523. Continuous Subarray Sum

Attempted

Medium | Topics | Companies

Given an integer array nums and an integer k, return `true` *if* `nums` *has a* ***good subarray*** *or* `false` *otherwise.*

A **good subarray** is a subarray where:

- its length is **at least two**, and
- the sum of the elements of the subarray is a multiple of `k`.

Note that:

- A **subarray** is a contiguous part of the array.
- An integer `x` is a multiple of `k` if there exists an integer `n` such that `x = n * k`. `0` is **always** a multiple of `k`.

Example 1:

```
Input: nums = [23,2,4,6,7], k = 6
Output: true
Explanation: [2, 4] is a continuous subarray of size 2 whose elements sum up to 6.
```

Example 2:

```
Input: nums = [23,2,6,4,7], k = 6
Output: true
Explanation: [23, 2, 6, 4, 7] is an continuous subarray of size 5 whose elements sum up to 42.
42 is a multiple of 6 because 42 = 7 * 6 and 7 is an integer.
```

Example 3:

```
Input: nums = [23,2,6,4,7], k = 13
Output: false
```

Constraints:

- `1 <= nums.length <= 10^5`
- `0 <= nums[i] <= 10^9`
- `0 <= sum(nums[i]) <= 2^31 - 1`
- `1 <= k <= 2^31 - 1`

Accepted **410.8K** | Submissions **1.4M** | Acceptance Rate **28.5%**

Approach 1: Brute force

$T(n) : O(n^2)$
$S(n) : O(1)$

Approach 2:

We are asked to find the subarray. So we can think of sliding window. But it won't work.

Q: Why sliding window won't work?

To implement sliding window we must be able to have clear protocol on which should be performed either shrinking or sliding based on some criteria. But here we can't have a criteria to decide.

So sliding window can't be implemented. This problem can be solved using mathematical intuition.

Intuition:

let $k = 4$

ie.
$5 = 1 + 4 \times 1$

a	a % k
1	1
2	2
3	3
4	0
5	1
6	2

As we can see we get same modulo after every 3 modulos.

i.e.

$9 = 1 + 4 \times 2$

7	3
8	0
9	1

if we have two numbers x and y such that

$$x \% k == y \% k$$

This only happens when

① $x == y$

(or)

② $y = x + a \times k$ i.e. x is added with multiple of k.

we use this idea to solve the given problem

let array is 23 2 4 6 7 $k = 6$

① Compute prefix Sum

23 25 29 35 42

② Compute $a_i \% k$

0	1	2	3	4
5	1	5	5	0

tells that subarray from $i: 0$ to 4 is a multiple of 6.

tells that $6a$ is added to 0th index element i.e. 6 is added

tells that $6a$ is added to 0th index element i.e. 12 is added

Some cases:

i) 6 1 2 1 4 $k = 6$

prefix Sum : 6 7 9 10 14

% k : 0 1 3 4 2

↳ here we got 0 but this is not answer becoz the subarray size must be atleast 2.

To handle this we insert 0 at the beginning with index -1.

unordered map m

m[0] = -1

prefix = 0

for (i : 0 to n-1)

{

prefix = prefix + nums[i]

r = Prefix % k

if (r is not in map)

m[r] = i

else if (r is in map && i - m[r] > 1)

return true

}

return false

here we are not modifying m[r] if r is in map.

i.e. 5 5 5 (indexes 0, 1, 2)

here 5 is in map but 1-0 is not greater than 1 so it goes to next iteration.

If we observe carefully the 5 index in map is not changed to 1. It is still 0 only.

↳ To check if subarray size is atleast 2.

The order of this is clearly the main catch here. The interchange of this will make some cases to fail.

$T(n) : O(n)$

$S(n) : O(n)$