Merge Sort

- Merge Sort follows the Divide and Conquer approach.
- It recursively divides the array into two halves until each sub-array contains a single element, then merges the sorted sub-arrays back together.

Steps of Merge Sort

1. **Divide:** If the array has more than one element, divide it into two halves.
2. **Conquer (Sort Recursively):** Recursively sort each half using Merge Sort.
3. **Merge:** Merge the two sorted halves into a single sorted array.

Example

Consider the array:

[8, 3, 7, 4, 9, 2, 6, 5]

Divide:

[8, 3, 7, 4] [9, 2, 6, 5]

[8, 3] [7, 4] [9, 2] [6, 5]

[8] [3] [7] [4] [9] [2] [6] [5] (Each element is now separate)

Merge (Sorting during merging):

[3, 8] [4, 7] [2, 9] [5, 6]

[3, 4, 7, 8] [2, 5, 6, 9]

[2, 3, 4, 5, 6, 7, 8, 9] (Final sorted array)

Time Complexity

- **Best Case:** O(n log n)
- **Average Case:** O(n log n)
- **Worst Case:** O(n log n)

Why Use Merge Sort?

✅ **Stable Sort** (Preserves order of equal elements)

✅ **Efficient for Large Data Sets**

✅ **Guaranteed O(n log n) Complexity**

🚫 **Consumes Extra Space** (Not in-place)

🚫 **Slower for Small Inputs** (Compared to Quick Sort or Insertion Sort)