Array : Video - 23

Medium
Leetcode
- 523

Continuous Subarray

MEDIUM

Leetcode - 523

Continuous Subarray Sum...

Companies :-

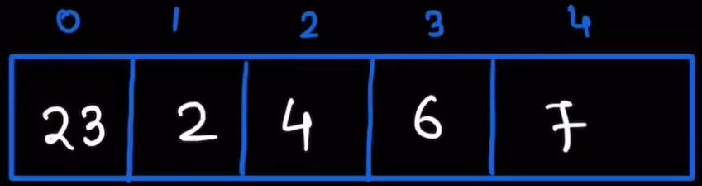
Input : {23, 2, 4, 6, 7} , K = 6

Output : True

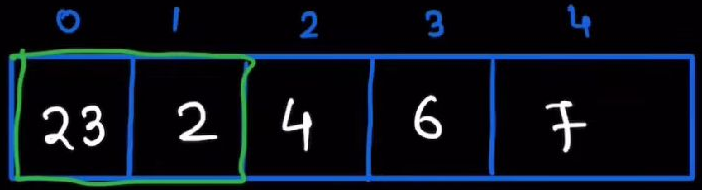
Input : {23, 2, 4, 6, 7} , K = 6

Output : True

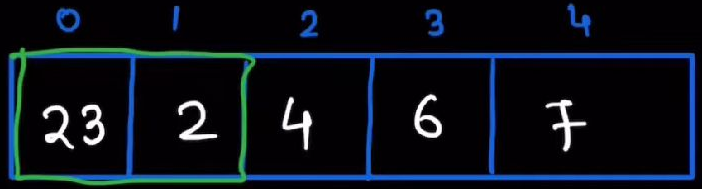
- (•) contiguous
- (•) size >= 2
- (•) sum % K = 0



K = 6

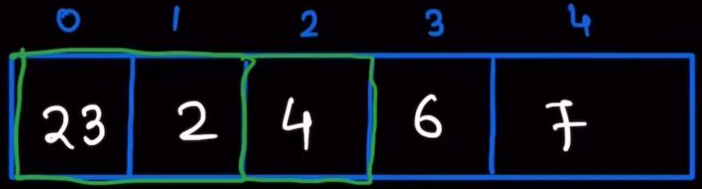


K = 6

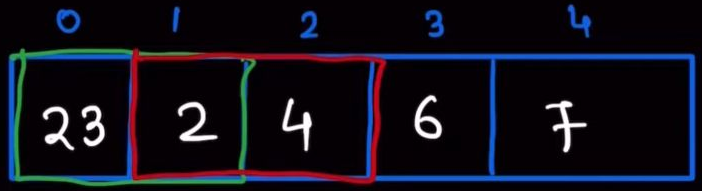


K = 6

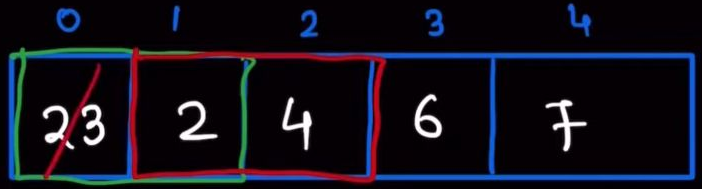
25/6 = 0



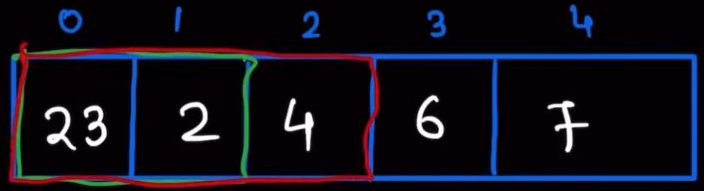
K = 6



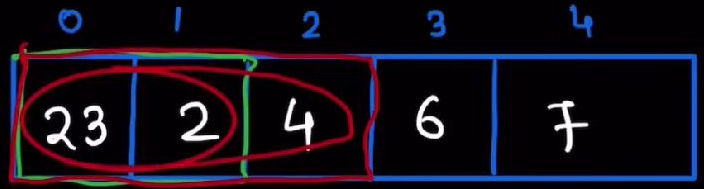
K = 6



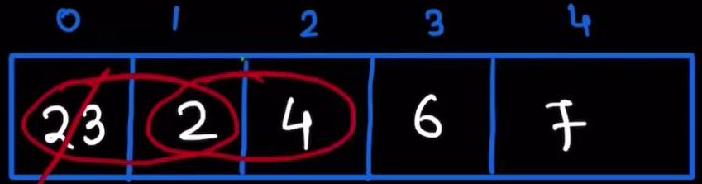
K = 6



K = 6

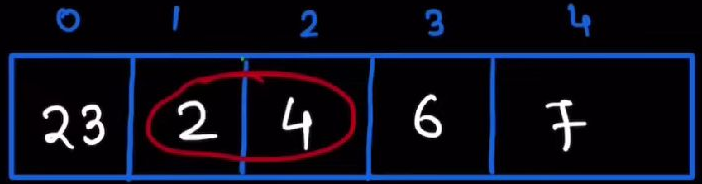


K = 6



K = 6

Sliding Window



K = 6

Basic Math

$31 \% 4 = 3$

$(31 + 8) \% 4 =$

$39 \% 4 = 3$

Basic Math

$31 \% 4 = 3$

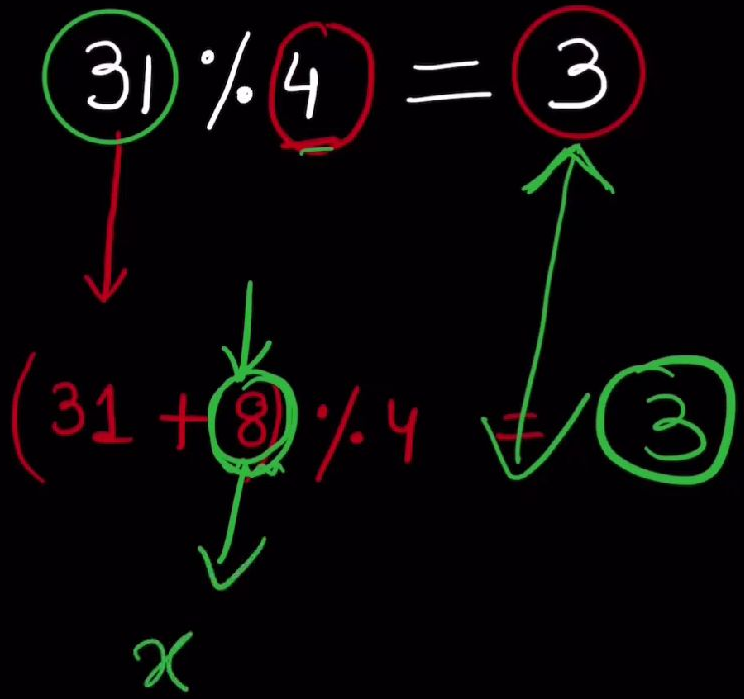
$(31 + 8) \% 4 =$

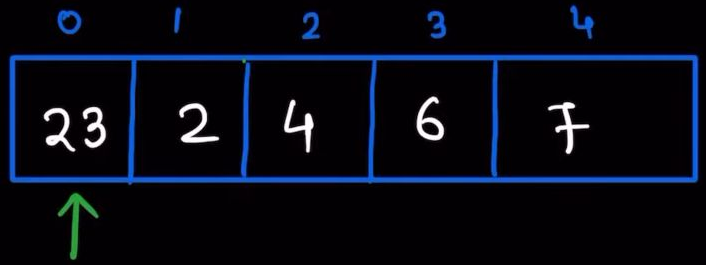
$39 \% 4 = 3$

$31 + 12$

$43 \% 4 = 3$

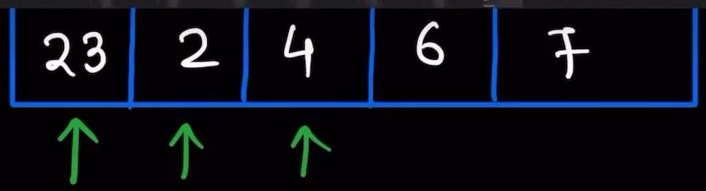
Basic Math





K = 6

23 % →

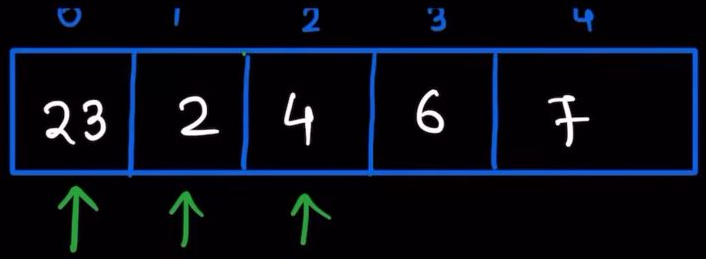


K = 6

23 $\xrightarrow{\%6}$ 5

23+2 $\xrightarrow{\%6}$ 1

23 +2+4 $\xrightarrow{\%6}$ 5

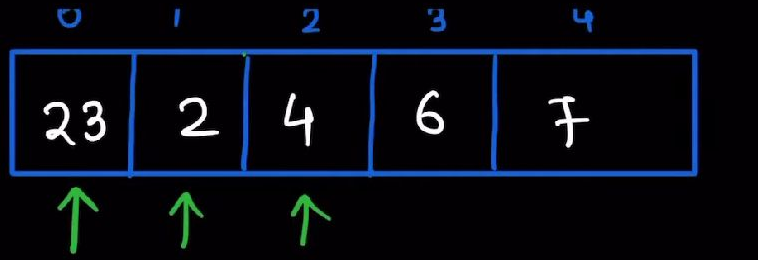


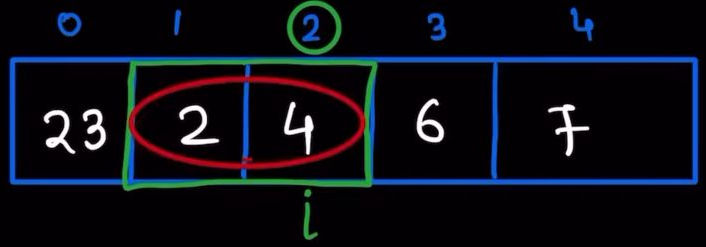
K = 6

$23 \xrightarrow{\%6} 5$

$23+2 \xrightarrow{\%6} 1$

$23+2+4 \xrightarrow{\%6} 5$


0 1 2 3 4
23 2 4 6 7
K = 6
23 %6 5
23+2 %6 1
23 +2+4 %6 5
multiple of 6



K = 6

map

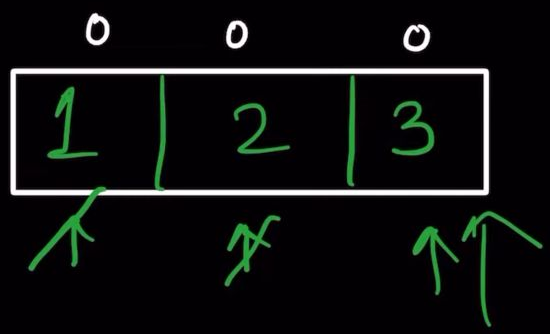
remain	index
5	0 ←
1	1

2 − 0 = 2

Sum = (23) % 6 = 5

Sum = (23 + 2) % 6 = 1

Sum = (23 + 2 + 4) % 6 = 5



K = 6

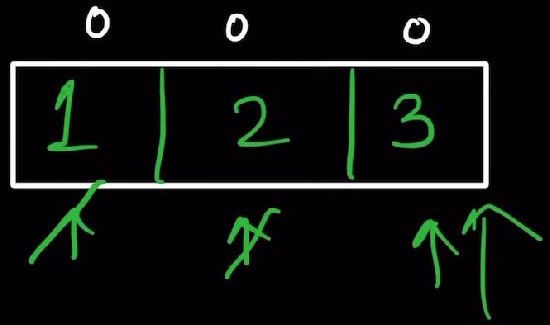
map

remainder	index
1	0
3	1

sum = (1) % 6 = 1

sum = (1+2) % 6 = 3

sum = (1+2+3) % 6 = 0



K = 6

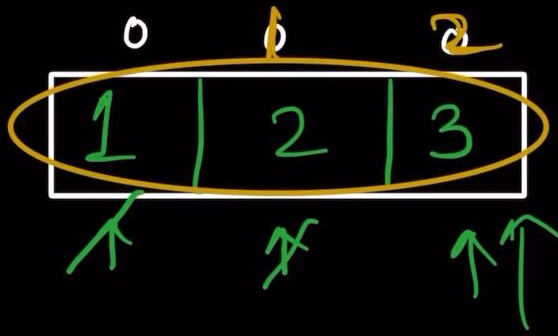
map

remainder	index
1	0
3	1
0	-1

Sum = (1) % 6 = 1

Sum = (1+2) % 6 = 3

Sum = (1+2+3) % 6 = 0



K = 6

map

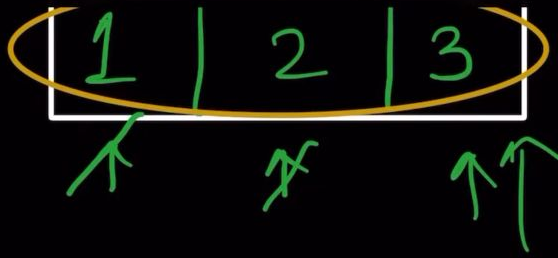
remainder	index
1	0
3	1
0	-1

sum = (1) % 6 = 1

sum = (1+2) % 6 = 3

sum = (1+2+3) % 6 = 0

2 - (-1) = 3



K = 6

map

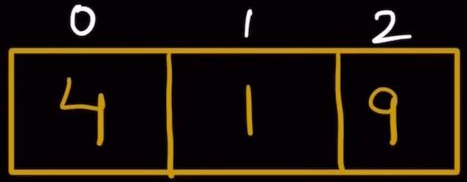
remainder	index
1	0
3	1
0	-1

sum = (1) % 6 = 1

sum = (1+2) % 6 = 3

sum = (1+2+3) % 6 = 0

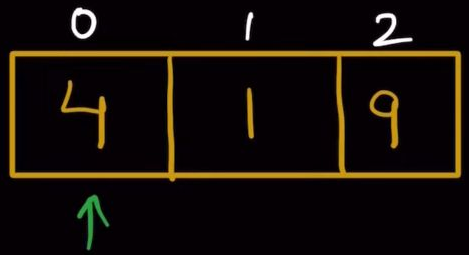
2 - (-1) = 3 >= 2



K = 4

map

rem	ind
0	-1



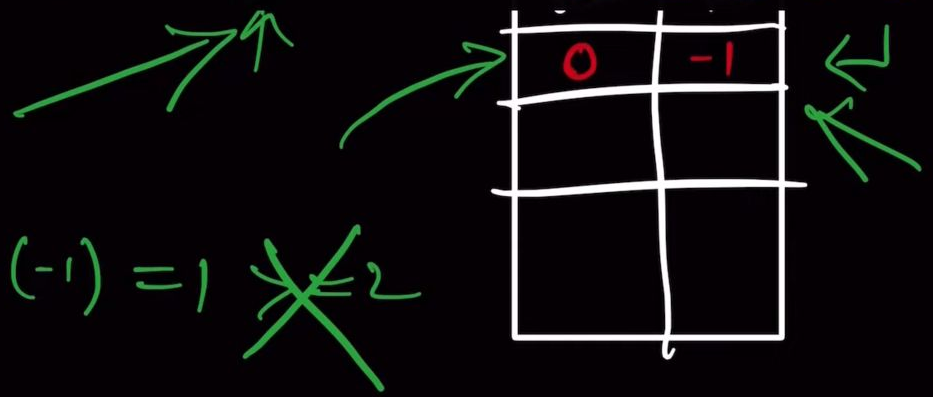
K = 4

Sum = 4 % 4 = 0

map

rem	ind
0	-1

0 - (-1) = 1 ~~2~~



$0 - (-1) = 1$ ~~$\neq 2$~~

"modulo की जय हो"

LeetCode Explore Problems Interview New Contest Discuss Store Premium 10

C++ Autocomplete

```cpp
class Solution {
public:
    bool checkSubarraySum(vector<int>& nums, int k) {
        int n = nums.size();

        unordered_map<int, int> mp;

        int sum = 0;
        mp[0] = -1;

        for(int i = 0; i<n; i++) {
            sum += nums[i];

            int remainder = sum%k;

            //check if it's present in map (past me kahi dekha hai ?)
            if(mp.find(remainder) != mp.end()) {

                if(i - mp[remainder] >= 2)
                    return true;

            } else {
                mp[remainder] = i;
            }
        }

        return false;
    }
};
```