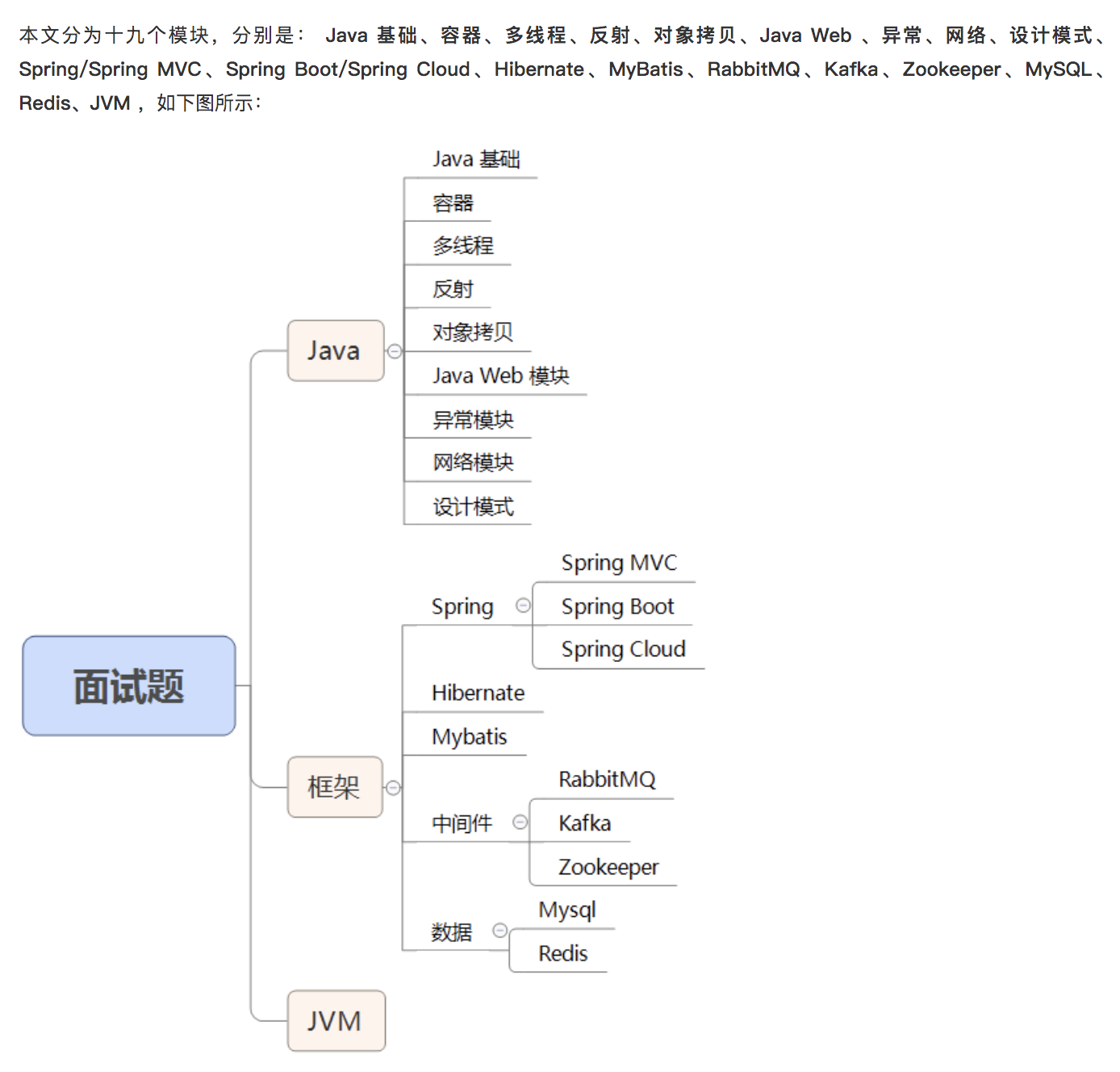
**陈中原：**

****

**王福成：**

JVM内存为什么要分成新生代，老年代，持久代。新生代中为什么要分为Eden和Survivor。

思路： 先讲一下JAVA堆，新生代的划分，再谈谈它们之间的转化，相互之间一些参数的配置（如： –XX:NewRatio，–XX:SurvivorRatio等），再解释为什么要这样划分，最好加一点自己的理解。

1）共享内存区划分

共享内存区 = 持久带 + 堆

持久带 = 方法区 + 其他

Java堆 = 老年代 + 新生代

新生代 = Eden + S0 + S1

2）一些参数的配置

默认的，新生代 ( Young ) 与老年代 ( Old ) 的比例的值为 1:2 ，可以通过参数 –XX:NewRatio 配置。

默认的，Edem : from : to = 8 : 1 : 1 ( 可以通过参数 –XX:SurvivorRatio 来设定)

Survivor区中的对象被复制次数为15(对应虚拟机参数 -XX:+MaxTenuringThreshold)

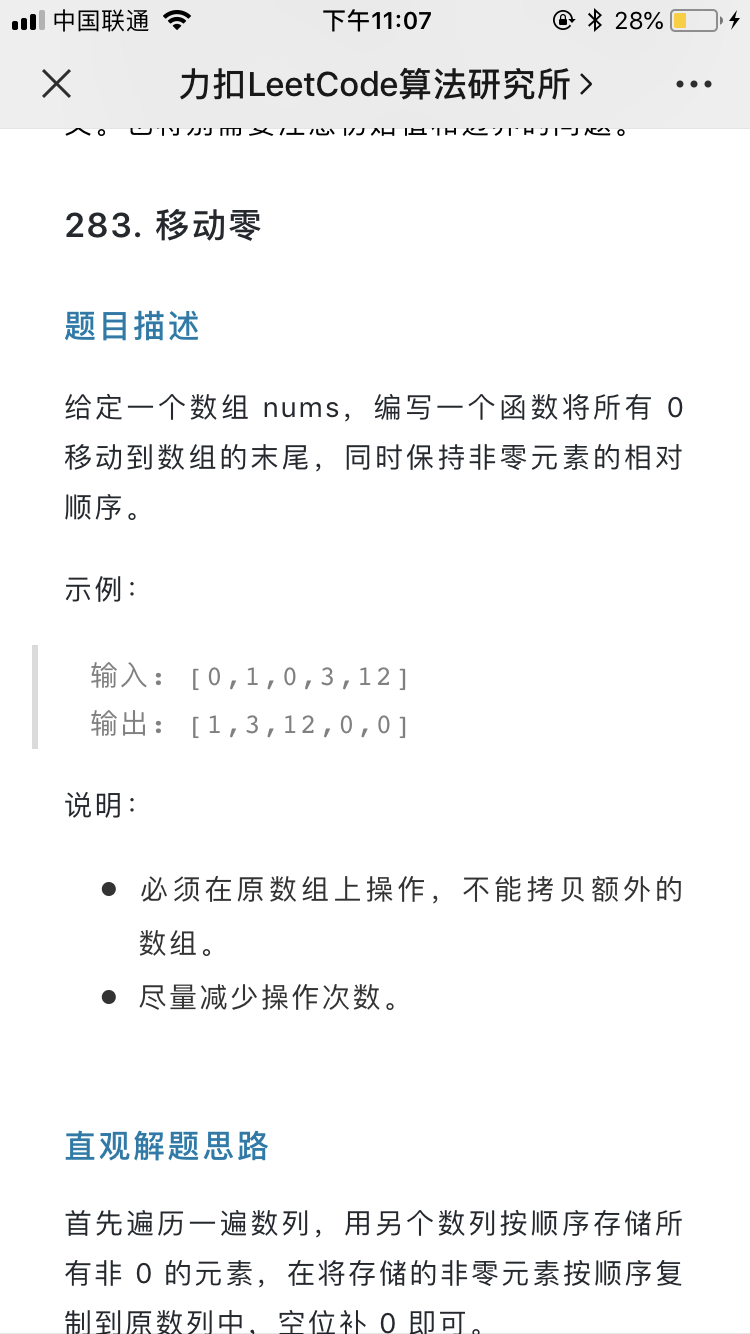
3)为什么要分为Eden和Survivor?为什么要设置两个Survivor区？

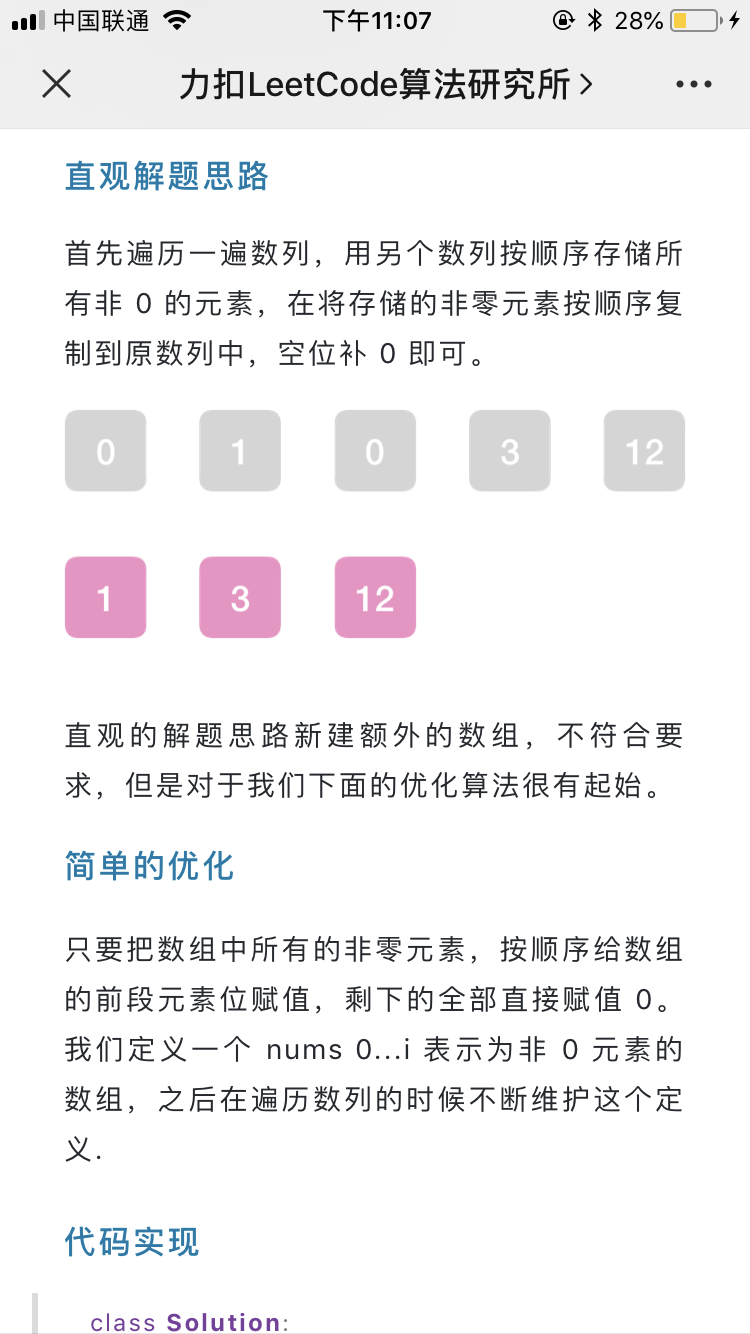
如果没有Survivor，Eden区每进行一次Minor GC，存活的对象就会被送到老年代。老年代很快被填满，触发Major GC.老年代的内存空间远大于新生代，进行一次Full GC消耗的时间比Minor GC长得多,所以需要分为Eden和Survivor。

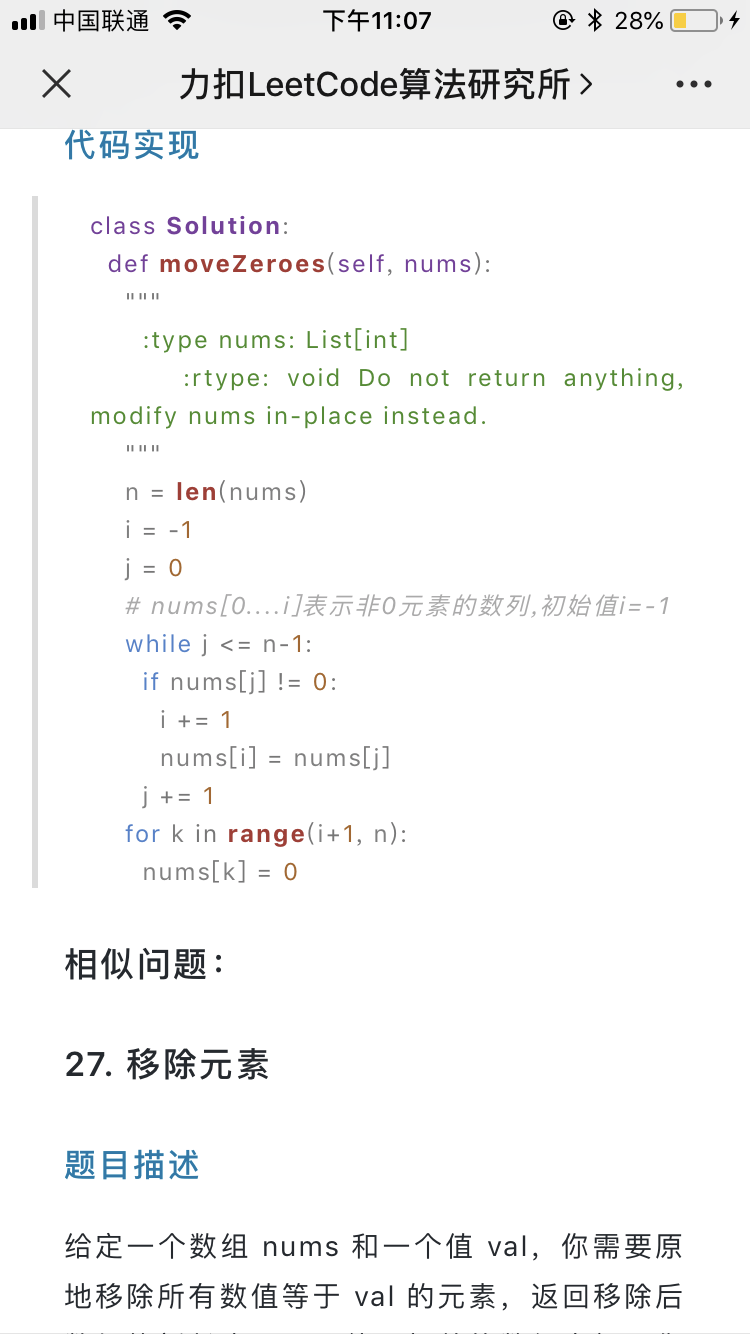
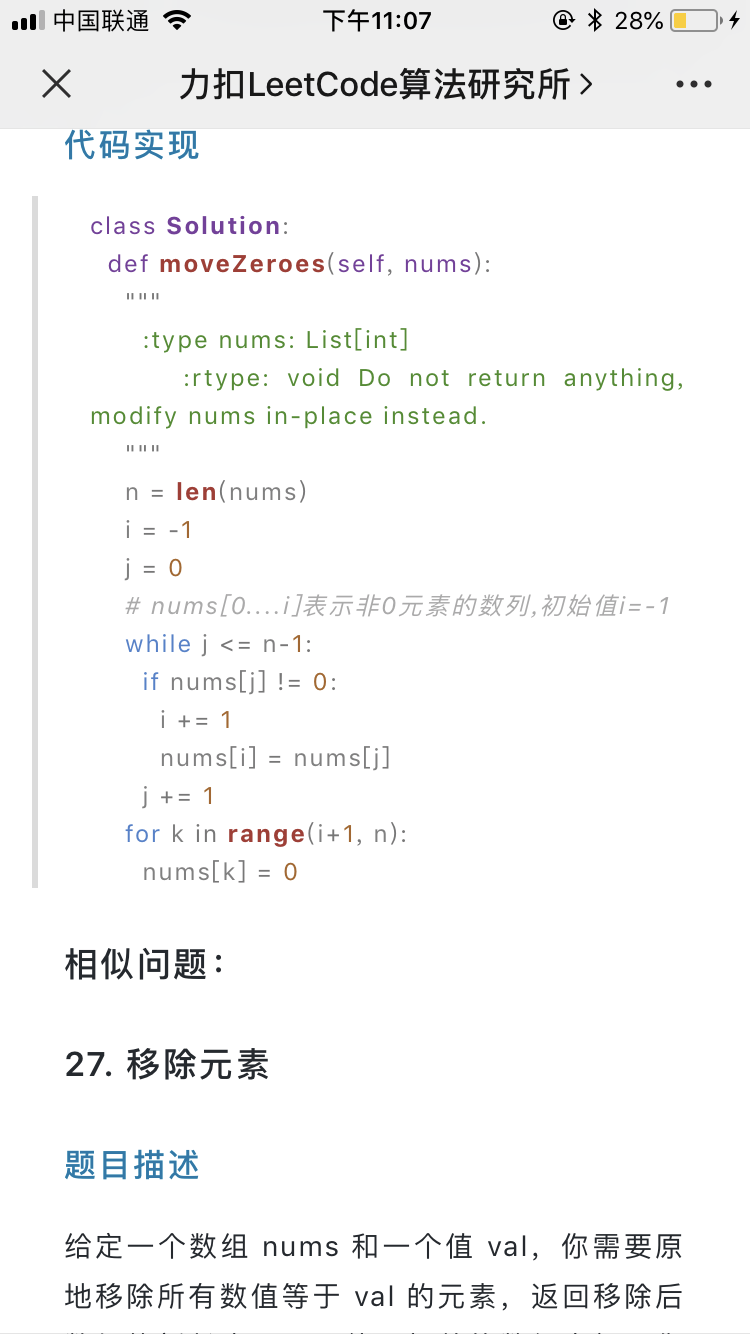
Survivor的存在意义，就是减少被送到老年代的对象，进而减少Full GC的发生，Survivor的预筛选保证，只有经历16次Minor GC还能在新生代中存活的对象，才会被送到老年代。

设置两个Survivor区最大的好处就是解决了碎片化，刚刚新建的对象在Eden中，经历一次Minor GC，Eden中的存活对象就会被移动到第一块survivor space S0，Eden被清空；等Eden区再满了，就再触发一次Minor GC，Eden和S0中的存活对象又会被复制送入第二块survivor space S1（这个过程非常重要，因为这种复制算法保证了S1中来自S0和Eden两部分的存活对象占用连续的内存空间，避免了碎片化的发生）

**丁昌茂：**

****

****

**** ****