

Combine overlapping Ranges

Approach:

1. Sort the Ranges: Start by sorting the ranges based on the starting value. This will make it easier to check for overlaps.
2. Initialize a Merged Array: Create an empty list to hold merged ranges.
3. Iterate through Sorted Ranges: Loop through each range:
 3. If the list of merged ranges is empty or if the current range does not overlap with the last added range in merged ranges, append it.
 3. If there is an overlap, combine the ranges by updating the last range in the merged array.
4. Return the Merged List: After processing all ranges, return the merged list.

Python Code:

```
def combine_overlapping_ranges(ranges):
    # Step 1: Sort the ranges based on the starting value
    ranges.sort(key=lambda x: x[0])

    merged = []

    for current in ranges:
        # If merged is empty or no overlap with the last range, add the current range
        if not merged or merged[-1][1] < current[0]:
            merged.append(current)
        else:
            # There is an overlap, so we merge the current range
            merged[-1][1] = max(merged[-1][1], current[1])

    return merged

# Example Cases
# Example 1
input1 = [[1,3],[2,6],[8,10],[15,18]]
print(combine_overlapping_ranges(input1))
# Expected Output: [[1,6],[8,10],[15,18]]

# Example 2
input2 = [[1,4],[4,5]]
print(combine_overlapping_ranges(input2))
# Expected Output: [[1,5]]
```

Explanation of the Code:

- **Sorting:** The ranges are sorted based on the starting point to ensure that they are processed in order.
- **Merging Logic:** The if condition inside the loop checks whether there is an overlap. If there is none, it simply adds the range. If there is an overlap, we merge by adjusting the end of the last added range to the maximum of the two overlapping ends.
- The final merged list is returned, containing non-overlapping ranges.

This solution is efficient with a time complexity of $O(n \log n)$ due to the sorting step, followed by $O(n)$ for the merging step, leading to an overall complexity of $O(n \log n)$.