# Merge Sort, Quick Sort, and Heap Sort Analysis

Given array: $A = [38, 27, 43, 3, 9, 82, 10, 14]$.

## Merge Sort

| | |
|---|---|
| Divide: | $[38], [27], [43], [3], [9], [82], [10], [14]$ |
| Merge: | $[27, 38], [3, 43], [9, 82], [10, 14]$ |
| Merge: | $[3, 27, 38, 43], [9, 10, 14, 82]$ |
| Merge: | $[3, 9, 10, 14, 27, 38, 43, 82]$ |

**Time Complexity:** Worst, average, and best: $O(n \log n)$.
**Space Complexity:** $O(n)$, additional space needed for the merging process.

## Quick Sort

We'll use the last element as the pivot in each step.

| | |
|---|---|
| Initial Array: | $A = [38, 27, 43, 3, 9, 82, 10, 14]$ |
| Pivot: 14, Partition: | $[10, 9, 3, 14, 43, 82, 38, 27]$ |
| Sort left sub-array: | |
| Pivot: 3, Partition: | $[3, 9, 10, 14, 43, 82, 38, 27]$ |
| Sort right sub-array of the initial pivot: | |
| Pivot: 27, Partition: | $[3, 9, 10, 14, 27, 82, 38, 43]$ |
| ... continue ... | |
| Final Sorted Array: | $[3, 9, 10, 14, 27, 38, 43, 82]$ |

**Time Complexity:** Worst: $O(n^2)$, average and best: $O(n \log n)$.
**Space Complexity:** $O(\log n)$ on average due to recursive call stack.

### Heap Sort

| | |
|---|---|
| Initial Array: | $A = [38, 27, 43, 3, 9, 82, 10, 14]$ |
| Build Max Heap: | $[82, 27, 43, 14, 9, 38, 10, 3]$ |
| Swap and Heapify: | $[3, 27, 43, 14, 9, 38, 10, 82]$ |
| Swap and Heapify: | $[10, 27, 38, 14, 9, 10, 43, 82]$ |
| Swap and Heapify: | $[9, 27, 10, 14, 9, 38, 43, 82]$ |
| ... continue ... | |
| Final Sorted Array: | $[3, 9, 10, 14, 27, 38, 43, 82]$ |

**Time Complexity:** Worst, average, and best: $O(n \log n)$.
**Space Complexity:** $O(1)$, in-place sort.

## Conclusion