CAS算法

什么叫做CAS

CAS机制就是使用了3个基本操作数：**内存地址V的当前值，旧的预期值A，要修改的新值B。**

当修改一个变量的时候，只有当变量的**旧的预期值A**和**内存地址V当前值**相同时，**要修改的值B**才会修改**内存地址V的值**

否则重复执行

Java内存模型中指出，Java内存模型规定了所有变量（不包括局部变量）存放在主存中，每条线程也有自己的工作内存。线程对变量的所有操作，均在工作内存中进行。

可以多个线程并发操作，无需等待

CAS的缺点

1. CPU开销大

在并发量比较高的情况下，如果大量线程反复重试更新某一个变量，却一直更新不成功，循环往复，会给CPU带来很大的压力。

2. 不能保证代码块的原子性

CAS机制所保证的只是一个变量的原子性操作，而不能保证整个代码块的原子性。如果需要保证3个变量共同进行原子性的更新，就不得不使用synchronize了。

3. ABA问题

这是CAS机制最大的问题所在。

ABA举例：

小明银行卡中有100元 ，要取50元，到atm去取款时出现故障，提交了俩次50的请求

开启了俩个线程，第一个线程将100改为50成功之后，第二个因某些原因阻塞

，此时小明妈妈通过转账转予小明50元，解除阻塞的第二个线程发现旧值与新值相同，则替换为50元，最后小明银行卡内仅50元了，原本应该有100

解决方案:

定制版本号，每次比较不但比较值，也同时比较版本号是否一致，每次更新成功升级一个版本号