**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。

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。

**实验代码如下：**

5.1

#include<iostream>

using namespace std;

class Time {

private:

int hour;

int minute;

int sec;

public:

void setTime(int h, int m, int s) {

hour = h;

minute = m;

sec = s;

}

void showTime() {

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

}

};

int main() {

Time t1;

int h, m, s;

cout << "Enter time (hour minute second): ";

cin >> h >> m >> s;

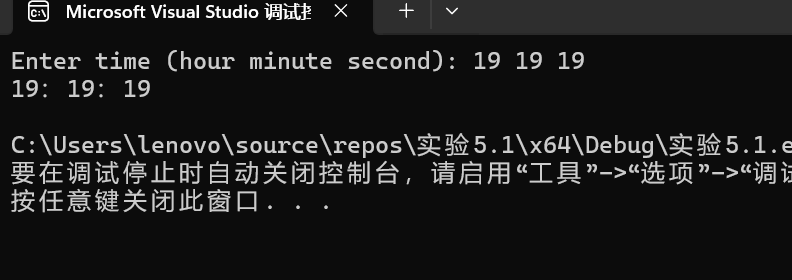
t1.setTime(h, m, s);

t1.showTime();

return 0;

}

实验结果：



5.2

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;

};

Student.cpp:

#include<iostream>

#include"student.h"

using namespace std;

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

num = n;

strcpy\_s(name, nm);

sex = s;

}

void Student::display() {

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

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

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

}

Main.cpp：

#include <iostream>

#include "student.h"

int main() {

Student stud;

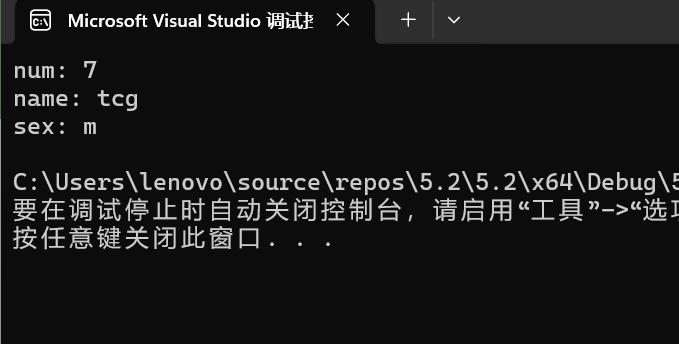
stud.set\_value(007, "tcg", 'm');

stud.display();

return 0;

}

实验结果：



5.3：

#include<iostream>

using namespace std;

class cuboid {

private:

int length;

int width;

int height;

public:

void input() {

cout << "请输入 length, width, height: ";

cin >> length >> width >> height;

}

int volume() {

return length \* width \* height;

}

void print() {

cout << "体积是: " << volume() << endl;

}

};

int main() {

cuboid cuboid1, cuboid2, cuboid3;

cuboid1.input();

cuboid2.input();

cuboid3.input();

cuboid1.print();

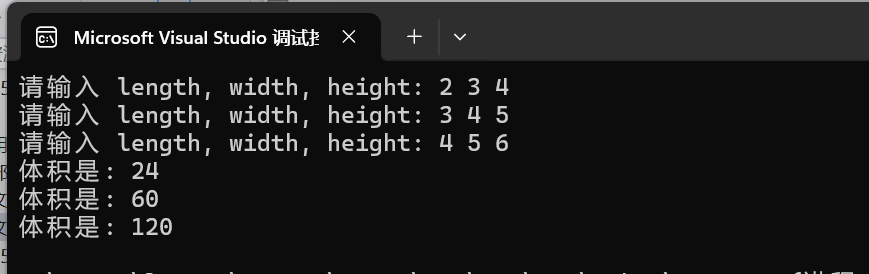
cuboid2.print();

cuboid3.print();

return 0;

}

实验结果：



5.4：

#include<iostream>

using namespace std;

class Student

{

public:

int index;

float grade;

Student(int idx,float grd) {

index = idx;

grade = grd;

}

static void max(Student\* students, int size) {

Student\* highest = &students[0];

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

if (students[i].grade > highest->grade) {

highest = &students[i];

}

}

cout << "学生成绩最高的是学生" << highest->index << "的成绩 " << highest->grade << endl;

}

};

int main() {

Student students[5] = {

Student(1, 89.5),

Student(2, 92),

Student(3, 88),

Student(4, 95),

Student(5, 91.5)

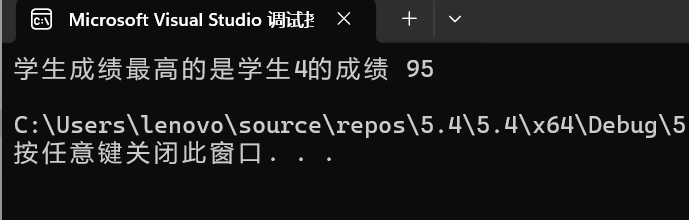
};

Student::max(students, 5);

return 0;

}

实验结果：



5.5：

#include<iostream>

using namespace std;

class Point

{

public:

float x;

float y;

Point(float X, float Y) {

x = X;

y = Y;

}

void setPoint(float i, float j) {

x += i;

y += j;

}

void display() {

cout << "坐标值为: " << x << "," << y << endl;

}

};

int main() {

Point point(60, 80);

float i, j;

cout << "请输入i,j的值:" << endl;

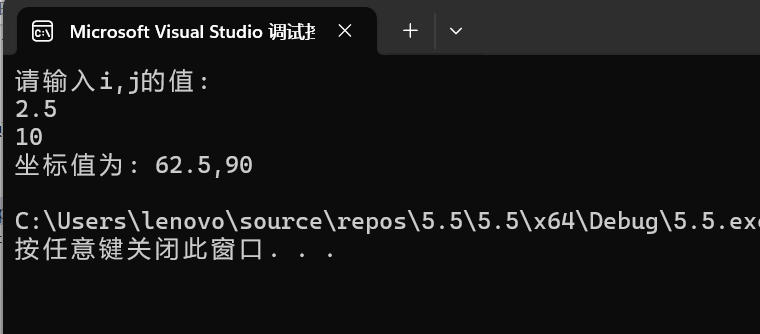
cin >> i >> j;

point.setPoint(i, j);

point.display();

}

实验结果：



**遇到的问题和解决办法：**

**遇到的问题：**

1. 在5.1Time 类中，私有成员变量无法在类外直接访问。
2. 在5.2 Student 类中，strcpy\_s 函数的使用不熟悉。
3. 在5.4 Student 类的 max 函数中，静态成员函数的使用可能存在困惑。

**解决办法：**

1. 使用公共成员函数来设置和获取私有成员变量的值。
2. 学习并使用 strcpy\_s 函数，确保目标数组有足够的空间。
3. 理解静态成员函数的使用，它不需要对象实例即可调用。

**体会：**

通过本次实验，我学会了如何在C++中封装类和对象，以及如何使用公共和私有成员来控制数据的访问。我也体会到了调试程序的重要性，以及如何通过成员函数来实现数据的封装和操作。