```
1  public final class PrioritySkipList<T> {
2    public static final class Node<T> {
3       final T item;
4       final int score;
5       AtomicBoolean marked;
6       final AtomicMarkableReference<Node<T>>[] next;
7       // sentinel node constructor
8       public Node(int myPriority) { ... }
9       // ordinary node constructor
10      public Node(T x, int myPriority) { ... }
11   }
12   boolean add(Node node) { ... }
13   boolean remove(Node<T> node) { ... }
14   public Node<T> findAndMarkMin() {
15      Node<T> curr = null;
16      curr = head.next[0].getReference();
17      while (curr != tail) {
18        if (!curr.marked.get()) {
19          if (curr.marked.compareAndSet(false, true))
20            return curr;
21          } else {
22            curr = curr.next[0].getReference();
23          }
24        }
25      }
26      return null; // no unmarked nodes
27   }
28   ...
29 }
```

FIGURE 15.13 The PrioritySkipList<T> class: inner Node<T> class.

Copyright © 2021 Elsevier Inc. All rights reserved.