**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。

**【实验结果】**

5.1（1）（2）

#include<iostream>

using namespace std;

class Time // 定义Time类

{

private: // 数据成员为私有的(1)

int hour;

int minute;

int sec;

public:

void in(){

cin >> hour;

cin >> minute;

cin >>sec;

}

void out() {

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

}

};

int main(){

Time t1;

t1.in();

t1.out();

return 0;

}

（3）

公有成员：用于与类的对象进行交互的成员函数

私有成员： 类的内部数据成员。为了隐藏类的内部实现细节，防止外部代码直接访问和修改这些数据成员，从而保证数据的完整性和安全性。

类内定义函数：短小、功能简单且与类的内部数据联系紧密的函数，应该在类中定义，使代码结构更加紧凑，并且可以直接访问类的私有成员。

类外定义函数 ：功能复杂、代码较长或者可能被多个类共享的函数，最好在类外定义。

5.2

#pragma once

//student.h

//(1)含类定义的头文件student.h，

class Student {

public:

void display();

void set\_value(int n, const char\* na, char s);

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

//student.cpp

#include <iostream>

#include "student.h"

void Student::display() {

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\* na, char s) {

num = n;

strcpy\_s(name, na);

sex = s;

}

//(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()

{

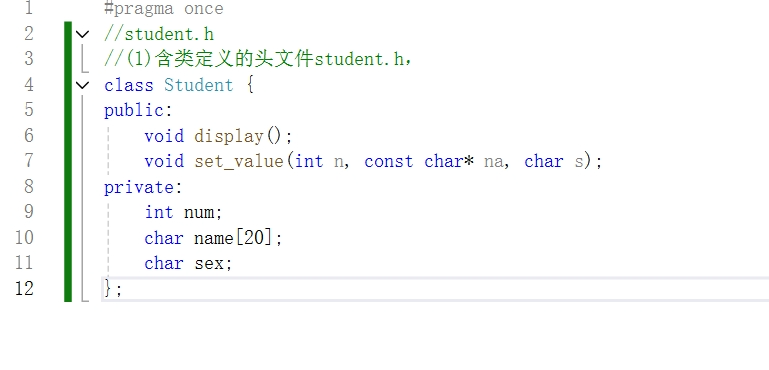
Student stud; //定义对象

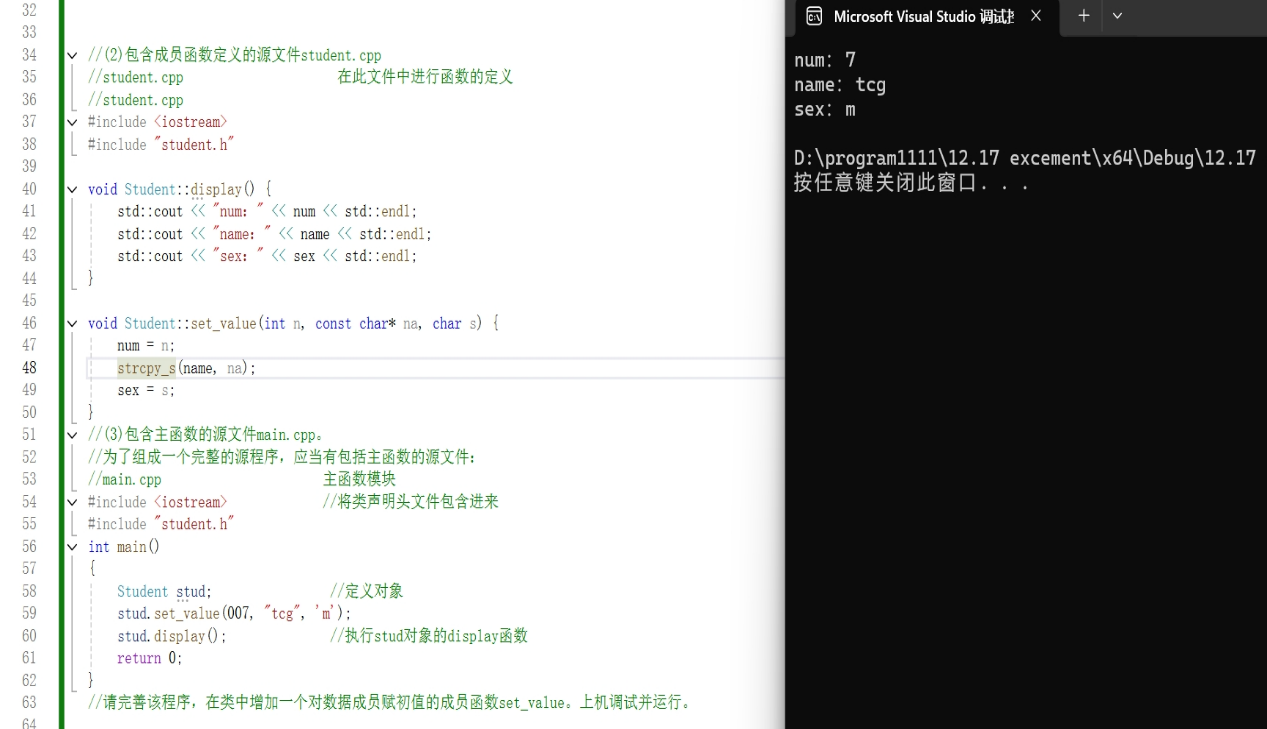
stud.set\_value(007, "tcg", 'm');

stud.display(); //执行stud对象的display函数

return 0;

}





5.3

#include<iostream>

using namespace std;

class cft {

private://数据成员包括length(长)、width(宽)、height(高)。

int length;

int width;

int height;

int v;

public:

void input() {

cin>> length;

cin>> width;

cin>> height;

}//(1)由键盘分别输入3个长方柱的长、宽、高；

void output() {

cout << "长方体的体积是" << v << endl;

}//输出长方柱的体积。

void nodun() {

v = length \* width \* height;

}

};

int main() {

int i = 1;//用循环来反复弄三次计算

for (;i<4;) {

cft t1;

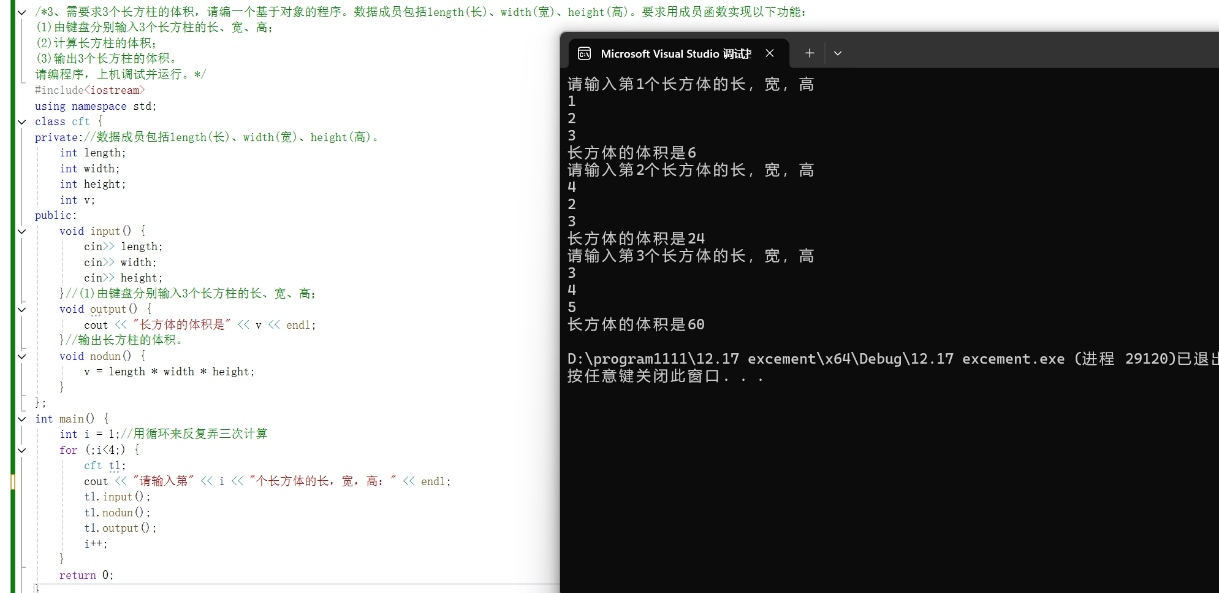
cout << "请输入第" << i << "个长方体的长，宽，高：" << endl;

t1.input();

t1.nodun();

t1.output();

i++;}

return 0;}

5.4#include <iostream>

#include <string>

class Student {

public:

Student(int n, int s) : num(n), score(s) {}

int num;

int score;

};

// 函数max，找出最高成绩者并输出其学号

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

if (size == 0) {

return;

}

Student\* maxStudent = students[0];

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

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

maxStudent = students[i];

}

}

std::cout << "最高成绩者的学号为: " << maxStudent->num << std::endl;

}

int main() {

Student\* students[5];

students[0] = new Student(1, 80);

students[1] = new Student(2, 90);

students[2] = new Student(3, 85);

students[3] = new Student(4, 95);

students[4] = new Student(5, 75);

max(students, 5);

// 释放动态分配的内存

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

delete students[i];

}

return 0;

}



5.5

#include<iostream>

using namespace std;

class Point {

private:

int x=60;

int y=80;

public:

// 构造函数

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

// 修改坐标的函数

void setPoint(int i, int j) {

x = 60 + i;

y = 80 + j;

}

// 显示坐标的函数

void display() {

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

}

};

int main() {

Point p;

p.display();

int i, j;

cout << "请输入坐标的增加量" << endl;

cin >> i >> j;

p.setPoint(i,j);

p.display();

return 0;

}

