

Subarray sum divisible by k

974. Subarray Sums Divisible by K

Medium

6012

252

Add to List

Share

Given an integer array `nums` and an integer `k`, return the number of non-empty **subarrays** that have a sum divisible by `k`.

A **subarray** is a **contiguous** part of an array.

Example 1:

Input: `nums = [4,5,0,-2,-3,1]`, `k = 5`

Output: 7

Explanation: There are 7 subarrays with a sum divisible by `k = 5`:

`[4, 5, 0, -2, -3, 1]`, `[5]`, `[5, 0]`, `[5, 0, -2, -3]`, `[0]`, `[0, -2, -3]`, `[-2, -3]`

Example 2:

Input: `nums = [5]`, `k = 9`

Output: 0



Brute force

Since we have to find sum of all subarray divisible by `k`

→ Nested loop

`[4, 5, 0, -2, -3, 1]` `k = 5`
i j

At `i = 0`

`j = 0`

`sum = 0 + 4 = 4`

`if (sum % k == 0) false`

`j = 1`

`sum = 4 + 5 = 9`

`if (sum % k == 0) false`

`j = 2`

`sum = 9 + 0 = 9`

`if (sum % k == 0) false`

`j = 3`

`sum = 9 + (-2) = 7`

`if (sum % k == 0) false`

`j = 4`

`sum = 7 - 3 = 4`

`if (sum % k == 0) false`

`j = 5`

`sum = 4 + 1 = 5`

`if (sum % k == 0) True cnt++`

→ `i++`, `a[i] = 5`

`j = 0`

`sum = 0 + 5`

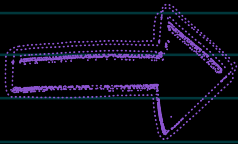
`if (sum % k == 0) True cnt++`

do on

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

S.C - $O(1)$

TLE



Optimal approach

Hashing :-

subarray sum divisible by k

$$arr = [a_0, a_1, a_2, \dots, a_j, \dots, a_{n-1}, a_n], k$$

Need

$$a[i:j] \text{ s.t. } \text{sum}(a[i:j]) \% k = 0$$

Calculate prefix sum for array

$$F = [a_0, a_1, a_2, \dots, a_i, \dots, a_j, \dots, a_n]$$

$\boxed{\begin{array}{c} S_i \\ S_j = S_i \% k \end{array}}$

 $\boxed{\begin{array}{c} S_j \\ S_j = S_j \% k \end{array}}$

$$\text{if } S_j = S_i$$

$$S_j \% k - S_i \% k = 0$$

$$(S_j - S_i) \% k = 0$$

so let us take example :-

$$\begin{array}{cccccc} 0 & 1 & 2 & 3 & 4 & 5 \\ [4, 5, 0, -2, -3, 1] \end{array}$$

$i=0$ $i=1$ $i=2$ $i=3$ $i=4$ $i=5$
 Prefix: 4, 4+5, 4+0, 4-2, 4-3, 4+1
 =9, =9, =7, =4, =5
 Find - 4%5, 9%5, 9%5, 7%5, 4%5, 5%5
 =4, =4, =4, =2, =4, =0
 False True True False True
 insert = 4, 4++, 4++, 3, 0++

Rem	Frequency
0	1
4	4 3
3	

$$cnt = \text{mpp}[\text{Find}]$$

$$= 0 + 1 + 2 = 3 + 3 = 6 + 1$$

$$= 7 \text{ ans.}$$

```
class Solution {
public:
    int subarraysDivByK(vector<int>& nums, int k) {
        unordered_map<int,int> mpp;
        mpp[0]=1;
        int sum=0,cnt=0;
        for(int i=0;i<nums.size();i++){
            sum+=nums[i];
            int findRem =sum%k ;
            if(findRem <0) findRem+=k;
            if(mpp.find(findRem)!=mpp.end()) cnt+=mpp[findRem];
            mpp[findRem]++;
        }
        return cnt;
    }
};
```