1. Describe the asymmetric solution. How does the asymmetric solution guarantee the philosophers
never enter a deadlocked state?

```
|<-0--|--0->|<-0--|--0->|<-0--|--0->

   0     1     2     3     4     5    philosopher
0     1     2     3     4     5       chopstick
```

The asymmetric solution makes it so even number philosophers grab the right chopstick first and odd number philosophers grab the left chopstick first. This means that philosopher 0 and 5 will both attempt to grab chopstick 0 first. One of them will be successful and the other will not be successful. If philosopher 0 is successful they will then try to grab chopstick 1. However it is possible that philosopher 1 has already managed to grab chopstick 2 and has grabbed chopstick 1 before philosopher 0. If this is the case then philosopher 0 will fail to grab chopstick 1. However if philosopher 1 grabs chopstick 2 and then chopstick 1, as dictated by this scheme, it will complete it's task and then release both chopsticks, this means that chopstick 1 will be released and philosopher 0 can then use it. Then philosopher 0 can release chopstick 0 and 1. Philosopher 5 can then reattempt to acquire chopstick 0. In this fashion, all philosophers next to each other will be unable to deadlock each other. Since chopsticks 0, 2, and 4 will be acquired first and chopsticks 1, 3, and 5 will always be acquired second. If a philosopher manages to grab chopstick 1, 3, or 5 it is always able to complete and then release its chopsticks.

2. Does the asymmetric solution prevent starvation? Explain.

No, this doesn't prevent starvation. There is nothing preventing a single philosopher from reclaiming the chopstick it just released over and over. Philosopher 0 could claim chopstick 0, then chopstick 1. It could then complete eating and release both chopsticks. It can then reattempt to claim chopstick 0 whenever it wants. It philosopher 5 hasn't claimed it, philosopher 0 will be able to reclaim it. It will then hold onto chopstick 0 until it is able to get chopstick 1 as well. It is possible for philosopher 5 to never have a turn with chopstick 0 and thus starve. This is true for any two philosophers competing for a single resource.

3. Describe the waiter's solution. How does the waiter's solution guarantee the philosophers never enter a deadlocked state?

The waiter solution makes it so whenever a philosopher wants chopsticks it locks the waiter's mutex and checks if both chopsticks it needs are available. If they are both available it claims both chopsticks it needs, unlocks the waiter mutex and eats. If either chopstick it needs is not available it waits until it receives a signal (can_eat) that it can retry to claim chopsticks. When it is done eating it locks the waiter mutex, releases the chopsticks it had, unlocks the waiter mutex, then sends signals to its neighboring philosophers (the ones who might use the chopsticks it had claimed) that they can retry to take chopsticks.

4. Does the waiter's solution prevent starvation? Explain.

Again, this doesn't stop starvation since a philosopher can release and reclaim the waiter mutex repeatedly without allowing the other philosophers to take turns.

5. Consider a scenario under a condition variable based solution where a philosopher determines at
the time it frees its chopsticks that both chopsticks of another philosopher (Phil) it shares with are free, and so it sends the (possibly) waiting Phil a signal.

Under what circumstances may Phil find that both of its chopsticks are NOT free when it checks?

Phil will find the chopsticks he wants are taken even after he received the signal if one of his neighbors has managed to take the chopstick between the time he received the signal and the time he checked if they were available.