Initial Array

```
{4, 5, 3, 2, 6, 1, 8, 7}
```

Step 1: Divide the Array in Half

```
merge-sort({4, 5, 3, 2, 6, 1, 8, 7})
          ↓
merge-sort({4, 5, 3, 2}) + merge-sort({6, 1, 8, 7})
```

Step 2: Continue Dividing into Pairs

```
merge-sort({4, 5}) + merge-sort({3, 2}) + merge-sort({6, 1}) + merge-sort({8,
7})
```

Step 3: Divide to Individual Elements (Base Case)

```
{4} + {5} + {3} + {2} + {6} + {1} + {8} + {7}
```

Step 4: Merge Pairs in Sorted Order

```
{4, 5} + {2, 3} + {1, 6} + {7, 8}
```

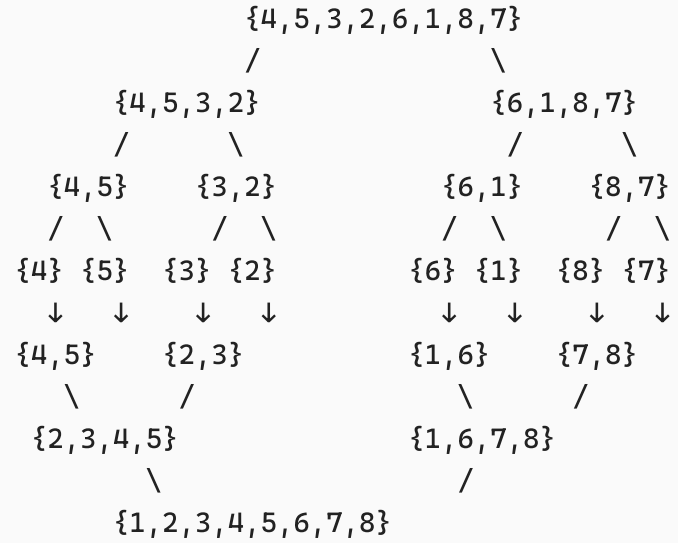
Step 5: Merge Groups of Four

```
{2, 3, 4, 5} + {1, 6, 7, 8}
```

Step 6: Final Merge

```
{1, 2, 3, 4, 5, 6, 7, 8}
```

Visual Tree Representation



Key Concepts

Divide: Recursively split the array into halves until each subarray contains a single element.

Conquer: Single elements are inherently sorted.

Merge: Combine sorted subarrays by comparing elements and placing them in order.

Time Complexity: O(n log n) in all cases