Merge Sort Cheatsheet

Overview

- Merge sort is a divide-and-conquer sorting algorithm.
- It works by dividing the input array into two halves, recursively sorting each half, and then merging the sorted halves.
- The merge operation takes two sorted sub-arrays and combines them into a single sorted array.

Algorithm

```
def merge_sort(arr):
    if len(arr) > 1:
        # Divide the array into two halves
        mid = len(arr) // 2
        left_half = arr[:mid]
        right_half = arr[mid:]

        # Recursively sort each half
        merge_sort(left_half)
        merge_sort(right_half)

        # Merge the sorted halves
        i = j = k = 0
        while i < len(left_half) and j < len(right_half):
            if left_half[i] < right_half[j]:
                arr[k] = left_half[i]
                i += 1
            else:
                arr[k] = right_half[j]
                j += 1
            k += 1

        while i < len(left_half):
            arr[k] = left_half[i]
            i += 1
            k += 1

        while j < len(right_half):
            arr[k] = right_half[j]
            j += 1
            k += 1
```

Time Complexity

- Worst-case performance: O(n log n)
- Best-case performance: O(n log n)
- Average-case performance: O(n log n)

Resources

- Merge Sort Wikipedia
- GeeksforGeeks: Merge Sort
- Visualgo: Merge Sort