CS 61B Spring 2026

Sorting

Exam-Level Discussion 13: April 20, 2026

1 Summary (Fill in Names of the Sorts)

Sorting. A sorting algorithm rearranges an array into non-decreasing order. This requires a **total order** on the keys: for any `a` and `b`, exactly one of `a < b`, `a = b`, `a > b` holds (trichotomy), and `<` is transitive.

An **inversion** is a pair of indices `i < j` with `a[i] > a[j]`. Sorting is the process of reducing inversions to zero.

_____. Repeatedly find the minimum of the unsorted suffix and swap it into the next sorted position. $\Theta(N^2)$ time on every input; $\Theta(1)$ extra space.

_____.

- **Naive:** add each item to a separate max-heap, then repeatedly delete-max into a new array. $\Theta(N \log N)$ time, $\Theta(N)$ extra space.
- **In-place:** heapify the input array itself using bottom-up sink (linear time, not proven in class), then repeatedly swap the root with the last unsorted slot and sink. $\Theta(N \log N)$ time, $\Theta(1)$ extra space.

_____. Recursively sort the two halves, then merge them into one sorted array. $\Theta(N \log N)$ worst case, but requires $\Theta(N)$ auxiliary memory for merging.

_____. For each index i, swap `a[i]` leftward while it is smaller than its left neighbor. Each swap removes exactly one inversion, so the total runtime is $\Theta(N + K)$ where K is the number of inversions. Best case $\Theta(N)$ (already sorted); worst case $\Theta(N^2)$ (reverse-sorted). Excellent on nearly-sorted arrays and on small arrays (roughly $N < 15$), so other sorts often switch to it as a base case.

_____. Pick a pivot, **partition** the array so that every item left of the pivot is $\leq$ pivot and every item right is $\geq$ pivot (the pivot ends up in its final sorted position), then recurse on the left and right sides.

- 3-Scan: scan three times into a new array. $\Theta(N)$ time but $\Theta(N)$ extra space.
- Hoare: walk one pointer in from the left (stopping on items $\geq$ pivot) and another in from the right (stopping on items $\leq$ pivot), swapping whenever both have stopped. In-place, $\Theta(N)$ per partition.

Runtime: best and average $\Theta(N \log N)$; worst case $\Theta(N^2)$, which arises on sorted or reverse-sorted input when you always pick the leftmost item as pivot. Even fairly unbalanced splits (pivot at the 10% mark) still give $\Theta(N \log N)$. Space: $\Theta(\log N)$ on the recursion stack — much less than _____'s $\Theta(N)$.

Avoiding the $\Theta(N^2)$ worst case. Four philosophies:

- **Randomness:** shuffle the input before sorting, or pick random pivots.
- **Smart pivot selection:** take the median, or an approximation such as median-of-three.
- **Introspection:** track recursion depth and switch to _____ if recursion goes too deep.
- **Preprocessing:** try to detect pathological inputs in advance (limited effectiveness).

Sort	Best	Worst	Extra space
Selection	$\Theta(N^2)$	$\Theta(N^2)$	$\Theta(1)$
Heapsort (in-place)	$\Theta(N \log N)$	$\Theta(N \log N)$	$\Theta(1)$
Mergesort	$\Theta(N \log N)$	$\Theta(N \log N)$	$\Theta(N)$
Insertion	$\Theta(N)$	$\Theta(N^2)$	$\Theta(1)$
Quicksort (Hoare)	$\Theta(N \log N)$	$\Theta(N^2)$	$\Theta(\log N)$

For interactive demos of every sort on this page, see `https://joshh.ug/61b/sorts/`.

2 All Sorts of Sorts

Show the steps taken by each sort on the following unordered list:

`0, 4, 2, 7, 6, 1, 3, 5`

(a) Insertion sort

(b) Selection sort

(c) Merge sort

(d) Use heapsort to sort the following array (hint: draw out the heap).
Draw out the array at each step: `0, 6, 2, 7, 4`

3 Conceptual Sorts

Answer the following questions regarding various sorting algorithms that we've discussed in class. If the question is T/F and the statement is true, provide an explanation. If the statement is false, provide a counterexample.

(a) We have a system running insertion sort and we find that it's completing faster than expected. What could we conclude about the input to the sorting algorithm?

(b) Give a 5 integer array that elicits the worst case runtime for insertion sort.

(c) (T/F) Heapsort is stable.

(d) Compare mergesort and quicksort in terms of (1) runtime, (2) stability, and (3) memory efficiency for sorting linked lists.

(e) You will be given an answer bank, each item of which may be used multiple times. You may not need to use every answer, and each statement may have more than one answer.

(A) Quicksort (in-place using Hoare partitioning and choose the leftmost item as the pivot)
(B) Merge Sort
(C) Selection Sort
(D) Insertion Sort
(E) Heapsort
(F) None of the above

For each of the statements below, list all letters that apply. Each option may be used multiple times or not at all. Note that all answers refer to the entire sorting process, not a single step of the sorting process, and assume that N indicates the number of elements being sorted.

_____ Bounded by $\Omega(N \log N)$ lower bound

_____ Worst case runtime that is asymptotically better than quicksort's worst case runtime.

_____ In the worst case, performs $\Theta(N)$ pairwise swaps of elements.

_____ Never compares the same two elements twice.

_____ Runs in best case $\Theta(\log N)$ time for certain inputs.

4 Sorted Runtimes

We want to sort an array of N **unique** numbers in ascending order. Determine the best case and worst case runtimes of the following sorts:

(a) Once the runs in merge sort are of size $<= \frac{N}{100}$, we perform insertion sort on them.

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

(b) We use a linear time median finding algorithm to select the pivot in quicksort.

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

(c) We implement heapsort with a min-heap instead of a max-heap. You may modify heapsort but must maintain constant space complexity.

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

(d) We use any algorithm to sort the array knowing that:

- There are at most N inversions.

  Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

- There is exactly 1 inversion.

Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$

- There are exactly $\frac{N(N-1)}{2}$ inversions.

  Best Case: $\Theta($ $)$, Worst Case: $\Theta($ $)$