**类与对象**

1. **实验目的**
2. 创建一个类和一个对象，调用这个对象的属性和方法。
3. 创建对象给对象定义无参构造方法和有参构造方法。
4. 通过所学知识完成选择测试题。
5. 使用this关键字，设置对象的成员变量。
6. 为“无名的粉”写一个类，编写该类的构造方法和成员变量。
7. 使用static关键词设置方法和变量的属性。
8. **实验内容**
9. 根据提示，在右侧编辑器Begin-End处补充代码：

声明一个Dog类，给Dog类添加三个String类型的属性，分别是name，color，variety，定义两个方法，分别是，eat，run。

在main方法中创建一个名为wuhuarou的Dog对象，并设置name为五花肉，color为棕色，variety为阿拉斯加。

使得调用wuhuarou对象的eat方法能输出啃骨头，调用run方法能输出叼着骨头跑。

1. 根据提示，在右侧编辑器Begin-End处补充代码:

创建一个Person类，给Person类定义一个无参构造方法和一个有参构造方法，有参构造方法定义两个String类型参数name和sex在main方法中调用无参构造和有参构造方法；

调用无参构造方法输出：一个人被创建了；

调用有参构造方法应根据输入的数据输出对应的结果，如输入：张三，男，则输出：姓名：张三，性别：男，被创建了。

3、 选择题

4. 右侧编辑器中已经编写好了代码，但是输出结果不能让我们满意，请根据所学知识修改代码，使其能输出如下结果： name：baby age：45 sex：女

5. 为“无名的粉”写一个类：class WuMingFen 要求：

有三个属性：面码:String theMa 粉的分量(两)：int quantity是否带汤：boolean likeSoup；

写一个构造方法，以便于简化初始化过程，如： WuMingFen f1 = new WuMingFen("牛肉",3,true);；

重载构造方法，使得初始化过程可以多样化： WuMingFen f2 = new WuMingFen("牛肉",2);；

如何使得下列语句构造出来的粉对象是酸辣面码、2两、带汤的？ WuMingFen f3 = new WuMingFen();；

写一个普通方法：check()，用于查看粉是否符合要求。即：将对象的三个属性打印在控制台上。

1. 现在编辑器中代码有异常，请利用本关所学知识修改右侧代码使其能输出如下结果：

hello educoder 我叫楚留香 我喜欢在educoder上学习java

注意： 在Java中是不能用static关键字修饰局部变量的，为什么？ 因为这个是老大说的，我们做小弟的不能不听呀。

1. **实验结果**

public class Test {

    public static void main(String[] args) {

        /\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

        //创建Dog对象

        Dog wuhuarou = new Dog();

        //设置Dog对象的属性

        wuhuarou.name = "五花肉";

        wuhuarou.color = "棕色";

        wuhuarou.variety = "阿拉斯加";

        //输出小狗的属性

        System.out.println("名字：" +  wuhuarou.name   + "，毛色：" +  wuhuarou.color  + "，品种：" + wuhuarou.variety );

        //调用方法

        wuhuarou.eat();

        wuhuarou.run();

        /\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

    }

}

//在这里定义Dog类

/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

class Dog{

    String name;

    String color;

    String variety;

    void eat(){

        System.out.println("啃骨头");

    }

    void run(){

            System.out.println("叼着骨头跑");

        }

}

/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

public class Test {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        String name = sc.next();

        String sex = sc.next();

        /\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

        //分别使用两种构造器来创建Person对象

        Person per = new Person();

        Person per1 = new Person(name,sex);

        /\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

    }

}

//创建Person对象，并创建两种构造方法

/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

class Person{

    public Person(){

        System.out.println("一个人被创建了");

    }

    public Person(String name,String sex){

        System.out.println("姓名："+name+"，性别："+sex+"，被创建了");

    }

}

/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

1.

下列关于构造方法的说法不正确的是（ C ）

A、Java语言规定构造方法名必须与类名相同

B、Java语言规定构造方法没有返回值，且不用void关键字声明

C、构造方法不可以重载

D、构造方法只能用new关键字来创建

2.

类Test定义如下：

public class Test{

public float method1(){

System.out.println("方法1");

return 1.5;

}

}

将下列哪些方法插入到第7行是合法的（ CD ）

A、public float method1(){

System.out.println("方法2");

return 1.1;

}

B、public Test1(){

System.out.println("hello");

}

C、public float method1( int a){

System.out.println("方法2");

return a+0.5;

}

D、public Test(){

System.out.println("hello");

}

    /\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

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

        this(age);

        this.name = name;

        this.sex = sex;

    }

    public Person(int age){

        this.age = age;

    }

    public void display(){

        System.out.println("name：" + name);

        System.out.println("age：" + age);

        System.out.println("sex：" + sex);

    }

    /\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

}

public class Test {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        String theMa = sc.next();

        int quantity = sc.nextInt();

        boolean likeSoup = sc.nextBoolean();

        /\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

        //使用三个参数的构造方法创建WuMingFen对象  取名 f1

        WuMingFen f1 = new WuMingFen(theMa, quantity, likeSoup);

        //使用两个参数的构造方法创建WuMingFen对象  取名 f2

        WuMingFen f2 = new WuMingFen(theMa, quantity);

        //使用无参构造方法创建WuMingFen对象  取名 f3

        WuMingFen f3 = new WuMingFen();

        //分别调用三个类的 check方法

        f1.check();

        f2.check();

        f3.check();

        /\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

    }

}

/\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

//在这里添加包名  step4

package step4;

//创建类 添加属性和方法

class WuMingFen{

  String theMa; // 面码

  int quantity; // 粉的分量(两)

  boolean likeSoup; // 是否带汤  boolean默认为false

// 4.如何使得下列语句构造出来的粉对象是酸辣面码、2两、带汤的？

    public WuMingFen() {

      this.theMa = "酸辣";

      this.quantity = 2;

      this.likeSoup = true;

    }

// 2.写一个构造方法，以便于简化初始化过程

    public WuMingFen(String theMa, int quantity, boolean likeSoup) {

        this.theMa = theMa;

        this.quantity = quantity;

        this.likeSoup = likeSoup;

    }

// 3.重载构造方法，使得初始化过程可以多样化：

public WuMingFen(String theMa, int quantity) {

this.theMa = theMa;

this.quantity = quantity;

}

// 5.写一个普通7a686964616fe58685e5aeb931333330336335方法：check()，用于查看粉是否符合要求。即：将对象的三个属性打印在控制台上。

public void check() {

System.out.println("面码：" + this.theMa + "，粉的份量：" + this.quantity

+ "两" + "，是否带汤：" + (this.likeSoup == true ? "true" : "false"));

}

}

/\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

public class Test {

    /\*\*\*\*\*\*\*\*\*\* Begin \*\*\*\*\*\*\*\*\*\*/

    static String name = "楚留香";

    static{

        System.out.println("hello educoder");

    }

    public static void main(String[] args) {

        System.out.println("我叫" + name);

        Test t1 = new Test();

        t1.study();

    }

    public void study(){

        System.out.println("我喜欢在educoder上学习java");

    }

    /\*\*\*\*\*\*\*\*\*\* End \*\*\*\*\*\*\*\*\*\*/

}

1. 有如下代码：

public class TestMain{

public static void main(String args[]){

MyClass mc1 = new MyClass();

MyClass mc2 = new MyClass();

mc1.a = 100;

mc1.b = 200;

mc2.a = 300;

mc2.b = 400;

System.out.println(mc1.a);

System.out.println(mc1.b);

System.out.println(mc2.a);

System.out.println(mc2.b);

}

}

class MyClass{

static int a;

int b;

}

请问输出结果是（ D ）

A、100

100

100

100

B、100

200

300

400

C、400

400

400

400

D、300

200

300

400

2.

class MyClass {

int a;

static int b;

void fa(){

}

static void fb(){

}

public void m1(){

System.out.println(a); //位置1

System.out.println(b); //位置2

fa(); //位置3

fb(); //位置4

}

public static void m2(){

System.out.println(a); //位置5

System.out.println(b); //位置6

fa(); //位置7

fb(); //位置8

}

}

上述代码会出错的位置有：（ EG ）

A、位置1

B、位置2

C、位置3

D、位置4

E、位置5

F、位置6

G、位置7

3.

class MyClass {

static int i = 10;

static {

i = 20;

System.out.println("In Static");

}

public MyClass() {

System.out.println("MyClass()");

}

public MyClass(int i) {

System.out.println("MyClass(int)");

this.i = i;

}

}

public class TestMain {

public static void main(String args[]) {

MyClass mc1 = new MyClass();

System.out.println(mc1.i);

MyClass mc2 = new MyClass(10);

System.out.println(mc2.i);

}

}

上述代码的运行结果是：（ B ）

A、MyClass()

20

MyClass(int)

10

B、In Static

MyClass()

20

MyClass(int)

10

C、In Static

MyClass(int)

20

MyClass()

10

D、In Static

MyClass()

10

MyClass(int)

20

1. **实验总结**
2. 类：类是一个模板，它描述一类对象的行为和属性。

对象：对象是类的一个实例，有属性和行为 。

1. 对象被创建的时候会调用的方法，对象在被创建的时候，也就是被new的时候，会自动调用构造方法。
2. 包如windows系统中的文件夹一样，只不过包是专门存放代码文件的。

包的主要作用就是给代码文件分类。

1. 如果一个类被定义在某一个包下那么在定义类的时候需要声明包名，否则程序会报错。
2. 关于静态代码块只需要记住一句话：在类被加载的时候运行且只运行一次。