

Merge Sort

Like [QuickSort](#), Merge Sort is a [Divide and Conquer](#) algorithm. It divides the input array into two halves, calls itself for the two halves, and then merges the two sorted halves. **The merge() function** is used for merging two halves. The merge(arr, l, m, r) is a key process that assumes that arr[l..m] and arr[m+1..r] are sorted and merges the two sorted sub-arrays into one. See the following C implementation for details.

MergeSort(arr[], l, r)

If $r > l$

1. Find the middle point to divide the array into two halves:

$$\text{middle } m = (l+r)/2$$

2. Call mergeSort for first half:

Call mergeSort(arr, l, m)

3. Call mergeSort for second half:

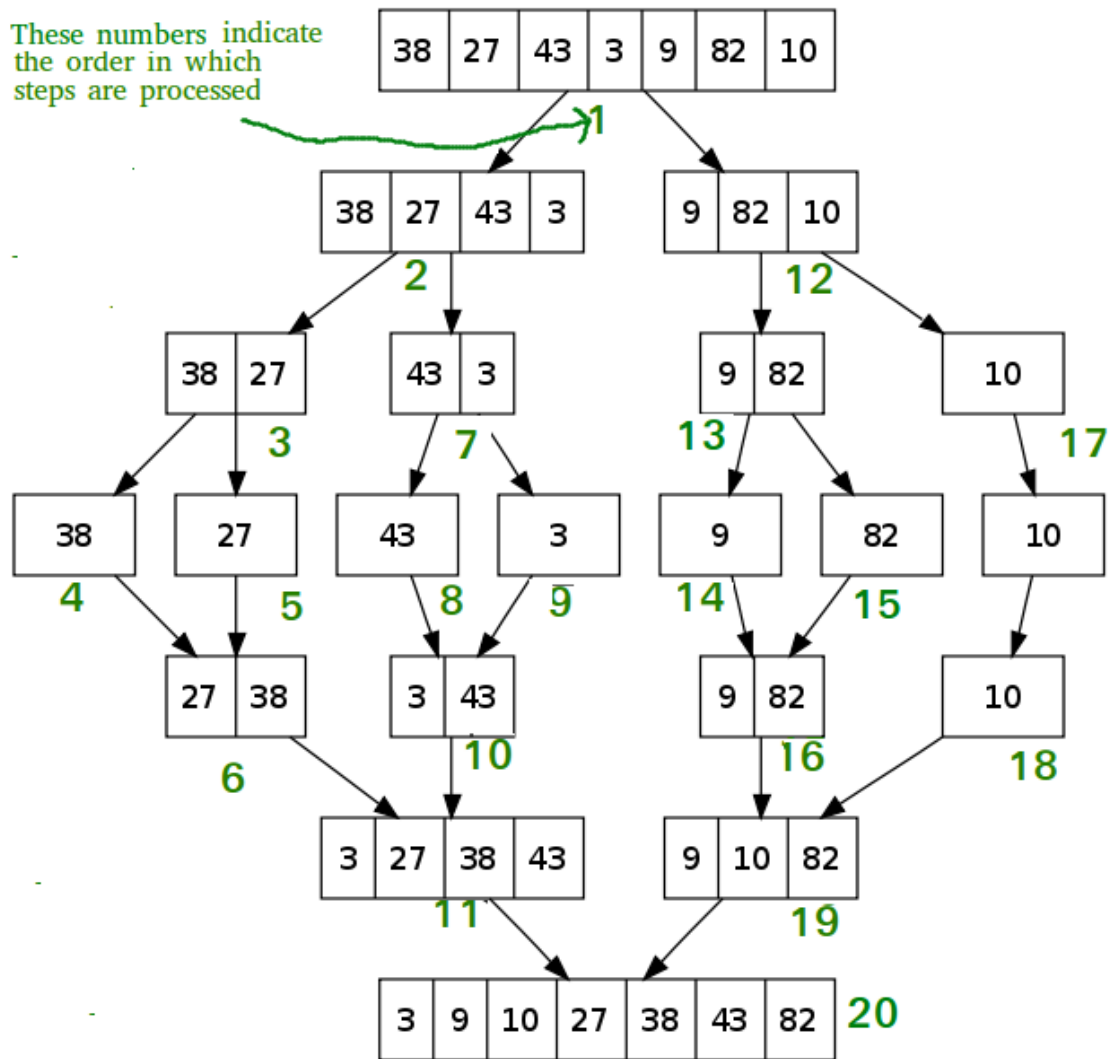
Call mergeSort(arr, m+1, r)

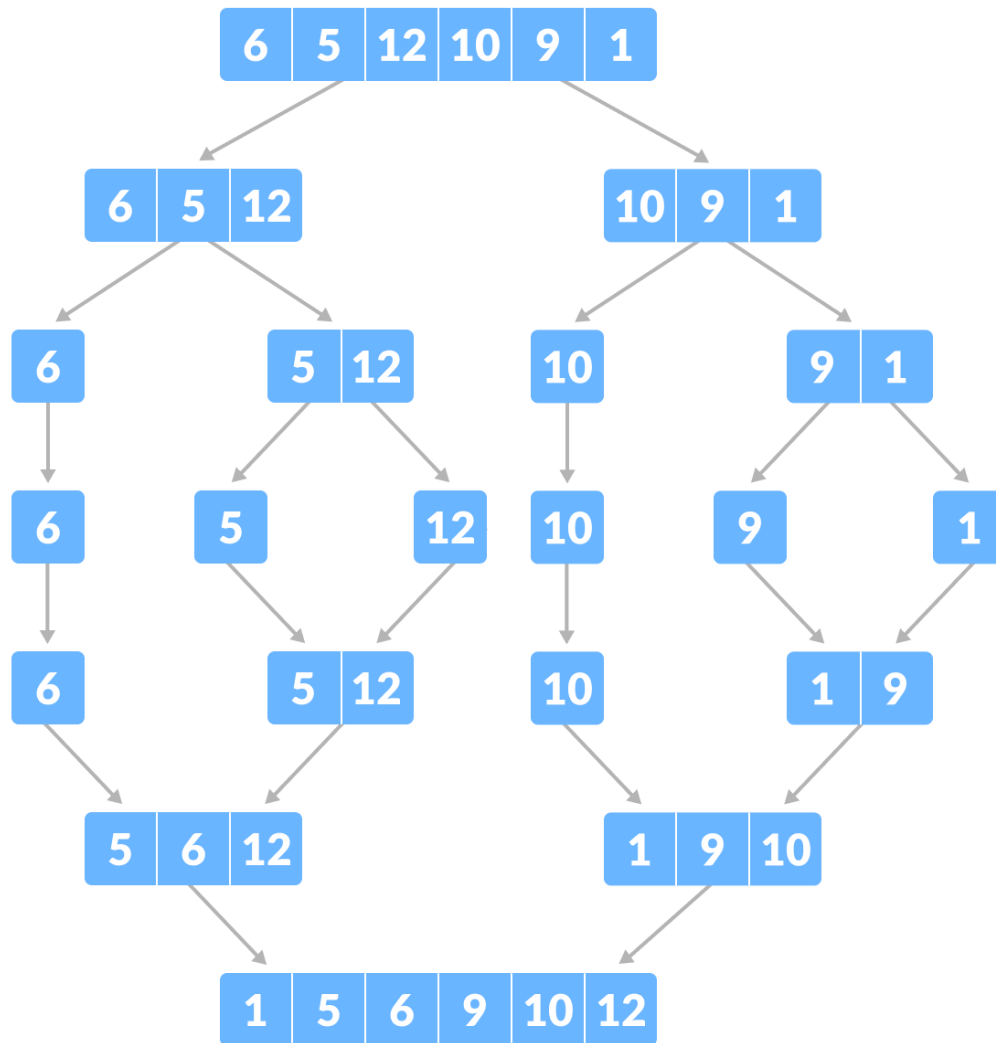
4. Merge the two halves sorted in step 2 and 3:

Call merge(arr, l, m, r)

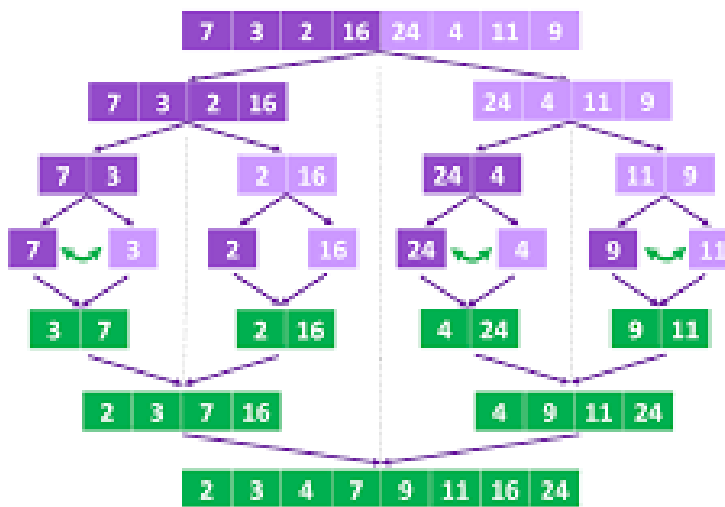
The following diagram shows the complete merge sort process for an example array {38, 27, 43, 3, 9, 82, 10}. If we take a closer look at the diagram, we can see that the array is recursively divided in two halves till the size becomes 1. Once the size becomes 1, the merge processes come into action and start merging arrays back till the complete array is merged.

These numbers indicate the order in which steps are processed





Merge Sort



Step 1: Split sub-lists in two until you reach pair of values.

Step 2: Sort/merge pair of values if needed.

Step 3: Merge and sort sub-lists and repeat process till you merge to the full list.