```
83    public void add(T item, int score) {
84      heapLock.lock();
85      int child = next++;
86      heap[child].lock();
87      heap[child].init(item, score);
88      heapLock.unlock();
89      heap[child].unlock();
90
91      while (child > ROOT) {
92        int parent = child / 2;
93        heap[parent].lock();
94        heap[child].lock();
95        int oldChild = child;
96        try {
97          if (heap[parent].tag == Status.AVAILABLE && heap[child].amOwner()) {
98            if (heap[child].score < heap[parent].score) {
99              swap(child, parent);
100             child = parent;
101           } else {
102             heap[child].tag = Status.AVAILABLE;
103             heap[child].owner = NO_ONE;
104             return;
105           }
106         } else if (!heap[child].amOwner()) {
107           child = parent;
108         }
109       } finally {
110         heap[oldChild].unlock();
111         heap[parent].unlock();
112       }
113     }
114     if (child == ROOT) {
115       heap[ROOT].lock();
116       if (heap[ROOT].amOwner()) {
117         heap[ROOT].tag = Status.AVAILABLE;
118         heap[child].owner = NO_ONE;
119       }
120       heap[ROOT].unlock();
121     }
122   }
```

FIGURE 15.11 The FineGrainedHeap class: the add() method.