2) Currently in our OS, each semaphore object has data statically allocated for a queue of (pointers to) threads blocked on that semaphore. This queue needs to be large enough to support all foreground threads being blocked on any single semaphore. This would be much more expensive in terms of memory if there were 100 foreground threads. We could address this by changing there to be a single queue shared by all semaphores.

4) If all threads are blocked or sleeping (the scheduler doesn’t make a distinction when looking for the next thread to run), the system will run a default thread added by the OS that that has the lowest priority and just calls OS\_Suspend() in a loop.

6) An advantage of spin-lock semaphores is that they are simple to implement and require less overhead in terms of memory. Blocking semaphores reduce time spent polling a value that can’t change until either an interrupt or another thread signals the semaphore.