

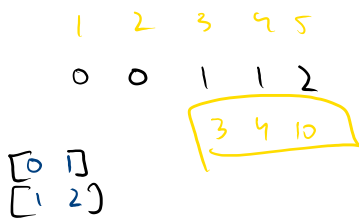
```

public static int maximumSum(int[] A, int[][] ops) {
    //Step 1: creating contri array
    int[] contri = new int[A.length];

    for(int[] op: ops){
        int sp = op[0];
        int ep = op[1];
        contri[sp]++;
        if(ep < A.length) contri[ep+1]--;
    }

    for(int i = 1; i < A.length; i++) contri[i] += contri[i-1];
    //Sort both
    Arrays.sort(A);
    Arrays.sort(contri);
    // summation
    int sum = 0;
    int md = 1000000007;
    for(int i = 0; i < A.length; i++) sum = (sum + md + (A[i] * contri[i]) % md) % md;
    return sum % md;
}

```



- ① Create contrib array
- ② Sort both array
2 1 5 3 4
- ③ $Cal(Ans) = \sum A[i] * contr[i]$

Range Queries
↳ Prefix Sum

10⁹ - 1
4 - N → O(N)

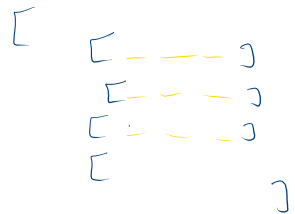
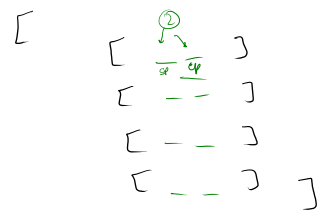
5	2	1	3	6	6	8
0	1	2	3	4	5	6

Prefix sum array

5	7	8	11	17	23	31
0	1	2	3	4	5	6

getSum(s, e, pref) <
if (s == 0) return pref[e];
else return pref[e] - pref[s-1];

int[] prefix = new int[n];
prefix[0] = arr[0];
for (i = 1; i < n; i++)
prefix[i] = prefix[i-1] + arr[i];
for (Query q, ans) <
SP, EP
System.out.println(getSum(s, e, pref));



15 + 11 - 7 + 2

A = B - C + D

A = 15 + 11 - 7 + 2

B = 15 + 11 - 7 + 2

C = 15 + 11 - 7 + 2

D = 15 + 11 - 7 + 2

A = B - C + D

A = 15 + 11 - 7 + 2

B = 15 + 11 - 7 + 2

C = 15 + 11 - 7 + 2

D = 15 + 11 - 7 + 2

15 + 11 - 7 + 2

A = B - C + D

A = 15 + 11 - 7 + 2

B = 15 + 11 - 7 + 2

C = 15 + 11 - 7 + 2

D = 15 + 11 - 7 + 2

A = B - C + D

A = 15 + 11 - 7 + 2

B = 15 + 11 - 7 + 2

C = 15 + 11 - 7 + 2

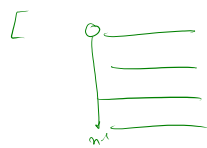
D = 15 + 11 - 7 + 2

0 1 1 2 3
1 2 3 4 5

1 2 3 1 0

0 2 0 1
1 3 4 9
2 2 3 5
Sum 22

A = arr[0][0]
B = arr[0][1]
C = arr[1][0]
D = arr[1][1]
arr[0][0] = A + B + C + D



3 3 7

2 1 5
3 3 7
5

K = 3

0 1 2 3 4 5 6 7

System.out.println(0 2 2 5 5 3 3 0)

O(N) O(N)