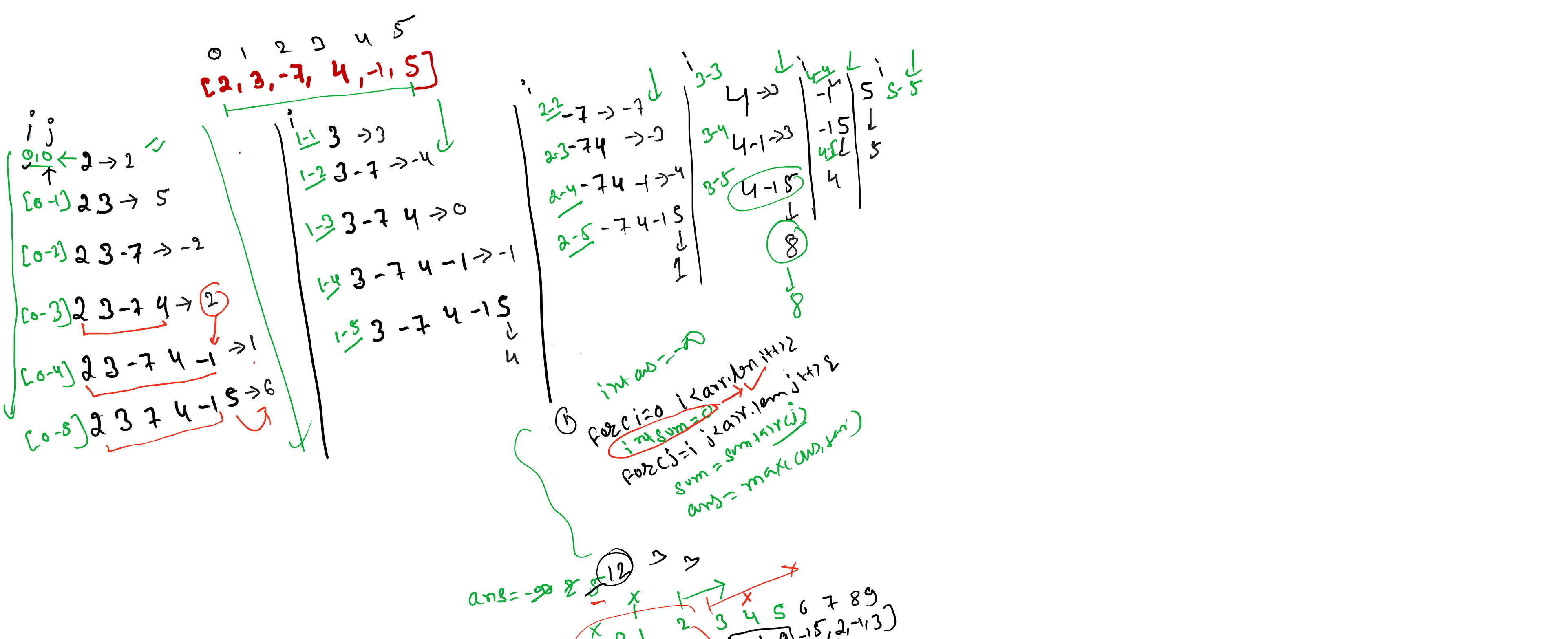
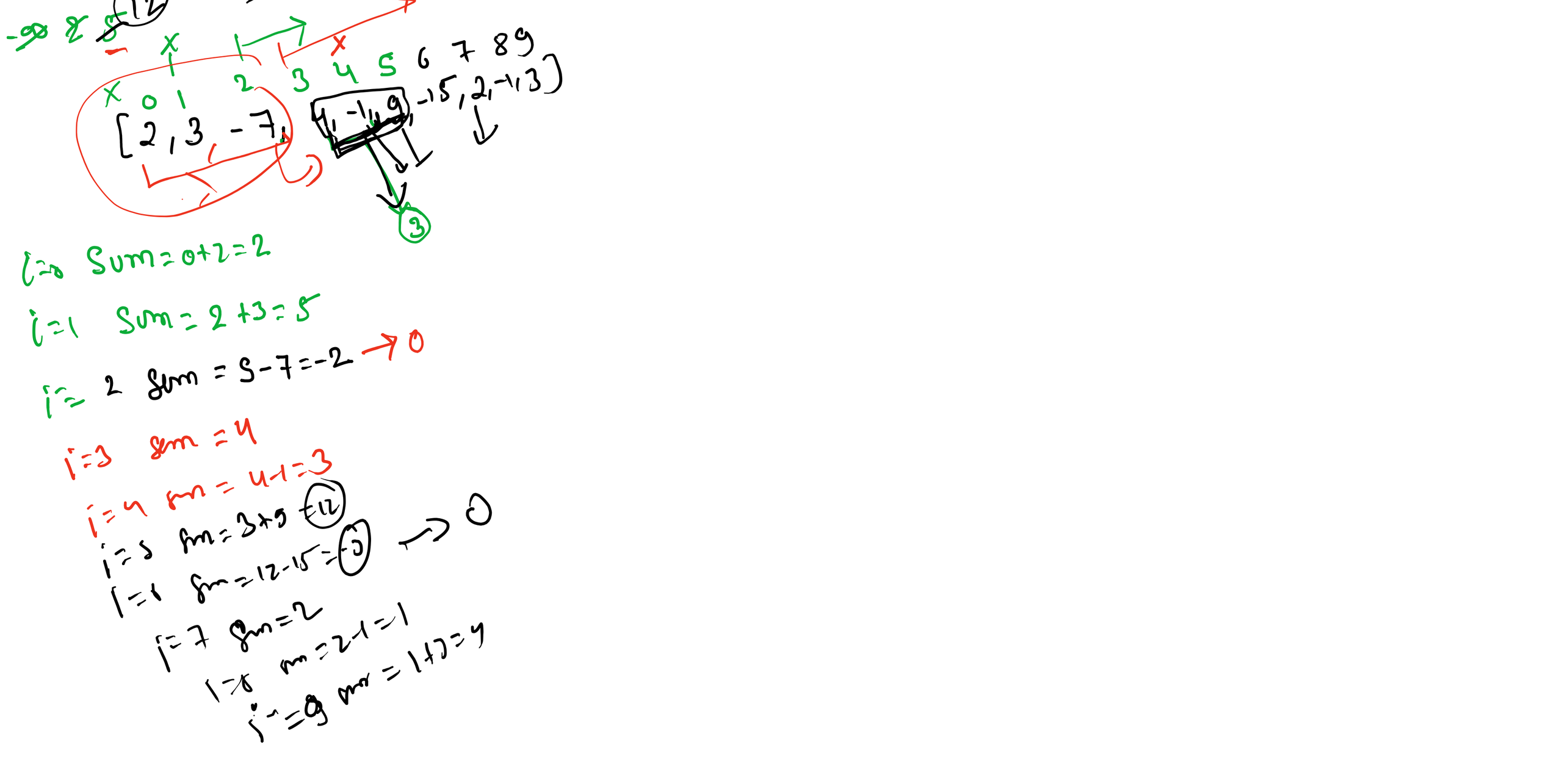
Lec-3



```java
public static int kadanes_Algo(int[] arr) {
        int sum = 0;
        int ans = Integer.MIN_VALUE;
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
            ans = Math.max(ans, sum);
        }
        return ans;
    }
```

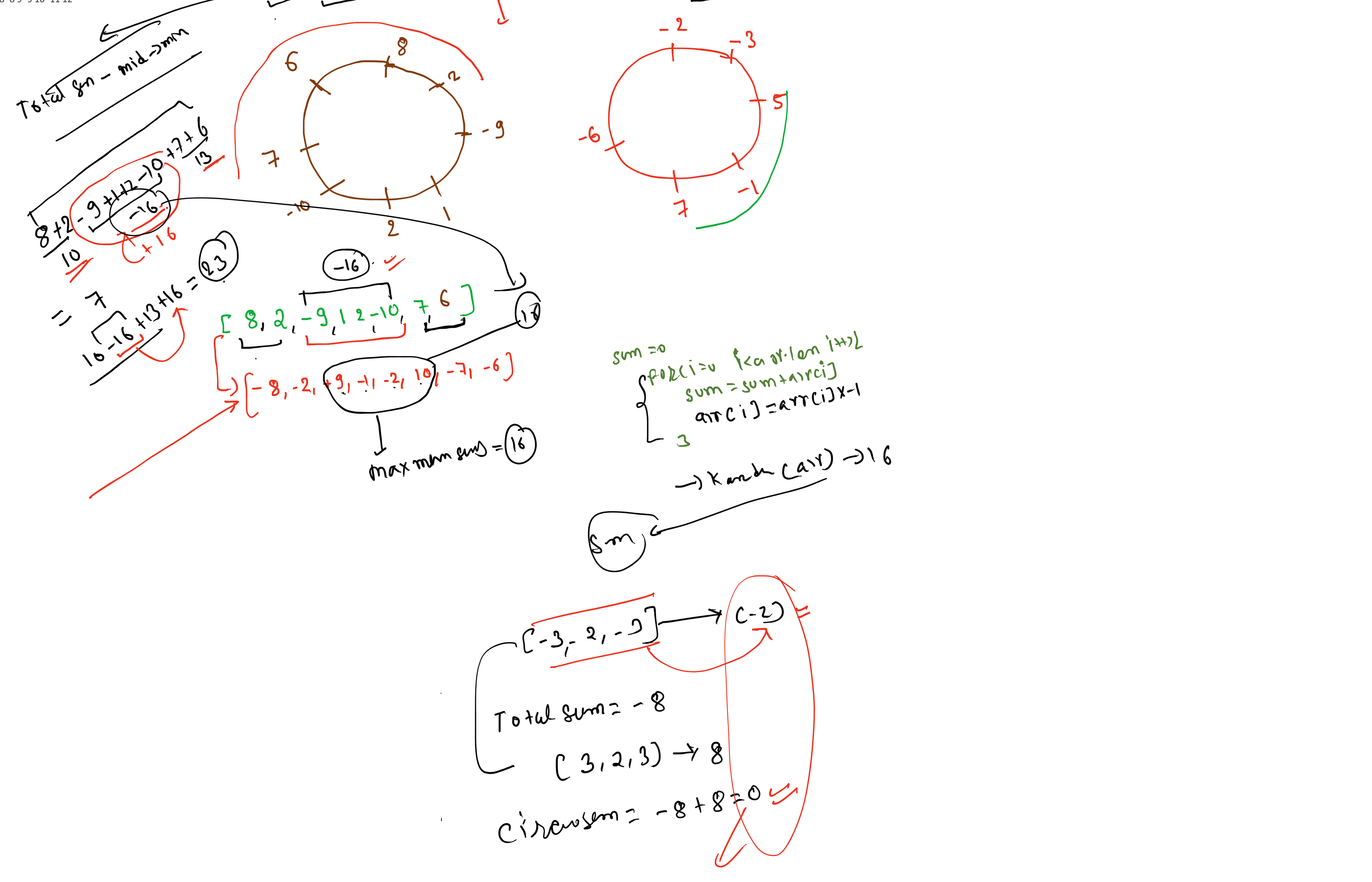


Given a **circular integer array** `nums` of length `n`, return *the maximum possible sum of a non-empty* ***subarray*** *of* `nums`.

A **circular array** means the end of the array connects to the beginning of the array. Formally, the next element of `nums[i]` is `nums[(i + 1) % n]` and the previous element of `nums[i]` is `nums[(i - 1 + n) % n]`.

A **subarray** may only include each element of the fixed buffer `nums` at most once. Formally, for a subarray `nums[i], nums[i + 1], ..., nums[j]`, there does not exist `i <= k1`, `k2 <= j` with `k1 % n == k2 % n`.

8 -8 9 -9 10 -11 12



You are given two arrays a and b, both of length n.

Let's define a function $f(l,r) = \sum_{l \le i \le r} a_i \cdot b_i$.

Your task is to reorder the elements (choose an arbitrary order of elements) of the array b to minimize the value of $\sum_{1 \le l \le r \le n} f(l,r)$. Since the answer can be very large, you have to print it modulo 998244353. Note that you should **minimize the answer but not its remainder**.

Input

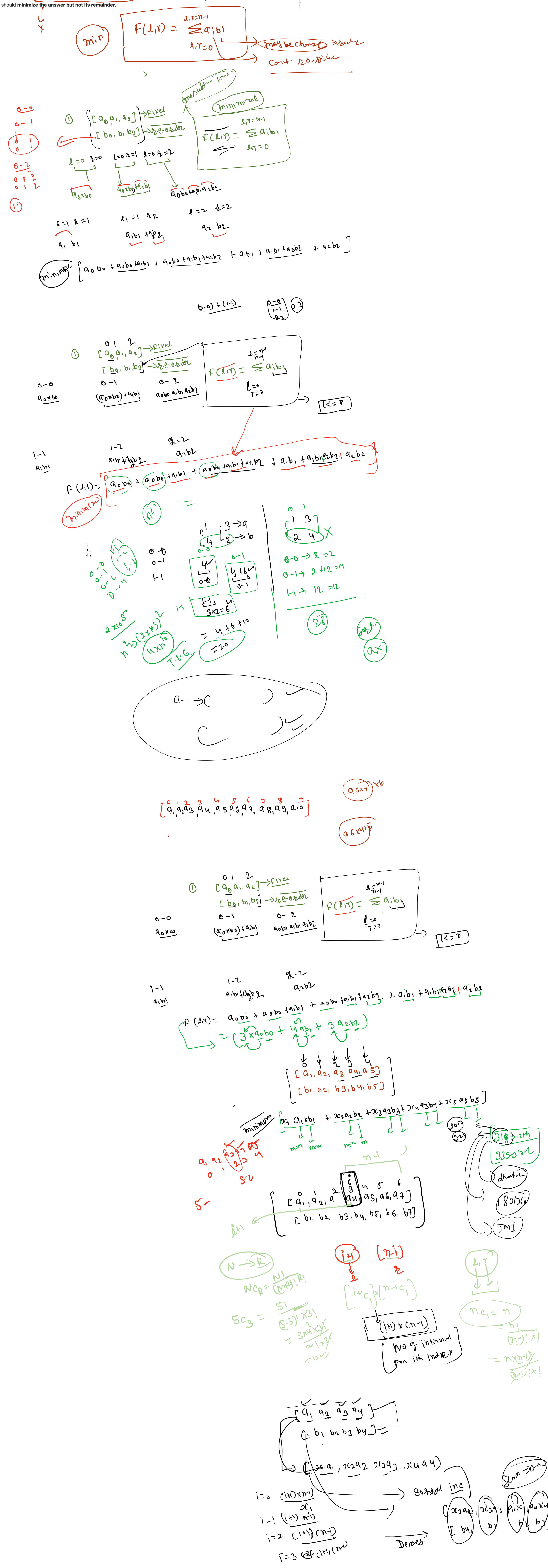
The first line of the input contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the number of elements in a and b.

The second line of the input contains n integers $a_1, a_2, \dots, a_n$ ($1 \le a_i \le 10^6$), where a_i is the i-th element of a.

The third line of the input contains n integers $b_1, b_2, \dots, b_n$ ($1 \le b_j \le 10^6$), where b_j is the j-th element of b.

Output

Print one integer — the minimum possible value of $\sum_{1 \le l \le r \le n} f(l,r)$ after rearranging elements of b, taken modulo 998244353. Note that you should **minimize the answer but not its remainder**.



```java
public static void main(String[] args) {
        // TODO Auto-generated method stub
        Scanner sc = new Scanner(System.in);
        int n = sc.nextInt();
        long[] a = new long[n];
        for (int i = 0; i < a.length; i++) {
            a[i] = sc.nextLong();
            a[i] = a[i] * (i + 1) * (n - i);// a[i] kitne times interval me aayega ye bhi store kr liya

        }
        Integer[] b = new Integer[n];
        for (int i = 0; i < b.length; i++) {
            b[i] = sc.nextInt();
        }
        Arrays.sort(a);// increasing order me
        Arrays.sort(b, Collections.reverseOrder());// dec order mei
        long ans = 0;
        int mod = 998244353;
        for (int i = 0; i < b.length; i++) {
            ans = (ans + (a[i]%mod * b[i]) % mod) % mod;
        }
        System.out.println(ans);

    }
```