生成随机数：



# 八、面向对象编程（中级）

**Idea**

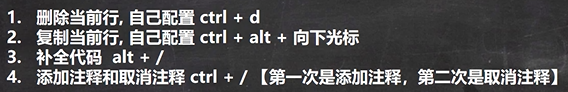
IDE：继承开发环境

Src: 存放 .java源代码

Out: 存放 .class字节码文件

Idea中run一个程序时，先编译成 .class文件，再运行

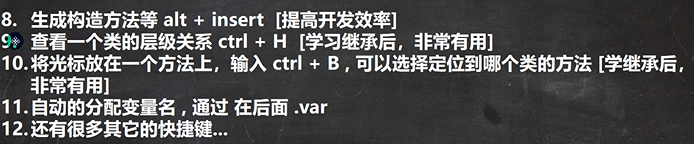
Idea常用快捷键：





6. 快速格式化代码： ctrl + shift + L





查看源码：ctrl + b 或 ctrl + 鼠标左键

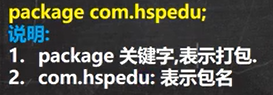
Idea模板：

Main sout fori

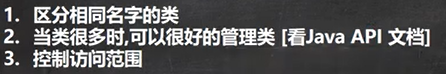


**包**

语法：



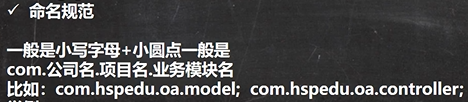
作用：



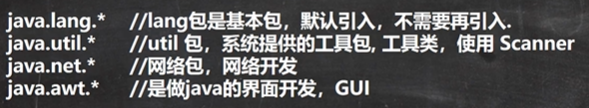
本质：不同文件夹，存放功能相似的类、接口

包的命名：



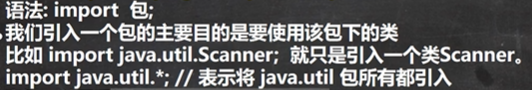


常用包：

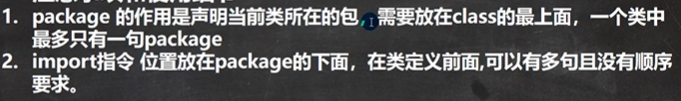


包使用细节

包导入：



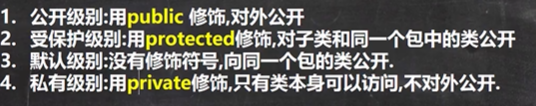
注意事项：

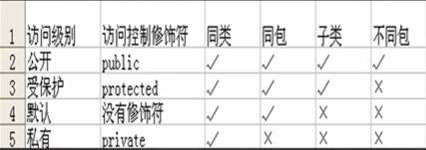


**访问修饰符**

作用：控制方法、属性的访问权限

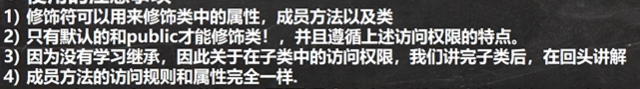
四种访问控制修饰符：





同包子类可访问父类属性、方法；不同包不可

使用细节：



**面向对象三大特征：封装、继承、多态**

**封装（encapsulation）：**

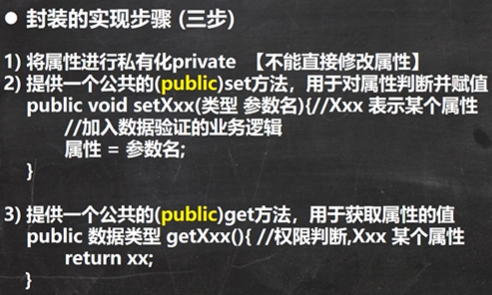
概念：



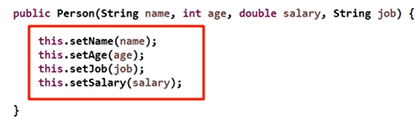
好处：



封装实现步骤：



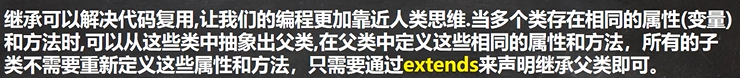
构造器与setXxx结合（用于初始化属性时，验证数据）：



**继承（extends）**

作用：解决代码复用性（多个类有多个属性、方法相同）

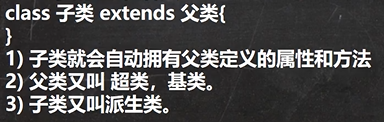
代码扩展性、维护性提高（直接改变父类）



示意图：

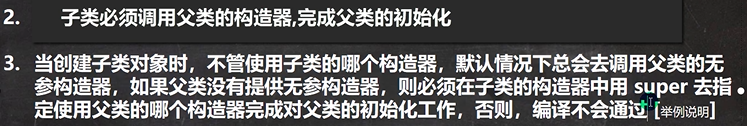


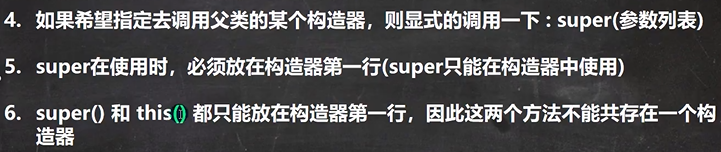
基本语法：



继承细节：



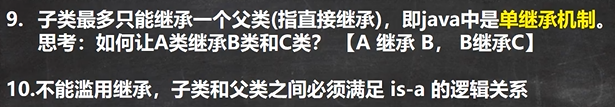




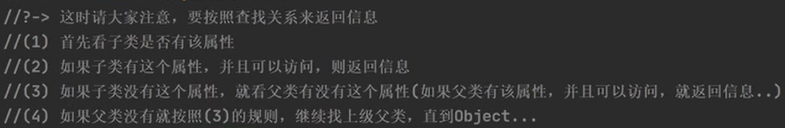
This(): 本类构造器



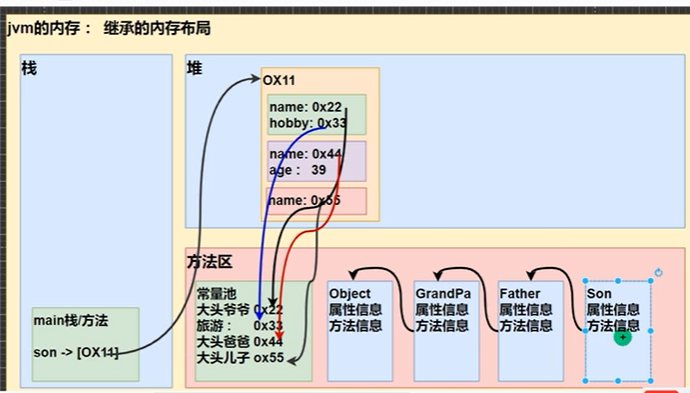
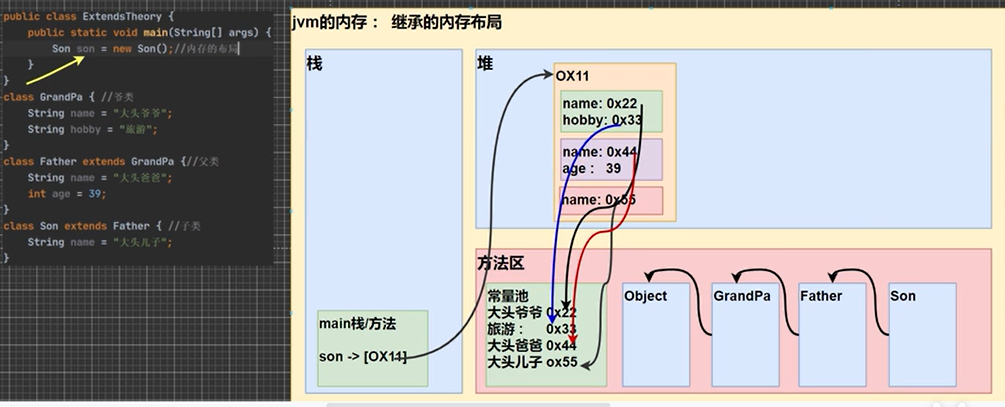




调用子类属性顺序：



继承内存调用机制

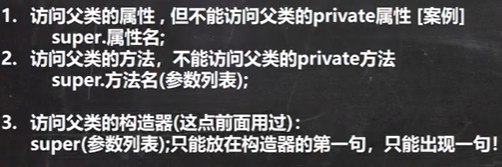


问题1：为什么不是三个对象空间

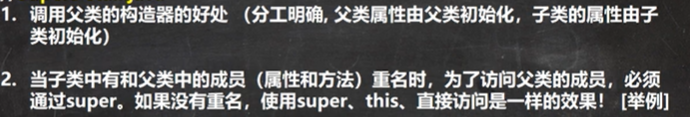
**Super**

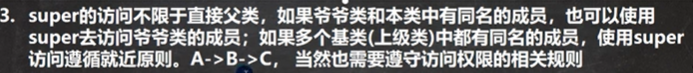
概念：代表父类引用，用于访问符类属性、方法、构造器

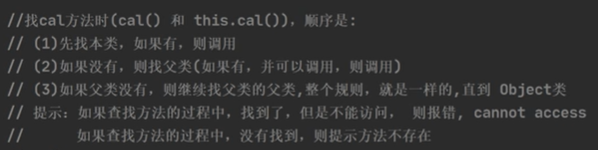
基本语法：



super关键字的好处：



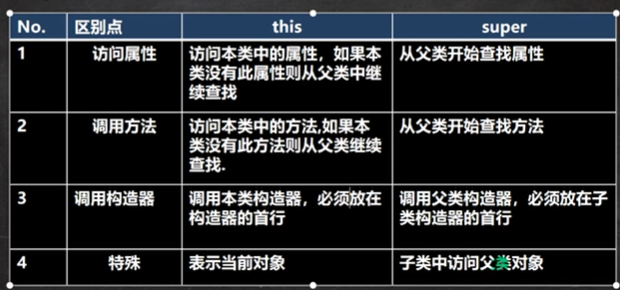




问题2：父类、子类属性是否在一个堆空间，为何不能访问父类私有的属性



Super与this关键字的区别：



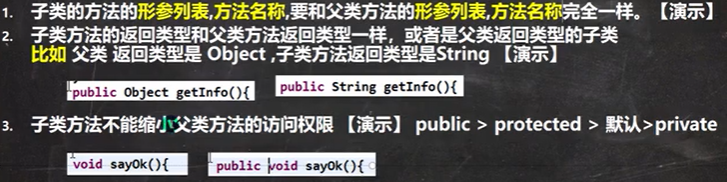
**方法重写(方法覆盖)（override）**

问题3：方法不能重名（除了重写），属性可以。重写a.方法调用A的，b.方法调用B的

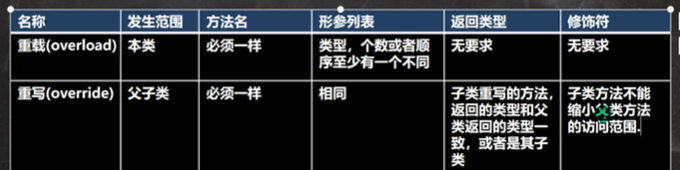
向上转型调用父类属性

概念：子类某方法的名称、返回类型、参数（方法三要素）一样，称子类的这个方法覆盖了父类（不一定是直接父类）的方法

细节：



重写与重载不同：



**多态（多种状态）**

作用：解决代码复用性不高（同一方法需传入不同类别的参数）、不利于维护的特点

概念：对象或方法（重载、重写）有多种形态，建立在封装、继承基础之上

**对象的多态**：



编译类型（表面）、运行类型（实际）：

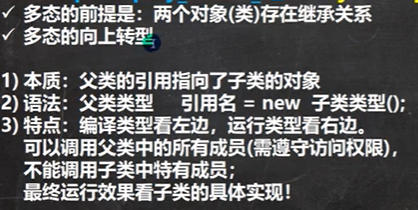


编译阶段能调用那些成员由编译类型决定

向上转型、向下转型（基本数据类型为自动类型转换、强制类型转换）

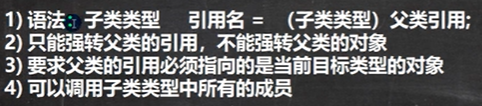
向上转型：父类引用 指向子类对象

向上转型细节：

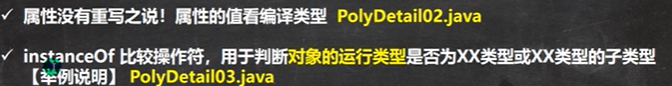


子类具体实现即重写了父类方法

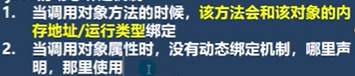
向下转型（常与instanceof结合使用）：



多态注意细节（instanceof）：



**Java动态绑定机制（方法才有，重要）**（属性看自己，方法看子类）



**多态应用：**

**1 多态数组**

概念：数组定义类型为父类类型，里面存放的实际元素类型为子类类型

**2 多态参数**

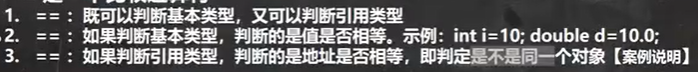
概念：方法定义形参类型为父类类型，实参类型允许为子类类型

**Object类（顶级类）**

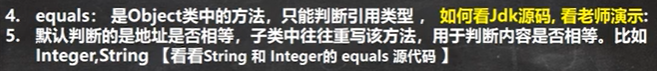


**Equals()方法**

==：一个比较运算符，非方法



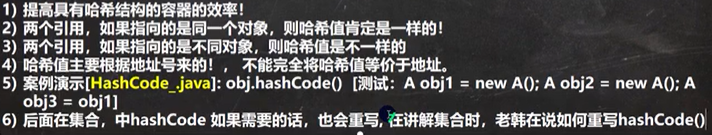
Equals：



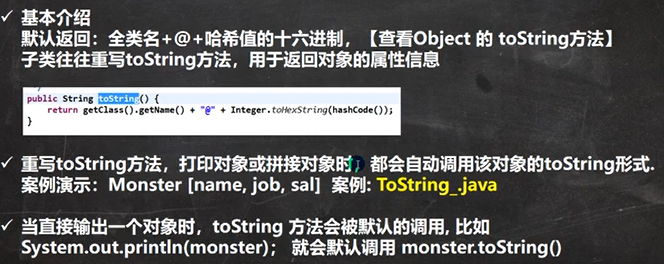
**HashCode()方法：**

作用：返回该对象的哈希码值（hashCode值）（不是内存中真正的地址）

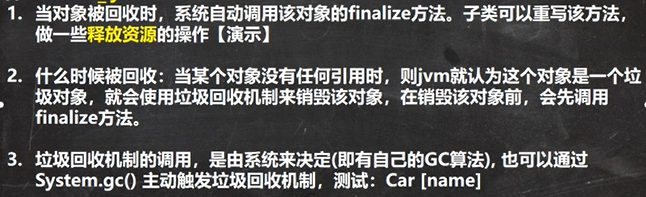
小节：



**toString()方法**

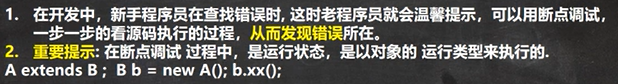


**Finalize()方法（开发中用的较少）：**

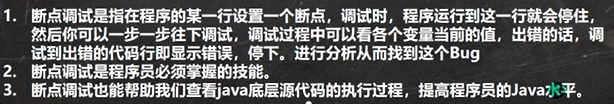


**断点调试（Debug）**

需求：



要点：

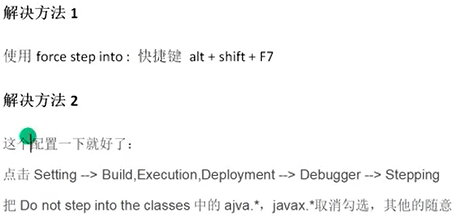


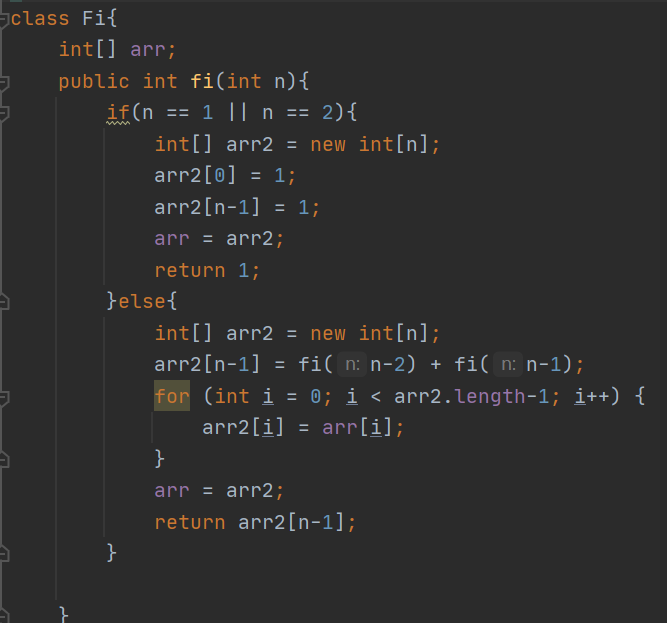
调试快捷键：



F9 resume可以动态下断点，检查代码是否会执行到断点处，进行逻辑检验

进入源码：





学校，学院信息管理系统