```
42    public void resize() {
43      int oldCapacity = capacity;
44      Thread me = Thread.currentThread();
45      if (owner.compareAndSet(null, me, false, true)) {
46        try {
47          if (capacity != oldCapacity) { // someone else resized first
48            return;
49          }
50          quiesce();
51          capacity = 2 * capacity;
52          List<T>[][] oldTable = table;
53          table = (List<T>[][]) new List[2][capacity];
54          locks = new ReentrantLock[2][capacity];
55          for (int i = 0; i < 2; i++) {
56            for (int j = 0; j < capacity; j++) {
57              locks[i][j] = new ReentrantLock();
58            }
59          }
60          for (List<T>[] row : table) {
61            for (int i = 0; i < row.length; i++) {
62              row[i] = new ArrayList<T>(PROBE_SIZE);
63            }
64          }
65          for (List<T>[] row : oldTable) {
66            for (List<T> set : row) {
67              for (T z : set) {
68                add(z);
69              }
70            }
71          }
72        } finally {
73          owner.set(null, false);
74        }
75      }
76    }
```

FIGURE 13.33 RefinableCuckooHashSet<T>: the resize() method.