**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。

算法分析，实验结果：

1.#include<iostream>

using namespace std;

class Time // 定义Time类

{

int hour;

int minute;

int sec;

public: // 数据成员为公用的

Time(int h = 0, int m = 0, int s = 0)

{

hour = h, minute = m, sec = s;

}

void get()

{

cin >> hour >> minute >> sec;

}

void show()

{

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

}

};

int main()

{

Time t1; //定义t1为Time类对象

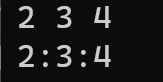
t1.get();

t1.show();

return 0;

}

运行结果如下：



用于改变类状态的成员应该定义为公有，用于保存类状态的成员应该定义为私有，简单得要死的函数最好在类中定义，有一定工作量的函数放在外面定义。

2.//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();

void set\_value(int Num, char Name[20], char Sex);

private:

int num;

char name[20];

char sex;

};

//main.cpp 主函数模块

#include <iostream> //将类声明头文件包含进来

#include "student.h"

using namespace std;

int main()

{

int num;

char name[20];

char sex;

cout << "请分别输入编号、姓名和性别:";

cin >> num;

cin>>name;

cin >> sex;

Student stud; //定义对象

stud.set\_value(num, name, sex);

stud.display(); //执行stud对象的display函数

return 0;

}

//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

using namespace std;

void Student:: set\_value(int Num, char Name[20], char Sex)

{

num = Num;

for (int i = 0; i < 20; i++)

{

name[i] = Name[i];

}

sex = Sex;

}

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

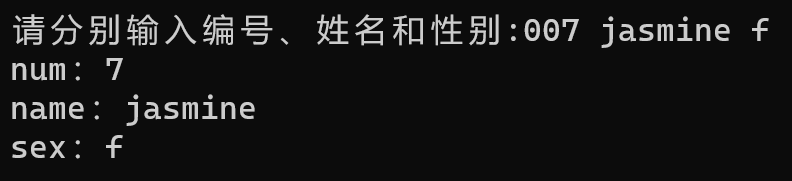
{

cout << "num：" <<num << endl;

cout << "name：" << name << endl;

cout << "sex：" << sex << endl;

}

运行结果如下：

3.#include<iostream>

using namespace std;

class VT

{

private:

double length;

double width;

double height;

public:

void get()

{

cin >> length >> width >> height;

}

void cou()

{

cout << "请分别输入长方体的长宽高:";

}

void show()

{

cout <<"该长方体的体积为："<< length \* width \* height << endl;

}

};

int main()

{

for (int i = 0; i < 3; i++)

{

VT cy;

cy.cou();

cy.get();

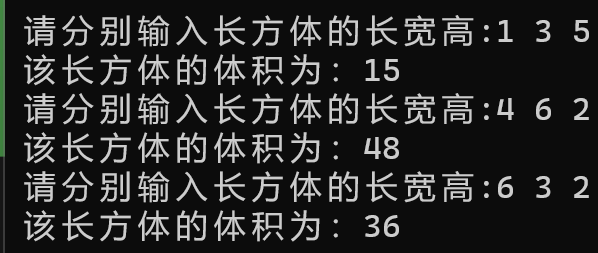
cy.show();

}

return 0;

}

运行结果如下：



4.#include<iostream>

using namespace std;

class stu

{

private:

int num\_;

int score\_;

public:

stu(int num, int score)

{

num\_ = num;

score\_ = score;

}

stu max(stu s)

{

if (s.score\_ < score\_)

return \*this;

else return s;

}

void show()

{

cout << "最高成绩的人是" << num\_ << "号" << endl;

}

};

int main()

{

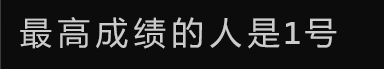
stu s1[5] = { stu(1, 89),stu(2, 85), stu(3,67), stu(4, 78), stu(5, 87) };

stu s = s1[0].max(s1[1]).max(s1[2]).max(s1[3]).max(s1[4]);

s.show();

}

运行结果如下：



5.#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 += i;

y += j;

}

void display()

{

cout << "更改后的坐标值为(" << x << "," << y << ")";

}

};

int main()

{

int i, j;

cout << "请分别输入i,j的值:";

cin >> i >> j;

Point p;

p.setPoint(i, j);

p.display();

return 0;

}

运行结果如下：

