一个程序就是一个世界，有很多事物(对象[属性, 行为])

类与对象：

1) 类是抽象的，概念的，代表一类事物,比如人类,猫类.., 即它是数据类型.

2) 对象是具体的，实际的，代表一个具体事物, 即 是实例.

3) 类是对象的模板，对象是类的一个个体，对应一个实例

属性：

属性是类的一个组成部分,一般是基本数据类型,也可是引用类型(对象,数组)。

属性细节：

1) 属性的定义语法同变量，示例：访问修饰符 属性类型 属性名;

有四种访问修饰符 public, proctected, 默认, private ,后面我会详细介绍

2) 属性的定义类型可以为任意类型，包含基本类型或引用类型

3) 属性如果不赋值，有默认值，规则和数组一致。具体说: int 0，short 0, byte 0, long 0, float 0.0,double 0.0，char \u0000，boolean false，String null

创建对象：

1) 先声明再创建

```
Cat cat ; //声明对象 cat
cat = new Cat(); //创建
```

2) 直接创建

```
Cat cat = new Cat();
```

访问属性：对象名.属性名;

Java 的内存结构：

1) 栈： 一般存放基本数据类型(局部变量)

2) 堆： 存放对象(Cat cat , 数组等)

3) 方法区：常量池(常量，比如字符串)， 类加载信息

成员方法：

在某些情况下，我们要需要定义成员方法(简称方法)。

访问修饰符 返回数据类型 方法名（形参列表..） {//方法体

语句；

return 返回值;

}

1）形参列表：表示成员方法输入

2）返回数据类型：表示成员方法输出，void 表示没有返回值

3）方法主体：表示为了实现某一功能代码块

4）return 语句不是必须的。

使用细节：

访问修饰符 （作用是控制 方法使用的范围)如果不写默认访问，[有四种: public, protected, 默认, private]

返回数据类型

1）一个方法最多有一个返回值［思考，如何返回多个结果 返回数组 ］

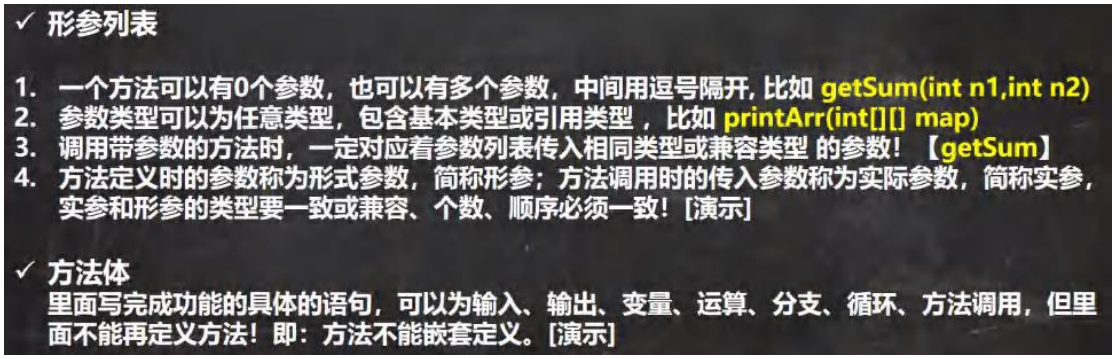
2）返回类型可以为任意类型，包含基本类型或引用类型(数组，对象)

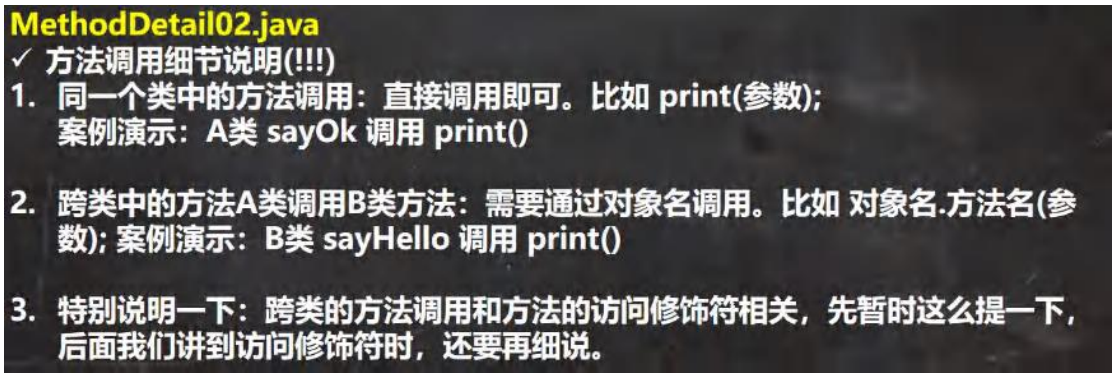
3） 如果方法要求有返回数据类型，则方法体中最后的执行语句必须为 return 值; 而且要求返回值类型必须和 return 的值类型一致或兼容

4）如果方法是 void，则方法体中可以没有 return 语句，或者 只写 return ;

方法名

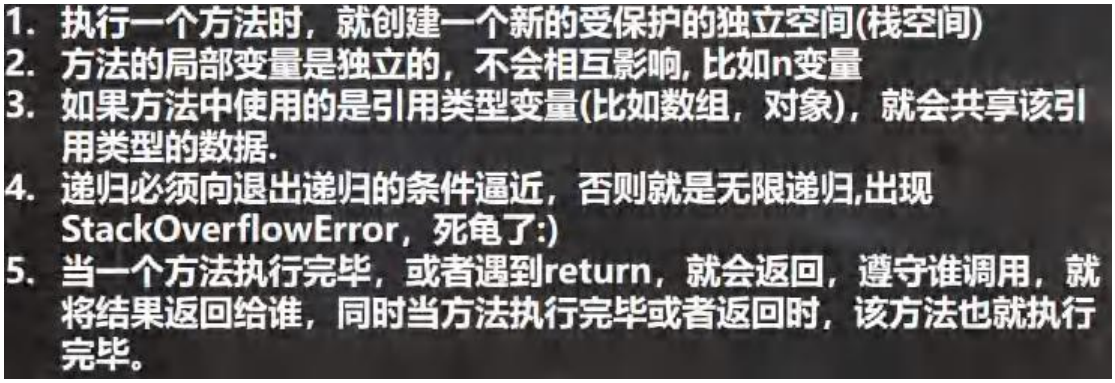
遵循驼峰命名法，最好见名知义，表达出该功能的意思即可。





方法的递归调用：

递归就是方法自己调用自己

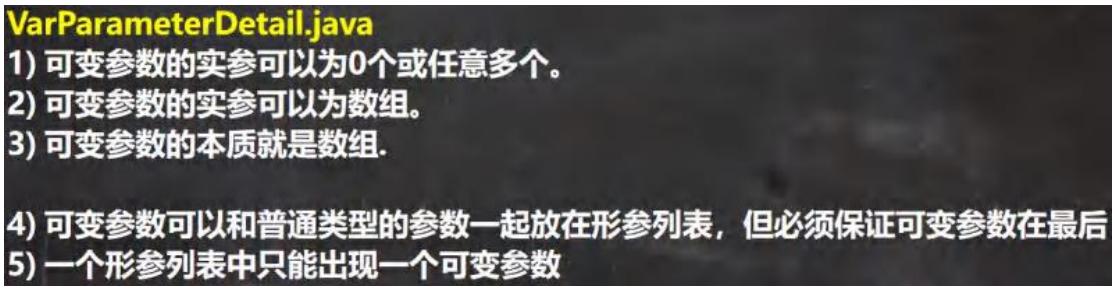


方法的重载：java 中允许同一个类中，多个同名方法的存在，但要求 形参列表不一致！

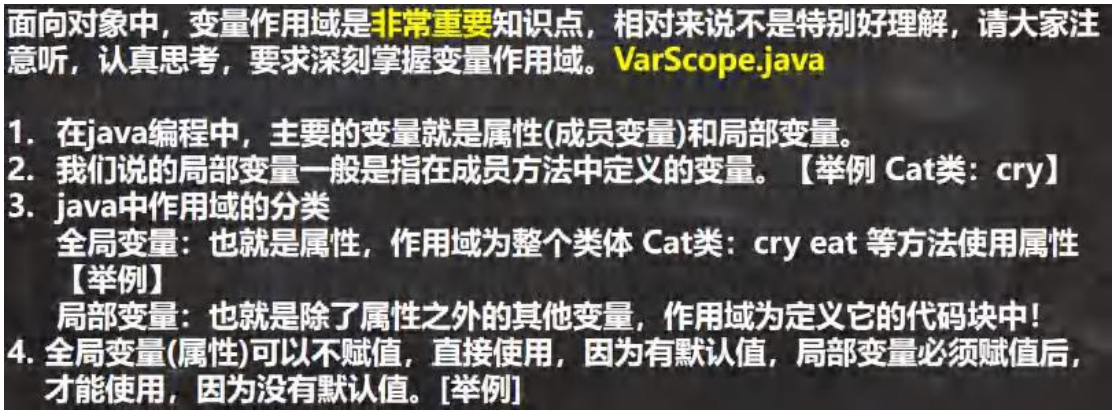
可变参数：java 允许将同一个类中多个同名同功能但参数个数不同的方法，封装成一个方法。

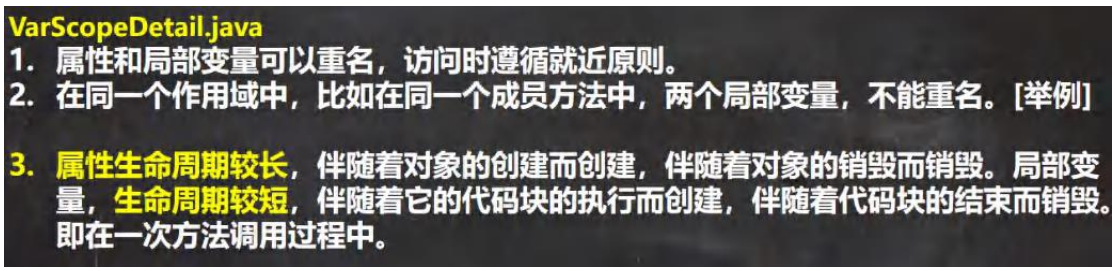
访问修饰符 返回类型 方法名(数据类型... 形参名) {

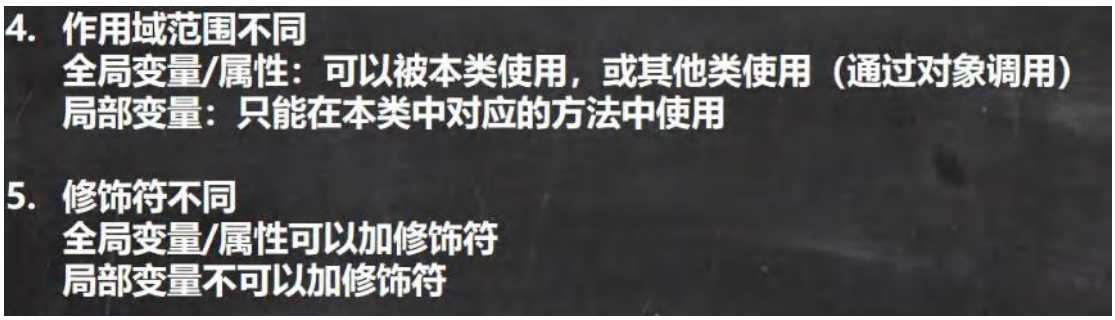
}



作用域：







构造器：

[修饰符] 方法名(形参列表){

方法体;

}

1) 构造器的修饰符可以默认， 也可以是 public protected private

2) 构造器没有返回值

3) 方法名 和类名字必须一样

4) 参数列表 和 成员方法一样的规则

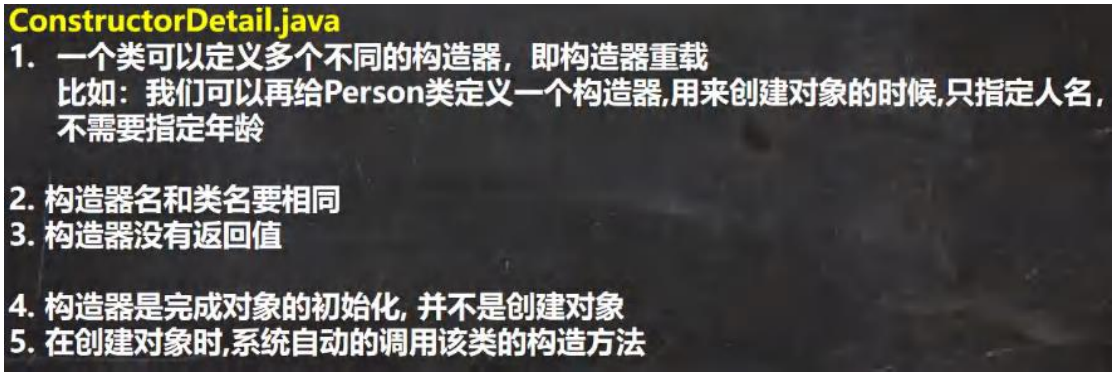
5) 构造器的调用，由系统完成

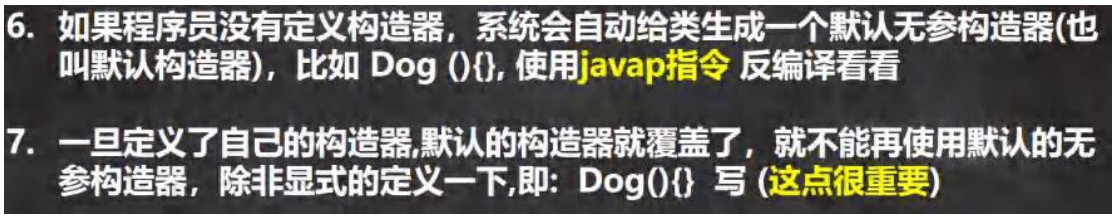
构造方法又叫构造器(constructor)，是类的一种特殊的方法，它的主要作用是完成对新对象的初始化。它有几个特点：

1) 方法名和类名相同

2) 没有返回值

3）在创建对象时，系统会自动的调用该类的构造器完成对象的初始化。





This 关键字：

1）this 关键字可以用来访问本类的属性、方法、构造器

2）this 用于区分当前类的属性和局部变量

3）访问成员方法的语法：this.方法名(参数列表)；

4）访问构造器语法：this(参数列表)；注意只能在构造器中使用(即只能在构造器中访问另外一个构造器，必须放在第一
条语句)

5）this 不能在类定义的外部使用，只能在类定义的方法中使用。