Chalkboard Notes on Sorting

A lecture-style Beamer starter

Prof. Dana Okafor (Department of Computer Science)

August 27, 2026

Outline

1. Why Sorting Matters
2. Algorithms
3. Worked Example
4. Wrap-up

Motivation

- Sorting is a building block for search, joins, and deduplication
- Comparing algorithms teaches asymptotic analysis directly
- We will trace three algorithms on the same input

Insertion Sort

Idea

- Grow a sorted prefix one element at a time
- Shift larger elements right to make room

Cost

- Best case: $O(n)$
- Worst case: $O(n^2)$

Merge Sort

Divide and conquer

Split the array in half, sort each half recursively, then merge the two sorted halves in linear time.

$$T(n) = 2T(n/2) + O(n) \implies T(n) = O(n \log n)$$

Comparing the Three

Algorithm	Best	Average	Worst
Insertion sort	$O(n)$	$O(n^2)$	$O(n^2)$
Merge sort	$O(n \log n)$	$O(n \log n)$	$O(n \log n)$
Quicksort	$O(n \log n)$	$O(n \log n)$	$O(n^2)$

Table: Time complexity by case.

Tracing Merge Sort on [5, 2, 8, 1]

- Split: [5, 2] and [8, 1]
- Split again: [5], [2], [8], [1]
- Merge: [2, 5] and [1, 8]
- Merge: [1, 2, 5, 8]

Summary

- Insertion sort is simple and fast on nearly-sorted input
- Merge sort guarantees $O(n \log n)$ but needs extra memory
- Quicksort is usually fastest in practice despite a worse worst case

Questions

Questions?

dana.okafor@example.edu