Spinlocks and WaitChannels are the low-level equivalents to Mutexes and Condition Variables. Spinlocks are designed to lock small sequences of code that do not include sleeps. WaitChannels maintain a list structure that contains a list of sleeping threads and coordinates sleeping and waking up with the CPU scheduler.

Below is a partial implementation of Semaphores that uses spinlocks and waitchannels.

Question 1. Use hand-over-hand locking to fill in the missing lines of `Semaphore_Wait`.

Question 2. Fill in the missing code in `Semaphore_Post`.

```
1  typedef struct Semaphore {
2    int sem_count;
3    Spinlock *sem_lock;
4    WaitChannel *sem_wchan;
5  } Semaphore;
6
7  Semaphore_Wait(Semaphore *sem) {
8    Spinlock_Lock(&sem->sem_lock);
9    while (sem->sem_count == 0) {
10     // Fill me in
11
12
13
14
15
16
17
18
19
20   }
21   sem->sem_count--;
22   Spinlock_Unlock(&sem->sem_lock);
23 }
24
25 Semaphore_Post(Semaphore *sem) {
26   // Fill me in
27
28
29
30
31
32
33 }
```