**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。

1. 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。

【程序设计，结果展示及问题回答】

1. （1）更改以后的程序如下：

#include<iostream>

using namespace std;

class Time // 定义Time类

{

private: // 数据成员为私有的

int hour;

int minute;

int sec;

public: // 函数是公用的

void out(); //输出

void in(); //输入

};

void Time::out() {

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

}

void Time::in() {

cout << "请按顺序输入小时、分钟、秒数" << endl;

cin >> hour >> minute >> sec;

cout << '\n';

}

int main()

{

Time t1; //定义t1为Time类对象

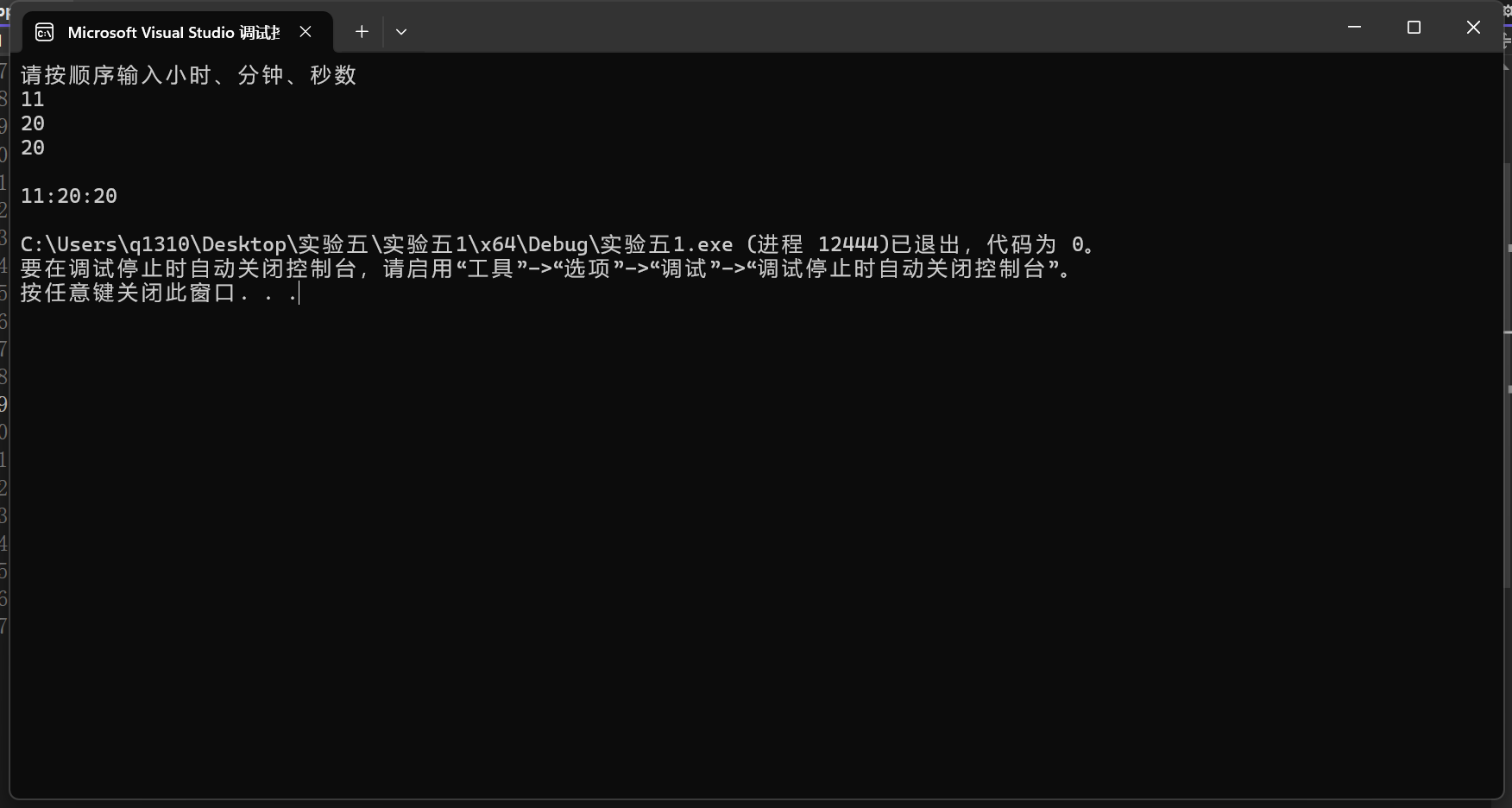
t1.in(); //.使用函数

t1.out();

return 0;

}

（2）结果如下



1. 问题及回答

问题：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?

回答：

一、成员指定为公用（public）或私有（private）的考量

（一）应指定为公用（public）的成员

接口相关成员  
类需要对外提供的操作方法或者能够被外部直接访问获取的属性，通常会被指定为 public。例如，对于一个表示图形的类，如果外部需要获取图形的面积、周长等关键信息，那相应计算并返回这些值的成员函数（方法）就适合设为 public，方便外部代码调用以获取图形的相关特征数据。再比如，一个实现栈数据结构的类，push（入栈）、pop（出栈）、top（获取栈顶元素）这些操作方法一般设为 public，因为它们是供外部使用者操作栈这个数据结构的基本接口。

需要被继承类访问的成员（在合适的继承体系下）  
当设计类的继承层次结构时，如果基类中的某些成员希望被子类直接访问并复用，那么可以将它们设为 public。这样，子类在扩展或修改基类行为时，能够方便地利用这些已有的公共成员来构建新的功能逻辑，符合面向对象中代码复用和扩展的设计理念。

（二）应指定为私有（private）的成员

类内部实现细节相关成员  
那些仅在类的内部实现各种方法时起辅助作用，而不应该被外部代码直接访问和操作的属性和方法，往往设为 private。比如，在上述图形类中，可能有一些用于内部计算图形顶点坐标、边长等中间变量或者辅助计算的函数，它们对于外部使用者来说没有直接意义，并且如果外部随意修改可能会破坏类内部的逻辑一致性，所以要设为 private 来隐藏这些实现细节，保证类的封装性。

用于维护类状态的关键属性  
类中一些控制自身状态的属性，像表示某个计数器的变量（例如记录一个对象被访问的次数等），如果任由外部代码随意更改，可能导致类处于不正确的状态，影响后续的功能执行，所以通常设为 private，并通过定义 public 的接口函数（例如get访问器、set修改器函数等，且在修改器函数中可以加入合法性校验逻辑）来有控制地访问和修改这些属性。

二、函数在类内或类外定义的考量

（一）最好放在类中定义的函数（成员函数）

与类的状态紧密相关的操作函数  
如果函数的执行依赖于类的具体状态（即类的成员变量的值），并且是对类自身进行操作、改变类状态或者基于类状态返回相关结果的，那么这类函数适合在类中定义为成员函数。例如，对于一个银行账户类，deposit（存款）、withdraw（取款）函数需要操作账户余额这个成员变量，根据当前余额情况进行相应业务逻辑处理，它们天然就属于类的内部操作，所以作为类的成员函数定义在类内是合适的。

体现类对象行为特征的函数  
每个类通常代表一种特定概念或者实体，那些能够体现该类对象特有行为的函数应该在类中定义。比如，对于一个动物类，如果有eat（进食）、move（移动）等行为函数，它们是和具体动物对象相关联的，不同动物子类（如猫、狗等）会有不同的实现方式，把它们定义在类中便于通过继承、多态等面向对象机制来灵活实现和扩展这些行为。

（二）最好在类外定义的函数

通用的辅助函数（与类本身关联性不强）  
有些函数只是执行一些通用的、不依赖于特定类状态的逻辑计算，它们可以在类外定义，作为独立的普通函数供需要的地方调用。例如，一个数学计算相关的函数，像计算两个数的最大公约数，它本身和某个具体的业务类没有直接关联，在类外定义更符合其通用性特点，方便在不同的代码模块中复用，而不用把它强行捆绑到某个类里面作为成员函数。

对多个类进行操作的函数（不属于任何单一类的核心行为）  
当存在函数需要对多个不同类的对象进行交互操作，且其功能并非是某个类专属的核心行为时，更适合在类外定义。比如，有一个函数用于比较两个不同类型图形类对象（如圆形和矩形）的面积大小，它不属于圆形类或者矩形类内部核心要定义的成员函数，放在类外定义能更清晰地体现其跨类操作的性质，并且避免破坏各个类自身的封装性和单一职责原则。

（查询资料得知）

2.(1)程序如下

含类定义的头文件student.h，

#pragma once

//student.h (这是头文件，在此文件中进行类的声明)

class Student //类声明

{

public:

void display();

void set\_value(int num1, const char name1[], char sex1); // 新增的设置值的成员函数声明

Student(int a = 0, const char name1[2]="张三", char c = 'm') {

num = a;//默认为零

strcpy\_s(name, name1);//默认张三

sex = c;//默认m

}

~Student()

{

}

private:

int num;

char name[20];

char sex;

};

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<cstring>

#include "student.h" //不要漏写此行，否则编译通不过

// 在类外定义display类函数

// student.cpp 在此文件中进行函数的定义

#include <iostream>

#include<cstring>

#include "student.h" //不要漏写此行，否则编译通不过

// 在类外定义display类函数

void Student::display()

{

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

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

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

}

// 定义set\_value函数，用于给类的数据成员赋初值

void Student::set\_value(int num1, const char name1[], char sex1)

{

num = num1;

strcpy\_s(name, name1);

sex = sex1;

}

包含主函数的源文件main.cpp。

//main.cpp 主函数模块

#include <iostream>

#include<cstring>

#include "student.h"

int main()

{

Student stud;

Student stud1;

// 使用set\_value函数给stud1对象的数据成员赋初值

stud1.set\_value(007, "tcg", 'm');

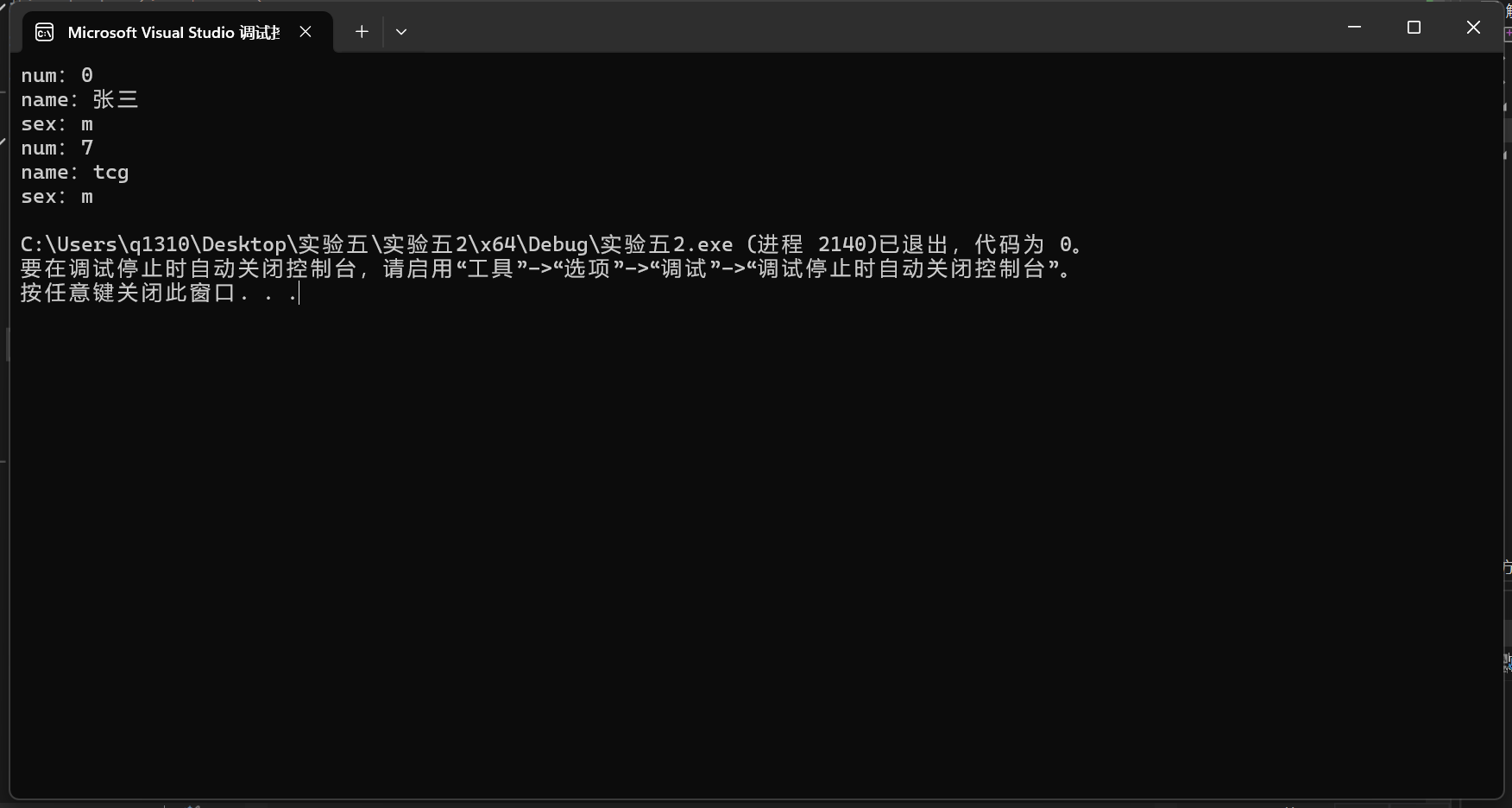
stud.display();

stud1.display();

return 0;

}

1. 结果如下



1. 遇到的问题
2. 思考

3.（1）程序如下

#include <iostream>

using namespace std;

class cuboid {

private:

int length;

int width;

int height;

public:

void vout();

void cin1();

};

void cuboid::vout() {

int v = length \* width \* height;

cout << "体积是" << v << endl;

}

void cuboid::cin1() {

cout << "length:";

cin >> length;

cout << '\n' << "width:";

cin >> width;

cout << '\n' << "height";

cin >> height;

cout << endl;

}

int main()

{

cuboid cub1;

cuboid cub2;

cuboid cub3;

cout << "1号长方体" << endl;

cub1.cin1();

cout << "2号长方体" << endl;

cub2.cin1();

cout << "3号长方体" << endl;

cub3.cin1();

cout << "1号长方体";

cub1.vout();

cout << "2号长方体";

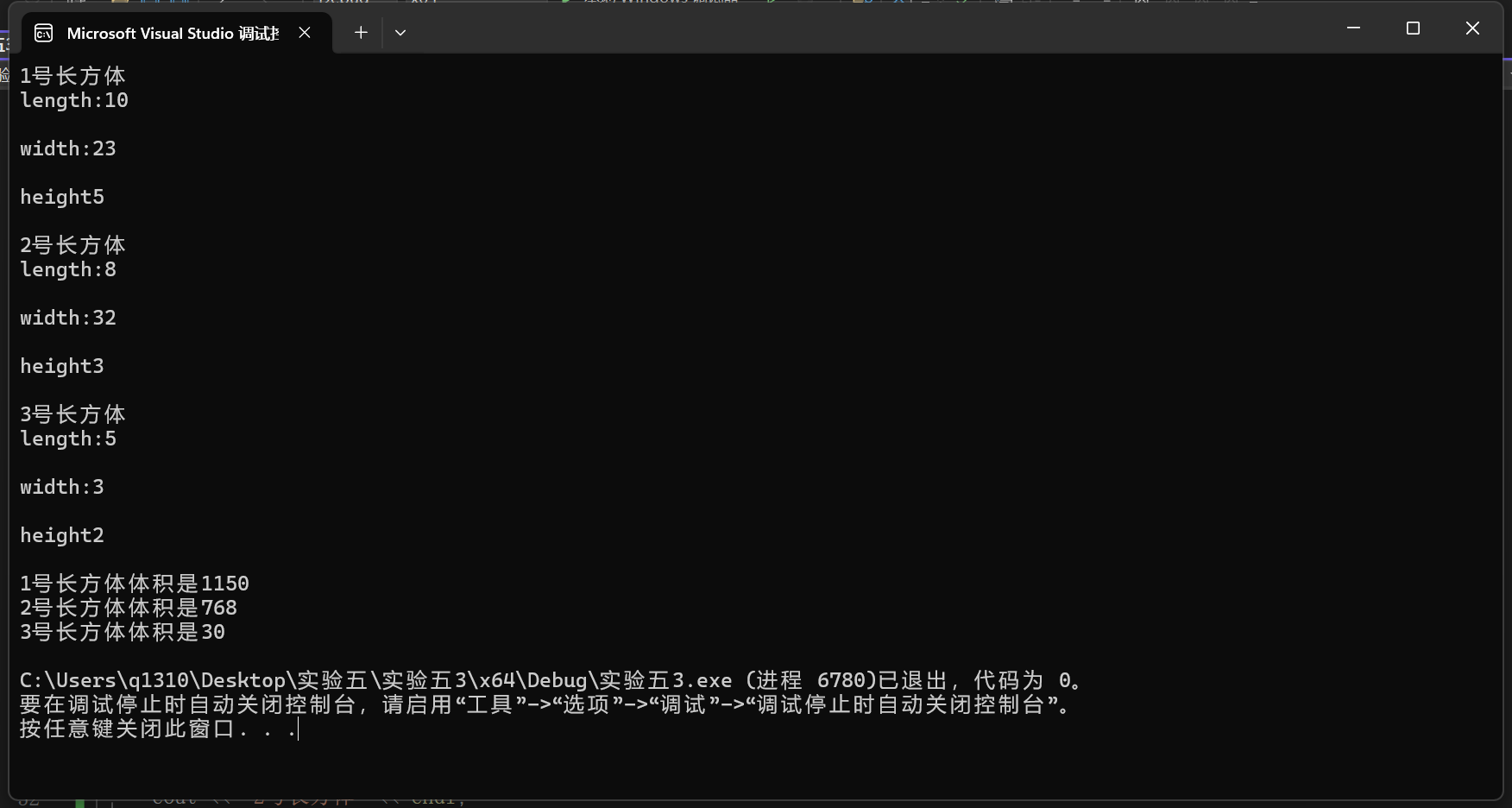
cub2.vout();

cout << "3号长方体";

cub3.vout();

}

1. 结果如下



（3）遇到的问题

（4）思考

4.（1）程序如下：

#include <iostream>

using namespace std;

class student {

private: int num;

int mark;

public: int getm() {

return mark;

};

int getn() {

return num;

}

student(int n, int s) {

num = n;

mark = s;

};

};

void max(student\* students[], int size) {

student\* maxStudent = students[0];

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

if (students[i]->getm() > maxStudent->getm()) {

maxStudent = students[i];

}

}

cout << "成绩最高的学生学号是：" << maxStudent->getn() << endl;

}

int main(){

student\* students[5];

int a[5], b[5];

for (int i = 0; i < 5; ++i) {

cin >> a[i];

cin >> b[i];

}

for (int i = 0; i < 5; ++i) {

students[i] = new student(a[i],b[i]);

}

max(students, 5);

for (int i = 0; i < 5; ++i) {

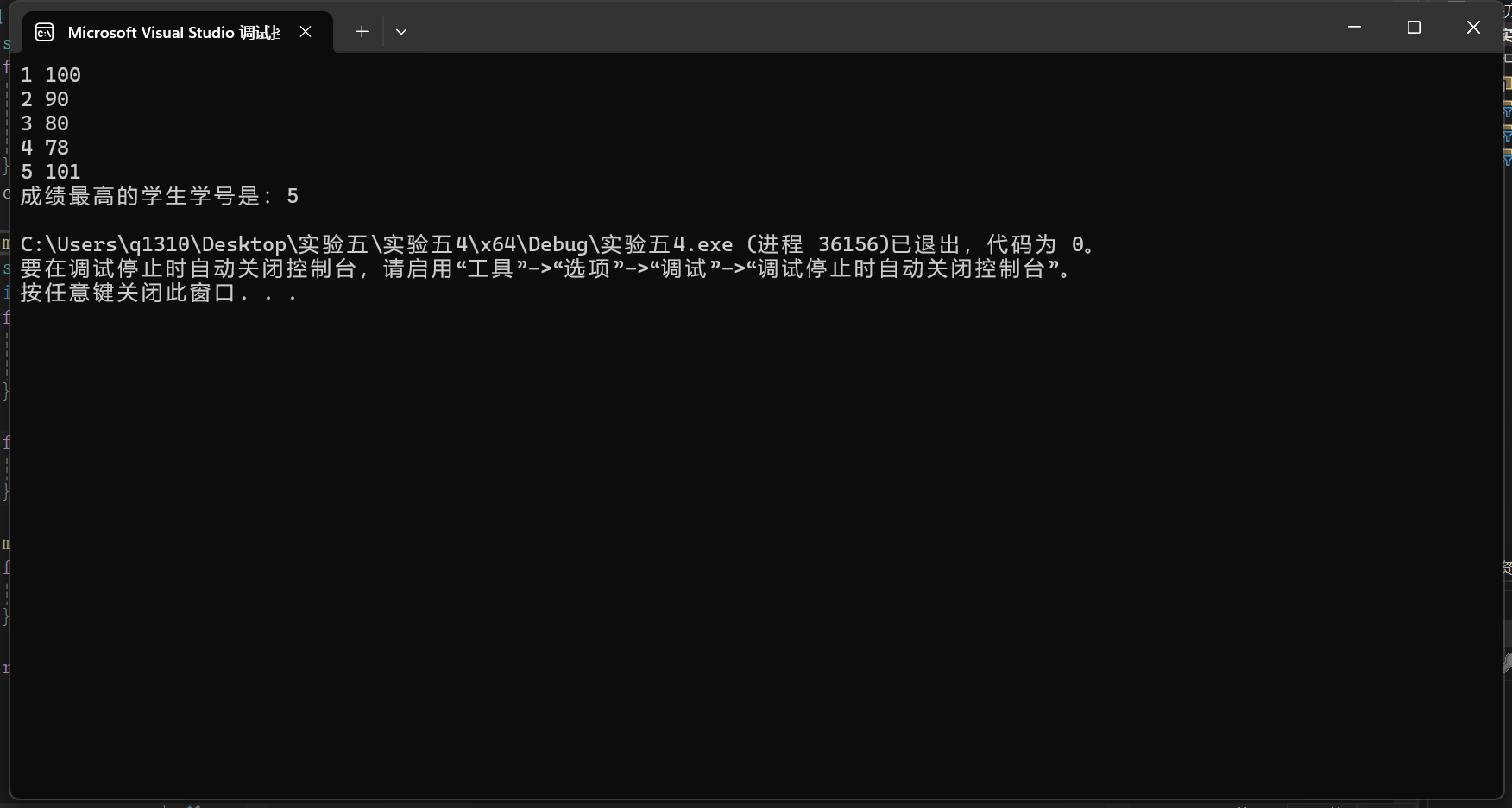
delete students[i];

}

return 0;

}

（2）结果如下



（3）遇到的问题

（4）思考

5（1）程序如下：#include <iostream>

using namespace std;

class point {

private:int x;

int y;

public: point() {

x = 60;

y = 80;

}

void setPoint(int i, int j);

void display();

};

void point::setPoint(int i, int j) {

x = 60 + i;

y = 80 + j;

};

void point::display() {

cout << "(" << x << "," << y << ")" << endl;

};

int main(){

point a;

int c, d;

a.display();

cin >> c >> d;

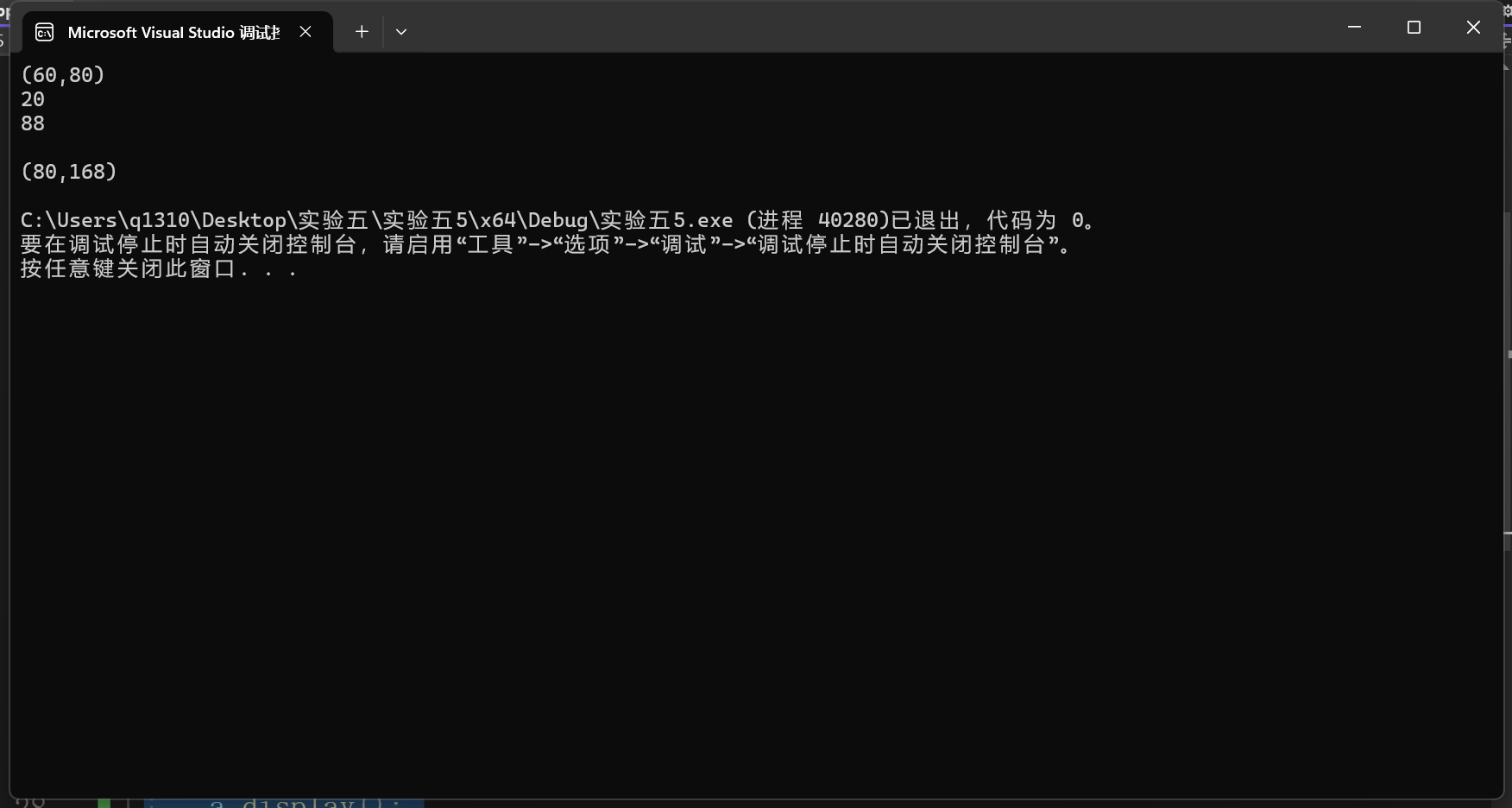
cout << endl;

a.setPoint(c, d);

a.display();

}

（2）结果如下



（3）遇到的问题

（4）思考