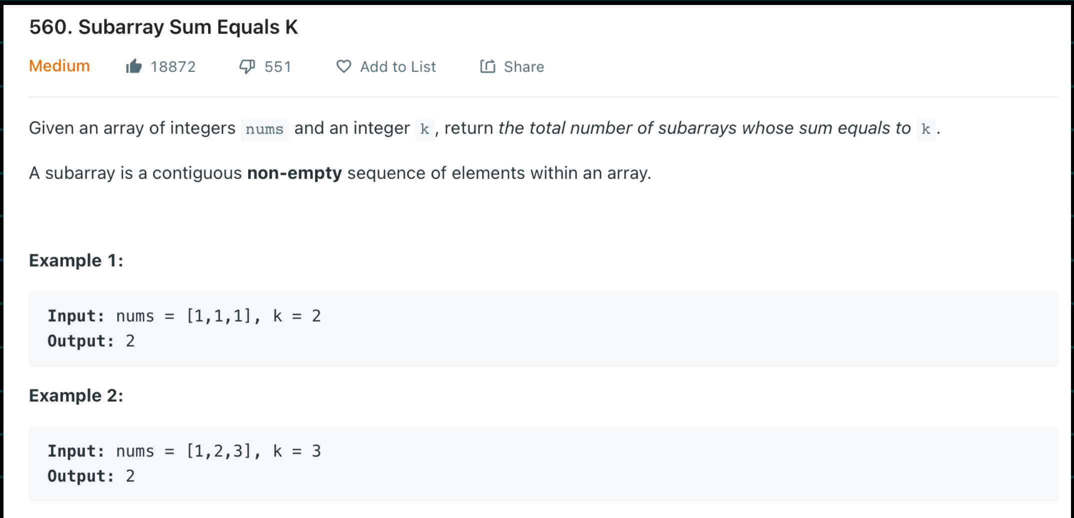
.Subarray Sum Equals K



560. Subarray Sum Equals K

Given an array of integers `nums` and an integer `k`, return *the total number of subarrays whose sum equals to* `k`.

A subarray is a contiguous **non-empty** sequence of elements within an array.

Example 1:

```
Input: nums = [1,1,1], k = 2
Output: 2
```

Example 2:

```
Input: nums = [1,2,3], k = 3
Output: 2
```

Brute :-

[1, 1, 1]   k=2

i j

sum += arr[j] // sum= 0+1
= 1

if (sum == k)
cnt++

[1, 1, 1]   j++

i j

sum = 1+1
= 2

cnt ++ True // 1

[1, 1, 1]   j++

i j

sum= 3

[1,1,1]   i++

i j

sum= 0+1
= 1

j++

[1,1,1]

i j

sum= 1+1=2
cnt++ // 2

[1,1,1]   sum= 0+1
= 1

i j

return cnt // 2

T.C - $O(n^2)$

S.C - $O(1)$

Optimal solution:-

Prefix sum / Hashing

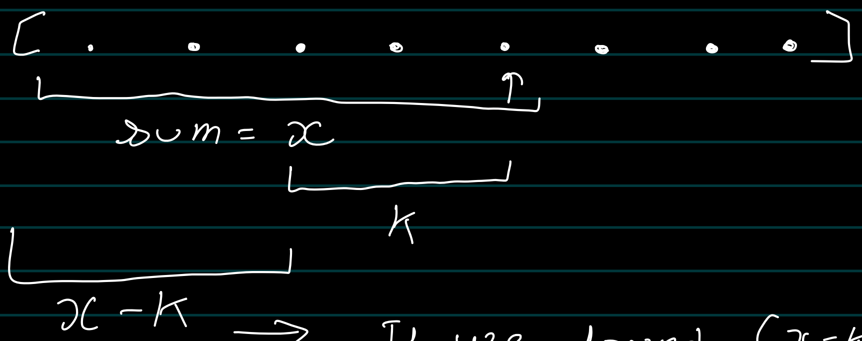
↓

[2, 5, 7, 8, 9]

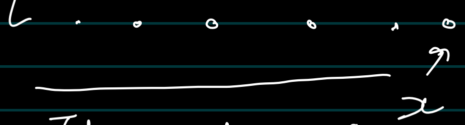
↑ standing right here

having a Hashmap which have sum of previous numbers

$[0+2, \underbrace{2+5}_{a}, \underbrace{7+a}_{b}, \underbrace{8+b}_{c}, 9+c]$



→ If we found (x-k) before x



If any found (x-k) then we have an subarray == k

Dry run:-

1st 2nd 3rd 4th 5th 6th 7th 8th 9th 10th

arr[] = [1, 2, 3, -3, 1, 1, 1, 4, 2, -3] k=3

insert(0) = 1

prefix sum:- $\cancel{0}$ $\cancel{1}$ $\cancel{3}$ $\cancel{6}$ $\cancel{3}$ $\cancel{4}$ $\cancel{5}$ $\cancel{6}$ $\cancel{10}$ 12 9

Find = 1-3 = -2 | $\boxed{3-3 = 0}$ $\boxed{3-6 = 3}$ 3-3 = 0 $\boxed{4-3 = 1}$ | 3-5 = 2 $\boxed{6-3 = 3}$ | 10-3 = 7 | 12-3 = 9 | 9-3 = 6

cnt = $\cancel{0}$ $\cancel{1}$ $\cancel{2}$ $\cancel{3}$ $\cancel{4}$ $\cancel{5}$ $\cancel{6}$ $\cancel{7}$ 8

↓ 0 3rd, ↓ 3 4th, ↓ -3 5th, ↓ 1, ↓ 1 7th, ↓ 1 7th, ↓ 10th, ↓ 10th

3 : 2   6 : 2

mpp
12 : 1
10 : 1
5 : 1
4 : 1
6 : $\cancel{1}$2
3 : $\cancel{1}$2
1 : 1
0 : 1

Pre / freq

return cnt

1st 2nd 3rd 4th 5th

arr[] = [3, -3, 1, 1, 1] k=3

insert(0) = 1

prefix = $\cancel{0}$ $\cancel{3}$ $\cancel{0}$ $\cancel{1}$ $\cancel{2}$ 3

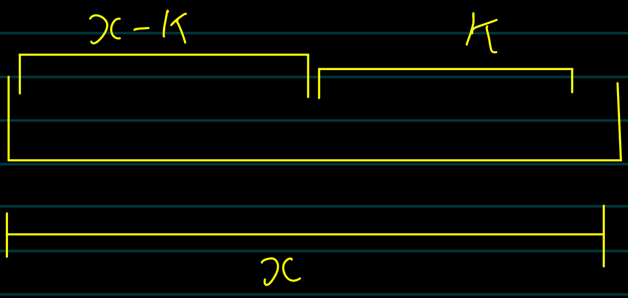
Find: $\boxed{3-3 = 0}$ | 0-3 = -3 | 1-3 = -2 | 2-3 = -1 $\boxed{3-3 = 0}$

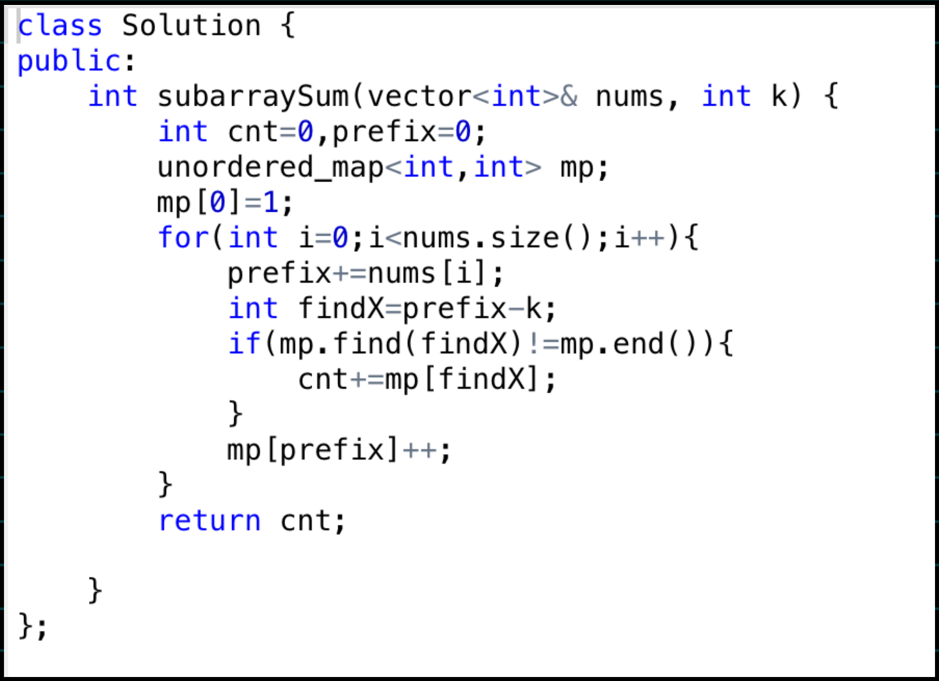
cnt = $\cancel{0}$ $\cancel{1}$ 2 3

↓ 1st, ↓ 5th, ↓ 5th

0 : 2

1 : 1
3 : $\cancel{1}$2
0 : $\cancel{1}$2





```
class Solution {
public:
    int subarraySum(vector<int>& nums, int k) {
        int cnt=0,prefix=0;
        unordered_map<int,int> mp;
        mp[0]=1;
        for(int i=0;i<nums.size();i++){
            prefix+=nums[i];
            int findX=prefix-k;
            if(mp.find(findX)!=mp.end()){
                cnt+=mp[findX];
            }
            mp[prefix]++;
        }
        return cnt;

    }
};
```

T.C - O(n)
+ O(logn)

Avg case O(1)

wrost case O(n)

S.C= O(1)