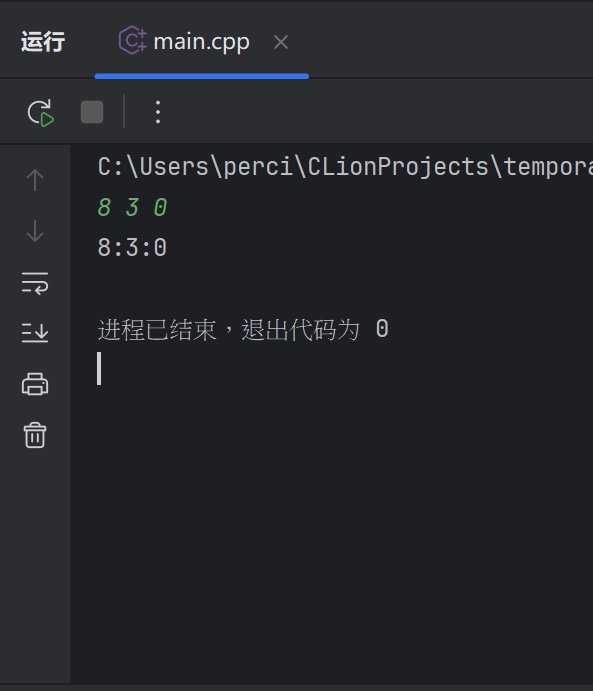
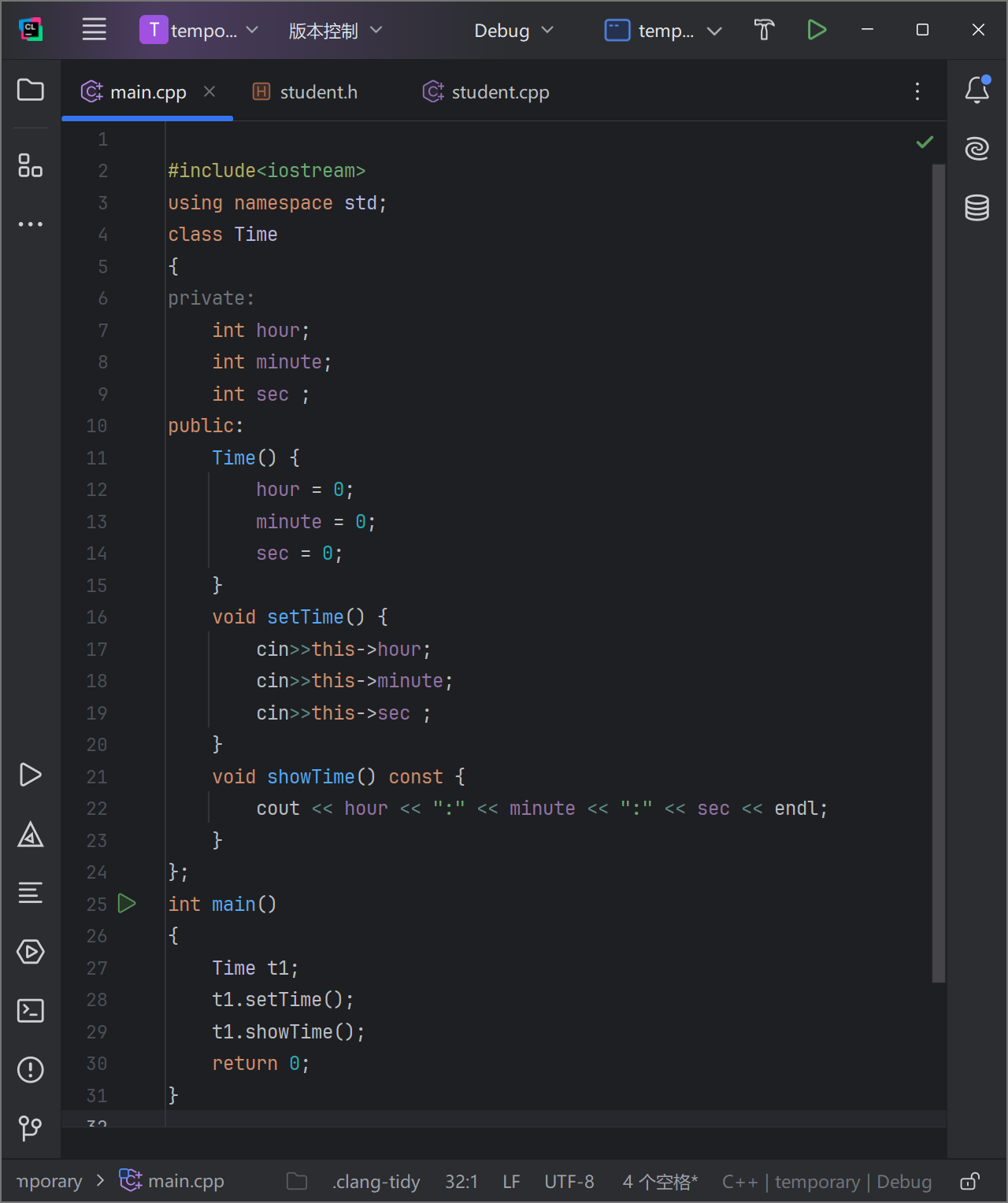
改写程序，要求：

(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?



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()

{

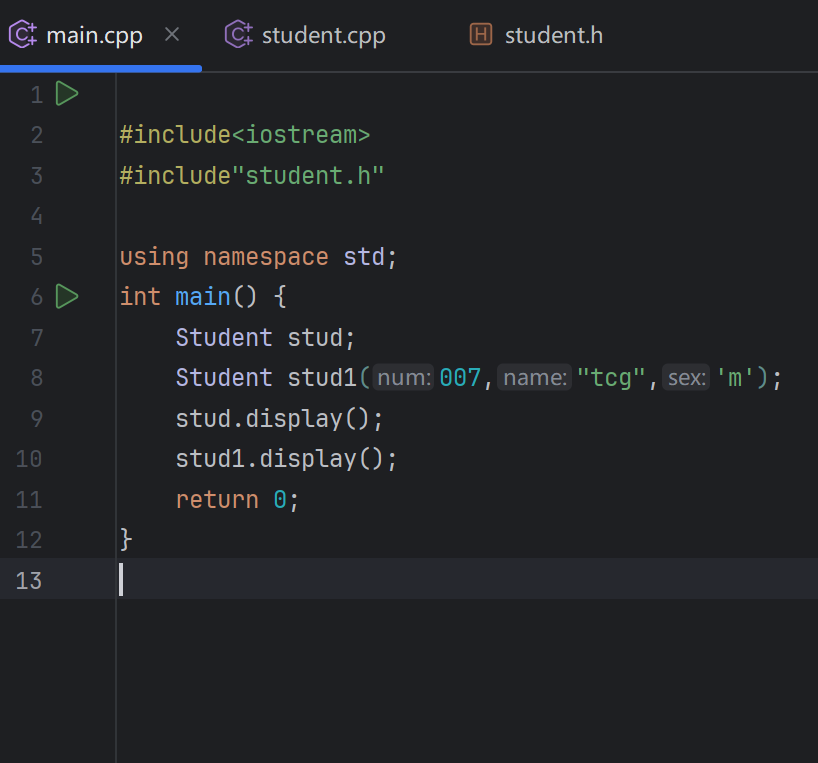
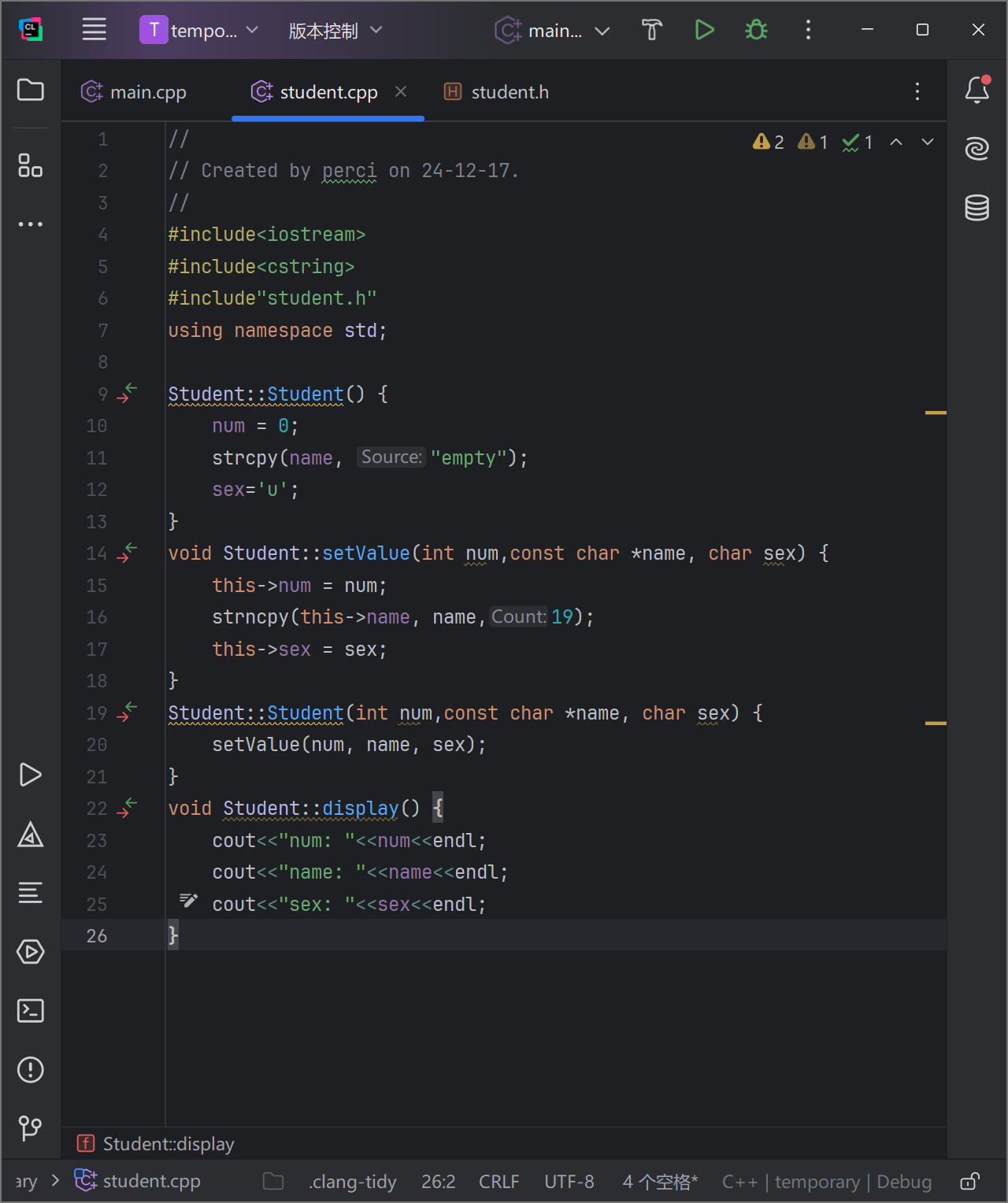
Student stud; //定义对象

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。

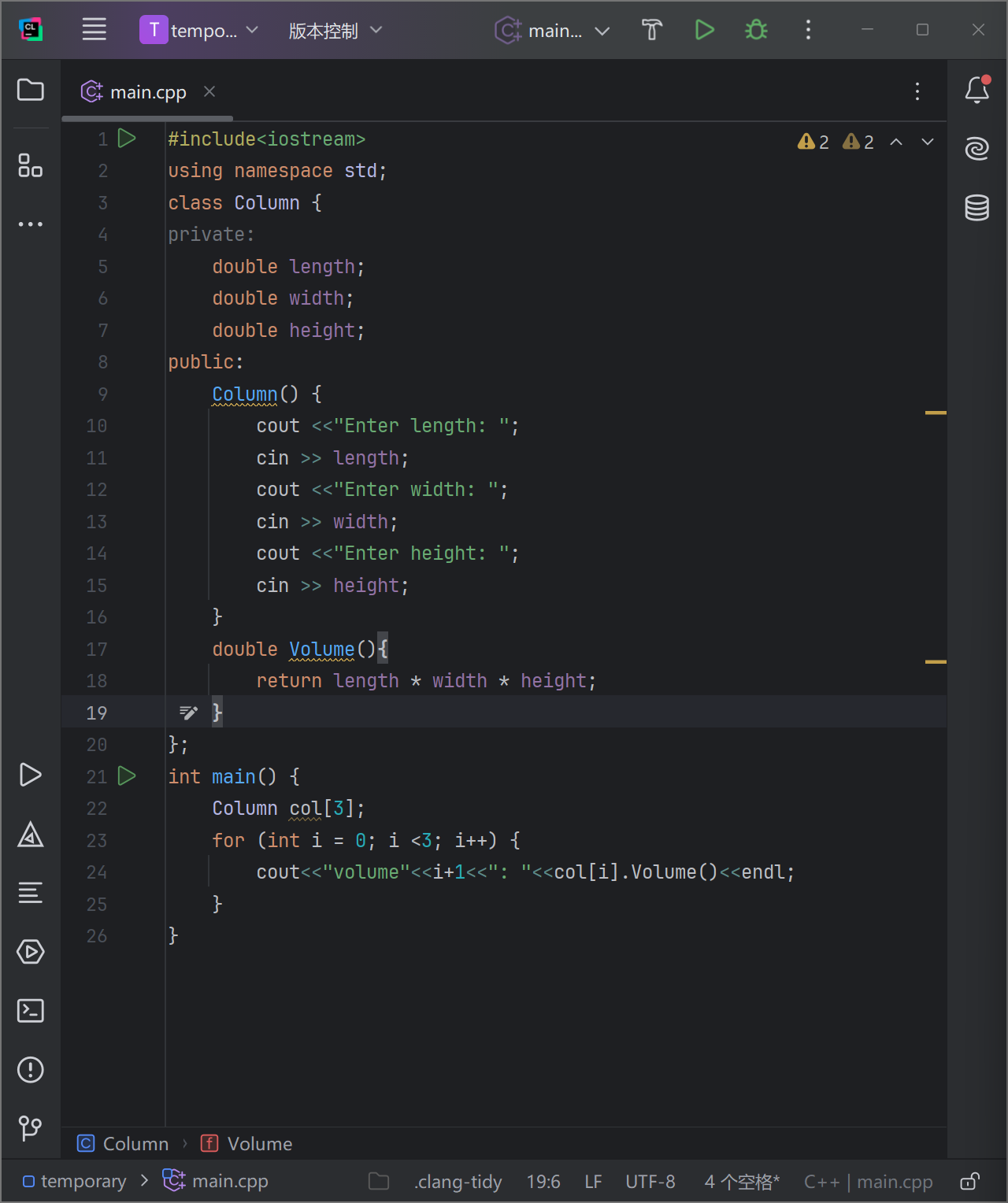
3、需要求3个长方柱的体积，请编一个基于对象的程序。数据成员包括length(长)、width(宽)、height(高)。要求用成员函数实现以下功能：

(1)由键盘分别输入3个长方柱的长、宽、高；

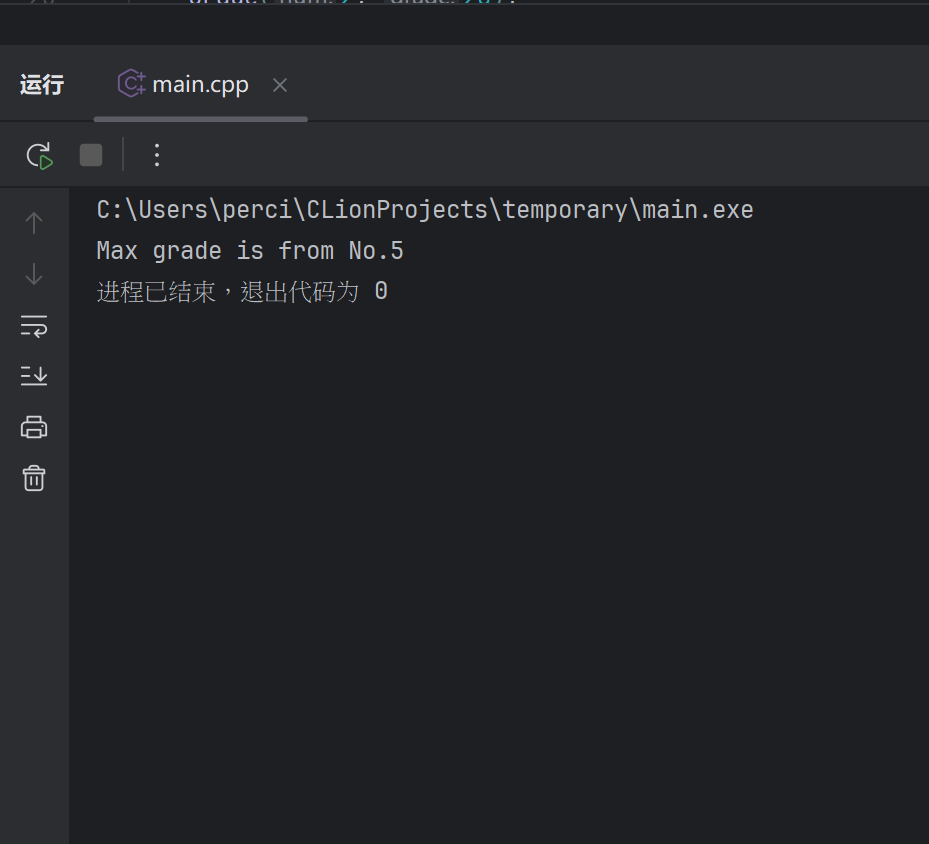
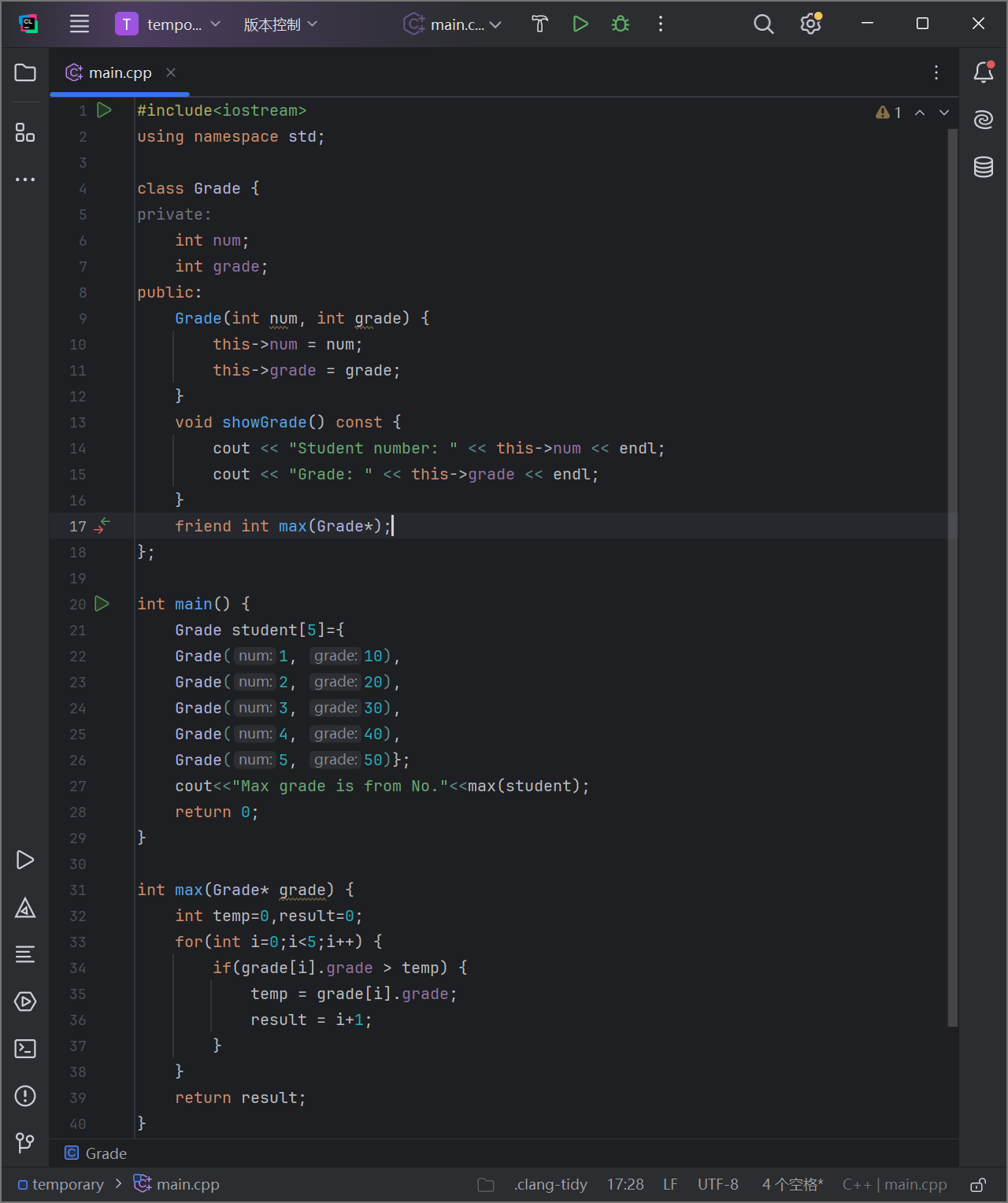
(2)计算长方柱的体积；

(3)输出3个长方柱的体积。

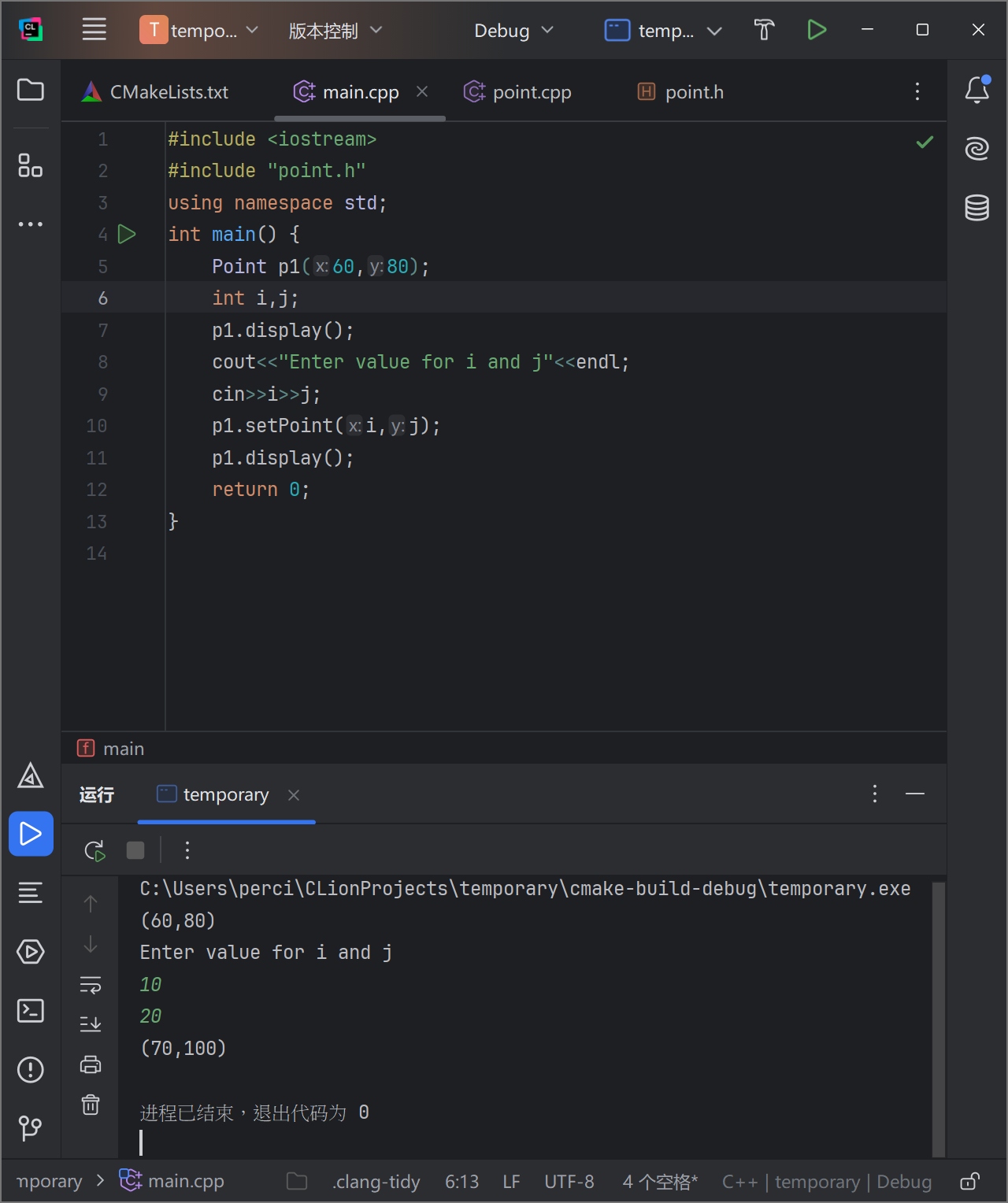
请编程序，上机调试并运行。

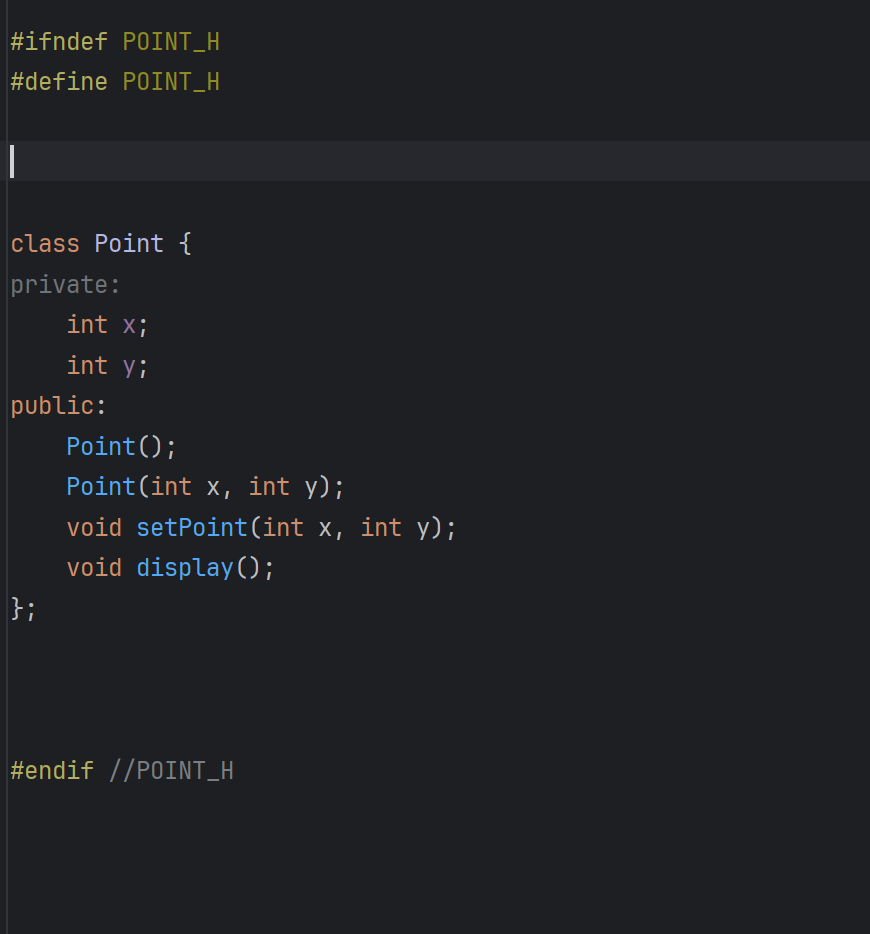


4、建立一个对象数组，内放5个学生的（学号，成绩），设立一个函数max，用指向对象的指针作函数参数，在max函数中找出5个学生的最高成绩者，并输出其学号。



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。





**问题与解决方案**

 **strncpy 和 strcpy 的区别**  
学到了 strncpy 和 strcpy 的主要区别在于拷贝时是否限制长度以及是否自动添加字符串的终止符。strcpy 更简单但不安全，容易导致缓冲区溢出；strncpy 更安全但需要手动处理终止符问题。

 **Student 类构造函数的 undefined reference 问题**  
遇到了链接器无法找到构造函数定义的问题，可能的原因包括：未实现构造函数、实现的文件未正确链接到项目中、命名空间或拼写错误等。

**体会**

今天的问题体现了编程学习过程中对基础知识掌握的重要性。C++ 是一门强大的语言，但它对细节非常严格，稍有疏忽就可能引发编译或链接问题。

**收获的体会：**

1. **基础知识是解决问题的关键**：  
   比如 strncpy 的安全特性和构造函数定义的必要性，只有理解原理才能快速定位问题。
2. **错误信息是排查问题的重要线索**：  
   链接器报错（undefined reference）提示了问题的根源——构造函数未实现或未正确链接。从这里出发一步步排查，非常高效。
3. **注重代码规范和细节**：  
   比如构造函数的定义和声明要一致，拼写和命名空间不能出错。这些“细节”在小项目中可能容易忽视，但在实际工程中很关键。
4. **调试和构建流程的重要性**：  
   学会清理并重新构建项目，确保所有文件都被正确编译，能避免很多不必要的错误。