**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?

hour, minute, sec。这些数据成员应该被设置为私有的，以确保数据的封装性和安全性，防止外部代码直接修改这些成员变量，从而避免潜在的错误。

set\_time() 和 show\_time()。这些成员函数应该被设置为公有的，因为它们需要被外部代码调用以实现输入和输出功能。

set\_time() 和 show\_time() 应该在类中定义。这些函数直接操作类的私有数据成员，因此将它们定义在类中可以更好地体现封装性。

在这个例子中，所有的成员函数都在类中定义了。如果类的实现变得非常复杂，可以考虑将成员函数的实现放在类外定义，以保持类定义的简洁性。但是在这个简单的例子中，将成员函数放在类中定义是合适的。

#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:

void set\_time(); // 输入时间的成员函数

void show\_time(); // 输出时间的成员函数

};

void Time::set\_time() {

cin >> hour >> minute >> sec;

}

void Time::show\_time() {

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

}

int main() {

Time t1; // 定义t1为Time类对象

t1.set\_time(); // 调用成员函数输入时间

t1.show\_time(); // 调用成员函数输出时间

return 0;

}

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。

#pragma once

//student.h (这是头文件，在此文件中进行类的声明)

#ifndef STUDENT\_H

#define STUDENT\_H

class Student //类声明

{

public: //公用成员函数原型声明

void display();

void set\_value(int n, const char\* nm, char s); // 新增成员函数声明

private:

int num;

char name[20];

char sex;

};

#endif // STUDENT\_H

//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

{

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

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

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

}

void Student::set\_value(int n, const char\* nm, char s) // 在类外定义set\_value类函数

{

num = n;

strncpy(name, nm, sizeof(name) - 1); // 使用strncpy防止缓冲区溢出

name[sizeof(name) - 1] = '\0'; // 确保字符串以null结尾

sex = s;

}

//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

stud.set\_value(007, "tcg", 'm'); // 使用set\_value函数初始化对象

stud.display(); //执行stud对象的display函数

Student stud1; // 定义另一个对象

stud1.set\_value(123, "xyz", 'f'); // 使用set\_value函数初始化对象

stud1.display(); // 执行stud1对象的display函数

return 0;

}

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。

#include <iostream>

using namespace std;

class Student {

public:

int id;

float score;

Student(int n, float s) : id(n), score(s) {}

};

void max(Student\* arr) {

int maxScore = 0;

int maxId = 0;

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

if (arr[i].score > maxScore) {

maxScore = arr[i].score;

maxId = arr[i].id;

}

}

cout << "成绩最高的学生是学号为 " << maxId << " 的学生。" << endl;

}

int main() {

Student students[5] = {

Student(101, 78.5),

Student(102, 85.3),

Student(103, 98.5),

Student(104, 100.0),

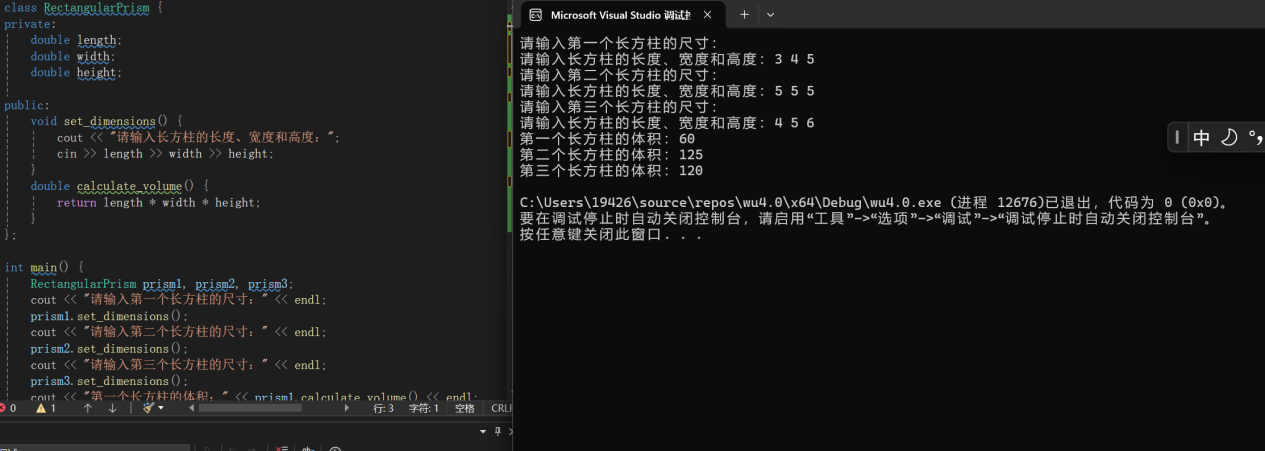
Student(105, 95.5)

};

max(students);

return 0;

}



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。

#include <iostream>

using namespace std;

class RectangularPrism {

private:

double length;

double width;

double height;

public:

void set\_dimensions() {

cout << "请输入长方柱的长度、宽度和高度：";

cin >> length >> width >> height;

}

double calculate\_volume() {

return length \* width \* height;

}

};

int main() {

RectangularPrism prism1, prism2, prism3;

cout << "请输入第一个长方柱的尺寸：" << endl;

prism1.set\_dimensions();

cout << "请输入第二个长方柱的尺寸：" << endl;

prism2.set\_dimensions();

cout << "请输入第三个长方柱的尺寸：" << endl;

prism3.set\_dimensions();

cout << "第一个长方柱的体积：" << prism1.calculate\_volume() << endl;

cout << "第二个长方柱的体积：" << prism2.calculate\_volume() << endl;

cout << "第三个长方柱的体积：" << prism3.calculate\_volume() << endl;

return 0;

}

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。

#include <iostream>

using namespace std;

class Point {

private:

int x;

int y;

public:

Point() : x(60), y(80) {}

void setPoint(int i, int j) {

x = 60 + i;

y = 80 + j;

}

void display() {

cout << "x: " << x << ", y: " << y << endl;

}

};

int main() {

Point p;

p.display();

p.setPoint(10, 20);

p.display();

return 0;

}