Quick Sort: Array-Based Lists

- Uses the divide-and-conquer technique
  - The list is partitioned into two sublists
  - Each sublist is then sorted
  - Sorted sublists are combined into one list in such a way so that the combined list is sorted

```
if (the list size is greater than 1)
{
   a. Partition the list into two sublists, say lowerSublist and
      upperSublist.
   b. Quick sort lowerSublist.
   c. Quick sort upperSublist.
   d. Combine the sorted lowerSublist and sorted upperSublist.
}
```

Quick Sort: Array-Based Lists (continued)

- To partition the list into two sublists, first we choose an element of the list called **`pivot`**
- The goal is to re-arrange the list so that the **`pivot`** divides the list into: `lowerSublist` and `upperSublist`
  - The elements in `lowerSublist` are < `pivot`
  - The elements in `upperSublist` are ≥ `pivot`

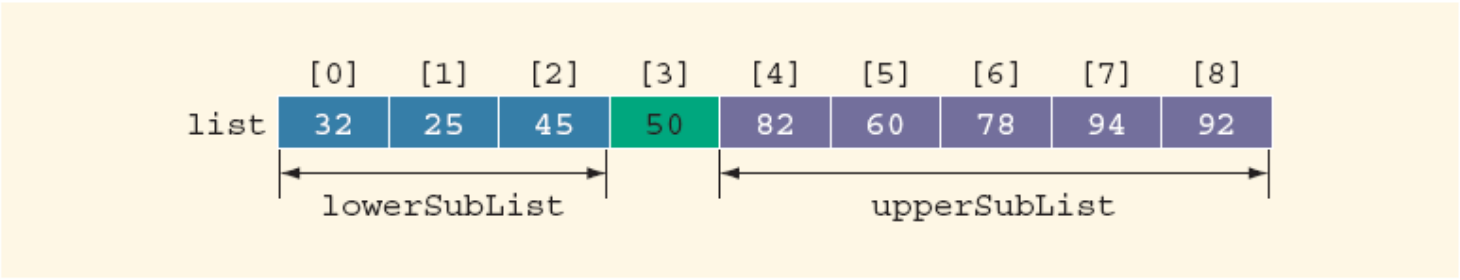

FIGURE 19-38 `list` after the partition

Quick Sort: Array-Based Lists (continued)

- Partition algorithm (we assume that `pivot` is chosen as the middle element of the list):
  - Determine `pivot`; swap it with the first element of the list
  - For the remaining elements in the list:
    - If the current element is less than pivot, (1) increment `smallIndex`, and (2) swap current element with element pointed by `smallIndex`
  - Swap the first element (`pivot`), with the array element pointed to by `smallIndex`

Quick Sort: Array-Based Lists (continued)

- Step 1 determines the pivot and moves `pivot` to the first array position
- During the execution of Step 2, the list elements get arranged

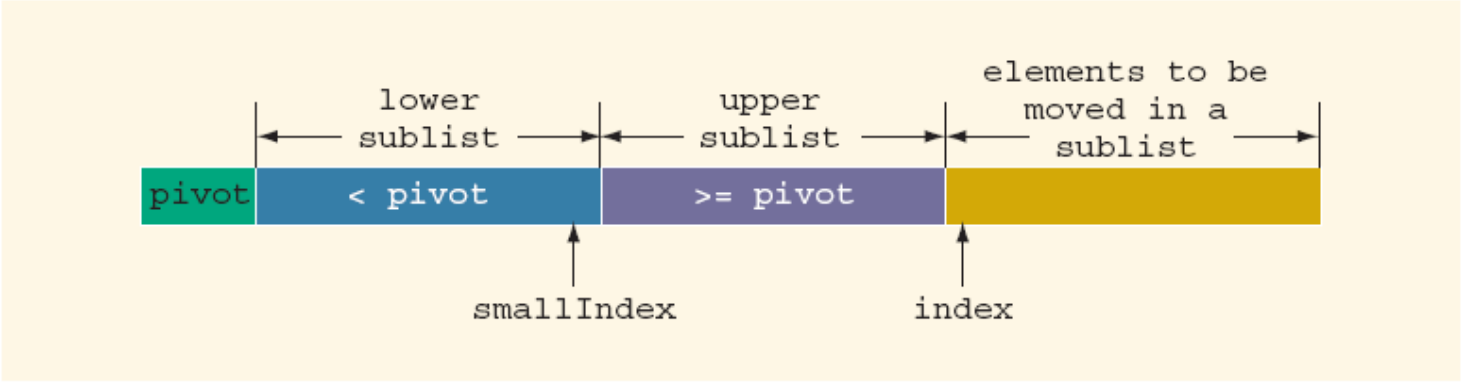

FIGURE 19-39 List during the execution of Step 2

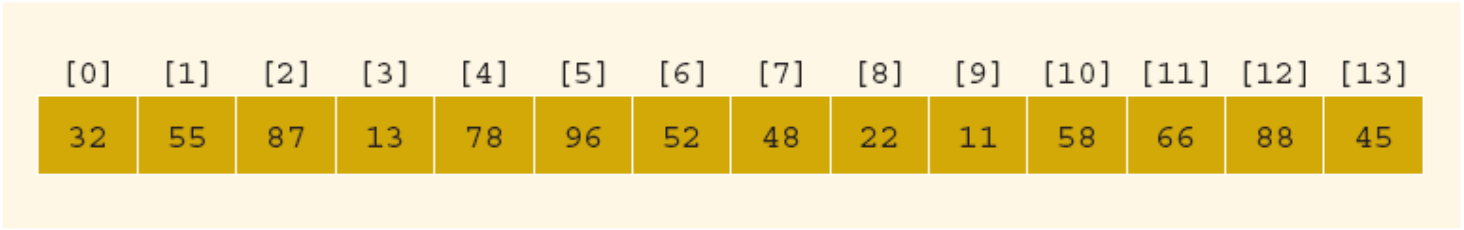

FIGURE 19-40 List before sorting

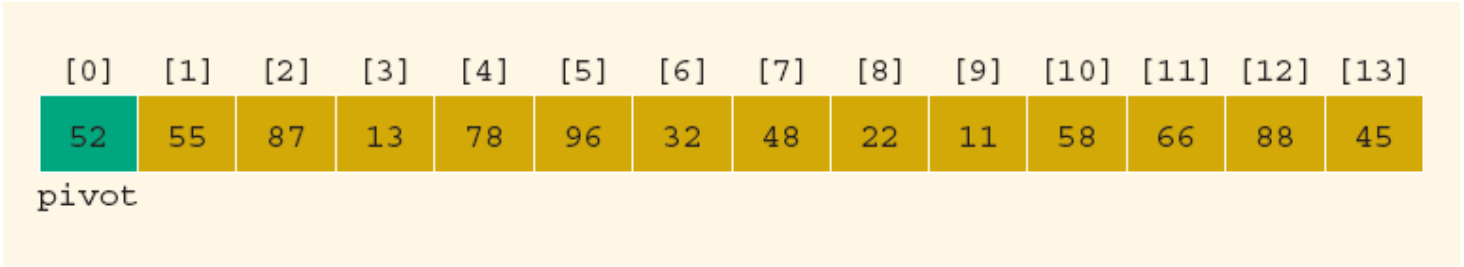

FIGURE 19-41 List after moving `pivot` to the first array position

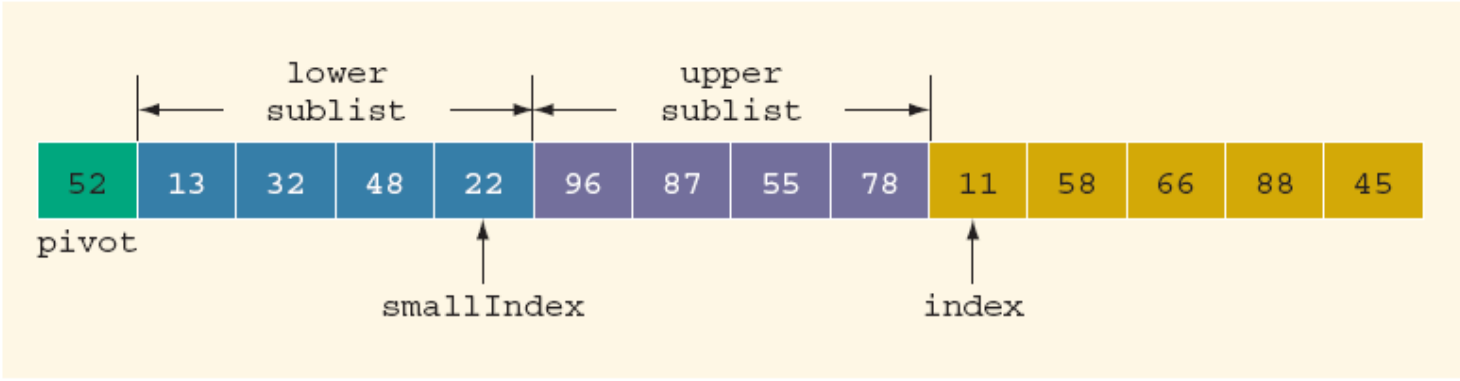

FIGURE 19-42 List after a few iterations of Step 2

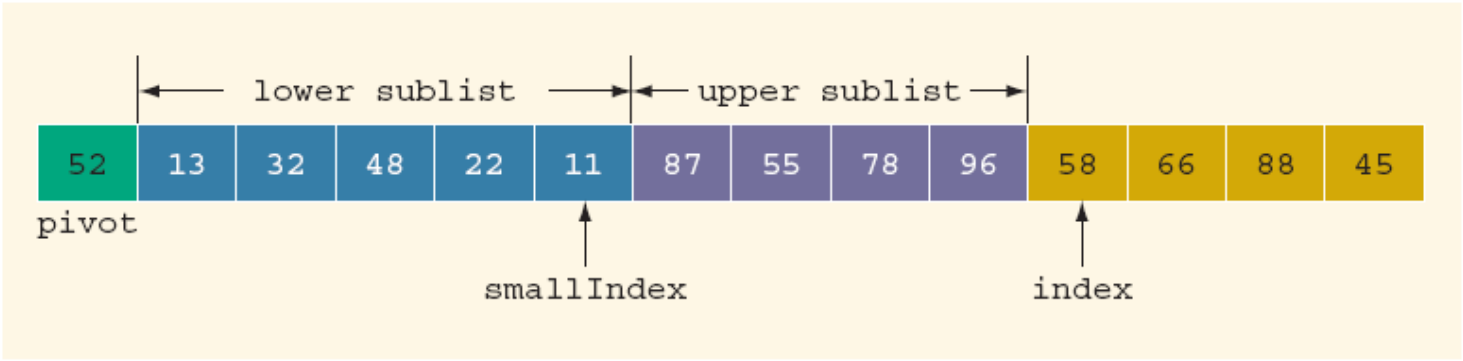

FIGURE 19-43 List after moving 11 into the lower sublist

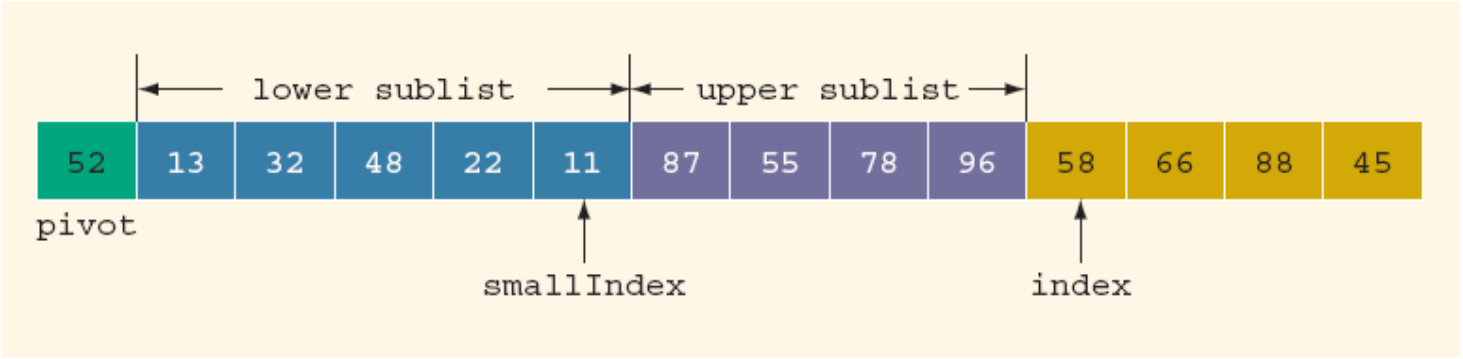

FIGURE 19-44 List before moving 58 into a sublist

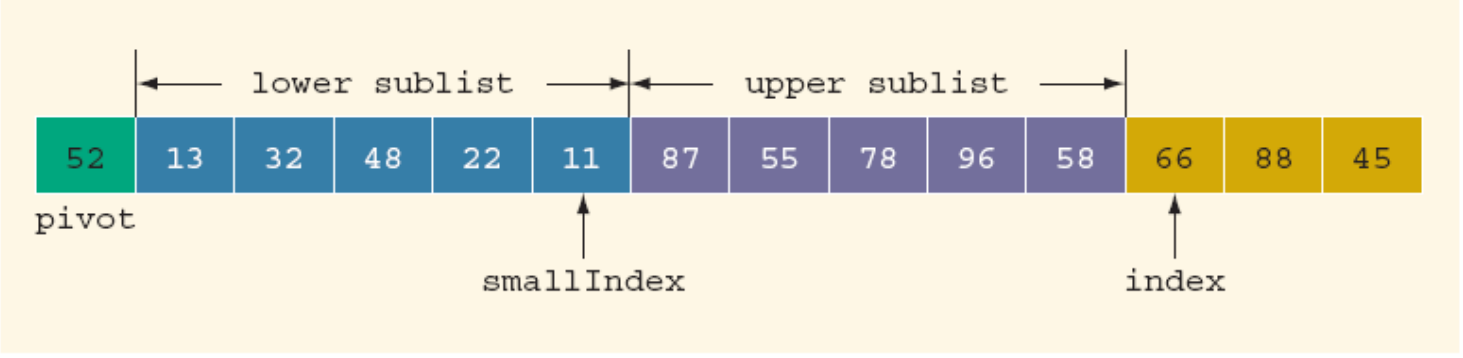

FIGURE 19-45 List after moving 58 into the upper sublist

Quick Sort: Array-Based Lists (continued)

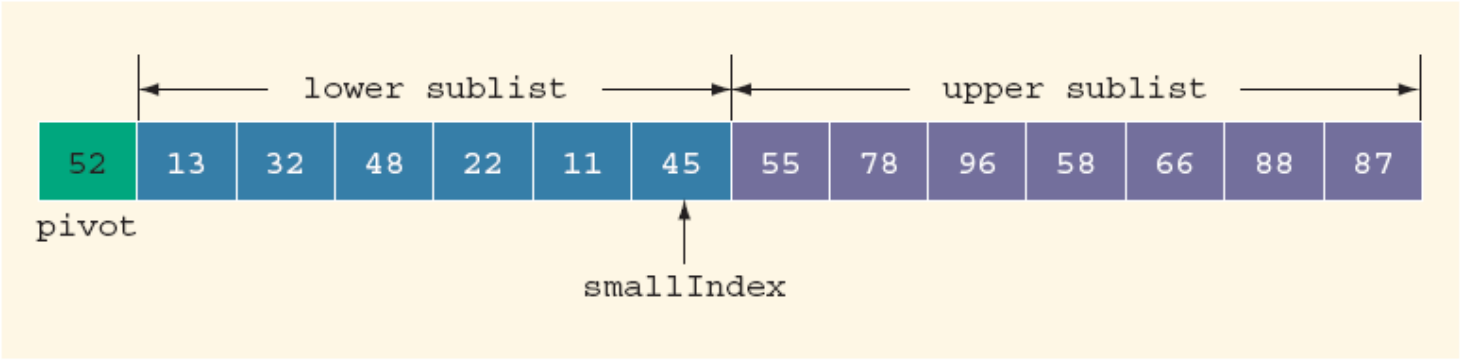

FIGURE 19-46 List elements after arranging into the lower sublist and upper sublist

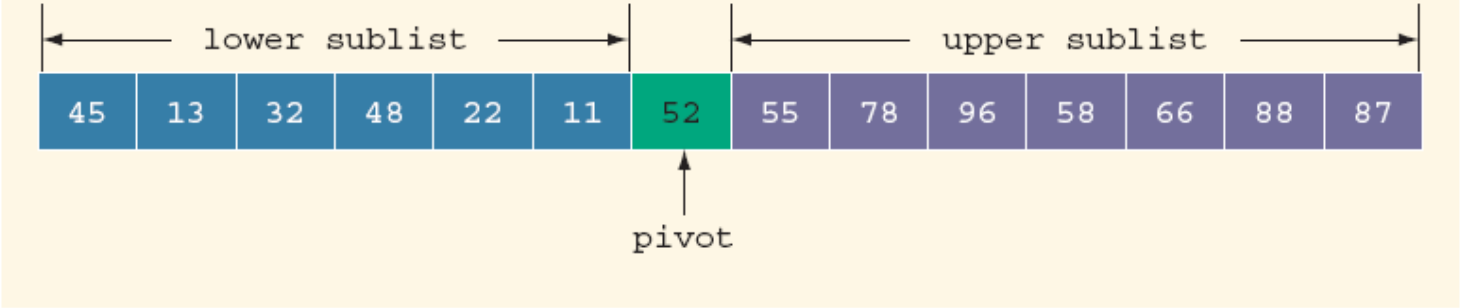

FIGURE 19-47 List after swapping 52 with 45

Analysis: Quick Sort

TABLE 19-10 Analysis of the Quick Sort Algorithm for a List of Length n

	Number of comparisons	Number of swaps
Average case	$(1.39)n\log_2 n + O(n) = O(n\log_2 n)$	$(0.69)n\log_2 n + O(n) = O(n\log_2 n)$
Worst case	$\frac{n^2}{2} - \frac{n}{2} = O(n^2)$	$\frac{n^2}{2} + \frac{3n}{2} - 2 = O(n^2)$