Array size: 5

	Merge sort runtime	Quick sort (left most pivot) runtime	Quick sort (median pivot) runtime	Quick sort (random pivot) runtime
Ascending	1.33	1.0	1.33	1.66
Descending	1.33	1.0	1.66	1.66
Random	2.33	2.66	1.66	1.66
Nearly	1.66	1.0	1.66	1.66

Array size: 60

	Merge sort runtime	Quick sort (left most pivot) runtime	Quick sort (median pivot) runtime	Quick sort (random pivot) runtime
Ascending	39.33	53.0	30.33	37.33
Descending	41.66	108.0	33.66	42.33
Random	32.0	39.0	37.66	33.0
Nearly	41.0	40.66	38.33	36.33

Array size: 200

	Merge sort runtime	Quick sort (left most pivot) runtime	Quick sort (median pivot) runtime	Quick sort (random pivot) runtime
Ascending	89.66	85.67	80.33	66.66
Descending	75.0	70.0	51.33	62.33
Random	16.33	17.0	41.33	39.66
Nearly	77.33	76.0	71.0	61.33

Array size: 1000

	Merge sort runtime	Quick sort (left most pivot) runtime	Quick sort (median pivot) runtime	Quick sort (random pivot) runtime
Ascending	72.33	972.33	831.0	934.0
Descending	54.0	1120.0	466	826.33
Random	77.0	58.0	80.0	79.0
Nearly	60.33	122.66	60.0	59.66

Merge sort

Merge sort is simpler than quick sort and has O(N log N) time complexity in best, average and worst cases. Merge sort is stable. So this is good if we care about comparisons. Merge sort requires a temp array of size N for merging. So it is in-place. The merge sort algorithm is built recursively. So it can theoretically cause a stack overflow. However, merge sort is a consistently very fast sorting algorithm.

Quick sort

In general, quick sort is the fastest sorting algorithm. It is in-place but unstable. In the worst case, it gives $O(N^2)$ time complexity. But it is unlikely to be worse than O(N log N). Pivot is the critical factor in quick sort. The speed of this algorithm depends on the pivot. Poor pivot can lead to the worst case of $O(N^2)$. So we have to use a good pivot selection strategy. This is also built recursively. So it can cause a stack overflow.

Finally, according to the above runtime results and previous (practical 01) bubble sort, selection sort and insertion sort runtime results, The merge sort is the fastest sorting algorithm.

	Pros	Cons	Big-O
Bubble sort	• Simple • Fast (if already sorted) • In-place • Stable	• Generally poor in speed. • Performs best only with already sorted data.	B: $O(N^2)$ A: $O(N^2)$ W: $O(N^2)$
Insertion sort	• Fast (with semi sorted data) • In-place • stable	• Complex to implement. • Slow when reversed.	B: O(N) A: $O(N^2)$ W: $O(N^2)$
Selection sort	• Simple • Only one swap per pass • In-place	• All cases are identical. • Unstable	B: $O(N^2)$ A: $O(N^2)$ W: $O(N^2)$

Merge sort	• Consistently very fast • Stable	• Not in-place • Fairly complex to implement.	B: O(N log N) A: O(N log N) W: O(N log N)
Quick sort	• Typically the fastest sorting algorithm. • In-place	• Complicated • Unstable • Could be stack overflowed.	B: O(N log N) A: O(N log N) W: $O(N^2)$