实验三 **Java**中的类继承机制、接口

实验日期： 2017 年 3 月 24 日 班级： 软件1501

学号（后四位）：\_\_\_\_\_0314\_\_\_\_\_ 姓名：\_\_\_\_\_马亚卿\_\_\_\_\_ 成绩：

1. 实验目的

1 实现Java中的类继承机制。

2 体会继承的好处：重用和封装。

1. 实验内容

编写能够满足如下条件的程序：

1. a 声明一个Person类，有name(String类型)、age(int类型)、sex(char类型)属性。

通过构造方法进行赋值。

一个show方法，返回String类型，内容如下：

某某 男（女） 年龄

**package** shiyan3;

**public** **class** Person {

**private** String name;

**private** **int** age;

**private** **char** sex;

**public** Person(String name, **int** age, **char** sex) {

**super**();

**this**.name = name;

**this**.age = age;

**this**.sex = sex;

}

**public** Person() {

**super**();

}

**public** String getName() {

**return** name;

}

**public** **void** setName(String name) {

**this**.name = name;

}

**public** **int** getAge() {

**return** age;

}

**public** **void** setAge(**int** age) {

**this**.age = age;

}

**public** **char** getSex() {

**return** sex;

}

**public** **void** setSex(**char** sex) {

**this**.sex = sex;

}

**public** String show(){

**return** **this**.getName() + " " + **this**.getSex() + " " + **this**.getAge();

}

}

b 声明一个Student类，继承Person类，增加id(int，学号)属性，通过构造方法，利用super调用父类构造方法来进行变量赋值。Override父类的show方法，返回String类型，内容如下：

某某 男（女） 年龄 学号

提示：利用super调用父类的show方法得到除学号部分的String，然后加上学号的信息。

**package** shiyan3;

**public** **class** Student **extends** Person {

**private** **int** id;

**public** Student(String name, **int** age, **char** sex, **int** id) {

**super**(name, age, sex);

**this**.id = id;

}

**public** Student() {

**super**();

}

**public** **int** getId() {

**return** id;

}

**public** **void** setId(**int** id) {

**this**.id = id;

}

**public** String show(){

**return** **super**.show() + " " + **this**.getId();

}

}

c 声明一个Teacher类，继承Person，增加course(String，所教课程)属性，通过构造方法，利用super调用父类构造方法来进行变量赋值。Override父类的show方法，返回String类型，内容如下：

某某 男（女）年龄 所教课程

提示：利用super调用父类的show方法得到除所教课程部分的String，然后加上所教课程的信息。

**package** shiyan3;

**public** **class** Teacher **extends** Person {

**private** String course;

**public** Teacher() {

**super**();

// **TODO** Auto-generated constructor stub

}

**public** Teacher(String name, **int** age, **char** sex, String course) {

**super**(name, age, sex);

**this**.course = course;

}

**public** String getCourse() {

**return** course;

}

**public** **void** setCourse(String course) {

**this**.course = course;

}

**public** String show(){

**return** **super**.show() + " " + **this**.getCourse();

}

}

d 声明PersonApp类，在其中的main方法中分别声明Person、Student、Teacher类型的变量，并通过构造方法初始化，然后显示各自的信息。

**package** shiyan3;

**public** **class** PersonApp {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Person []per = **new** Person[3];

per[0] = **new** Person("Tom" , 18 , 'w');

per[1] = **new** Student("Jack", 20,'w' , 201501);

per[2] = **new** Teacher("Mary" , 40 , 'm', "Java");

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

System.***out***.println(per[i].show());

}

}

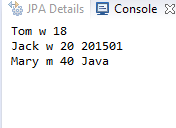


图1 测试图

1. 声明一个Shape接口，其中有计算面积（area）、周长（perimeter）的方法，有以下几个实现：Circle(圆)，Rectangle(矩形)，Triangle(三角形)，都有计算面积、周长的方法。

1. Shape接口

**package** shiyan3;

**public** **interface** Shape {

**double** area();

**double** perimeter();

}

2. Circle实现

**package** shiyan3;

**public** **class** Circle **implements** Shape {

**private** **double** r;

**private** **final** **double** PI = 3.14;

**public** Circle(**double** r) {

**super**();

**this**.r = r;

}

@Override

**public** **double** area() {

// **TODO** Auto-generated method stub

**return** PI \* r \* r;

}

@Override

**public** **double** perimeter() {

// **TODO** Auto-generated method stub

**return** 2.0 \* PI \* r;

}

}

3. Rectangle实现

**package** shiyan3;

**public** **class** Rectangle **implements** Shape {

**private** **double** length;

**private** **double** width;

**public** Rectangle(**double** length, **double** width) {

**super**();

**this**.length = length;

**this**.width = width;

}

@Override

**public** **double** area() {

// **TODO** Auto-generated method stub

**return** **this**.length \* **this**.width;

}

@Override

**public** **double** perimeter() {

// **TODO** Auto-generated method stub

**return** 2.0 \* (**this**.length + **this**.width);

}

}

4. Triangle实现

**package** shiyan3;

**import** java.lang.Math;

**public** **class** Triangle **implements** Shape {

**private** **double** a;

**private** **double** b;

**private** **double** c;

**public** Triangle(**double** a, **double** b, **double** c) {

**super**();

**this**.a = a;

**this**.b = b;

**this**.c = c;

}

@Override

**public** **double** area() {

// **TODO** Auto-generated method stub

**double** p = (**this**.a + **this**.b + **this**.c) / 2.0;

**return** Math.*sqrt*( p \* (p - **this**.a) \* (p - **this**.b) \* (p - **this**.c));

}

@Override

**public** **double** perimeter() {

// **TODO** Auto-generated method stub

**return** **this**.a + **this**.b + **this**.c;

}

}

5.测试类

**package** shiyan3;

**public** **class** ShapeTest {

**public** **static** **void** main(String[] args) {

// **TODO** Auto-generated method stub

Shape []s = **new** Shape[3];

s[0] = **new** Circle(1);

s[1] = **new** Rectangle(3,2);

s[2] = **new** Triangle(3 ,4,5);

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

System.***out***.println(s[i].area() + " " + s[i].perimeter());

}

}

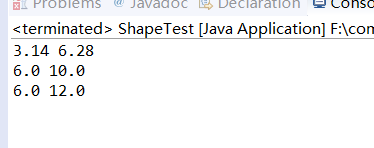


图2 ShapeTest测试结果

三．在本次实验中的问题和解决方法

在第一个试验的时候，Teacher类继承了Person类，但是起初只给出了有参数的构造方法。没有写没有参数的构造方法，导致在测试类之中定义Person对象的时候，编译不通过，后来经过调试，发现了java不像C++那样不管有没有定义构造函数，都给出默认的构造函数，而是如果显式给出构造函数后就会覆盖默认的构造函数，因而在测试类中无法new一个无参数的Person类。

四．IV心得体会

**本次试验让我认识到了类的继承机制，可以大量的节约时间，实现代码的重用以及封装。大大的减少了代码的书写量以及安全性。**

**接口是特殊的抽象类，没有实现方法。需要进行实现后才可以定义对象。**

**通过此次试验，还让我知道了Java的与C++的区别之处。以后要多多注意Java和C++的区别，不然会把C++的习惯带入到Java中造成不可预料的错误。**