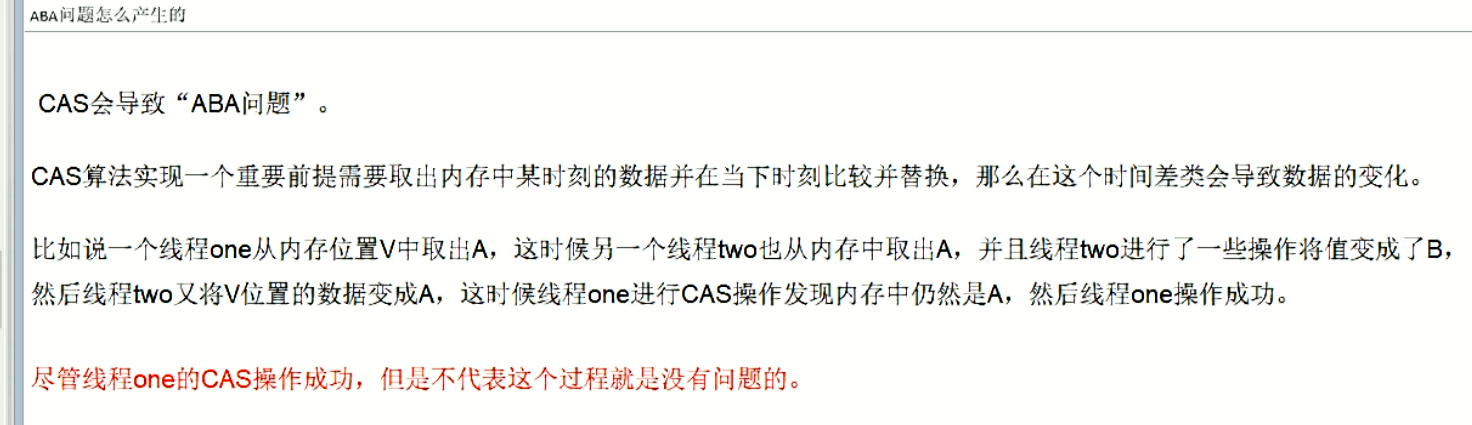
ABA:狸猫换太子



两个线程同时获取主内存的值k , a加载需要10秒,b需要两秒,b加载之后回写主内存数据把k编程v,又把v变回了k ,这之后a机型cas操作时 发现内存中仍然是a,然后替换变成线程a需要改变的值.,尽管a线程操作成功,但是不代表这个过程没有问题.

高并发下我们需要保证数据的一致性,操作时不要有中断和破坏 , 我们可以使用juc包下的类例如atomicInteger 但是我们需要自定义的类型,如原子用户类型,原子订单类型,这是就需要atomicReference

AtomicReferend 原子引用 读法 ao to mei ke rui fe men si

如果你想对某个类进行原子包装,在泛型中添加类

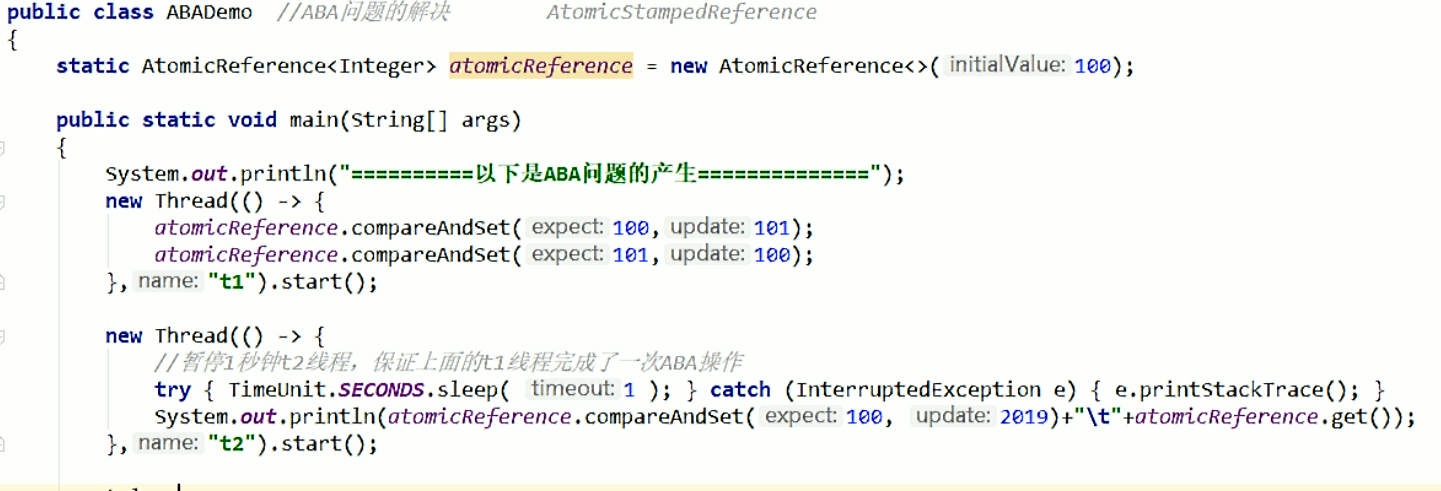


解决ABA问题 , 理解原子引用 + 新增一种机制,那就是修改版本号(类似于时间戳)

时间戳的原子引用 -----解决ABA问题

AtomicStampedReference

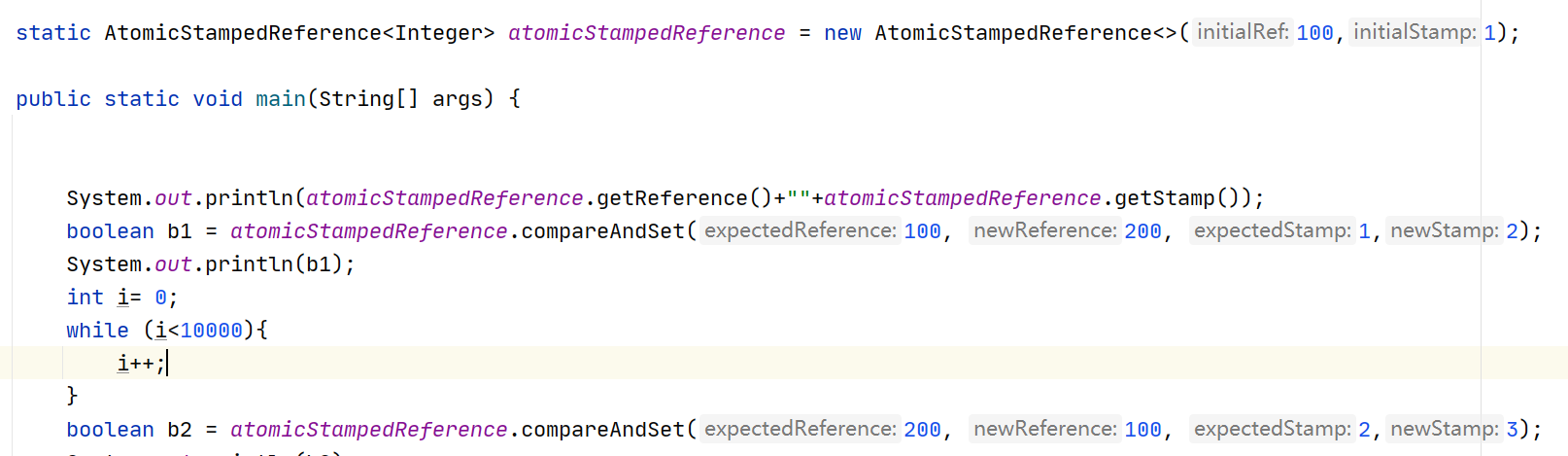
ABA问题的产生



ABA问题的解决

使用

AtomicStampedReference<Integer> *atomicStampedReference* = new AtomicStampedReference<>(100,1);



利用版本号时间戳,如果时间戳不同,不可以回写数据