**实验五 类与对象**

专业班级： 软件工程2402

学 号： 8209240230

姓 名： 周靖宇

**【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。

**二 算法分析，程序结果**

**1**

**分析：**

**1 类的外部接口（用于简介访问类中私有属性的函数等）和一些需要被外界访问的常量数据应作为公有成员，类所具有的独特属性（内部数据变量）以及不能被外界随便改变的属性，还有一些仅用于类内部实现的辅助函数应作为私有成员。**

**2 类中定义函数为内联函数，适合一些简单的函数，如访问和修改私有成员，以及一些与类紧密相关的小型函数，可以放在类中定义。 而类外定义函数，只在类中声明，适合一些复杂或会被多次使用的函数，这些函数在类外定义，避免定义类时需要多次重复定义。**

#include<iostream>

using namespace std;

class Time // 定义Time类

{

private: // 数据成员为公用的

int hour;

int minute;

int sec;

public:

void settime(int Hour,int Minute,int Sec) {

hour = Hour;

minute = Minute;

sec = Sec;

}

void showtime() {

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

}

};

int main()

{

int Hour;

int Minute, Sec;

Time t1; //定义t1为Time类对象

cin >>Hour; //输入设定的时间

cin >> Minute;

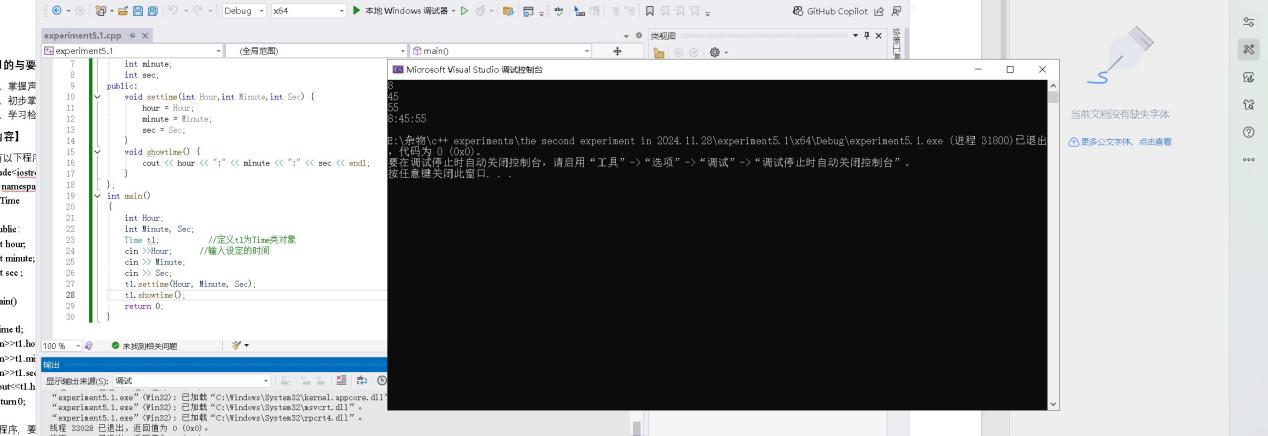
cin >> Sec;

t1.settime(Hour, Minute, Sec);

t1.showtime();

return 0;

}



2

main.cpp:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream> //将类声明头文件包含进来

#include"student.h"

using namespace std;

int main()

{

Student stud; //定义对象

Student stud1;

stud1.setvalue(7, "tcg", 'm');

stud.setvalue(7, "tcg", 'm');

stud.display(); //执行stud对象的display函数

return 0;

}

Student.cpp:

#define \_CRT\_SECURE\_NO\_WARNINGS

#include <iostream>

#include"student.h" //不要漏写此行，否则编译通不过

using namespace std;

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

{

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

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

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

}

void Student::setvalue(int Num, const char\* Name, char Sex)

{

num = Num;

strcpy(name, Name);

sex = Sex;

}

Student.h:

#pragma once

//student.h (这是头文件，在此文件中进行类的声明)

#include<cstring>

using namespace std;

class Student //类声明

{

public: //公用成员函数原型声明

void display();

void setvalue(int Num, const char\* Name, char Sex);

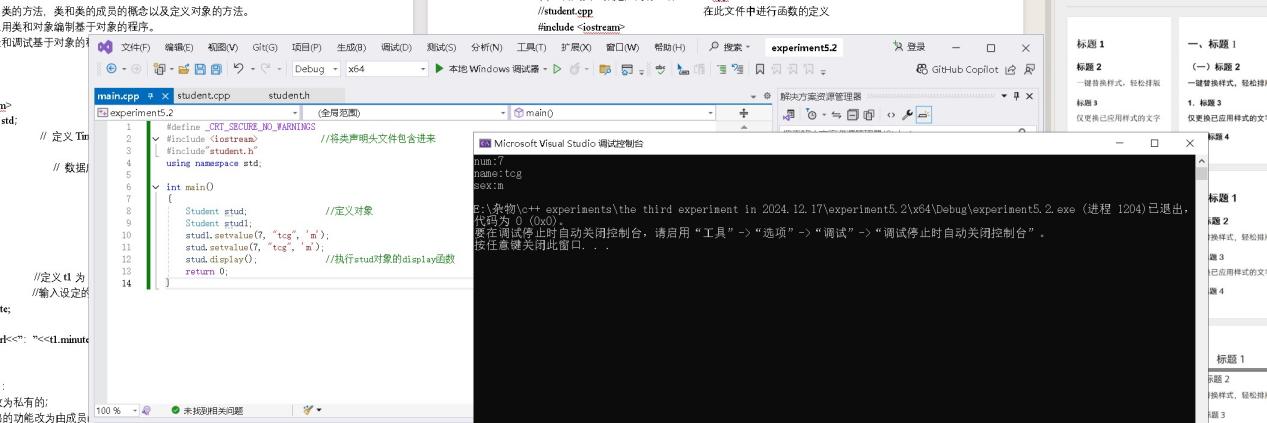
private:

int num;

char name[20];

char sex;

};

3

#include<iostream>

using namespace std;

class chang

{

private:

double length, width, height;

double v;

public:

void set() {

int len, wid, hei;

cout << "请输入长" << endl;

cin >> len;

length = len;

cout << "请输入宽" << endl;

cin >> wid;

width = wid;

cout << "请输入高" << endl;

cin >> hei;

height = hei;

}

void countv()

{

v=length \* width \* height;

}

void showv() {

cout << "体积为"<<v << endl;

}

};

int main() {

cout << "这是第一个长方柱" << endl;

chang c1;

c1.set();

c1.countv();

c1.showv();

cout << "这是第二个长方柱" << endl;

chang c2;

c2.set();

c2.countv();

c2.showv();

cout << "这是第三个长方柱" << endl;

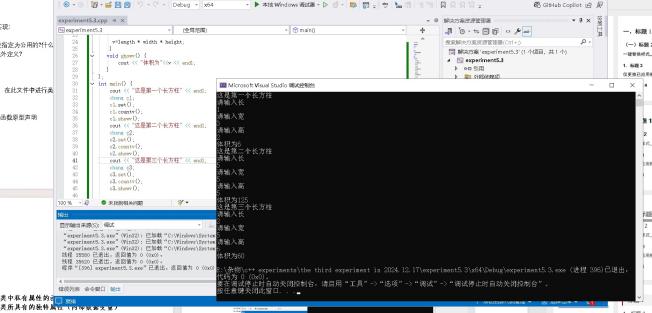
chang c3;

c3.set();

c3.countv();

c3.showv()；

}



4

#include<iostream>

using namespace std;

class Student

{

public:

int num;

double ch;

Student(int n=0, double s=0)

{

num = n;

ch = s;

}

};

void max(Student \*stu){

double maxch = stu[0].ch;

int k = 0;

for (int i = 1;i < 5;i++)

{

if (stu[i].ch > maxch)

{

maxch = stu[i].ch;

k = i;

}

}

cout << "成绩最高的学生学号为" << stu[k].num << endl;

}

int main() {

Student stu[5] = {

Student(1,32),

Student(2,21),

Student(3,70),

Student(4,69),

Student(5,82)

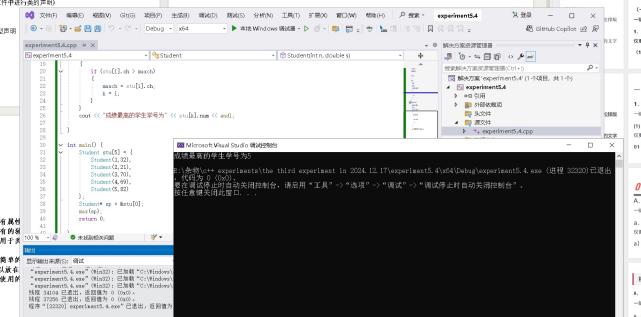
};

Student\* sp = &stu[0];

max(sp);

return 0;

}



5

#include<iostream>

using namespace std;

class Point

{

private:

double x, y;

public:

void setPoint(int i, int j)

{

x = x + i;

y = y + j;

};

void display() {

cout << "坐标为" << '(' << x << ',' << y << ')' << endl;

};

Point(int X = 60, int Y = 80)

{

x = X;

y = Y;

};

};

int main()

{

double i, j;

cout << "更改的值为" << endl;

cin >> i >> j;

Point p1;

cout << "初始化的坐标：" << endl;

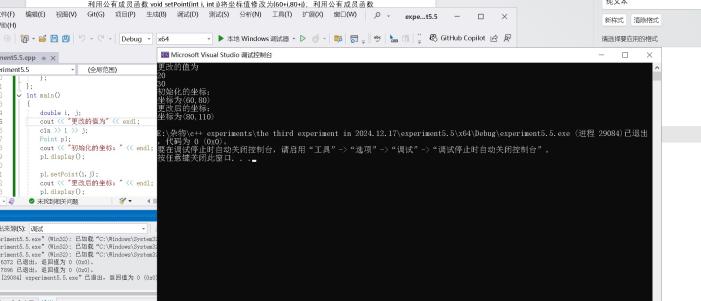
p1.display();

p1.setPoint(i,j);

cout << "更改后的坐标：" << endl;

p1.display();

}



**三 遇到的问题与解决方法**

**1 （3）中传参数“tcg”给函数setvalue时发现数据类型不对，查阅资料后得知，在c++将char类型数组设定为函数参数时会出现数组衰减为指针的情况，导致无法将const char类型的“tcg”传给char\*（原本为char数组，衰减为了指针），后通过使用strcpy解决。之后使用strcpy，出现系统报错，再次查阅资料后得知，strcpy有数据泄露的风险，使用宏定义将该报错消除后解决。**

**2 （4）中需输出数组最高成绩者的学号，参考了以往程序的设计，使用类似气泡算法的方法，从第一个元素开始比较，得出输出结果。**

**四 体会**

**1 程序设计中即使程序本身没问题，也有可能出现一些无关程序的报错问题，比如编译器的问题，这次实验中的c++的数组衰减为指针就是一个例子，这种问题只有实际操作才能发现并积累经验，因此，实践操作是锻炼程序技术必不可少的一环。**

**2 程序设计中必须考虑周全，一些没有考虑到的可能性有可能会使程序不被允许，比如使用strcpy时有可能会因为源字符超过目标字符组而导致数据泄露，因此会使用strcpys函数等，因此，在程序设计中，考虑到可能出现的问题，并确保不同问题时都有解决方案和对应处理分支也是很重要的一环。**