**Understanding Sorting Algorithms**

**🔹 Bubble Sort**

* Simple comparison-based sorting.
* Repeatedly steps through the list, compares adjacent elements, and swaps them if they are in the wrong order.
* **Time Complexity**:
  + Best Case: O(n)
  + Average & Worst Case: O(n²)

**🔹 Insertion Sort**

* Builds the final sorted array one item at a time.
* Efficient for small data or nearly sorted data.
* **Time Complexity**:
  + Best Case: O(n)
  + Average & Worst Case: O(n²)

**🔹 Quick Sort (Efficient)**

* Uses **divide-and-conquer**.
* Picks a "pivot", partitions the array into elements < pivot and > pivot, then recursively sorts subarrays.
* **Time Complexity**:
  + Best & Average: O(n log n)
  + Worst: O(n²) (rare, with poor pivot selection)

**🔹 Merge Sort**

* Also divide-and-conquer.
* Divides array, recursively sorts halves, then merges them.
* **Time Complexity**:
  + Always O(n log n)
  + Needs extra space for merging.