Figure 18.2 Fixing the bug in *sort()*.

If a list contains identical items, our **sort**() doesn't work:

i	list[i]	countSmaller(list[i], list)	sortedList[i]
0	Chan	1	
1	Cassidy	0	
2	Clarke	2	
3	Smith	4	
4	Clarke	2	

One way of fixing this is to move the second "Clarke" down to the next empty slot

- that slot *had* to belong to Clarke as well—why?

Here's our sorting method with the bug fixed:

```
/**
 * sort: sorts Comparables in "list" in increasing order
 *    – correctly handles equal-elements problem
 */
public static void sort (Comparable[] list) {
    Comparable[] sortedList = new Comparable[list.length];
    for (int i = 0; i < list.length; i++)      // this loop is NEW
        sortedList[i] = null;
    for (int i = 0; i < list.length; i++) {
        int smallerCount = 0;
        for (int j = 0; j < list.length; j++)
            if (list[j].compareTo(list[i]) < 0)
                smallerCount++;
        while (sortedList(smallerCount) != null)      // this loop is NEW
            smallerCount++;
        sortedList(smallerCount) = list[i];
    }
    for (int i = 0; i < list.length; i++)
        list[i] = sortedList[i];
}
```