Day05

java课程-李彦伯

作业

1. 定义一个People类,有name,height,weight三个属性,有一个方法判断是否肥胖,启动程序录入3个学生的信息,依次输入姓名,身高,体重,当输入完成后,显示”xx同学,你的身材标准|偏胖|偏瘦,录入成功”,当三个学生录入完成后将三个学生的姓名依次打印

```
public class People {

    private String name;
    private double height;
    private double weight;
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public double getHeight() {
        return height;
    }
    public void setHeight(double height)
{
        this.height = height;
    }
    public double getWeight() {
        return weight;
    }
    public void setWeight(double weight)
{
        this.weight = weight;
    }
    //我们不能保证theHeight就是对象的身高,the
Weight就是对象的体重

    //有同学说:我可以将100赋值给对象的身高,将1
```

```
20赋值给对象的体重,调用方法的时候把对象的身高和
体重做为参数传递
    //我们可以直接在非静态方法中使用对象的身高和
体重,所以这个方法是不需要参数的
    //必须先要对对象的属性赋值
    public void orFat(){
        if (height - 105 > weight){
            System.out.println("偏瘦");
        }else if(height - 105 < weight){
            System.out.println("偏胖");
        }else{
            System.out.println("标准");
        }

    }

    @Override
    public String toString() {
        return "姓名是:"+name+"身高是:"+hei
ght+"体重是:"+weight;
    }

}
```

```
import java.util.Scanner;

public class HomeworkDemo {

    public static void main(String[] arg
s) {

        /*
         * 1.创建一个对象数组,People类型,数组
的长度3
         * 2.for循环3次,每次输入3个信息,将3个
信息放入对象的3个属性中
         * 3.对象调用orFat方法
         */
        People [] arr = new Peopl
e[3];//默认是null,需要需要再出创建
        for (int i = 0; i < 1; i++){
            Scanner sc = new Scanner(Syst
em.in);
            People pe = new People();
            System.out.println("输入姓
名");
            pe.setName(sc.nextLine());
            System.out.println("输入身
高");
            pe.setHeight(sc.nextDoubl
e());
            System.out.println("输入体
重");
```

```
                pe.setWeight(sc.nextDoubl
e());;
                pe.orFat();
                arr[i] = pe;
            }
            for(People pe : arr){
            //正常情况下,使用System.out.printl
n(pe);去打印引用类型的时候,会打印内存地址,其实
系统会调用
            //对象的toString方法,如果我们需要打印
对象的特定的信息,可以重新去写这个toString方法实
现目的
            //toString方法的返回值是String,返回
值是什么就会在打印的时候显示什么
                System.out.println(pe);
            }
        }
}
```

创建一个People类型,有年龄,工资,性别三个属性.定义一个方法叫做找对象,找对象方法传过来一个人;首先如果性别相同,就输出”不是同性恋”;如果对方是男的,年龄小于28,工资大于8000就输出”我们结婚吧”,如果年龄太大就输出”太老了”,如果钱太少就输出”屌丝,滚开”;如果对方是女的,如果年龄比自己小,工资大于2000,就输出”结婚吧”,如果年龄太大就输出”不当小白脸”,如果工资太少就输出”不合适”.另一个方法是计算个人所得税:

个人税率表

2011年9月1日起调整后，也就是2012年实行的7级超额累进个人所得税税率表

应纳个人所得税税额= 应纳税所得额× 适用税率- 速算扣除数

扣除标准3500元/月（2011年9月1日起正式执行）（工资、薪金所得适用）

个税免征额3500元　（工资薪金所得适用）

级数	全月应纳税所得额（含税级距）	全月应纳税所得额（不含税级距）	税率(%)	速算扣除数
1	不超过1,500元	不超过1455元的	3	0
2	超过1,500元至4,500元的部分	超过1455元至4155元的部分	10	105
3	超过4,500元至9,000元的部分	超过4155元至7755元的部分	20	555
4	超过9,000元至35,000元的部分	超过7755元至27255元的部分	25	1,005
5	超过35,000元至55,000元的部分	超过27255元至41255元的部分	30	2,755
6	超过55,000元至80,000元的部分	超过41255元至57505元的部分	35	5,505
7	超过80,000元的部分	超过57505元的部分	45	13,505

养老保险单位缴费费率为20%，个人缴费费率为8%；
医疗保险单位缴费比例为10%，个人缴费比例为2%；
工伤保险单位缴费比例为0.2-1.9%，个人不缴费；
生育保险单位缴费比例为1%，个人不缴费；
失业保险单位缴费比例为1%，个人缴费比例为0.5%；
住房公积金单位和个人缴费比例各为7%。

五险一金	合计缴费	雇主缴费	雇员缴费
养老保险	28%	20%	8%
医疗保险	12%	10%	2%+3元
失业保险	1.2%	1%	0.2%
工伤保险	0.3%	0.3%	0
生育保险	0.8%	0.8%	0
住房公积金	24%	12%	12%
合计费率	66.3%	44.1%	22.2%

```java
public class People {
    private int salary;
    private boolean sex;//男的就是true,女的是false
    private int age;

    public int getSalary() {
        return salary;
    }
    public void setSalary(int salary) {
        this.salary = salary;
    }
    public boolean isSex() {
        return sex;
    }
    public void setSex(boolean sex) {
        this.sex = sex;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
    public void findFriend(People pe){
        //调用方法的对象的性别
        if(this.sex == pe.isSex()) {
            System.out.println("不是同性恋");
```

```
            return;
        }
        if(pe.isSex()){
            if(pe.getAge()<28 && pe.getSa
lary() > 8000){
                System.out.println("结婚
吧");
            }else if(pe.getAge() >= 28){
                System.out.println("太老
了");
            }else if(pe.getSalary() <= 80
00){
                System.out.println("屌丝滚
开");
            }
        }else{
            if (pe.getAge() < this.getAg
e() && pe.salary > 2000){
                System.out.println("结婚
吧");
            }else if (pe.getAge() >= thi
s.getAge()){
                System.out.println("不当小
白脸");
            }else if(pe.getSalary() <= 20
00){
                System.out.println("不合
适");
            }
        }
```

```
    }
    public double getTax(){
        double wuXianYiJin = salary * 0.2
22;
        double yingNaShuiSuoDeE = salary
- wuXianYiJin - 3500;
        double shuiLv = 0;
        double suSuanKouChuShu = 0;
        if(yingNaShuiSuoDeE <= 0){
            return 0.0;
        }
        if(yingNaShuiSuoDeE <= 1500){
            shuiLv = 0.03;
            suSuanKouChuShu = 0;
        }else if (yingNaShuiSuoDeE <= 450
0){
            shuiLv = 0.1;
            suSuanKouChuShu = 105;
        }else if (yingNaShuiSuoDeE <= 900
0){
            shuiLv = 0.2;
            suSuanKouChuShu = 555;
        }else if (yingNaShuiSuoDeE <= 350
00){
            shuiLv = 0.25;
            suSuanKouChuShu = 1005;
        }else if (yingNaShuiSuoDeE <= 550
00){
            shuiLv = 0.3;
            suSuanKouChuShu = 2755;
```

```
        }else if (yingNaShuiSuoDeE <= 800
00){
            shuiLv = 0.35;
            suSuanKouChuShu = 5505;
        }else {
            shuiLv = 0.45;
            suSuanKouChuShu = 13505;
        }
    return yingNaShuiSuoDeE * shuiLv - su
SuanKouChuShu;
    }

}
```

```
public class TestPeople {

    public static void main(String[] arg
s) {
        People wangFeng = new People();
        People zhangZiYi = new People();

        wangFeng.setAge(25);
        wangFeng.setSalary(10000);
        wangFeng.setSex(true);

        zhangZiYi.setAge(18);
        zhangZiYi.setSalary(5000);
        zhangZiYi.setSex(false);

        wangFeng.findFriend(zhangZiYi);

        System.out.println(wangFeng.getTa
x());
        System.out.println(zhangZiYi.getT
ax());
    }
}
```

- 可以在声明类的时候直接对成员变量进行赋值,那么所有通过这个类创建出来的对象,成员变量的值都是相等的,都是之前赋值的值

继承

> 继承是描述类与类之间的所属关系,通过类的继承可以形成一个关系体系
> 注意:java的class是单继承不是多继承;因为子类可以继承父类的所有功能,如果有两个父类的功能名称都相同,那么子类继承后调用的方法到底是哪个类我们就不清楚

- 一个文件中可以同时声明多个类,但是最多只能有一个public的类,如果是public则必须和文件名称保持一致
- 格式为:`class 子类 extends 父类{}`
- 子类拥有父类的成员变量和方法

继承关系的成员变量特点

- 父类的成员变量非私有,子类才能直接继承
- 如果子类定义了相同的成员变量,那么子类用的是自己的,而不是父类的
- 如果子类要使用父类的成员变量,使用super关键字进行调用,格式为`super.成员变量名称`

继承关系的成员方法特点

- 父类的成员方法非私有,子类才能直接继承

- 如果子类定义了父类相同的方法,叫做方法的重写,当子类对象进行方法调用的时候优先调用子类的方法,如果子类中没有就回去父类中
- 如果我们子类重写父类方法的时候,使用eclipse的提示功能让系统帮我们生成重写方法的时候会出现@Override,是为了增强编译性的,@Override下的方法父类必须有,如果没有就报错
- 如果子类要使用父类的成员方法,使用super关键字进行调用,格式为`super.方法名称(方法参数);`

> 注意:子类重写父类的方法,权限必须要大于等于父类才行,一般我们写的方法的修饰符都是public

作业

定义一个父类People,定义一个属性weight,定义一个方法eat有一个参数表示吃了多少,每吃一次,体重增加相应的值.定义一个Man是People的子类,定义一个属性是Woman类型叫做女朋友,定义一个属性叫工资.定义一个sport方法,没有参数,运动一次体重减少5,父类中eat方法的功能Man中不变,如果体重超过200,调用一下sport方法.写一个方法lol,男人每玩一次lol,他的女朋友调用自己购物方法去shopping定义一个Woman,是People的子类,定义一个属性是Man类型叫男朋友,定义一个方法shopping,女人每调用一次shopping,男朋友的工资减100

```
public class People {

    double weight;

    public void eat(double shiLiang){
        weight += shiLiang;
    }
}
```

```java
public class Woman extends People{
    //null
    Man bf;
    public void shopping(){
        System.out.println("花男朋友的钱好开心");
        bf.salary -= 100;
    }
}
```

```java
public class Man extends People {
    //gf的默认值是null
    Woman gf;
    double salary;

    public void sport(){
        System.out.println("运动一下");
        weight -= 5;
    }
    @Override
    public void eat(double shiLiang) {
        super.eat(shiLiang);
        if(weight > 200){
            System.out.println("有点胖
了");
            sport();
        }
    }

    public void lol(){
        System.out.println("游戏玩的好爽
啊,但是女朋友生气了");
        gf.shopping();
    }

}
```

```
public class TestPeople {
    public static void main(String[] arg
s) {
        /*
         * 1.程序要运行,需要main方法
         * 2.新建类,简历main方法
         * 3.在main中创建Man对象
         * 4.创建Woman对象
         * 5.给他们各自付成员变量的值
         * 6.让他们各自成为朋友
         * 7.调用他们的方法进行测试
         */
        Man man = new Man();
        Woman woman = new Woman();
        man.weight = 180;
        man.salary = 1000;
        man.gf = woman;
        woman.bf = man;
        man.lol();
        System.out.println(man.salary);
        man.eat(100);
        System.out.println(man.weight);
    }
}
```