**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?

#include<iostream>

using namespace std;

class Time

{

private:

int hour;

int minute;

int sec;

public:

void setTime()

{

cin >> hour>>minute>> sec;

}

void showTime()

{

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

}

};

int main()

{

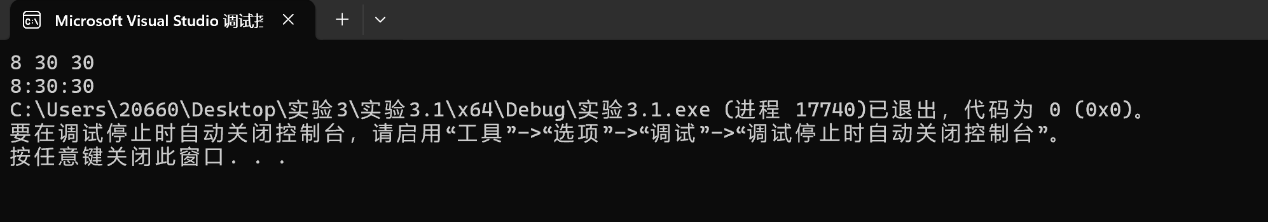
Time t1;

t1.setTime();

t1.showTime();

return 0;

}



类的使用者可以直接访问和操纵的成员应定义为公用，只在类内操作和访问的，不希望使用者直接访问和修改的成员应指定为私有。

较为简单且与类成员联系紧密的函数最好在类内定义，而比较复杂，功能比较独立可在多个类中使用的函数最好在类内定义。

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。

#pragma once

class Student

{

public:

void display();

void set\_value();

private:

int num;

char name[20];

char sex[7];

};

#include<iostream>

#include"student.h"

using namespace std;

void Student::set\_value()

{

cin >> num;

cin >> name;

cin >> sex;

}

void Student::display()

{

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

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

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

}

#include<iostream>

#include"student.h"

using namespace std;

int main()

{

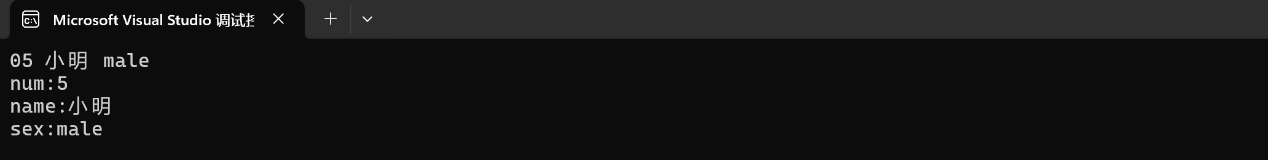
Student stud;

stud.set\_value();

stud.display();

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 cylinder

{

private:

int height;

int length;

int width;

public:

void setnumber()

{

cout << "Enter the height,length and width:";

cin >> height;

cin >> length;

cin >> width;

}

void volume()

{

int volume;

volume = height \* length \* width;

cout <<"The volume is:"<< volume;

}

};

int main()

{

cylinder c1, c2, c3;

c1.setnumber();

c1.volume();

cout << endl;

c2.setnumber();

c2.volume();

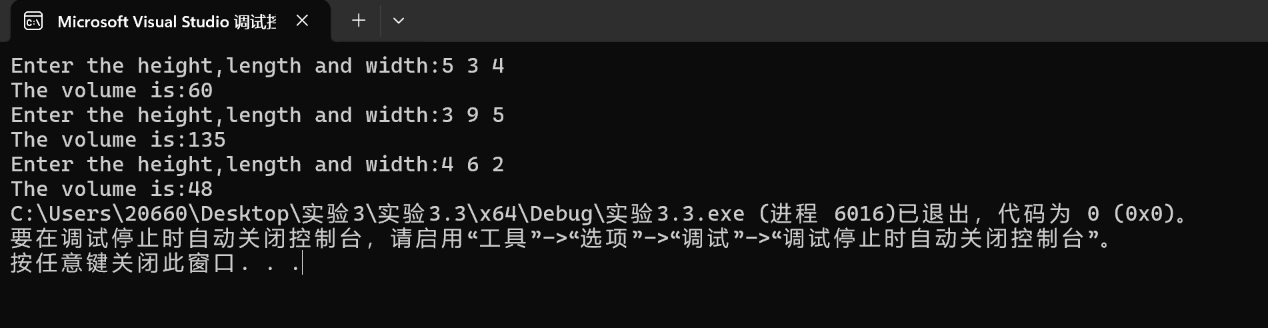
cout << endl;

c3.setnumber();

c3.volume();

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 student

{

private:

char num[4];

int score;

public:

student(const char Num[4], int Score)

{

for (int i = 0; i < 4; i++)

num[i] = Num[i];

score = Score;

}

char\* get\_num()

{

return num;

}

int get\_score()

{

return score;

}

};

void max(student \*ptr)

{

student\* max\_stu = ptr;

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

if (ptr[i].get\_score() > max\_stu->get\_score())

max\_stu = &ptr[i];

cout << "成绩最高的学生学号是：";

cout<< max\_stu->get\_num()<<" ";

}

int main()

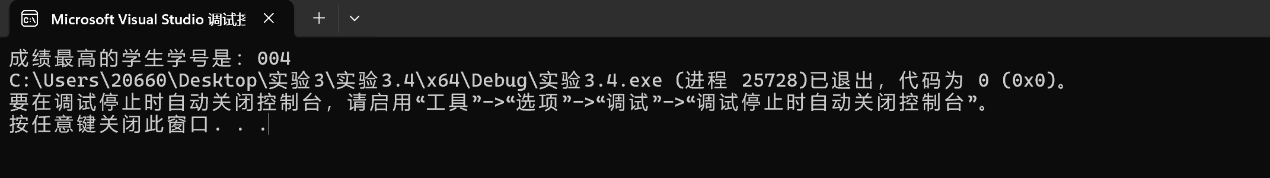
{

student students[5] = { {"001", 87}, {"002", 91},{"003", 84},{"004",95},{"005",86} };

max(students);

return 0;

}



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。

#include<iostream>

using namespace std;

class point

{

private:

int x=60;

int y=80;

public:

void setpoint(int i,int j)

{

x = x + i;

y = y + j;

}

void display()

{

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

}

};

int main()

{

point p1;

p1.setpoint(6, 8);

cout << "The new point is:";

p1.display();

return 0;

}

