**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。

1. #include<iostream>

class Time

{

private:

int hour, minute, sec;

public:

Time(int hour,int minute,int sec)

{

this->hour = hour;

this->minute = minute;

this->sec = sec;

}

void time()

{

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

}

};

int main()

{

int hour, minute, sec;

std::cin >> hour >> minute >> sec;

Time time(hour, minute, sec);

time.time();

return 0;

}

要保证安全的数据应该私有，要与外部交互的接口需要设成共有；

代码量较小且联系紧密的当设置成类内，代码量大的则设置成类外

2. class Student //类声明 student.h

{

public: //公用成员函数原型声明

void display();

void set\_value();

private:

int num;

char name[20];

char sex;

};

#include <iostream> //student.cpp

#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()

{

std::cin >> num >> name >> sex;

}

#include <iostream> //main.cpp

#include"student.h"

int main()

{

Student stud; //定义对象

stud.set\_value();

/\*Student stud1(007, "tcg", 'm');\*/

stud.display(); //执行stud对象的display函数

return 0;

}

3. #include<iostream>

class object

{

private:

int length, width, height;

public:

void set\_value()

{

std::cin >> length >> width >> height;

}

void volume()

{

std::cout<<length \* width \* height<<std::endl;

}

};

int main()

{

object o1, o2, o3;

o1.set\_value();

o2.set\_value();

o3.set\_value();

o1.volume();

o2.volume();

o3.volume();

return 0;

}

4. #include<iostream>

class Student

{

public:

int no, grade;

Student(int no, int grade)

{

this->no = no;

this->grade = grade;

}

};

void max(Student\* student[])

{

int maxno = student[0]->no;

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

{

if (student[i - 1]->grade < student[i]->grade)

{

maxno = student[i]->no;

}

}

std::cout << "分最高:" << maxno;

}

int main()

{

Student student0(0, 95);

Student student1(1,97);

Student student2(2,95);

Student student3(3,98);

Student student4(4,96);

Student\*student[5];

student[0] = &student0;

student[1] = &student1;

student[2] = &student2;

student[3] = &student3;

student[4] = &student4;

max(student);

}

5. #include<iostream>

class Point

{

private:

int x, y;

public:

Point(int x = 60, int y = 80)

{

this->x = x;

this->y = y;

}

void setPoint(int i, int j)

{

x += i;

y += j;

}

void display()

{

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

}

};

int main()

{

Point point;

point.setPoint(10, 10);

point.display();

return 0;

}