In the program in Section 9.1, is it possible that both threads are sleeping at

the same time? That neither of the two threads is sleeping at a particular time? Explain.

Yes. It is possible for both threads to be sleeping at the same time. This is because the scheduler will activate a new thread when one is blocked or sleeping. As such, one thread can be blocked (by going to sleep), at which point the scheduler will activate the other thread. This would not be possible if there are locks or if the threads are synchronized.

2. A race condition can occur in this scenario. If one thread attempts to read when the other thread is attempting to write, the data may not be written to the list. The data may also be corrupted because the list will not be able to resize itself properly. The action of reading, or writing, involves both reading and writing to memory, and the list will definitely be corrupted. Synchronizing the threads will ensure that a race condition does not arise. Another method is to use locks, which will prevent the threads from concurrently trying to access/write to the list. A concurrent LinkedQueue (provided within the java library) would also work. The list can also be wrapped with a Collections.synchronizedList method.