Subarray Sum Equals K

Solution

Given an array of integers and an integer **k**, you need to find the total number of continuous subarrays whose sum equals to **k**.

Example 1:

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

Note:

1. The length of the array is in range [1, 20,000].
2. The range of numbers in the array is [-1000, 1000] and the range of the integer **k** is [-1e7, 1e7].

Hide Hint #1

Will Brute force work here? Try to optimize it.

Hide Hint #2

Can we optimize it by using some extra space?

Hide Hint #3

What about storing sum frequencies in a hash table? Will it be useful?

Hide Hint #4

sum(i,j)=sum(0,j)-sum(0,i), where sum(i,j) represents the sum of all the elements from index i to j-1. Can we use this property to optimize it.

Java

```java
class Solution {
    public int subarraySum(int[] nums, int k) {
        int sum = 0;
        int count = 0;
        for(int i = 0; i < nums.length; i++) {
            sum += nums[i];
            if(map.containsKey(sum - k)) {
                count += map.get(sum - k);
            }
            map.put(sum, map.getOrDefault(sum, 0) + 1);
        }

        return count;
    }
}
```

Custom Testcase (Contribute)

Run Code Submit