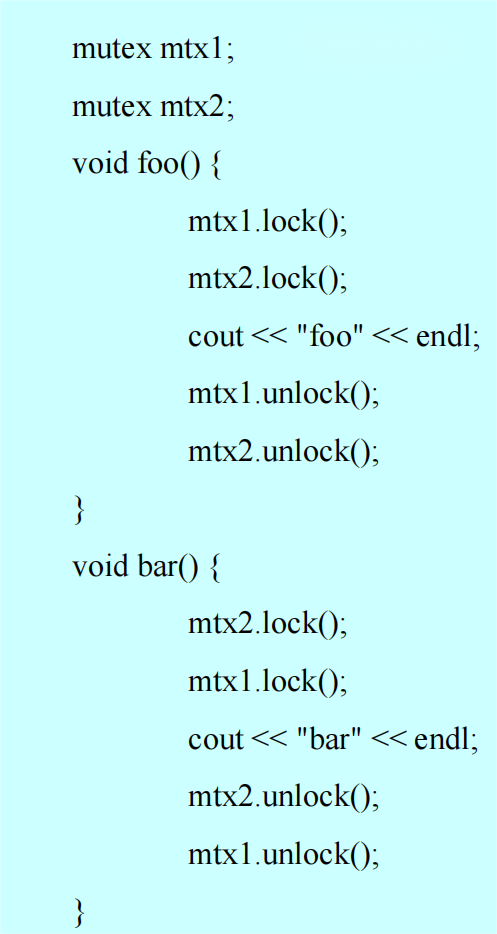
Part1:

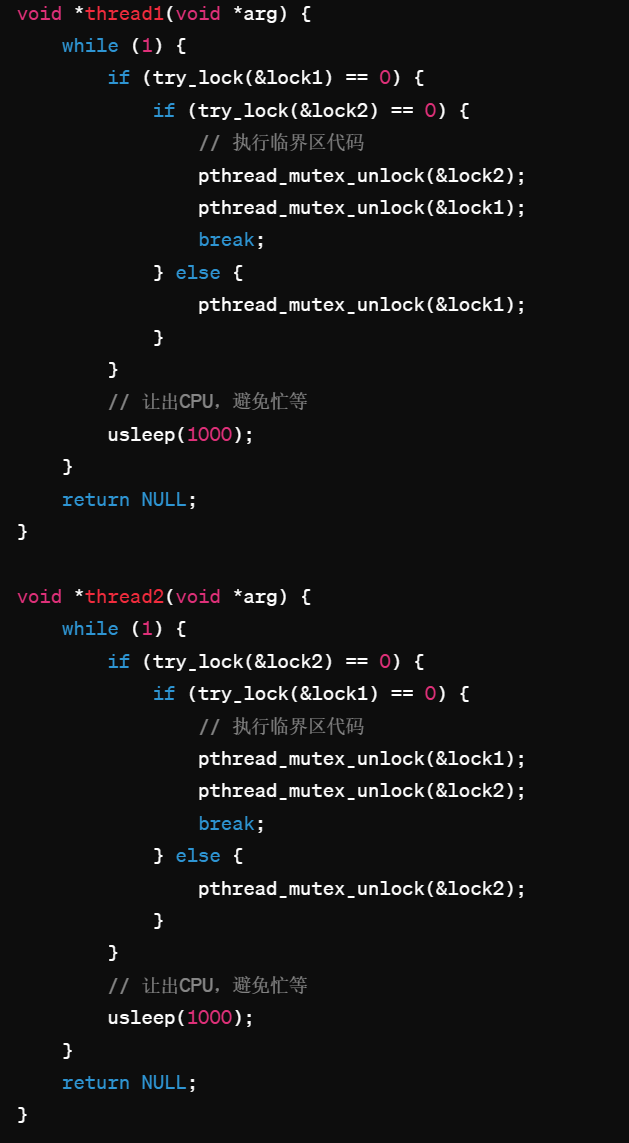
deadlock:两个或多个线程互相等待对方释放资源的锁，而都无法继续执行的状态。每个线程都持有一个资源并等待其他进程释放锁。

示例：两个线程尝试获取两个锁，但顺序相反，如下图，如果foo获取了mtx1的锁，而bar获取了mtx2的锁，那么接下来两个线程都无法继续执行，而会阻塞住，进入死锁状态。



livelock:两个或多个线程不断地改变自己的状态以响应其他进程或线程的状态变化，但由于这些改变，任何线程都无法继续向前推进

示例：如下图，两个线程都获取到第一个锁后，想要获取第二个锁，但如果无法获取第二个锁，它们会释放第一个锁并重试,这种行为可能导致两个线程不断地获取和释放锁，进入活锁状态。



Part2:

1. 可以使用lock\_guard锁，只设置一个全局锁，在pop,push,isEmpty函数的开头用lock\_guard<mutex> lock()函数上锁，避免有多个操作同时对栈进行处理。
2. 性能瓶颈：
   * 1. 由于并发栈只有栈顶指针，所以对全局锁的竞争会非常激烈，大大降低性能
     2. 可能会被内核数限制，当并行的线程达到某一值后，性能不再提高
     3. 频繁的内存分配和释放可能会成为性能瓶颈

3、不会，我的设计中只采用一个全局锁，当一个操作进行时其他操作都无法进行，而活锁和死锁是两个或多个线程才有可能触发的。