CS112 Data Structures

Recitation 13

Yu Yang
yu.yang@rutgers.edu
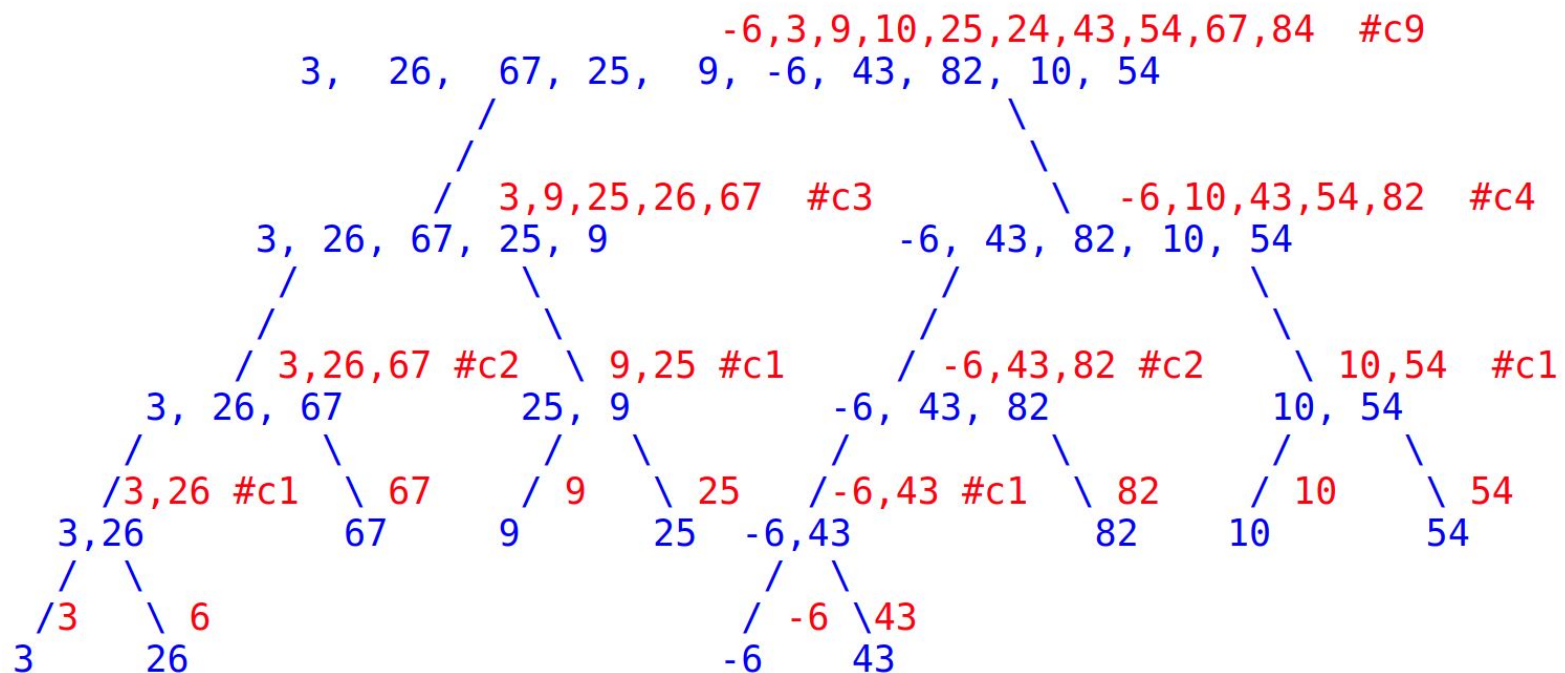Office: CoRE 331
Office Hour: 3-5pm, Wed.

1. Trace the mergesort algorithm on the following list:

3, 26, 67, 25, 9, -6, 43, 82, 10, 54

Show the resulting recursion tree, with the to-be-sorted original and sub-lists at each node, and the number comparions for each merge. (Assume that if there is an odd number of entries in an array, the left part has one more entry than the right after the split.)
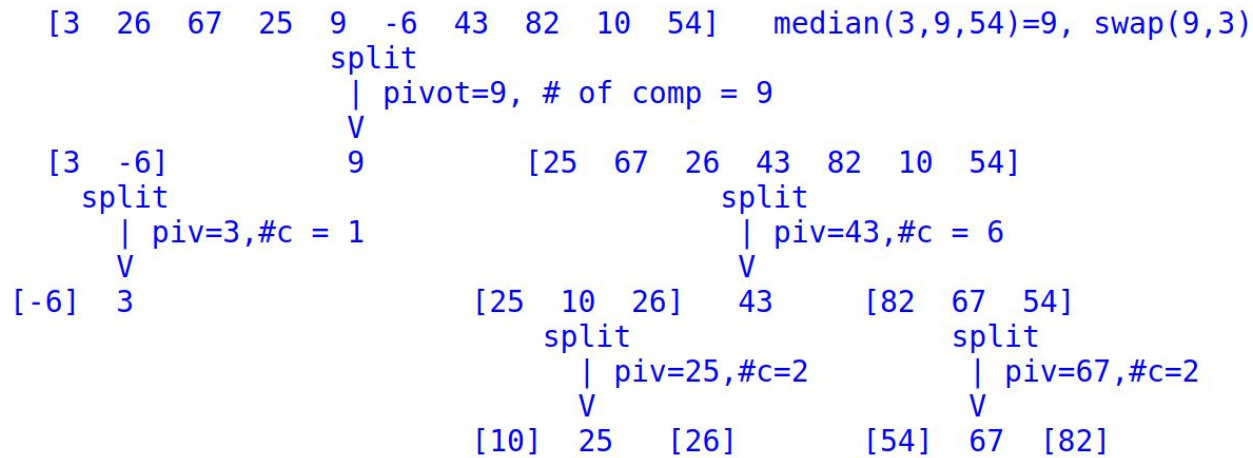
The original and sorted result indicated above and below:

```
                                          -6,3,9,10,25,24,43,54,67,84  #c9
                        3,   26,  67, 25,  9, -6, 43, 82, 10, 54
                                  /                       \
                                 /                          \
                                /   3,9,25,26,67  #c3         \    -6,10,43,54,82  #c4
                      3, 26, 67, 25, 9                     -6,  43, 82, 10, 54
                          /          \                       /              \
                         /            \                     /                \
                        / 3,26,67 #c2  \ 9,25 #c1          / -6,43,82 #c2      \ 10,54  #c1
                    3, 26, 67          25, 9          -6, 43, 82              10, 54
                    /       \          /    \          /       \             /      \
                   /3,26 #c1 \ 67    / 9     \ 25    /-6,43 #c1 \ 82       / 10      \ 54
                3,26          67    9         25  -6,43          82      10           54
                /   \                             /    \
               /3    \ 6                         / -6  \43
              3       26                        -6      43
```

4. Trace the quicksort algorithm on the following array:

3, 26, 67, 25, 9, -6, 43, 82, 10, 54

Use the median of the first, middle, and last entries as the pivot to split a subarray. (If a subarray has fewer than 3 entries, use the first as the pivot.) Show the quicksort tree and the number of comparisons at each split. **SOLUTION**

```
          [3  26  67  25  9  -6  43  82  10  54]   median(3,9,54)=9, swap(9,3)
                           split
                           | pivot=9, # of comp = 9
                           V
       [3  -6]             9          [25  67  26  43  82  10  54]
         split                                    split
           | piv=3,#c = 1                         | piv=43,#c = 6
           V                                      V
     [-6]  3                         [25  10  26]    43    [82  67  54]
                                        split                split
                                        | piv=25,#c=2        | piv=67,#c=2
                                        V                    V
                                   [10]  25  [26]       [54]  67  [82]
```

Total number of comparisons = 20

5. A *stable* sorting algorithm is one which preserves the order of duplicate elements when sorted. For instance, if the following (key,value) pairs are sorted on the keys:

`(3,sun)` `(2,mars)` `(4,moon)` `(3,venus)`

then the output of a stable sorting algorithm would be:

`(2,mars)` `(3,sun)` `(3,venus)` `(4,moon)`

Notice that `(3,sun)` comes before `(3,venus)`, preserving the order of the input for elements that have the same key (i.e. 3), hence *stable*.

However, if the output is this:

`(2,mars)` `(3,venus)` `(3,sun)` `(4,moon)`

then the sorting algorithm is not stable since it does not preserve the input order of `(3,sun)` before `(3,venus)`.

For each of insertion sort, mergesort, and quicksort, tell whether the algorithm is stable or not.

SOLUTION

- Insertion sort is a stable sort.
- Mergesort is a stable sort.
- Quicksort is not a stable sort.

All Slides:
https://devyang.xyz/cs112.html