

Merge Sort

Recursive sorting algorithm

Merge function used with recursion

Input: ① array of integers to be sorted

② starting index (usually 0)

③ starting end index (usually array length-1)

Output: sorted input array

↳ start to end values be in non-decreasing order

1 2 3 4
sorted

4 1 3 2
not sorted

* Assume array of length 1 is sorted

Steps: ① Sort the first half of the input array (used midpoint)

② Sort the second half of the array

③ Merge the sorted halves of the input array

④ Stop ~~at~~ merging when the ^{input} ~~merged~~ array is sorted

1 4 2 | 5 3
0 1 2 | 3 4

integer
↓

$$\text{mid} = \frac{\text{start index} + \text{end index}}{2}$$

$$\text{mid} = \frac{4+0}{2} = 2$$

Merge Function

Combine subarrays into one array; values are sorted when subarrays are being combined

- Steps:
- ① Find the smallest values in each subarray
 - ② Compare the smallest values in each subarray
 - ③ Place smallest value from a subarray in leftmost position of the larger (new) array
 - ④ Find new minimum value in subarray which had placed minimum value
 - ⑤ Repeat steps until all values from each subarray are in the larger array in sorted order

Array A	Array B	Array C
2 7 16 44 55 89	1 6 9 13 15 49	1 2 6 7 ...
↑ ↑ ↑	↑ ↑ ↑	

1 4 6 0
↑ ↑ ↑ ↑

3 2 5
↑ ↑ ↑

0 1 2 3 4 5 6

Example of Merge Sort

8 4 6 3 | 2 7 1 5
0 1 2 3 4 5 6 7

mid=3
MS(0,7)

2 7 | 1 5
MS(4,7)
mid=5

MS(0,3)
mid=1
8 4 | 6 3

MS(2,3)
mid=2
6 3

MS(0,1)
mid=0
8 4

MS(4,5)
mid=4
2 7

MS(6,7)
mid=6
1 5

M(0,0,1)
8 4

M(2,2,3)
6 3

M(4,4,5)
2 7

M(6,6,7)
1 5

M(0,0,1)
4 8

M(2,2,3)
3 6

M(4,4,5)
2 7

M(6,6,7)
1 5

M(0,0,1)
3 4 6 8

1 2 5 7

M(0,1,3)

M(4,5,7)

1 2 3 4 5 6 7 8