Lin/Snyder, *Principles of Parallel Programming*, Figure 6.12: Corrected

```
1   int readers;                      // Neg value=> active writer
2   int readWaiters = 0;
3   pthread_mutex_t lock;
4   pthread_cond_t rBusy, wBusy;      // Use separate conditional vars
5                                     // for readers and writers
6   AcquireExclusive()
7   {
8     pthread_mutex_lock(&lock);
9     while(readers !=0)
10     {
11      pthread_cond_wait(&wBusy, &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(&rBusy, &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    int doSignal;
44
45    pthread_mutex_lock(&lock);
46    readers--;
47    doSignal=(readers==0)
48    pthread_mutex_unlock(&lock);
49    if(doSignal)                      // Signal executes outside
50    {                                 // of critical section
51      pthread_cond_signal(&wBusy);    // Wake up writer
52    }
53  }
```