The Selection Sort

The **selection sort** improves on the bubble sort by making only one exchange for every pass through the list. In order to do this, a selection sort looks for the largest value as it makes a pass and, after completing the pass, places it in the proper location. As with a bubble sort, after the first pass, the largest item is in the correct place. After the second pass, the next largest is in place. This process continues and requires $n - 1$ passes to sort n items, since the final item must be in place after the $(n - 1)$ st pass.

Figure 3 shows the entire sorting process. On each pass, the largest remaining item is selected and then placed in its proper location. The first pass places 93, the second pass places 77, the third places 55, and so on. The function is shown in ActiveCode 1.
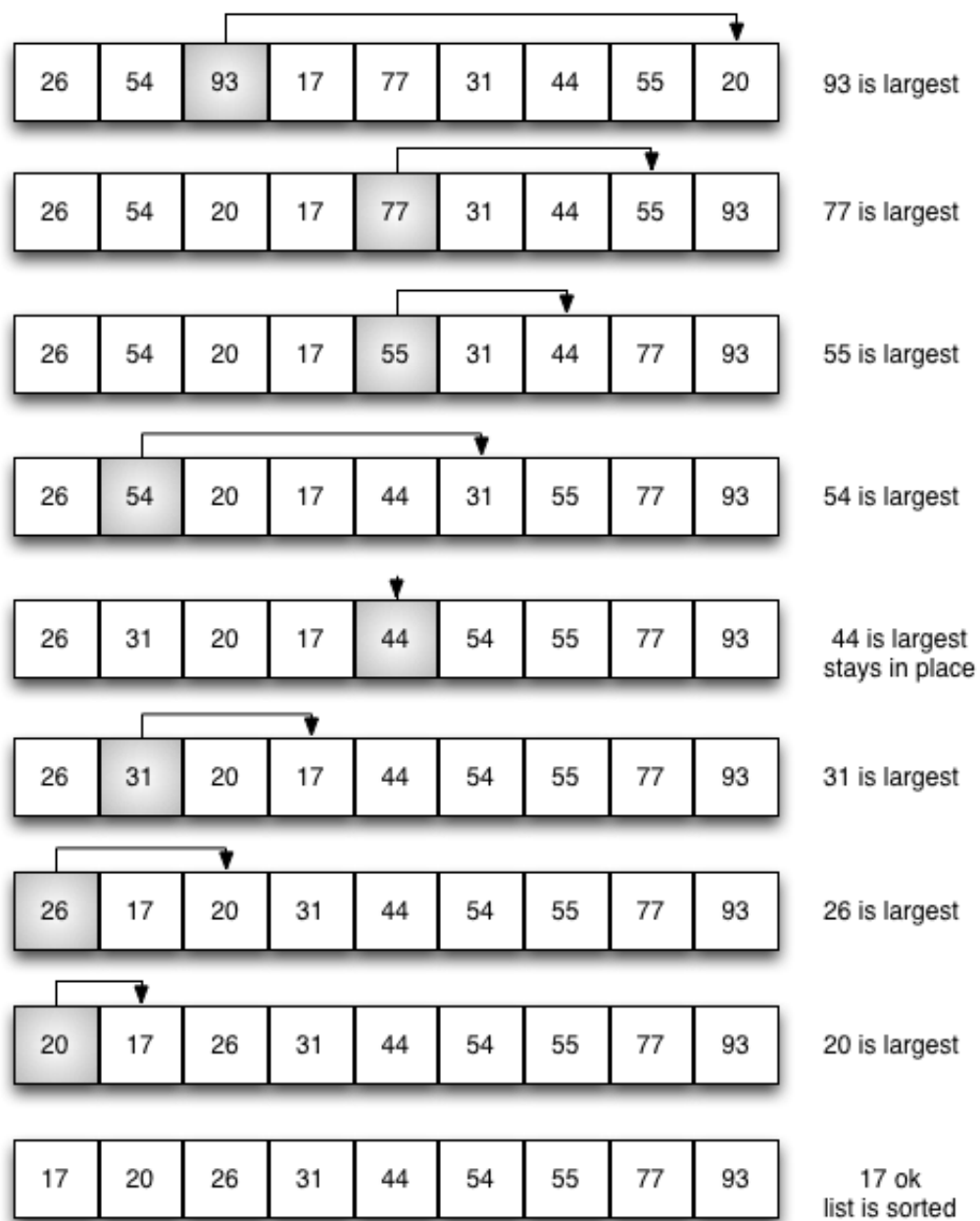
bbleSort.html)

| 26 | 54 | 93 | 17 | 77 | 31 | 44 | 55 | 20 | 93 is largest |

| 26 | 54 | 20 | 17 | 77 | 31 | 44 | 55 | 93 | 77 is largest |

| 26 | 54 | 20 | 17 | 55 | 31 | 44 | 77 | 93 | 55 is largest |

| 26 | 54 | 20 | 17 | 44 | 31 | 55 | 77 | 93 | 54 is largest |

| 26 | 31 | 20 | 17 | 44 | 54 | 55 | 77 | 93 | 44 is largest stays in place |

| 26 | 31 | 20 | 17 | 44 | 54 | 55 | 77 | 93 | 31 is largest |

| 26 | 17 | 20 | 31 | 44 | 54 | 55 | 77 | 93 | 26 is largest |

| 20 | 17 | 26 | 31 | 44 | 54 | 55 | 77 | 93 | 20 is largest |

| 17 | 20 | 26 | 31 | 44 | 54 | 55 | 77 | 93 | 17 ok list is sorted |

Figure 3: selectionSort

[Run] [Save] [Load] [Show CodeLens]

```
 1 def selectionSort(alist):
 2    for fillslot in range(len(alist)-1,0,-1):
 3        positionOfMax=0
 4        for location in range(1,fillslot+1):
 5            if alist[location]>alist[positionOfMax]:
 6                positionOfMax = location
 7
 8        temp = alist[fillslot]
 9        alist[fillslot] = alist[positionOfMax]
10        alist[positionOfMax] = temp
11
12 alist = [54,26,93,17,77,31,44,55,20]
13 selectionSort(alist)
14 print(alist)
15
```

ActiveCode: 1 Selection Sort (lst_selectionsortcode)

| Initialize | Run | Stop |

| Beginning | Step Forward | Step Backward | End |

You may see that the selection sort makes the same number of comparisons as the bubble sort and is therefore also $O(n^2)$. However, due to the reduction in the number of exchanges, the selection sort typically executes faster in benchmark studies. In fact, for our list, the bubble sort makes 20 exchanges, while the selection sort makes only 8.

Self Check

Q-49: Suppose you have the following list of numbers to sort: [11, 7, 12, 14, 19, 1, 6, 18, 8, 20] which list represents the partially sorted list after three complete passes of selection sort?

○ [7, 11, 12, 1, 6, 14, 8, 18, 19, 20]

○ [7, 11, 12, 14, 19, 1, 6, 18, 8, 20]

○ [11, 7, 12, 14, 1, 6, 8, 18, 19, 20]

○ [11, 7, 12, 14, 8, 1, 6, 18, 19, 20]

| Check Me | Compare me |

❮ (TheBubbleSort.html) ❯ (TheInsertionSort.html)

user not logged in