```
45    private QNode acquireQNode(Backoff backoff, long startTime, long patience)
46        throws TimeoutException, InterruptedException {
47      QNode node = waiting[ThreadLocalRandom.current().nextInt(SIZE)];
48      QNode currTail;
49      int[] currStamp = {0};
50      while (true) {
51        if (node.state.compareAndSet(State.FREE, State.WAITING)) {
52          return node;
53        }
54        currTail = tail.get(currStamp);
55        State state = node.state.get();
56        if (state == State.ABORTED || state == State.RELEASED) {
57          if (node == currTail) {
58            QNode myPred = null;
59            if (state == State.ABORTED) {
60              myPred = node.pred;
61            }
62            if (tail.compareAndSet(currTail, myPred, currStamp[0], currStamp[0]+1)) {
63              node.state.set(State.WAITING);
64              return node;
65            }
66          }
67        }
68        backoff.backoff();
69        if (timeout(patience, startTime)) {
70          throw new TimeoutException();
71        }
72      }
73    }
```

FIGURE 7.27 The CompositeLock class: the acquireQNode() method.