Count the number of nice subarrays

nums = [1 1 2 1 1] k = 3

Count the number of nice subarrays

nums = [1 1 2 1 1] k = 3

odd = 3

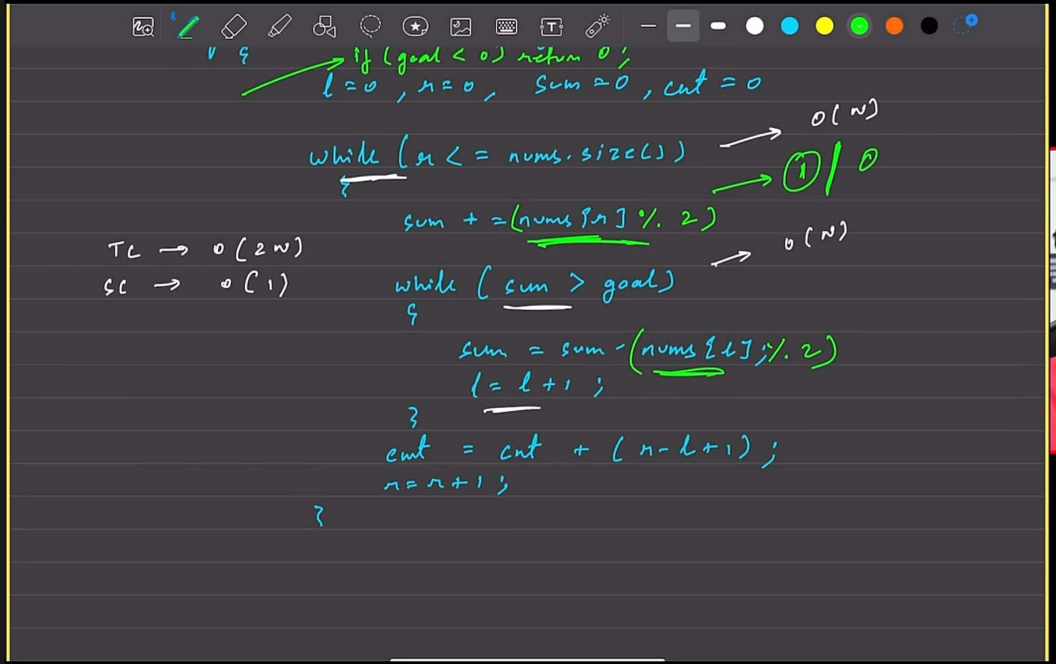
Count the number of nice subarrays

odd → 1
even → 0

nums = [1 5 2 1 1] k = 3

[1 1 0 1 1] k = 3

sum == k


if (goal < 0) return 0;
l = 0, r = 0, sum = 0, cnt = 0
while (r <= nums.size())
O(N)
sum += (nums[r] % 2)
TC → O(2N)
SC → O(1)
while (sum > goal)
O(N)
sum = sum - (nums[l] % 2)
l = l + 1;
cnt = cnt + (r - l + 1);
r = r + 1;

no. of subarrays where | sum <= goal

func(nums, goal)

func(nums, goal - 1)

TC → O(2 × 2N)

SC → O(1)

func(list<int> nums, goal)

{

if (goal < 0) return 0;

l = 0, r = 0, sum = 0, cnt = 0

while (r <= nums.size())

{

sum += (nums[r] % 2)

while (sum > goal)

{

O(N)

1 | 0

O(N)

TC → O(2N)

SC → O(1)