# 作业安排

1. 1、请列出Happens-before的几种规则
2. 程序顺序规则：一个线程中的每个操作，happens-before 于该线程中的任意后续操作
3. volatile 变量规则：对于 volatile 修饰的变量的写的操作，一定 happen-before 后续对于 volatile 变量的读操作
4. 传递性规则：如果 1 happens-before 2; 3happens-before 4; 那么传递性规则表示: 1 happens-before 4
5. start 规则:如果线程 A 执行操作 ThreadB.start(),那么线程 A 的 ThreadB.start()操作 happens-before 线程 B 中的任意操作
6. join 规则:如果线程 A 执行操作 ThreadB.join()并成功返回，那么线程 B 中的任意操作 happens-before 于线程A 从 ThreadB.join()操作成功返回
7. 监视器锁的规则:对一个锁的解锁，happens-before 于随后对这个锁的加锁
8. 2、volatile 能使得一个非原子操作变成原子操作吗？为什么？

不能，因为volatile保证的是多线程的可见性和一定的有序性，无法保证共享数据的原子性，如果一个共享数据自增，线程1读取到缓存中后，进行更新操作时线程1被阻塞了，而线程2继续读取到该共享数据未变化，然后更新数据到内存中，线程1继续执行保证不了数据操作的原子性。

1. 3、哪些场景适合使用Volatile
2. 状态标志：实现 volatile 变量的规范使用仅仅是使用一个布尔状态标志，用于指示发生了一个重要的一次性事件。
3. 一次性安全发布：单例模式的双重检查锁定。
4. 独立观察模式：系统后台有一个采集数据线程，每个一段时间会采集并刷新共享变量，可以设置volatile进行可见性观察。
5. Volatile Bean模式
6. 开销较低的”读-写锁”策略：如果读操作远远超过写操作，您可以结合使用内部锁和 volatile 变量来减少公共代码路径的开销。
7. 4、如果对一个数组修饰volatile，是否能够保证数组元素的修改对其他线程的可见？为什么？

可以，数据元素存储到线程内容中的是引用地址，数据存储在堆内存中，修改数组中元素，实际上是修改堆内存数据，修改会导致可见性，所以对其他线程可见。

# 作业提交方式

作业统一提交到：<https://gper.gupaoedu.com/homework/subjects/5493>