4. Introduction to Sorting

4.1 What is Sorting?

Sorting is the process of arranging elements in a specific order (ascending or descending).

Why Sort?

1. **Faster Searching:** Enables binary search
2. **Data Organization:** Makes data easier to understand
3. **Algorithm Requirements:** Many algorithms require sorted input
4. **Data Presentation:** Better user experience
5. **Finding Duplicates:** Easier with sorted data

Real-World Examples:

- Sorting contacts alphabetically
- Arranging files by date
- Organizing products by price
- Ranking search results
- Leaderboards in games

4.2 Sorting Algorithm Categories

1. By Complexity:

- **Simple:** Bubble, Selection, Insertion Sort - $O(n^2)$
- **Efficient:** Merge, Quick, Heap Sort - $O(n \log n)$
- **Special:** Counting, Radix, Bucket Sort - $O(n)$

2. By Method:

- **Comparison-based:** Compare elements to sort
- **Non-comparison:** Use element properties

3. By Stability:

- **Stable:** Preserves relative order of equal elements
- **Unstable:** May change relative order

4. By Memory:

- **In-place:** O(1) extra space
- **Out-of-place:** O(n) extra space

4.3 Sorting Algorithm Comparison Table

Algorithm	Best	Average	Worst	Space	Stable	Method
Bubble Sort	O(n)	O(n²)	O(n²)	O(1)	Yes	Exchange
Selection Sort	O(n²)	O(n²)	O(n²)	O(1)	No	Selection
Insertion Sort	O(n)	O(n²)	O(n²)	O(1)	Yes	Insertion
Merge Sort	O(n log n)	O(n log n)	O(n log n)	O(n)	Yes	Divide & Conquer
Quick Sort	O(n log n)	O(n log n)	O(n²)	O(log n)	No	Divide & Conquer
Heap Sort	O(n log n)	O(n log n)	O(n log n)	O(1)	No	Selection
Counting Sort	O(n+k)	O(n+k)	O(n+k)	O(k)	Yes	Non-comparison
Radix Sort	O(d(n+k))	O(d(n+k))	O(d(n+k))	O(n+k)	Yes	Non-comparison

Revision #1
Created 2025-11-03 02:03:15 UTC by DS
Updated 2025-11-03 02:03:46 UTC by DS