1. <https://leetcode.com/problems/merge-intervals/>

Given a collection of intervals, merge all overlapping intervals.

**Example 1:**

**Input:** [[1,3],[2,6],[8,10],[15,18]]

**Output:** [[1,6],[8,10],[15,18]]

**Explanation:** Since intervals [1,3] and [2,6] overlaps, merge them into [1,6].

**Example 2:**

**Input:** [[1,4],[4,5]]

**Output:** [[1,5]]

**Explanation:** Intervals [1,4] and [4,5] are considered overlapping.

2. <https://leetcode.com/problems/insert-interval/>

Given a set of non-overlapping intervals, insert a new interval into the intervals (merge if necessary).

You may assume that the intervals were initially sorted according to their start times.

**Example 1:**

**Input:** intervals = [[1,3],[6,9]], newInterval = [2,5]

**Output:** [[1,5],[6,9]]

**Example 2:**

**Input:** intervals = [[1,2],[3,5],[6,7],[8,10],[12,16]], newInterval = [4,8]

**Output:** [[1,2],[3,10],[12,16]]

**Explanation:** Because the new interval [4,8] overlaps with [3,5],[6,7],[8,10].

3. <https://leetcode.com/problems/count-of-range-sum/>

Given an integer array nums, return the number of range sums that lie in [lower, upper] inclusive.  
Range sum S(i, j) is defined as the sum of the elements in nums between indices i and j (i ≤ j), inclusive.

**Note:**  
A naive algorithm of *O*(*n*2) is trivial. You MUST do better than that.

**Example:**

**Input:** *nums* = [-2,5,-1], *lower* = -2, *upper* = 2,

**Output:** 3

**Explanation:** The three ranges are : [0,0], [2,2], [0,2] and their respective sums are: -2, -1, 2.