类与对象

# 引言

1. 类与对象的定义；
2. 对象的产生于内存划分；
3. 初步认识对象的引用传递。

# 具体内容

## 类与对象基本概念

在现实生活之中，类指的就是具备某一共性的群体集合。而对象是一种个性的描述。如果要想进行更简单的理解，对象包含的是具体的信息内容，例如：陶先生、李先生。

对象所能够具备的操作行为实际上是由类来定义的，也就是说要使用类就必须要有一个具体的描述对象，而对象的操作完全由类来定义。对象是可以明确使用的，而类是不能够直接使用的。

真实存在的可以使用的就是对象，而定义概念的就是类。

## 类与对象的定义

在开发之中一定要记住，是先有类，再有对象。在Java中如果要定义类，则可以使用class关键字来描述，而在类中的组成主要有两种：属性（变量）、方法（此方法不是在主类中定义，不由主方法直接调用，所以不static）。

类本身不能够直接使用，所有的类必须通过对象才可以使用。由于类属于引用数据类型（内存关系），所以对象的产生定义语法现在有两种：

* 声明并实例化对象：类名称 对象名称 = new 类名称()；
* 分布进行：
  + 声明对象：类名称 对象名称 = null；
  + 实例化对象： 对象名称 = new 类名称()。

引用数据类型在使用前必须开辟空间，而基本数据类型可以直接赋值，那么开辟空间的关键字就是new。

那么一旦有了对象之后，就可以利用对象来实现类中属性和方法调用，使用原则：

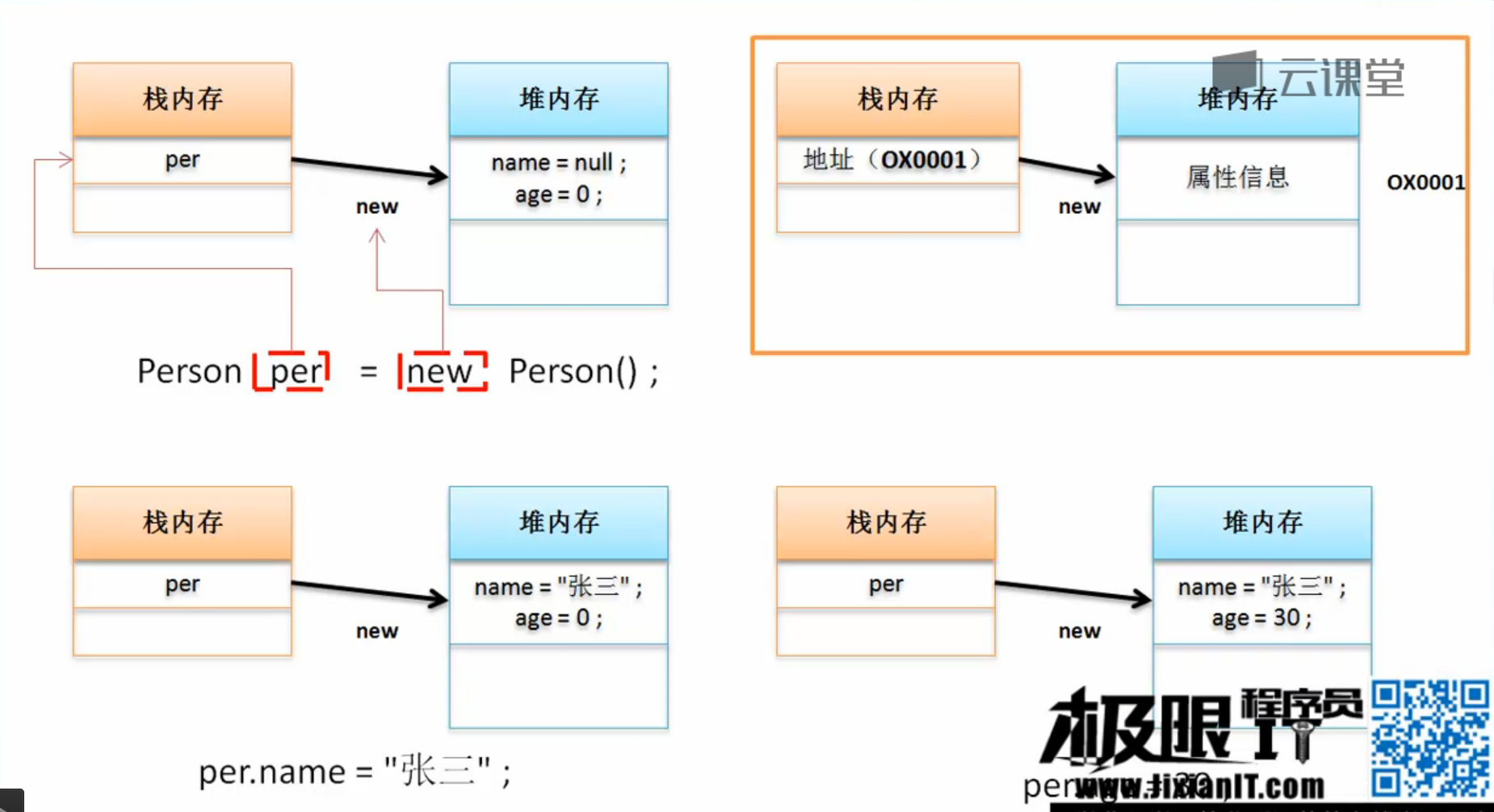
* 调用属性：实例化对象.属性名称；
* 调用方法：实例化对象.方法名称()；

## 对象产生分析

类本身属于引用数据类型，所以对于引用数据类型就必须为其进行内存分析，那么在分析之前，首先给出两块内存空间的定义：

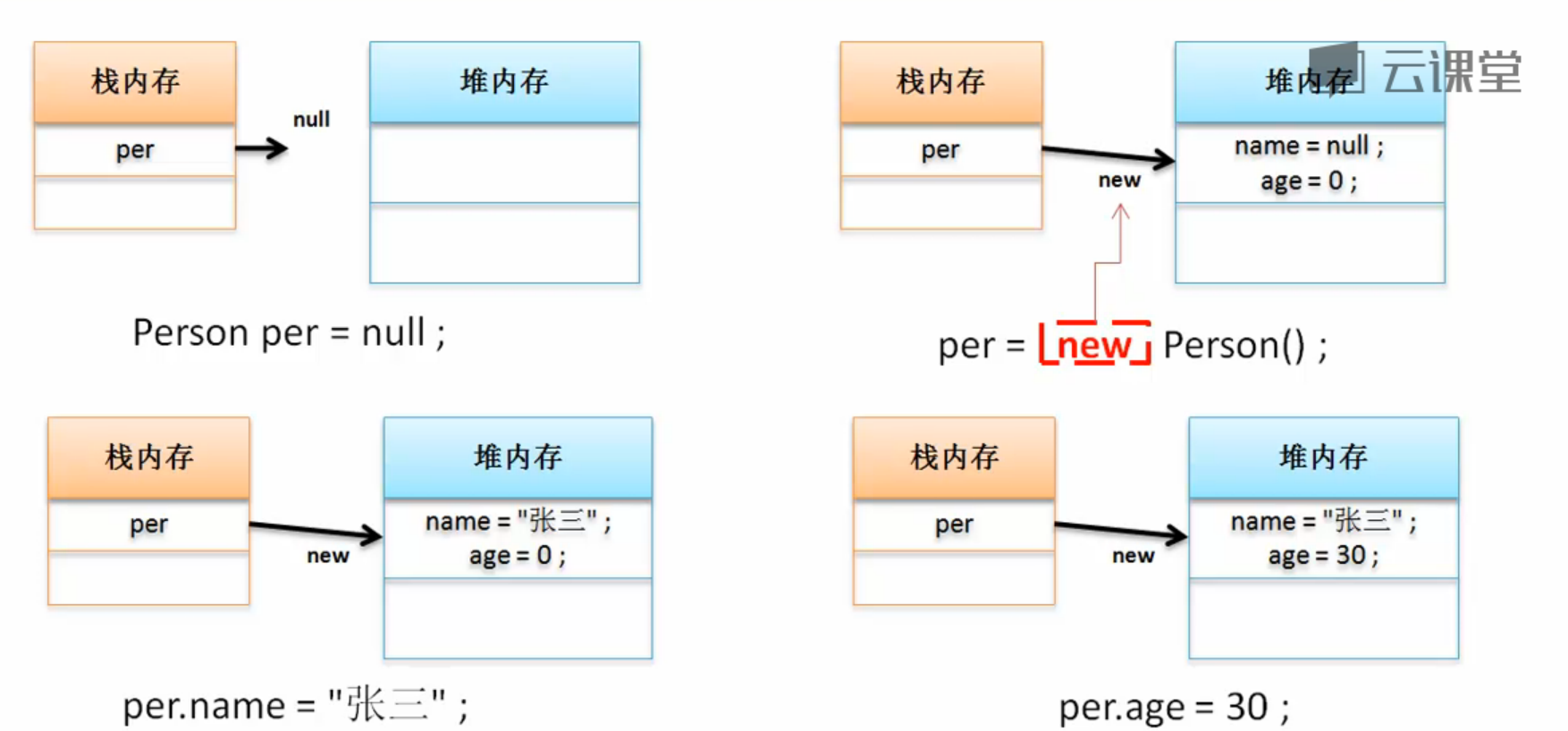
* 堆内存空间（Heap）：保存的是对象中具体的属性信息；
* 栈内存空间（Stack）：是保存的堆内存的地址数值，所以现在可以简化一点，假设保存在栈内存中的是对象的名称，一个栈内存空间只能保存一个地址。

所以现在按照以上的分析，就可以得出如下的内存关系图。



引用数据使用与C语言是完全一样的，所以这个就是Java的灵魂所在。

实际上对象的产生格式本次只使用了一种，还有另外一种分布的方式完成。

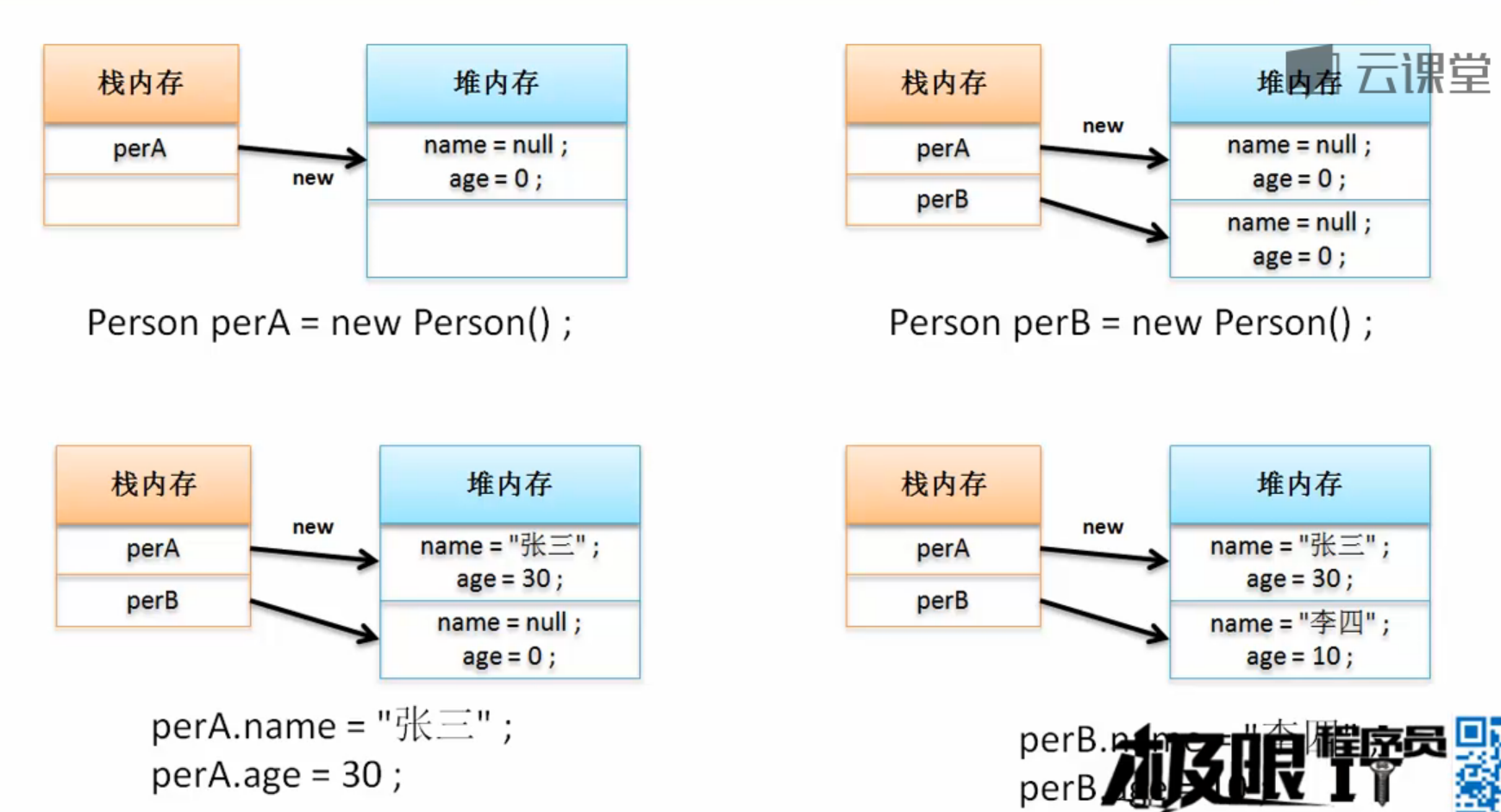


**注意：关于引用数据类型操作存在的重要问题。**

理论上当对象开辟堆内存（实例化对象）那么属性才会进行内存的分配，那么如果说使用了没有实例化的对象呢？

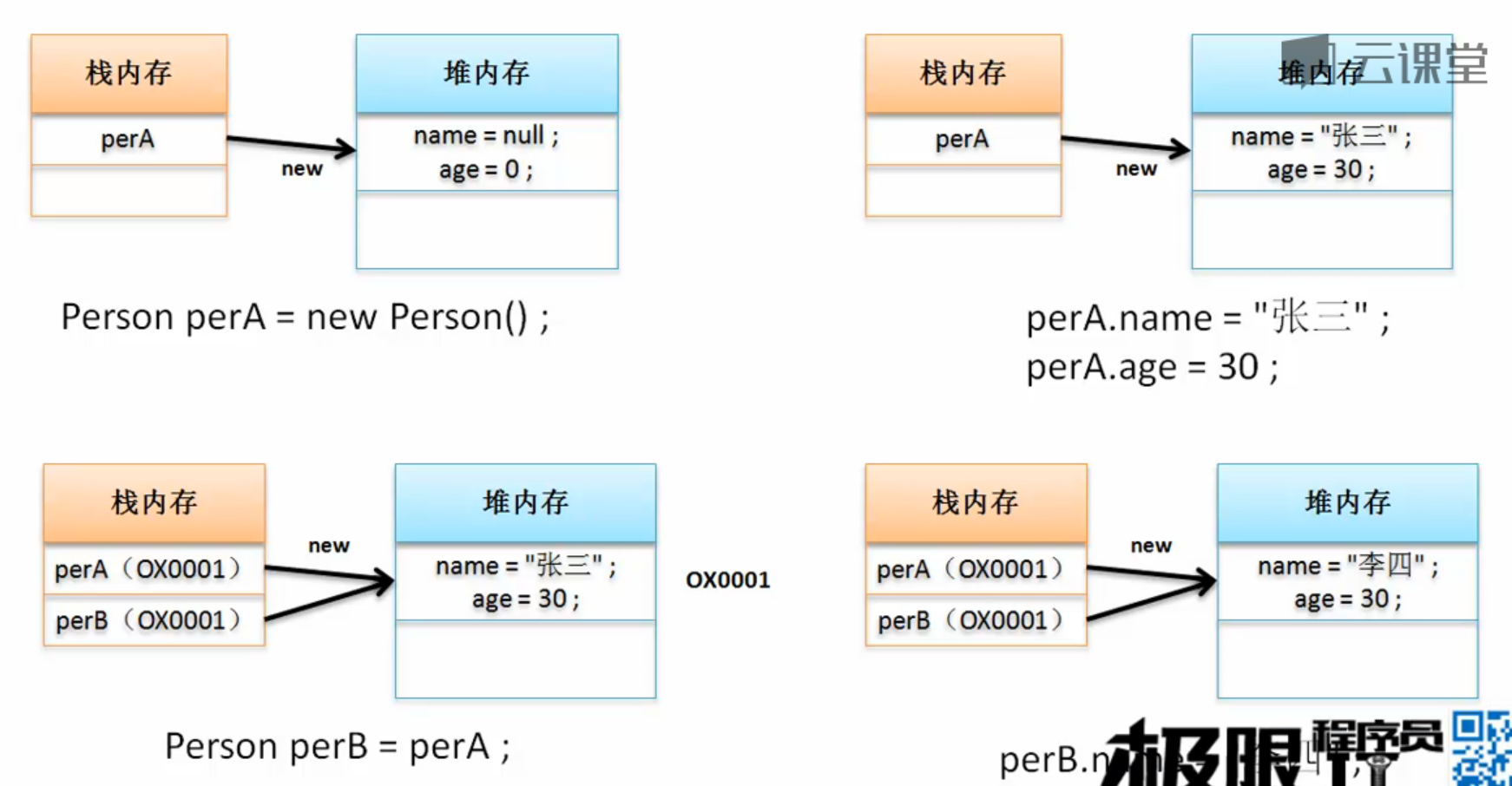
那么此时返回的是“NullPointerException”异常，翻译就是空指向异常。这个异常只有引用数据类型会出现，出现的原因只有：使用了没有开辟堆内存空间的引用对象。

现在既然已经可以开辟一个对象了，那么也就可以开辟两个对象。

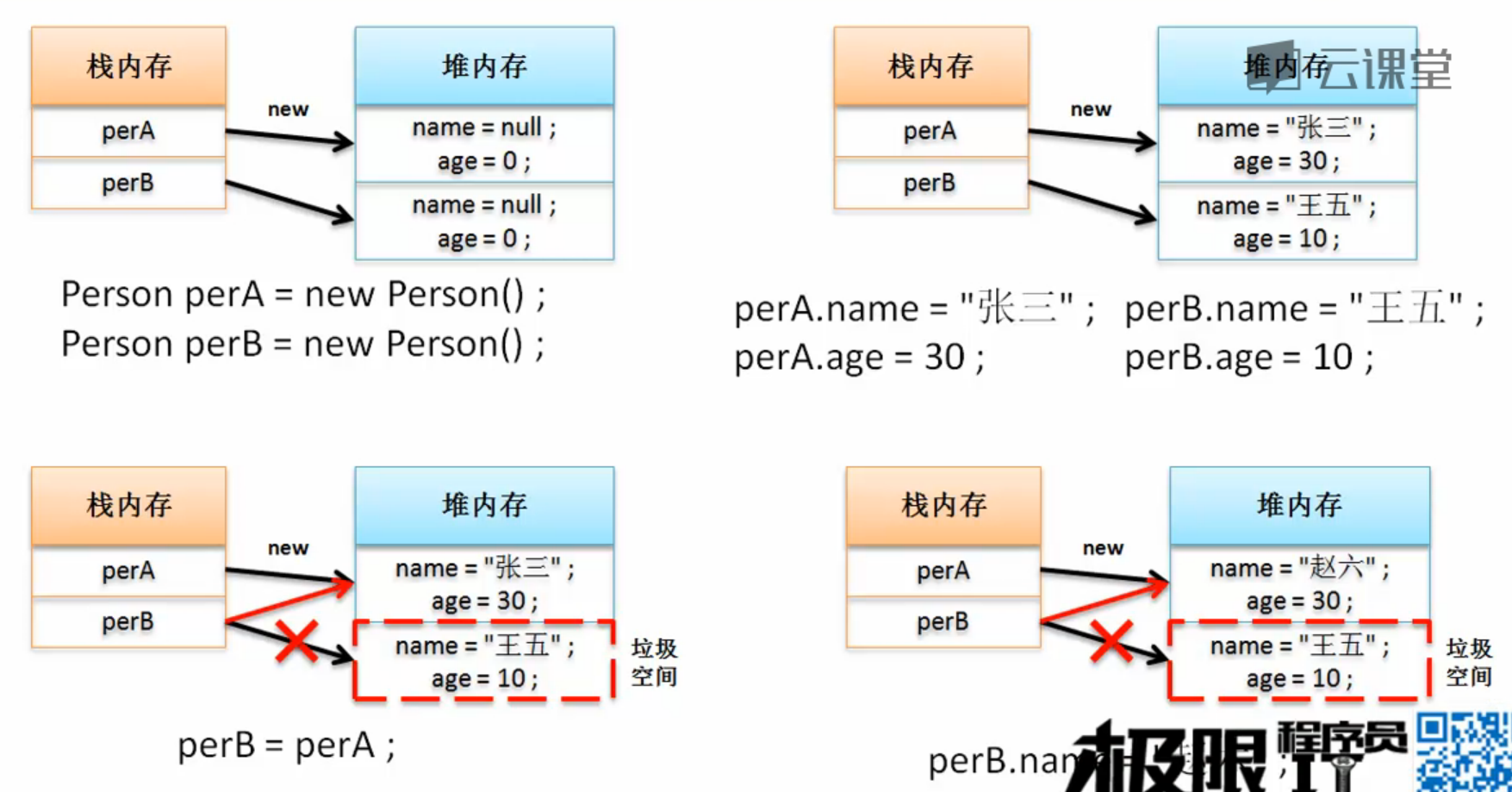


## 初步分析引用传递

引用传递是在引用数据类型上所使用的一个操作定义，其实Java的精髓，其操作的性质与C语言指针是相同的，进行内存的操作。换到程序中，那么就是一块堆内存空间可以同时被多个栈内存所指向。



以上是采用了声明对象的方式进行了引用数据类型的接受，那么如果说此时两个对象都已经明确实例化并设置内容了呢？



通过以上的分析应该可以发现几点：

* 使用关键字new永恒可以开辟新的堆内存空间，堆内存空间保存的是属性；
* 栈内存只能保存一块堆内存的使用地址；
* 引用传递的本质在于同一块堆内存空间可以被不同的栈内存所指向；
* 在发生引用传递时，如果操作的栈内存有堆内存指向，那么改变堆空间就意味着改变内存指向；
* 如果某一块堆内存没有被任何的栈内存指向，那么此空间将成为垃圾空间，所有的垃圾空间会自动被GC（垃圾收集器，Garbage Collector）回收并且释放。由于垃圾的回收是不确定的执行时间，所以最好的做法是尽量少产生垃圾空间。

如果要产生垃圾，除了改变引用之外，也可以设置为null。

那么就是表示此对象放弃原本的指向，变为一个没有指向的栈内存。

# 总结

1. 清楚类与对象的定义格式；
2. 熟练掌握对象的内存分配，以及内存图的描述。