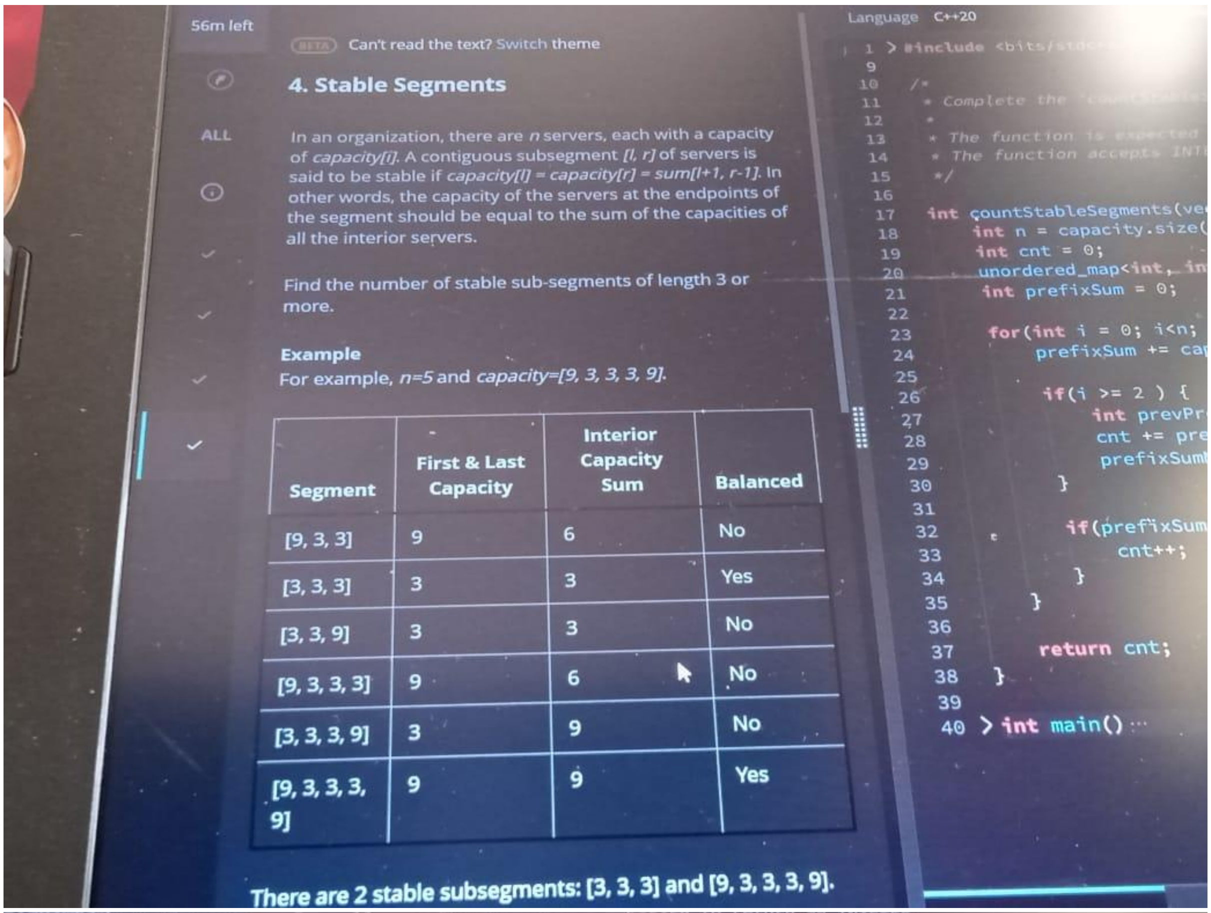


56m left

Can't read the text? Switch theme

4. Stable Segments

In an organization, there are *n* servers, each with a capacity of *capacity[i]*. A contiguous subsegment *[l, r]* of servers is said to be stable if *capacity[l] = capacity[r] = sum[l+1, r-1]*. In other words, the capacity of the servers at the endpoints of the segment should be equal to the sum of the capacities of all the interior servers.

Find the number of stable sub-segments of length 3 or more.

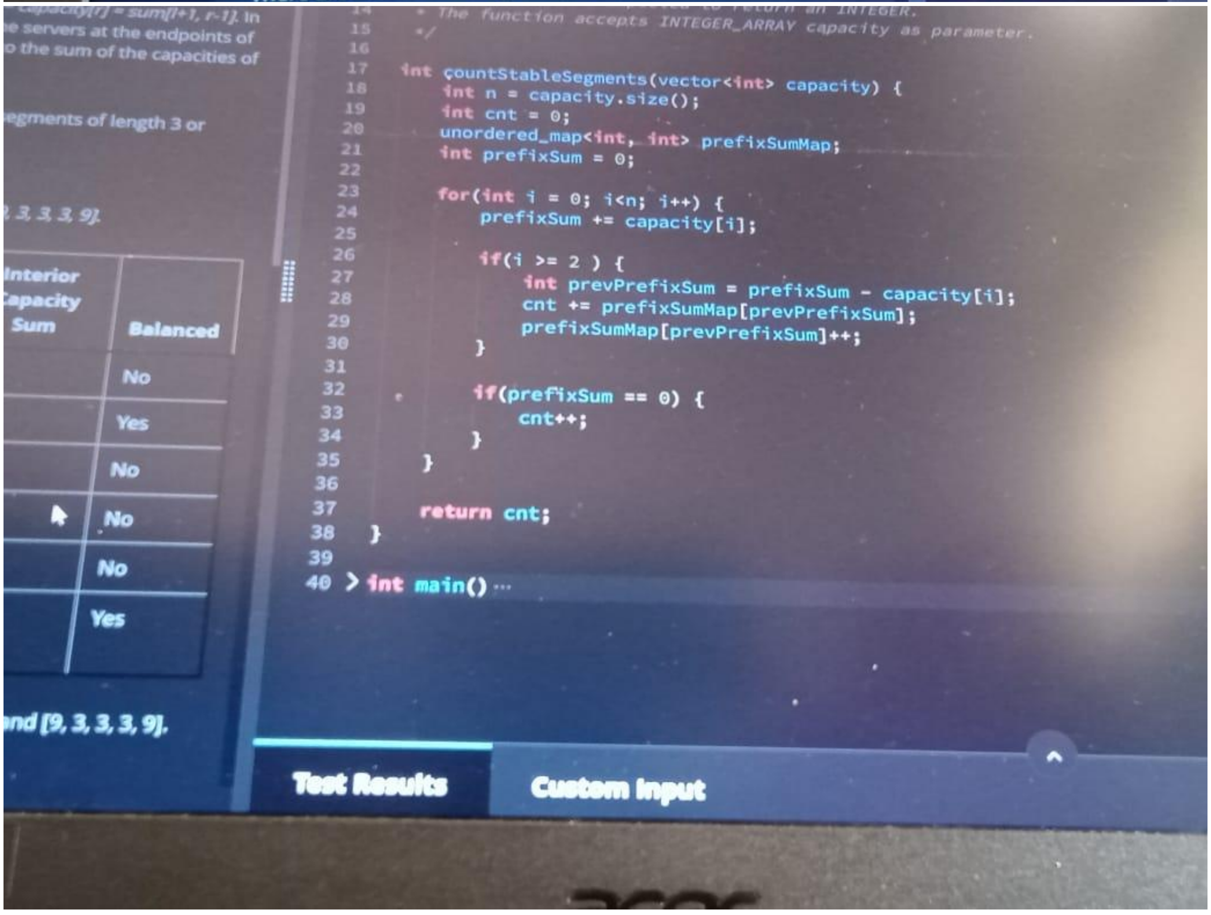
Example

For example, *n=5* and *capacity=[9, 3, 3, 3, 9]*.

Segment	First & Last Capacity	Interior Capacity Sum	Balanced
[9, 3, 3]	9	6	No
[3, 3, 3]	3	3	Yes
[3, 3, 9]	3	3	No
[9, 3, 3, 3]	9	6	No
[3, 3, 3, 9]	3	9	No
[9, 3, 3, 3, 9]	9	9	Yes

There are 2 stable subsegments: [3, 3, 3] and [9, 3, 3, 3, 9].

Language C++20



```
#include <bits/stdc

/*
 * Complete the 'countStable
 *
 * The function is expected to return an INTEGER.
 * The function accepts INTEGER_ARRAY capacity as parameter.
 */

int countStableSegments(vector<int> capacity) {
    int n = capacity.size();
    int cnt = 0;
    unordered_map<int, int> prefixSumMap;
    int prefixSum = 0;

    for(int i = 0; i<n; i++) {
        prefixSum += capacity[i];

        if(i >= 2 ) {
            int prevPrefixSum = prefixSum - capacity[i];
            cnt += prefixSumMap[prevPrefixSum];
            prefixSumMap[prevPrefixSum]++;
        }

        if(prefixSum == 0) {
            cnt++;
        }
    }

    return cnt;
}

int main() ⋯
```