**每日作业卷答案**

**就业班JavaSE第2天**

# 关卡1

## 训练案例1

### 训练描述

请阐述你对重写的理解

### 操作步骤答案

1. 子类中出现与父类一模一样的方法时，会出现覆盖操作，也称为override重写、复写或者覆盖。
2. 当子类需要父类的功能，而功能主体子类有自己特有内容时，可以重写父类中的方法，这样，即沿袭了父类的功能，又定义了子类特有的内容
   1. 比如手机，当描述一个手机时，它具有发短信，打电话，显示来电号码功能，后期由于手机需要在来电显示功能中增加显示姓名和头像，这时可以重新定义一个类描述智能手机，并继承原有描述手机的类。并在新定义的类中覆盖来电显示功能，在其中增加显示姓名和头像功能。
3. 重写注意事项有
   1. 子类方法覆盖父类方法，必须要保证权限大于等于父类权限。
   2. 子类与父类的方法声明必须一模一样:函数的返回值类型 函数名 参数列表都要一样

## 训练案例2

### 训练描述

请阐述当子类的成员与父类的成员变量重名时,如何访问成员变量 ?

### 操作步骤答案

1. 当子类的成员与父类的成员变量重名时,在子类中直接成员变量或使用this.成员变量名称访问本类的成员变量
2. 子类的成员与父类的成员变量重名时,在子类中,super.成员变量访问父类的成员变量

## 训练案例3

### 训练描述

请阐述this与super关键字的作用和注意事项.

### 操作步骤答案

1. 请说如何使用this关键字访问本类的成员变量?
   1. this.成员变量名
2. 请说如何使用this关键字访问本类的成员方法?
   1. this.成员方法名(实参列表);
3. 请说如何使用this关键字在构造方法中,调用本类的其他构造方法?
   1. this(实参列表);
4. 请说如何使用super关键字访问父类的成员变量?
   1. super.成员变量名
5. 请说如何使用super关键字访问父类的成员方法?
   1. super.成员变量名
6. 请说如何使用super关键字在构造方法中,调用父类的构造方法?
   1. super(实参列表);
7. 请说如果在构造方法中我们没有this调用父类的构造方法,也没有super调用父类的构造方法,会怎么样?
   1. 系统会提供一个默认super();
8. 请问在构造方法中可以同时使用this调用本类构造方法和super调用父类构造方法吗?
   1. 不可以,编译会报错.

## 训练案例4

### 训练描述

请阐述你对抽象方法和抽象类的理解

### 操作步骤答案

1. 请说出什么抽象方法 ?
   1. 被abstract修饰,只有方法声明没有方法主体的方法就是抽象方法
2. 请说出什么抽象类 ?
   1. 被abstract修饰类就是抽象类
3. 请说出抽象方法的特点是什么 ?
   1. 只有方法声明没有方法主体
   2. 只能必须在抽象类中
4. 请问什么时候需要使用抽象方法 ?
   1. 某个父类只是知道子类应该包含怎么样的方法，但是无法准确知道子类如何实现这些方法。比如一个图形类应该有一个求周长的方法，但是不同的图形求周长的算法不一样;那么父类的这个方法就定义为抽象方法.
5. 请抽象类可以直接创建对象吗? 为什么?
   1. 抽象类不可以直接创建对象,因为抽象类中可以有抽象方法,调用抽象方法没有意义.
6. 请问什么是方法实现 ?
   1. 子类重写类父类的抽象方法,实现了方法主体,这就是方法实现.
7. 请问抽象类可以有构造方法吗,构造方法在抽象类有意义吗?
   1. 可以,有意义,子类的构造方法中可以调用抽象父类的构造方法给成员变量赋值.
8. 请问抽象类一定是父类吗?
   1. 抽象类一定是个父类，因为抽象类时不断抽取共性需求而来的
9. 请问抽象类中可以没有抽象方法吗 ?
   1. 抽象类中是可以不定义抽象方法的，此时仅仅是不让该类创建对象，用于某些特殊的设计需要。
10. 请问是先有抽象类还是先有子类?
    1. 设计时由具体类抽取出抽象类，而开发阶段应该先定义抽象父类，再根据不同需求由父类定义子类。

## 训练案例5

### 训练描述

请用代码描述:

一只黄色的加菲猫和一条白色名称为洛洛的哈士奇狗

要求: 把猫和狗共性抽取到动物类中

### 操作步骤答案

/\*

1.定义动物类(Animal)

a)成员变量 名称(name), 颜色(color)

b)要求:成员变量私有,提供空参和有参的构造方法

\*/

**public** **class** Animal {

// 名称(name)

**private** String name;

// 颜色(color)

**private** String color;

// 无参构造

**public** Animal() {

}

// 带参构造

**public** Animal(String name, String color) {

**this**.name = name;

**this**.color = color;

}

// setXxx和getXxx方法

**public** String getName() {

**return** name;

}

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

**this**.name = name;

}

**public** String getColor() {

**return** color;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

}

/\*

2.定义猫类(Cat),继承Animal类

a)提供空参,有参构造方法,在有参构造中调用父类的构造方法

\*/

**public** **class** Cat **extends** Animal {

// 无参构造

**public** Cat() {

}

// 有参构造

**public** Cat(String name, String color) {

// 调用父类的有参构造,给父类的成员变量赋值

**super**(name, color);

}

}

/\*

3.定义狗类(Dog)

a)成员变量: 品种(breed)

b)提供空参,有参构造方法,在有参构造中调用父类的构造方法

\*/

**public** **class** Dog **extends** Animal {

// 品种(breed)

**private** String bread;

**public** Dog() {

}

**public** Dog(String name,String color, String bread) {

// 调用父类的有参构造方法,给父类的成员变量赋值

**super**(name, color);

// 给自己成员变量赋值

**this**.bread = bread;

}

}

/\*

请用代码描述:

4.创建测试类Test

a)提供main方法

b)在main方法中

i.使用Cat类的带参构造创建猫对象 c

ii.使用 Dog 类的带参构造创建狗对象 d

\*/

**public** **class** Test {

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

// 使用Cat类的带参构造创建猫对象 c

Cat c = **new** Cat("加菲","黄色");

// 使用 Dog 类的带参构造创建狗对象 d

Dog d = **new** Dog("洛洛", "白色", "哈士奇");

}

}

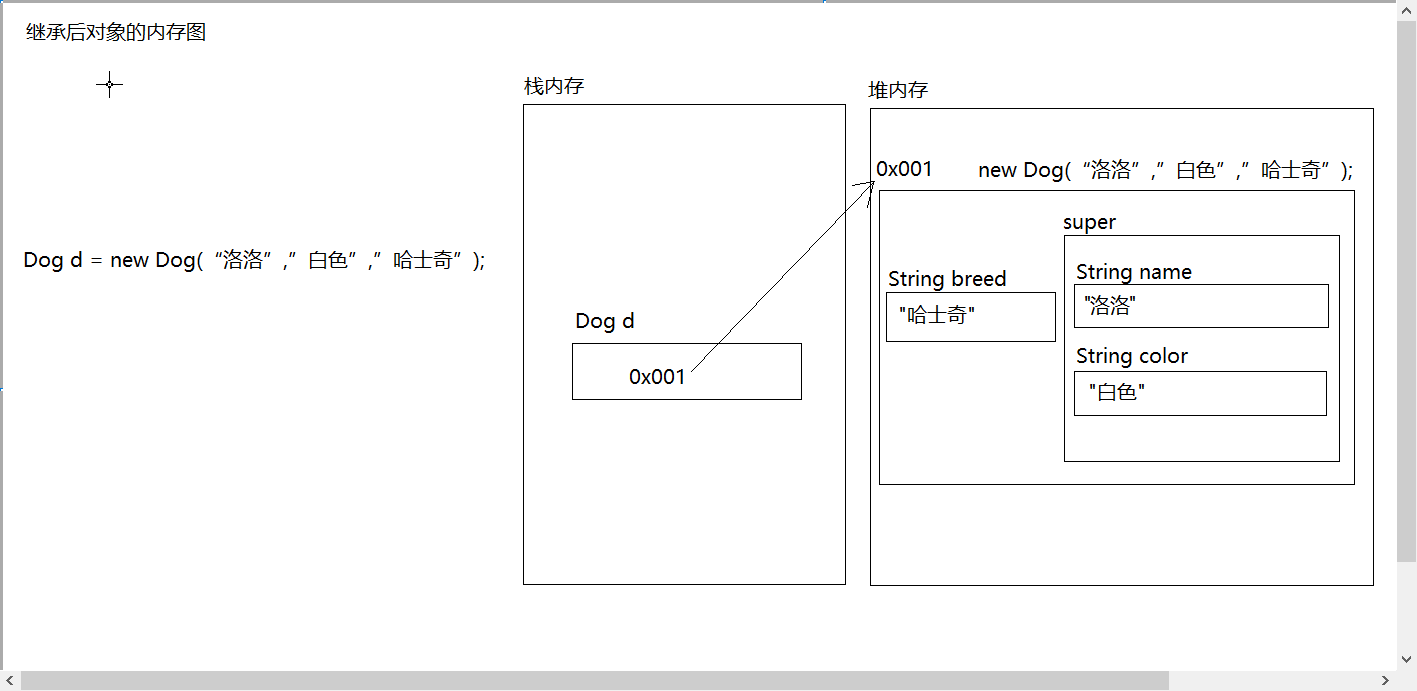
## 训练案例6

### 训练描述

请绘制Dog d = new Dog(“洛洛”,”白色”,”哈士奇”); 这句代码的内存图

说明: 这里Dog类是第5题中的Dog类

### 操作步骤答案



# 关卡2

## 训练案例1

### 训练描述

请使用代码描述:

工资为8000元的30岁的王小平老师,会吃饭(吃工作餐)和讲课.

成绩为90分的15岁的李小乐学生,会吃饭(吃学生餐)和学习.

提示: 把老师和学生的共性抽取人类中,人类不使用抽象类

### 操作步骤答案

/\*

1.定义Person类

i.成员变量(私有): 名称(name),年龄(age)

ii.成员方法: 吃饭(void eat())

1.输出格式: 30岁的王小平在吃饭

iii.提供空参和带参构造方法

iv.提供setXxx和getXxx方法

\*/

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

// 名称(name)

**private** String name;

// 年龄(age)

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

// 饭(void eat())

**public** **void** eat(){

System.***out***.println(age+"岁的"+name+"在吃饭");

}

// 空参构造

**public** Person() {

}

// 带参构造

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

**this**.name = name;

**this**.age = age;

}

// setXxx和getXxx方法

**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;

}

}

/\*

2.定义老师类(Teacher),继承Person类

i.成员变量: salary(工资)

ii.成员方法:

1. 重写父类的 eat()方法

a)输出: 工资为8000元的30岁的王小平老师在吃工作餐

2. 特有方法: lecture() 讲课方法

a)输出: 工资为8000元的30岁的王小平老师在讲课

iii.提供空参和带参构造方法

iv.提供setXxx和getXxx方法

\*/

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

// salary(工资)

**private** **int** salary;

// 重写父类的 eat()方法

**public** **void** eat() {

System.***out***.println("工资为"+salary+"元的"+getAge()+"岁的"+getName()+"老师在吃工作餐");

}

// lecture() 讲课方法

**public** **void** lecture(){

System.***out***.println("工资为"+salary+"元的"+getAge()+"岁的"+getName()+"老师在讲课");

}

// 空参构造

**public** Teacher() {

// 没有写会默认提供一个super();

}

// 带参构造方法

**public** Teacher(String name,**int** age, **int** salary) {

**super**(name,age);

**this**.salary = salary;

}

// 提供setXxx和getXxx方法

**public** **int** getSalary() {

**return** salary;

}

**public** **void** setSalary(**int** salary) {

**this**.salary = salary;

}

}

/\*

3.定义学生类(Student),继承Person类

i.成员变量: score(成绩)

ii.成员方法:

1. 重写父类的 eat()方法

a)输出: 成绩为90分的15岁的李小乐学生在吃学生餐

2. 特有方法: study() 学习方法

a)输出: 成绩为90分的15岁的李小乐学生在学习

iii.提供空参和带参构造方法

iv.提供setXxx和getXxx方法

\*/

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

// score(成绩)

**private** **int** score;

// 重写父类的 eat()方法

**public** **void** eat() {

System.***out***.println("成绩为"+score+"分的"+getAge()+"岁的"+getName()+"学生在吃学生餐");

}

// study() 学习方法

**public** **void** study(){

System.***out***.println("成绩为"+score+"分的"+getAge()+"岁的"+getName()+"学生在学习");

}

// 无参构造

**public** Student() {

**super**();

}

// 带参构造

**public** Student(String name, **int** age,**int** score) {

**super**(name, age);

**this**.score = score;

}

// 提供setXxx和getXxx方法

**public** **int** getScore() {

**return** score;

}

**public** **void** setScore(**int** score) {

**this**.score = score;

}

}

/\*

4.定义Test类

a)提供main方法

b)在main方法中

i.创建老师对象t,并把名称赋值为”王小平”,年龄赋值为30,工资赋值为8000

ii.调用老师对象t的吃饭方法

iii.调用老师对象t的讲解方法

iv.创建学生对象 s,并把名称赋值为”李小乐”,年龄赋值为14,成绩赋值为90分.

v.调用学生对象 s 的吃饭方法

vi.调用学生对象 s 的学习方法

\*/

**public** **class** Test {

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

// i.创建老师对象t,并把名称赋值为”王小平”,年龄赋值为30,工资赋值为8000

Teacher t = **new** Teacher("王小平", 30, 8000);

// ii.调用老师对象t的吃饭方法

t.eat();

// iii.调用老师对象t的讲解方法

t.lecture();

// iv.创建学生对象 s,并把名称赋值为”李小乐”,年龄赋值为14,成绩赋值为90分.

Student s = **new** Student("李小乐", 14, 90);

// v.调用学生对象 s 的吃饭方法

s.eat();

// vi.调用学生对象 s 的学习方法

s.study();

}

}

## 训练案例2

### 训练描述

请使用代码描述:

2岁的红色的公鸡会吃饭(啄米)和打鸣

1岁的黑色的鸭子会吃饭(吃鱼)和游泳.

提示: 把公鸡和鸭子的共性抽取家禽类中,家禽类不使用抽象类

### 操作步骤答案

/\*

1.定义家禽类(Poultry)

i.成员变量(私有): 颜色(color),年龄(age)

ii.成员方法: 吃饭(void eat())

1.输出格式: 2岁的红色家禽在吃饭

iii.提供空参和带参构造方法

iv.提供setXxx和getXxx方法

\*/

**public** **class** Poultry {

// i.成员变量(私有):

// 颜色(color)

**private** String color;

// 年龄(age)

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

// ii.成员方法: 吃饭(void eat())

// 1.输出格式: 2岁的红色家禽在吃饭

**public** **void** eat(){

System.***out***.println(age+"岁的"+color+"的家禽在吃饭");

}

// iii.提供空参和带参构造方法

**public** Poultry() {

**super**();

}

**public** Poultry(String color, **int** age) {

**super**();

**this**.color = color;

**this**.age = age;

}

// iv.提供setXxx和getXxx方法

**public** String getColor() {

**return** color;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

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

**return** age;

}

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

**this**.age = age;

}

}

/\*

2.定义公鸡类(Cock),继承Poultry类

i.成员方法:

1. 重写父类的 eat()方法

a)输出: 2岁的红色公鸡在啄米

2. 特有方法: 打鸣(crow)

a)输出: 2岁的红色公鸡在打鸣

ii.提供空参和带参构造方法

\*/

**public** **class** Cock **extends** Poultry{

// i.成员方法:

// 1. 重写父类的 eat()方法

// a)输出: 2岁的红色公鸡在啄米

**public** **void** eat() {

System.***out***.println(getAge()+"岁的"+getColor()+"公鸡在啄米");

}

// 2. 特有方法: 打鸣(crow)

// a)输出: 2岁的红色公鸡在打鸣

**public** **void** crow(){

System.***out***.println(getAge()+"岁的"+getColor()+"公鸡在打鸣");

}

// ii.提供空参和带参构造方法

**public** Cock() {

**super**();

}

**public** Cock(String color, **int** age) {

**super**(color, age);

}

}

/\*

3.定义鸭子类(Duck),继承Poultry类

i.成员方法:

1. 重写父类的 eat()方法

a)输出: 1岁的黑色的鸭子在吃鱼

2. 特有方法: swim() 游泳方法

a)输出: 1岁的黑色的鸭子在游泳

ii.提供空参和带参构造方法

\*/

**public** **class** Duck **extends** Poultry {

// i.成员方法:

// 1. 重写父类的 eat()方法

// a)输出: 1岁的黑色的鸭子在吃鱼

**public** **void** eat() {

System.***out***.println(getAge()+"岁"+getColor()+"的鸭子在吃鱼");

}

// 2. 特有方法: swim() 游泳方法

// a)输出: 1岁的黑色的鸭子在游泳

**public** **void** swim(){

System.***out***.println(getAge()+"岁"+getColor()+"的鸭子在游泳");

}

// ii.提供空参和带参构造方法

**public** Duck() {

**super**();

}

**public** Duck(String color, **int** age) {

**super**(color, age);

}

}

/\*

4.定义Test类

c)提供main方法

d)在main方法中

i.创建公鸡对象c,并把颜色赋值为”红色”,年龄赋值为2

ii.调用公鸡对象c的吃饭方法

iii.调用公鸡对象c的打鸣方法

iv.创建鸭子对象 d,并把颜色赋值为”黑色”,年龄赋值为1.

v.调用鸭子对象 d 的吃饭方法

vi.调用鸭子对象 d 的游泳方法

\*/

**public** **class** Test {

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

// i.创建公鸡对象c,并把颜色赋值为”红色”,年龄赋值为2

Cock c = **new** Cock("红色", 2);

// ii.调用公鸡对象c的吃饭方法

c.eat();

// iii.调用公鸡对象c的打鸣方法

c.crow();

// iv.创建鸭子对象 d,并把颜色赋值为”黑色”,年龄赋值为1.

Duck d = **new** Duck("黑色", 1);

// v.调用鸭子对象 d 的吃饭方法

d.eat();

// vi.调用鸭子对象 d 的游泳方法

d.swim();

}

}

## 训练案例3

### 训练描述

请使用代码描述:

18岁的演员景甜会吃饭(吃小龙虾)和跳舞

30岁的歌手薛之谦会吃饭(吃大闸蟹)和唱歌.

要求: 把演员和歌手的共性抽取人类中,使用抽象类和抽象方法

### 操作步骤答案

/\*

1.定义抽象类 人类(Person)

i.成员变量(私有): 姓名(name),年龄(age)

ii.抽象方法: 吃饭(void eat())

iii.提供空参和带参构造方法

iv.提供setXxx和getXxx方法

\*/

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

// i.成员变量(私有): 姓名(name),年龄(age)

**private** String name;

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

// ii.抽象方法: 吃饭(void eat())

**public** **abstract** **void** eat();

// iii.提供空参和带参构造方法

**public** Person() {

}

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

**this**.name = name;

**this**.age = age;

}

// iv.提供setXxx和getXxx方法

**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;

}

}

/\*

2.定义演员类(Actor),继承Person类

i.成员方法:

1.重写父类的 eat()方法

a)输出: 18岁的演员景甜在吃小龙虾

2. 特有方法: 跳舞(dance)

a)输出: 18岁的演员景甜在跳白天鹅.

ii.提供空参和带参构造方法

\*/

**public** **class** Actor **extends** Person {

// i.成员方法:

// 1.重写父类的 eat()方法

// a)输出格式: 18岁的演员景甜在吃小龙虾

**public** **void** eat(){

System.***out***.println(getAge()+"岁的演员"+getName()+"在吃小龙虾");

}

// 2. 特有方法: 跳舞(dance)

// a)输出格式: 18岁的演员景甜在跳白天鹅.

**public** **void** dance(){

System.***out***.println(getAge()+"岁的演员"+getName()+"在跳白天鹅");

}

// ii.提供空参和带参构造方法

**public** Actor() {

**super**();

}

**public** Actor(String name, **int** age) {

**super**(name, age);

}

}

/\*

3.定义歌手类(Singer),继承Person类

i.成员方法:

1. 重写父类的 eat()方法

a)输出: 30岁的歌手薛之谦在吃大闸蟹

2. 特有方法: sing() 唱歌

a)输出: 30岁的歌手薛之谦在演唱丑八怪

ii.提供空参和带参构造方法

\*/

**public** **class** Singer **extends** Person {

// i.成员方法:

// 1. 重写父类的 eat()方法

// a)输出: 30岁的歌手薛之谦在吃大闸蟹

**public** **void** eat() {

System.***out***.println(getAge()+"岁的歌手"+getName()+"在吃大闸蟹");

}

// 2. 特有方法: sing() 唱歌

// a)输出: 30岁的歌手薛之谦在演唱丑八怪

**public** **void** sing(){

System.***out***.println(getAge()+"岁的歌手"+getName()+"在演唱丑八怪");

}

// ii.提供空参和带参构造方法

**public** Singer() {

**super**();

}

**public** Singer(String name, **int** age) {

**super**(name, age);

}

}

/\*

4.定义Test类

e)提供main方法

f)在main方法中

i.创建演员对象a,并把名称赋值为”景甜”,年龄赋值为18

ii.调用演员对象a的吃饭方法

iii.调用演员对象a的跳舞方法

iv.创建歌手对象 s,并把姓名赋值为”薛之谦”,年龄赋值为30.

v.调用歌手对象 s 的吃饭方法

vi.调用歌手对象 s 的唱歌方法

\*/

**public** **class** Test {

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

// i.创建演员对象a,并把名称赋值为”景甜”,年龄赋值为18

Actor a = **new** Actor("景甜", 18);

// ii.调用演员对象a的吃饭方法

a.eat();

// iii.调用演员对象a的跳舞方法

a.dance();

// iv.创建歌手对象 s,并把姓名赋值为”薛之谦”,年龄赋值为30.

Singer s = **new** Singer("薛之谦",30);

// v.调用歌手对象 s 的吃饭方法

s.eat();

// vi.调用歌手对象 s 的唱歌方法

s.sing();

}

}

## 训练案例4

### 训练描述

请使用代码描述:

四条腿黑色的狗会吃饭(吃骨头)和看家

两条腿的绿色鹦鹉吃饭(吃小米)和说话.

要求: 把狗和鹦鹉的共性抽取动物类中,使用抽象类和抽象方法

### 操作步骤答案

/\*

1.定义抽象类 动物类(Animal)

i.成员变量(私有): 颜色(color),腿的个数(numOfLegs)

ii.抽象方法: 吃饭(void eat())

iii.提供空参和带参构造方法

iv.提供setXxx和getXxx方法

\*/

**public** **abstract** **class** Animal {

// i.成员变量(私有): 颜色(color),腿的个数(numOfLegs)

**private** String color;

**private** **int** numOfLegs;

// ii.抽象方法: 吃饭(void eat())

**public** **abstract** **void** eat();

// iii.提供空参和带参构造方法

**public** Animal() {

}

**public** Animal(String color, **int** numOfLegs) {

**this**.color = color;

**this**.numOfLegs = numOfLegs;

}

// iv.提供setXxx和getXxx方法

**public** String getColor() {

**return** color;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

**public** **int** getNumOfLegs() {

**return** numOfLegs;

}

**public** **void** setNumOfLegs(**int** numOfLegs) {

**this**.numOfLegs = numOfLegs;

}

}

/\*

2.定义狗类(Dog),继承Animal类

i.成员方法:

1.重写父类的 eat()方法

a)输出格式:: 4条腿黑色的狗在啃骨头

2. 特有方法: 看家(lookHome)

a)输出格式:: 4条腿黑色的狗在看家.

ii.提供空参和带参构造方法

\*/

**public** **class** Dog **extends** Animal {

// i.成员方法:

// 1.重写父类的 eat()方法

// a)输出格式:: 4条腿黑色的狗在啃骨头

**public** **void** eat() {

System.***out***.println(getNumOfLegs()+"条腿"+getColor()+"的狗在啃骨头");

}

// 2. 特有方法: 看家(lookHome)

// a)输出格式:: 4条腿黑色的狗在看家.

**public** **void** lookHome(){

System.***out***.println(getNumOfLegs()+"条腿"+getColor()+"的狗在看家");

}

// ii.提供空参和带参构造方法

**public** Dog() {

**super**();

}

**public** Dog(String color, **int** numOfLegs) {

**super**(color, numOfLegs);

}

}

/\*

3.定义鹦鹉类(Parrot),继承Animal类

i.成员方法:

1. 重写父类的 eat()方法

a)输出格式: 2条腿的绿色鹦鹉在吃小米

2. 特有方法: say() 说明

a)输出格式: 2条腿的绿色鹦鹉在说你好,丑八怪

ii.提供空参和带参构造方法

\*/

**public** **class** Parrot **extends** Animal {

// i.成员方法:

// 1. 重写父类的 eat()方法

// a)输出格式: 2条腿的绿色鹦鹉在吃小米

**public** **void** eat(){

System.***out***.println(getNumOfLegs()+"条腿的"+getColor()+"鹦鹉在吃小米");

}

// 2. 特有方法: say() 说明

// a)输出格式: 2条腿的绿色鹦鹉在说你好,丑八怪

**public** **void** say(){

System.***out***.println(getNumOfLegs()+"条腿的"+getColor()+"鹦鹉在说 你好,丑八怪");

}

// ii.提供空参和带参构造方法

**public** Parrot() {

**super**();

}

**public** Parrot(String color, **int** numOfLegs) {

**super**(color, numOfLegs);

}

}

/\*

4.定义Test类

a)提供main方法

b)在main方法中

i.创建Dog对象d,并把颜色赋值为”黑色”,腿的个数赋值为4

ii.调用Dog对象d的吃饭方法

iii.调用Dog对象d的看家方法

iv.创建Porrot对象 p,并把颜色赋值为”绿色”,腿的个数赋值为2.

v.调用Porrot对象 p 的吃饭方法

vi.调用Porrot对象 p 的说话方法

\*/

**public** **class** Test {

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

// i.创建Dog对象d,并把颜色赋值为”黑色”,腿的个数赋值为4

Dog d = **new** Dog("黑色",4);

// ii.调用Dog对象d的吃饭方法

d.eat();

// iii.调用Dog对象d的看家方法

d.lookHome();

// iv.创建Porrot对象 p,并把颜色赋值为”绿色”,腿的个数赋值为2.

Parrot p = **new** Parrot("绿色",2);

// v.调用Porrot对象 p 的吃饭方法

p.eat();

// vi.调用Porrot对象 p 的说话方法

p.say();

}

}

## 训练案例5

### 训练描述

请使用代码描述

在动物园中有多种动物,每种动物都年龄和腿的个数,按吃的不同分为食肉动物和食草动物

食肉动物: 有老虎,啄木鸟

食草动物: 有长颈鹿,兔子

吃的东西:

老虎: 年龄为 2的4条腿的老虎在吃羊

啄木鸟: 年龄为 1岁的2条腿的啄木鸟的在吃昆虫

长颈鹿: 年龄为 3 岁的4条腿的长颈鹿在吃树叶

兔子: 年龄为 1岁的4条腿的兔子在吃狗尾草

### 操作步骤答案

/\*

1.定义抽象类动物类(Animal)

a)成员变量:年龄(age),腿的个数(numOfLegs)

b)抽象方法: void eat()

c)提供空参和有参构造方法和setXxx和getXxx方法

\*/

**public** **abstract** **class** Animal {

// a)成员变量:年龄(age),腿的个数(numOfLegs)

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

**private** **int** numOfLegs;

// b)抽象方法: void eat()

**public** **abstract** **void** eat();

// c)提供空参和有参构造方法和setXxx和getXxx方法

**public** Animal() {

**super**();

}

**public** Animal(**int** age, **int** numOfLegs) {

**super**();

**this**.age = age;

**this**.numOfLegs = numOfLegs;

}

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

**return** age;

}

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

**this**.age = age;

}

**public** **int** getNumOfLegs() {

**return** numOfLegs;

}

**public** **void** setNumOfLegs(**int** numOfLegs) {

**this**.numOfLegs = numOfLegs;

}

}

/\*

2.定义抽象类食肉类动物(Carnivore)继承自动物类(Animal)

a)提供空参和有参构造方法

\*/

**public** **abstract** **class** Carnivore **extends** Animal {

// a)提供空参和有参构造方法

**public** Carnivore() {

**super**();

}

**public** Carnivore(**int** age, **int** numOfLegs) {

**super**(age, numOfLegs);

}

}

/\*

3.定义抽象类食草类动物(Herbivore)继承自动物类(Animal)

a)提供空参和有参构造方法

\*/

**public** **abstract** **class** Herbivore **extends** Animal {

// a)提供空参和有参构造方法

**public** Herbivore() {

**super**();

}

**public** Herbivore(**int** age, **int** numOfLegs) {

**super**(age, numOfLegs);

}

}

/\*

4.定义老虎类(Tiger)继承食肉类动物(Carnivore)

a)重写eat()方法

i.输出格式: 年龄为 2的4条腿的老虎在吃羊

b)提供空参和有参构造方法

\*/

**public** **class** Tiger **extends** Carnivore {

// a)重写eat()方法

// i.输出格式: 年龄为 2的4条腿的老虎在吃羊

**public** **void** eat() {

System.***out***.println("年龄为 "+getAge()+"的"+getNumOfLegs()+"条腿的老虎在吃羊");

}

// b)提供空参和有参构造方法

**public** Tiger() {

**super**();

}

**public** Tiger(**int** age, **int** numOfLegs) {

**super**(age, numOfLegs);

}

}

/\*

5.定义啄木鸟(Pecker)继承食肉类动物(Carnivore)

a)重写eat()方法

i.输出格式: 年龄为 1岁的2条腿的啄木鸟的在吃昆虫

b)提供空参和有参构造方法

\*/

**public** **class** Pecker **extends** Carnivore {

//a)重写eat()方法

// i.输出格式: 年龄为 1岁的2条腿的啄木鸟的在吃昆虫

**public** **void** eat(){

System.***out***.println("年龄为 "+getAge()+"岁的"+getNumOfLegs()+"条腿的啄木鸟的在吃昆虫");

}

//b)提供空参和有参构造方法

**public** Pecker() {

**super**();

}

**public** Pecker(**int** age, **int** numOfLegs) {

**super**(age, numOfLegs);

}

}

/\*

6.定义长颈鹿(Giraffe)继承食草动物(Herbivore)

a)重写eat()方法

i.输出格式: 年龄为 3 岁的4条腿的长颈鹿在吃树叶

b)提供空参和有参构造方法

\*/

**public** **class** Giraffe **extends** Herbivore {

//a)重写eat()方法

// i.输出格式: 年龄为 3 岁的4条腿的长颈鹿在吃树叶

**public** **void** eat(){

// 子类中访问没有被重写父类的成员方法: 可以直接调用,this.方法名(实际参数列表)调用,也可以super.方法名(实际参数列表)调用

System.***out***.println("年龄为 "+**this**.getAge()+" 岁的"+**super**.getNumOfLegs()+"条腿的长颈鹿在吃树叶");

}

//b)提供空参和有参构造方法

**public** Giraffe() {

**super**();

}

**public** Giraffe(**int** age, **int** numOfLegs) {

**super**(age, numOfLegs);

}

}

/\*

7.定义兔子(Rabbit)继承食草动物(Herbivore)

a)重写eat()方法

i.输出格式: 年龄为 1岁的4条腿的兔子在吃狗尾草

b)提供空参和有参构造方法

\*/

**public** **class** Rabbit **extends** Herbivore {

//a)重写eat()方法

// i.输出格式: 年龄为 1岁的4条腿的兔子在吃狗尾草

**public** **void** eat(){

System.***out***.println("年龄为 "+getAge()+"岁的"+getNumOfLegs()+"条腿的兔子在吃狗尾草");

}

//b)提供空参和有参构造方法

**public** Rabbit() {

**super**();

}

**public** Rabbit(**int** age, **int** numOfLegs) {

**super**(age, numOfLegs);

}

}

/\*

8.定义测试类Test

a)提供main方法

b)在main方法中

i.创建老虎对象 t,年龄赋值为2,腿的个数赋值为4;调用老虎对象t的eat()方法

ii.创建啄木鸟对象 p,年龄赋值为1,腿的个数赋值为2;调用啄木鸟对象p的eat()方法

iii.创建长颈鹿对象 g,年龄赋值为3,腿的个数赋值为4;调用长颈鹿对象g的eat()方法

iv.创建兔子对象 r,年龄赋值为1,,腿的个数赋值为4;调用兔子对象r的eat()方法

\*/

**public** **class** Test {

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

// i.创建老虎对象 t,年龄赋值为2,腿的个数赋值为4;调用老虎对象t的eat()方法

Tiger t = **new** Tiger(2, 4);

t.eat();

// ii.创建啄木鸟对象 p,年龄赋值为1,腿的个数赋值为2;调用啄木鸟对象p的eat()方法

Pecker p = **new** Pecker(1, 2);

p.eat();

// iii.创建长颈鹿对象 g,年龄赋值为3,腿的个数赋值为4;调用长颈鹿对象g的eat()方法

Giraffe g = **new** Giraffe(3, 4);

g.eat();

// iv.创建兔子对象 r,年龄赋值为1,,腿的个数赋值为4;调用兔子对象r的eat()方法

Rabbit r = **new** Rabbit(1,4);

r.eat();

}

}

## 训练案例6

### 训练描述

请用代码描述：

20岁1.75米的男孩慕容紫英有一个18岁1.6米瓜子脸女朋友欧阳青青

欧阳青青在洗一件白色的李宁牌子的衣服

慕容紫英去散步(和欧阳青青)

慕容紫英用带有阳刚之气的声音对欧阳青青说:我会守护你一生一世

欧阳青青微笑着用于甜美的声音对慕容紫英说:有你在我就安心.

要求: 男孩和女孩的共有成员提取到人类中,使用抽象类和抽象方法

### 操作步骤答案

/\*

1.定义衣服类(Clothes)

a)成员变量:

i.颜色(color)

ii.品牌(brand)

b)提供 无参,带参构造和setXxx和getXxx方法

\*/

**public** **class** Clothes {

// a)成员变量:

// i.颜色(color)

**private** String color;

// ii.品牌(brand)

**private** String brand;

// b)提供 无参,带参构造和setXxx和getXxx方法

**public** Clothes() {

**super**();

}

**public** Clothes(String color, String brand) {

**super**();

**this**.color = color;

**this**.brand = brand;

}

**public** String getColor() {

**return** color;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

**public** String getBrand() {

**return** brand;

}

**public** **void** setBrand(String brand) {

**this**.brand = brand;

}

}

/\*

2.抽象类(Person)

a)成员变量

i.姓名 name

ii.年龄 age

iii.身高 height

b)成员方法

i.抽象方法 void say(String content)

c)要求

i.提供无参和带参构造

ii.提供setXxx和getXxx方法

\*/

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

//a)成员变量

// i.姓名 name

**private** String name;

// ii.年龄 age

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

// iii.身高 height

**private** **double** height;

//b)成员方法

// i.抽象方法 void say(String content)

**public** **abstract** **void** say(String content);

//c)要求

// i.提供无参和带参构造

**public** Person() {

**super**();

}

**public** Person(String name, **int** age, **double** height) {

**super**();

**this**.name = name;

**this**.age = age;

**this**.height = height;

}

// ii.提供setXxx和getXxx方法

**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** **double** getHeight() {

**return** height;

}

**public** **void** setHeight(**double** height) {

**this**.height = height;

}

}

/\*

3. 定义Girl类继承Person类

a)成员变量

i.脸型(String face)

b)成员方法:

i.重写父类的 void say(String content) 方法

1.输出格式: 欧阳青青微笑着用于甜美的声音对慕容紫英说:有你在我就安心.

ii.特有方法: 洗衣服 wash(Clothes c)

输出格式: 欧阳青青在洗一件白色的李宁牌子的衣服

c)要求:

i.提供无参和带参构造

ii.提供setXxx和getXxx方法

\*/

**public** **class** Girl **extends** Person {

//a)成员变量

// i.脸型(String face)

**private** String face;

//b)成员方法:

// i.重写父类的 void say(String content) 方法

// 1.输出格式: 欧阳青青微笑着用于甜美的声音对慕容紫英说:有你在我就安心.

**public** **void** say(String countent){

System.***out***.println(getName()+"微笑着用于甜美的声音对慕容紫英说:"+countent);

}

// ii.特有方法: 洗衣服 wash(Clothes c)

// 输出格式: 欧阳青青在洗一件白色的李宁牌子的衣服

**public** **void** wash(Clothes c){

System.***out***.println(getName()+"在洗一件"+c.getColor()+"的"+c.getBrand()+"牌子的衣服");

}

//c)要求:

// i.提供无参和带参构造

// ii.提供setXxx和getXxx方法

**public** Girl() {

**super**();

}

**public** Girl(String name, **int** age, **double** height,String face) {

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

**this**.face = face;

}

}

/\*

4.定义Boy类继承Person类

a)成员变量

i.女朋友 (girlFriend)

b)成员方法:

i.重写父类的 void say(String content) 方法

输出格式:容紫英用带有阳刚之气的声音对欧阳青青说:我会守护你一生一世

ii.特有方法: 散步 walking()

输出格式: 慕容紫英和欧阳青青在河边的林荫小道上手牵着手散步

c)要求:

i.提供无参和带参构造

iii.提供setXxx和getXxx方法

\*/

**public** **class** Boy **extends** Person {

//a)成员变量

// i.女朋友 (girlFirend)

**private** Girl girlFriend;

//b)成员方法:

// i.重写父类的 void say(String content) 方法

// 输出格式:容紫英用带有阳刚之气的声音对欧阳青青说:我会守护你一生一世

**public** **void** say(String content){

System.***out***.println(getName()+"带有阳刚之气的声音对"+girlFriend.getName()+"说:"+content);

}

// ii.特有方法: 散步 walking()

// 输出格式: 慕容紫英和欧阳青青在河边的林荫小道上手牵着手散步

**public** **void** walking(){

System.***out***.println(getName()+"和"+girlFriend.getName()+"在河边的林荫小道上手牵着手散步");

}

//c)要求:

// i.提供无参和带参构造

**public** Boy() {

**super**();

}

**public** Boy(String name, **int** age, **double** height,Girl girlFriend) {

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

**this**.girlFriend = girlFriend;

}

// iii.提供setXxx和getXxx方法

**public** Girl getGirlFriend() {

**return** girlFriend;

}

**public** **void** setGirlFriend(Girl girlFriend) {

**this**.girlFriend = girlFriend;

}

}

/\*

5.定义测试类Test

a)提供main方法

b)在main方法中

i.创建衣服对象 c,品牌初始化为李宁,颜色初始化为白色

ii.创建女孩对象 girl,姓名赋值为欧阳青青,年龄赋值为18,身高赋值为1.6,脸型赋值为瓜子脸

iii.创建男孩对象 boy,姓名赋值为慕容紫英,年龄赋值以为20,身高赋值1.75,女朋友赋值为girl

iv.调用对象girl的洗衣服方法

v.调用对象boy的散步方法

vi.调用对象boy的说话方法,传入:我会守护你一生一世.

vii.调用对象 girl 的说话方法,传入: 有你在我就安心.

\*/

**public** **class** Test {

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

// i.创建衣服对象 c,品牌初始化为李宁,颜色初始化为白色

Clothes c = **new** Clothes("白色", "李宁");

// ii.创建女孩对象 girl,姓名赋值为欧阳青青,年龄赋值为18,身高赋值为1.6,脸型赋值为瓜子脸

Girl girl = **new** Girl("欧阳青青", 18, 1.6, "瓜子脸");

// iii.创建男孩对象 boy,姓名赋值为慕容紫英,年龄赋值以为20,身高赋值1.75,女朋友赋值为girl

Boy boy = **new** Boy("慕容紫英", 20, 1.75, girl);

// iv.调用对象girl的洗衣服方法

girl.wash(c);

// v.调用对象boy的散步方法

boy.walking();

// vi.调用对象boy的说话方法,传入:我会守护你一生一世

boy.say("我会守护你一生一世.");

// vii.调用对象 girl 的说话方法,传入: 有你在我就安心.

girl.say("有你在我就安心.");

}

}

## 训练案例7

### 训练描述

请用代码描述

四个轮子的白色的汽车可以跑和加油

两个轮子的红色电动自行车可以跑和充电

要求: 把汽车和电动自行车的共性抽取到交通工具类中

### 操作步骤答案

/\*

1.定义交通工具类(Vehicle)

a)成员变量

i.轮子个数 wheel

ii.颜色 color

b)成员方法

i.跑 run()

1.打印格式: 4个轮子白色的车在跑

c)要求:

i.成员变量不带权限修饰符,方法是公共权限的.

\*/

**public** **class** Vehicle {

// a)成员变量

// i.轮子个数 wheel

**int** wheel;

// ii.颜色 color

String color;

//b)成员方法

// i.跑 run()

// 1.打印格式: 四个轮子白色的车在跑

**public** **void** run(){

System.***out***.println(wheel+"个轮子"+color+"的车在跑");

}

}

/\*

2.定义小汽车类(Car),继承Vehicle类

a)成员方法: 加油 (addGasoline() ))

i.在方法中的输出格式为: 4个轮子的白色小汽车在加油

\*/

**public** **class** Car **extends** Vehicle{

// a)成员方法: 加油 (addGasoline() ))

// i.在方法中的输出格式为: 4个轮子的白色小汽车在加油

**public** **void** addGasoline(){

System.***out***.println(wheel+"个轮子的"+color+"小汽车在加油");

}

}

/\*

3.定义电动车类(ElectricBicycle),继承Vehicle类

a)成员方法: 充电 (charge())

i.在方法中的输出格式为: 2个轮子的黑色电动自行车在充电

\*/

**public** **class** ElectricBicycle **extends** Vehicle {

// a)成员方法: 充电 (charge())

// i.在方法中的输出格式为: 2个轮子的黑色电动自行车在充电

**public** **void** charge(){

System.***out***.println(wheel+"个轮子的"+color+"电动自行车在充电");

}

}

/\*

4.定义测试类Test

c)提供main方法

d)在main方法中

i.使用空参构造,创建Car 对象c

ii.把c对象的成员wheel 赋值为 4

iii.把c对象的成员color赋值为 白色

iv.调用c对象成员方法run方法

v.调用c对象的成员方法addGasoline()

vi.使用空参构造,创建ElectricBicycle对象 eb

vii.把eb对象的成员wheel 赋值为2

viii.把ed对象的成员color赋值为 黑色

ix.调用ed对象成员方法run方法

x.调用ed对象成员方法 charge方法

\*/

**public** **class** Test {

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

// i.使用空参构造,创建Car 对象c

Car c = **new** Car();

// ii.把c对象的成员wheel 赋值为 4

c.wheel = 4;

// iii.把c对象的成员color赋值为 白色

c.color = "白色";

// iv.调用c对象成员方法run方法

c.run();

// v.调用c对象的成员方法addGasoline()

c.addGasoline();

// vi.使用空参构造,创建ElectricBicycle对象 eb

ElectricBicycle eb = **new** ElectricBicycle();

// vii.把eb对象的成员wheel 赋值为2

eb.wheel = 2;

// viii.把ed对象的成员color赋值为 黑色

eb.color = "黑色";

// ix.调用ed对象成员方法run方法

eb.run();

// x.调用ed对象成员方法 charge方法

eb.charge();

}

}

# 关卡3

## 训练案例1

### 训练描述

请用代码描述:

白色4条腿的北极熊(Bear)会吃(吃蜂蜜)和抓鱼(catchFish)

黑色4条腿的大熊猫(Panda)会吃(吃竹子)和爬树(climbTree)

要求: 把北极熊和大熊猫的共性提取动物类(Animal)中,不使用抽象类

### 操作步骤

1. 定义动物类（Animal）类

1. 属性： 颜色（color），腿的个数(numOfLegs)

1. 成员方法: eat()
   1. 输出格式: 白色4条腿的动物在吃东西
2. 提供带参构造和setXxx和getXxx方法

2. 定义北极熊（Bear）类 继承 动物（Animal）类

1. 成员方法：

1. 重写父类的eat()方法

输出格式： 白色4条腿的北极熊在吃蜂蜜

2. 特有方法: 抓鱼(catchFish)

输出格式: 白色4条腿的北极熊在抓鱼

2. 提供：带参构造方法

3. 定义大熊猫（Panda）类 继承动物（Animal）类

1. 成员方法：

1. 重写父类的eat()方法

输出格式： 白色4条腿的大熊猫在吃竹子

2. 特有方法: 爬树(climbTree)

输出格式: 黑色4条腿的大熊猫在爬树

2. 提供：带参构造方法

4. 定义测试类Test

1. 提供main方法

2. 在main方法中

1. 创建北极熊对象 b,颜色赋值为白色,腿的个数赋值为4

2. 调用北极熊对象b的吃方法

1. 调用北极熊对象b的抓鱼方法

4. 创建大熊猫对象 p,颜色赋值为黑色,腿的个数赋值为4

5. 调用大熊猫对象p的吃方法

6. 调用大熊猫对象p的爬树方法

### 操作步骤答案

/\*

1. 定义动物类（Animal）类

1. 属性： 颜色（color），腿的个数(numOfLegs)

2.成员方法: eat()

a)输出格式: 白色4条腿的动物在吃东西

3. 提供带参构造和setXxx和getXxx方法

\*/

**public** **class** Animal {

// 1. 属性： 颜色（color），腿的个数(numOfLegs)

**private** String color;

**private** **int** numOfLegs;

// 2.成员方法: eat()

// a)输出格式: 白色4条腿的动物在吃东西

**public** **void** eat(){

System.***out***.println(color+numOfLegs+"条腿的动物在吃东西");

}

// 3. 提供带参构造和setXxx和getXxx方法

**public** Animal() {

**super**();

}

**public** Animal(String color, **int** numOfLegs) {

**super**();

**this**.color = color;

**this**.numOfLegs = numOfLegs;

}

**public** String getColor() {

**return** color;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

**public** **int** getNumOfLegs() {

**return** numOfLegs;

}

**public** **void** setNumOfLegs(**int** numOfLegs) {

**this**.numOfLegs = numOfLegs;

}

}

/\*

2. 定义北极熊（Bear）类 继承 动物（Animal）类

1. 成员方法：

1. 重写父类的eat()方法

输出格式： 白色4条腿的北极熊在吃蜂蜜

2. 特有方法: 抓鱼(catchFish)

输出格式: 白色4条腿的北极熊在抓鱼

2. 提供：带参构造方法

\*/

**public** **class** Bear **extends** Animal {

// 1. 成员方法：

// 1. 重写父类的eat()方法

// 输出格式： 白色4条腿的北极熊在吃蜂蜜

**public** **void** eat() {

System.***out***.println(getColor()+getNumOfLegs()+"腿的北极熊在吃蜂蜜");

}

// 2. 特有方法: 抓鱼(catchFish)

// 输出格式: 白色4条腿的北极熊在抓鱼

**public** **void** catchFish(){

System.***out***.println(getColor()+getNumOfLegs()+"腿的北极熊在抓鱼");

}

// 2. 提供：带参构造方法

**public** Bear() {

**super**();

}

**public** Bear(String color, **int** numOfLegs) {

**super**(color, numOfLegs);

}

}

/\*

3. 定义大熊猫（Panda）类 继承 动物（Animal）类

1. 成员方法：

1. 重写父类的eat()方法

输出格式： 白色4条腿的大熊猫在吃竹子

2. 特有方法: 爬树(climbTree)

输出格式: 黑色4条腿的大熊猫在爬树

2. 提供：带参构造方法

\*/

**public** **class** Panda **extends** Animal {

// 1. 成员方法：

// 1. 重写父类的eat()方法

// 输出格式： 白色4条腿的大熊猫在吃竹子

**public** **void** eat() {

System.***out***.println(getColor()+getNumOfLegs()+"条腿的大熊猫在吃竹子");

}

// 2. 特有方法: 爬树(climbTree)

// 输出格式: 黑色4条腿的大熊猫在爬树

**public** **void** climbTree(){

System.***out***.println(getColor()+getNumOfLegs()+"条腿的大熊猫在爬树");

}

// 2. 提供：带参构造方法

**public** Panda() {

**super**();

}

**public** Panda(String color, **int** numOfLegs) {

**super**(color, numOfLegs);

}

}

/\*

4. 定义测试类Test

1. 提供main方法

2. 在main方法中

1. 创建北极熊对象 b,颜色赋值为白色,腿的个数赋值为4

2. 调用北极熊对象b的吃方法

3.调用北极熊对象b的抓鱼方法

4. 创建大熊猫对象 p,颜色赋值为黑色,腿的个数赋值为4

5. 调用大熊猫对象p的吃方法

6. 调用大熊猫对象p的爬树方法

\*/

**public** **class** Test {

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

// 1. 创建北极熊对象 b,颜色赋值为白色,腿的个数赋值为4

Bear b = **new** Bear("白色", 4);

// 2. 调用北极熊对象b的吃方法

b.eat();

// 3.调用北极熊对象b的抓鱼方法

b.catchFish();

// 4. 创建大熊猫对象 p,颜色赋值为黑色,腿的个数赋值为4

Panda p = **new** Panda("黑色", 4);

// 5. 调用大熊猫对象p的吃方法

p.eat();

// 6. 调用大熊猫对象p的爬树方法

p.climbTree();

}

}

## 训练案例2

### 训练描述

请使用代码描述:

4个轮子的白色的汽车可以跑(开着跑)和加油

2个轮子的红色电动自行车可以跑(骑着跑)和充电

要求: 把汽车和电动自行车的共性抽取到交通工具类中,使用抽象类和抽象方法

### 操作步骤

1. 定义抽象交通工具类(Vehicle)
   1. 成员变量
      1. 轮子个数 wheel
      2. 颜色 color
   2. 抽象方法
      1. 跑 run()
   3. 要求： 提供空参构造和带参构造以及setters和getters方法
2. 定义小汽车类(Car),继承Vehicle类
   1. 成员方法
      1. 重写父类的run方法
         1. 输出格式： 4个轮子的白色的汽车开着跑
      2. 特有方法: 加油 (addGasoline() )
         1. 在方法中的输出格式为: 4个轮子的白色汽车在加油
   2. 要求：提供空参构造和带参构造
3. 定义电动车类(ElectricBicycle),继承Vehicle类
   1. 成员方法
      1. 重写父类的run方法
         1. 输出格式： 2个轮子的黑色的电动车骑着跑
      2. 特有成员方法: 充电 (charge())
         1. 在方法中的输出格式为: 2个轮子的黑色电动自行车在充电
   2. 要求：提供提供空参构造和带参构造
4. 定义测试类Test
   1. 提供main方法
   2. 在main方法中
5. 创建Car 对象c，把颜色赋值为白色，把轮子个数赋值为4
6. 调用c对象成员方法run方法
7. 调用c对象的成员方法addGasoline()
8. 创建ElectricBicycle对象 eb，吧颜色赋值为黑色，轮子个数赋值为2
9. 调用ed对象成员方法run方法
10. 调用ed对象成员方法 charge方法

### 操作步骤答案

/\*

1. 定义抽象交通工具类(Vehicle)

a)成员变量

i.轮子个数 wheel

ii.颜色 color

b)抽象方法

i.跑 run()

c)要求： 提供提供空参构造和带参构造以及getters和setters方法

\*/

**public** **abstract** **class** Vehicle {

// a)成员变量

// i.轮子个数 wheel

**private** **int** wheel;

// ii.颜色 color

**private** String color;

//b)抽象方法

// i.跑 run()

**public** **abstract** **void** run();

// c) 提供提供空参构造和带参构造以及getters和setters方法

**public** Vehicle() {

**super**();

}

**public** Vehicle(**int** wheel, String color) {

**this**.wheel = wheel;

**this**.color = color;

}

**public** **int** getWheel() {

**return** wheel;

}

**public** **void** setWheel(**int** wheel) {

**this**.wheel = wheel;

}

**public** String getColor() {

**return** color;

}

**public** **void** setColor(String color) {

**this**.color = color;

}

}

/\*

2.定义小汽车类(Car),继承Vehicle类

a)成员方法

i.重写父类的run方法

1.输出格式： 4个轮子的白色的汽车开着跑

ii.特有方法: 加油 (addGasoline() ))

1.在方法中的输出格式为: 4个轮子的白色汽车在加油

b) 要求：提供空参构造和带参构造

\*/

**public** **class** Car **extends** Vehicle{

//a)成员方法

// i.重写父类的run方法

// 1.输出格式： 4个轮子的白色的汽车开着跑

**public** **void** run(){

System.***out***.println(getWheel()+"个轮子"+getColor()+"的汽车开着跑");

}

// ii.特有方法: 加油 (addGasoline())

// 1.在方法中的输出格式为: 4个轮子的白色小汽车在加油

**public** **void** addGasoline(){

System.***out***.println(getWheel()+"个轮子"+getColor()+"的汽车在加油");

}

//b) 要求：提供空参构造和带参构造

**public** Car() {

**super**();

}

**public** Car(**int** wheel, String color) {

**super**(wheel, color);

}

}

/\*

3.定义电动车类(ElectricBicycle),继承Vehicle类

a)成员方法

i.重写父类的run方法

1.输出格式： 2个轮子的黑色的电动车骑着跑

ii.特有成员方法: 充电 (charge())

1.在方法中的输出格式为: 2个轮子的黑色电动自行车在充电

b) 要求：提供提供空参构造和带参构造

\*/

**public** **class** ElectricBicycle **extends** Vehicle{

// a)成员方法

// i.重写父类的run方法

// 1.输出格式： 2个轮子的黑色的电动车骑着跑

**public** **void** run() {

System.***out***.println(getWheel()+"个轮子的"+getColor()+"的电动车骑着跑");

}

// ii.特有成员方法: 充电 (charge())

// 1.在方法中的输出格式为: 2个轮子的黑色电动自行车在充电

**public** **void** charge(){

System.***out***.println(getWheel()+"个轮子的"+getColor()+"的电动车在充电");

}

// b) 要求：提供提供空参构造和带参构造

**public** ElectricBicycle() {

**super**();

}

**public** ElectricBicycle(**int** wheel, String color) {

**super**(wheel, color);

}

}

/\*

4.定义测试类Test

a)提供main方法

b)在main方法中

①　创建Car 对象c，把颜色赋值为白色，把轮子个数赋值为4

②　调用c对象成员方法run方法

③　 调用c对象的成员方法addGasoline()

④　 创建ElectricBicycle对象 eb，吧颜色赋值为黑色，轮子个数赋值为2

⑤　调用ed对象成员方法run方法

⑥　 调用ed对象成员方法 charge方法

\*/

**public** **class** Test {

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

// ①　创建Car 对象c，把颜色赋值为白色，把轮子个数赋值为4

Car c = **new** Car(4, "白色");

// ②　调用c对象成员方法run方法

c.run();

// ③　 调用c对象的成员方法addGasoline()

c.addGasoline();

// ④　 创建ElectricBicycle对象 eb，吧颜色赋值为黑色，轮子个数赋值为2

ElectricBicycle eb = **new** ElectricBicycle(2, "黑色");

// ⑤　调用ed对象成员方法run方法

eb.run();

// ⑥　 调用ed对象成员方法 charge方法

eb.charge();

}

}

## 训练案例3

### 训练描述

请使用代码描述

在极客营有很多员工(Employee),按照工作内容不同分教研部员工(Teacher)和行政部员工(AdminStaff)

教研部根据教学的方式不同又分为讲师(Lecturer)和助教(Tutor)

行政部根据负责事项不同,有分为维护专员(Maintainer),采购专员(Buyer)

公司的每一个员工都编号,姓名和其负责的工作

工作内容:

讲师: 工号为 666 的讲师 傅红雪 在讲课

助教: 工号为 668的助教 顾棋 在帮助学生解决问题

维护专员: 工号为 686 的维护专员 庖丁 在解决不能共享屏幕问题

采购专员: 工号为 888 的采购专员 景甜 在采购音响设备

提示: 参考今天的综合案例

### 操作步骤描述

一. 分析继承关系

1. 顶层类： 员工类(Employee)
2. 教研部员工（Teacher）和行政部员工(AdminStaff)继承员工类(Employee)
3. 讲师(Lecturer)和助教(Tutor)继承教研部员工(Teacher)
4. 维护专员(Maintainer),采购专员(Buyer) 继承行政部员工(AdminStaff)
5. 代码实现步骤:
6. 定义抽象类员工类(Employee)
   1. 成员变量: 工号(id),姓名(name)
   2. 抽象方法: void work();
   3. 提供无参和带参的构造方法以及setters和getters
7. 定义抽象类教研部员工（Teacher）类继承员工类(Employee)
   1. 提供无参和带参的构造方法
8. 定义抽象类行政部员工(AdminStaff)类继承员工类(Employee)
   1. 提供无参和带参的构造方法
9. 定义讲师(Lecturer)类继承研部员工（Teacher）类
   1. 提供无参和带参的构造方法
   2. 实现抽象方法: void work();

输出格式: 工号为 666 的讲师 傅红雪 在讲课

1. 定义助教(Tutor)类继承研部员工（Teacher）类
   1. 提供无参和带参的构造方法
   2. 实现抽象方法: void work();
      1. 输出格式: 工号为 668的助教 顾棋 在帮助学生解决问题
2. 定义维护专员(Maintainer)类继承行政部员工(AdminStaff)类
   1. 提供无参和带参的构造方法
   2. 实现抽象方法: void work();
      1. 输出格式: 工号为 686 的维护专员 庖丁 在解决不能共享屏幕问题
3. 定义采购专员(Buyer) 类继承行政部员工(AdminStaff)类
   1. 提供无参和带参的构造方法
   2. 实现抽象方法: void work();

输出格式: 工号为 888 的采购专员 景甜 在采购音响设备

1. 定义测试类Test
   1. 创建讲师对象 l, 把工号赋值为666,姓名赋值为”傅红雪”
   2. 调用讲师对象l的工作方法
   3. 创建助教对象 t, 把工号赋值为668,姓名赋值为”顾棋”
   4. 调用助教对象t的工作方法
   5. 创建维护专员对象 m, 把工号赋值为686,姓名赋值为”庖丁”
   6. 调用维护专员对象m的工作方法
   7. 创建采购专员对象 b, 把工号赋值为888,姓名赋值为”景甜”
   8. 调用采购专员对象b的工作方法

### 操作步骤答案

/\*

1.定义抽象类员工类(Employee)

a)成员变量: 工号(id),姓名(name)

b)抽象方法: void work();

c)提供无参和带参的构造方法以及setters和getters

\*/

**public** **abstract** **class** Employee {

// a)成员变量: 工号(id),姓名(name)

**private** String id;

**private** String name;

// b)抽象方法: void work();

**public** **abstract** **void** work();

// c)提供无参和带参的构造方法以及setters和getters

**public** Employee() {

**super**();

}

**public** Employee(String id, String name) {

**super**();

**this**.id = id;

**this**.name = name;

}

**public** String getId() {

**return** id;

}

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

**this**.id = id;

}

**public** String getName() {

**return** name;

}

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

**this**.name = name;

}

}

/\*

2.定义抽象类教研部员工（Teacher）类继承员工类(Employee)

a)提供无参和带参的构造方法

\*/

**public** **abstract** **class** Teacher **extends** Employee{

// a)提供无参和带参的构造方法

**public** Teacher() {

**super**();

}

**public** Teacher(String id, String name) {

**super**(id, name);

}

}

/\*

3.定义抽象类行政部员工(AdminStaff)类继承员工类(Employee)

a)提供无参和带参的构造方法

\*/

**public** **abstract** **class** AdminStaff **extends** Employee {

// a)提供无参和带参的构造方法

**public** AdminStaff() {

**super**();

}

**public** AdminStaff(String id, String name) {

**super**(id, name);

}

}

/\*

4.定义讲师(Lecturer)类继承研部员工（Teacher）类

a)提供无参和带参的构造方法

b)实现抽象方法: void work();

输出格式: 工号为 666 的讲师 傅红雪 在讲课

\*/

**public** **class** Lecturer **extends** Teacher {

// a)提供无参和带参的构造方法

**public** Lecturer() {

**super**();

}

**public** Lecturer(String id, String name) {

**super**(id, name);

}

// b)实现抽象方法: void work();

// 输出格式: 工号为 666 的讲师 傅红雪 在讲课

**public** **void** work() {

System.***out***.println("工号为 "+getId()+" 的讲师 "+getName()+" 在讲课");

}

}

/\*

5.定义助教(Tutor)类继承研部员工（Teacher）类

a)提供无参和带参的构造方法

b)实现抽象方法: void work();

i.输出格式: 工号为 668的助教 顾棋 在帮助学生解决问题

\*/

**public** **class** Tutor **extends** Teacher{

// a)提供无参和带参的构造方法

**public** Tutor() {

**super**();

}

**public** Tutor(String id, String name) {

**super**(id, name);

}

// b)实现抽象方法: void work();

// i.输出格式: 工号为 668的助教 顾棋 在帮助学生解决问题

**public** **void** work() {

System.***out***.println("工号为 "+getId()+"的助教 "+getName()+" 在帮助学生解决问题");

}

}

/\*

6.定义维护专员(Maintainer)类继承行政部员工(AdminStaff)类

a)提供无参和带参的构造方法

b)实现抽象方法: void work();

i.输出格式: 工号为 686 的维护专员 庖丁 在解决不能共享屏幕问题

\*/

**public** **class** Maintainer **extends** AdminStaff {

// a)提供无参和带参的构造方法

**public** Maintainer() {

**super**();

}

**public** Maintainer(String id, String name) {

**super**(id, name);

}

// b)实现抽象方法: void work();

// i.输出格式: 工号为 686 的维护专员 庖丁 在解决不能共享屏幕问题

**public** **void** work() {

System.***out***.println("工号为 "+getId()+" 的维护专员 "+getName()+" 在解决不能共享屏幕问题");

}

}

/\*

7.定义采购专员(Buyer) 类继承行政部员工(AdminStaff)类

a)提供无参和带参的构造方法

b)实现抽象方法: void work();

输出格式: 工号为 888 的采购专员 景甜 在采购音响设备

\*/

**public** **class** Buyer **extends** AdminStaff {

// a)提供无参和带参的构造方法

**public** Buyer() {

**super**();

}

**public** Buyer(String id, String name) {

**super**(id, name);

}

// b)实现抽象方法: void work();

// 输出格式: 工号为 888 的采购专员 景甜 在采购音响设备

**public** **void** work() {

System.***out***.println("工号为 "+getId()+" 的采购专员 "+getName()+" 在采购音响设备");

}

}

/\*

8.定义测试类Test

a)创建讲师对象 l, 把工号赋值为666,姓名赋值为”傅红雪”

b)调用讲师对象l的工作方法

c)创建助教对象 t, 把工号赋值为668,姓名赋值为”顾棋”

d)调用助教对象t的工作方法

e)创建维护专员对象 m, 把工号赋值为686,姓名赋值为”庖丁”

f)调用维护专员对象m的工作方法

g)创建采购专员对象 b, 把工号赋值为888,姓名赋值为”景甜”

h)调用采购专员对象b的工作方法

\*/

**public** **class** Test {

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

// a)创建讲师对象 l, 把工号赋值为666,姓名赋值为”傅红雪”

Lecturer l = **new** Lecturer("666", "傅红雪");

// b)调用讲师对象l的工作方法

l.work();

// c)创建助教对象 t, 把工号赋值为668,姓名赋值为”顾棋”

Tutor t = **new** Tutor("668", "顾棋");

// d)调用助教对象t的工作方法

t.work();

// e)创建维护专员对象 m, 把工号赋值为686,姓名赋值为”庖丁”

Maintainer m = **new** Maintainer("686", "庖丁");

// f)调用维护专员对象m的工作方法

m.work();

// g)创建采购专员对象 b, 把工号赋值为888,姓名赋值为”景甜”

Buyer b = **new** Buyer("888", "景甜");

// h)调用采购专员对象b的工作方法

b.work();

}

}