Lin/Snyder, *Principles of Parallel Programming*, Figure 6.13: Correct

```
1  int readers;                    // Negative value => active writer
2  int readWaiters = 0;
3  pthread_mutex_t lock;
4  pthread_cond_t busy;            // Use one condition variable to
5                                  // indicate whether data if busy
6  AcquireExclusive()
7  {
8    pthread_mutex_lock(&lock);    // This code suffers from spurious
9    while(readers!=0)             // wake-ups!!!
10    {
11      pthread_cond_wait(&busy, &lock);
12    }
13    readers=-1;
14    pthread_mutex_unlock(&lock);
15  }
16
17  AcquireShared()
18  {
19    pthread_mutex_lock(&lock);
20    while(readers<0)
21    {
22      readWaiters++;
23      pthread_cond_wait(&busy, &lock);
24      readWaiters--;
25    }
26    readers++;
27    pthread_mutex_unlock(&lock);
28  }
29
30  ReleaseExclusive()
31  {
32    pthread_mutex_lock(&lock);
33    readers=0;
34     if (readWaiters==0)                // If there are no waiting readers
35         pthread_cond_signal(&wBusy);    // Wake up a writer
36     else
37           pthread_cond_broadcast(&rBusy); // Wake up all readers
38    pthread_mutex_unlock(&lock);
39  }
40
41  ReleaseShared()
42  {
43    pthread_mutex_lock(&lock);
44    readers--;
45    if(readers==0)
46    {
47      pthread_cond_signal(&busy);
48    }
49    pthread_mutex_unlock(&lock);
50  }
```