**Problem#1508: range Sum of Sorted Subarray Sums (Medium)**

<https://leetcode.com/problems/range-sum-of-sorted-subarray-sums/description/>

**My Solution:**

1. Initialize alist as an empty list to hold the results.
2. Create an outer for loop and Iterate in range of (length of nums minus 1).

Set total to be nums at index i.

Append total to alist.

1. Create an inner for loop and iterate in the range of I + 1 to the length of nums

Add element at index j of nums to total and add it to alist.

1. Since the last element in nums has not been added to alist, append this element to alist.
2. Sort alist in ascending order.
3. Return the sum of alist from index (left -1 ) to right modulo (10\*\*9 + 7).

Note: index left and right are indexed from 1 in alist. So to adjust to 0-based index, subtract 1.

class Solution:

def rangeSum(self, nums: List[int], n: int, left: int, right: int) -> int:

alist = []

for i in range(len(nums) - 1):

total = nums[i]

alist.append(total)

for j in range(i + 1, len(nums)):

total += nums[j]

alist.append(total)

alist.append(nums[-1])

alist = sorted(alist)

return sum(alist[left - 1: right]) % (10 \*\* 9 + 7)