Subarray Range With Given Sum

solved by	Senan
Platform	GeeksForGeeks
difficulty	Medium
tags	Hash Map, Prefix Sum
language	C++
solved on	@15/10/2024
link	https://www.geeksforgeeks.org/problems/subarray-range-with-given-sum0128/1
Completion	✓

Intuition

We are tasked with finding the number of subarrays that sum up to a given target. The key observation here is that if the sum of elements from index `i` to `j` equals the target, then the cumulative sum at `j` minus the cumulative sum at `i-1` equals the target. This means we can efficiently track cumulative sums and check if the difference between the current cumulative sum and the target exists in a hash map of previously seen cumulative sums.

Approach

We iterate over the array while maintaining a running cumulative sum of the elements encountered so far. For each element, we check if the difference between the current cumulative sum and the target exists in the map. If it does, it means that there is a subarray ending at the current index that sums to the target. We then add the frequency of this difference (as stored in the map) to our count. At each step, we also update the map to include the current cumulative sum.

Complexity

Time Complexity:

The time complexity is $O(n)$, where n is the size of the array. We only iterate through the array once, and all operations involving the hash map (insertions and lookups) take $O(1)$ on average.

Space Complexity:

The space complexity is $O(n)$, as we are using an unordered map to store cumulative sums. In the worst case, all the cumulative sums could be different, requiring space proportional to the size of the array.

Code

```
class Solution {
  public:
    // Function to count the number of subarrays which add up to the given sum.
    int subArraySum(vector<int>& arr, int tar) {
        int n = arr.size();
        int count = 0;
        unordered_map<int, int> mpp;

        int sum = 0;
        mpp[0] = 1;
        for(int i = 0; i < arr.size(); i++) {
            sum += arr[i];
            int target = sum - tar;
            if(mpp.count(target))
                count += mpp[target];
            mpp[sum]++;  /
                 }
        return count;
    }
};
```