Subarray sum equals k in C++

```
#include <iostream>
#include <vector>
#include <unordered_map>
using namespace std;
class SubarraySumEqualsK {
public:
    static int sol(const std::vector<int>& arr,
int target) {
        int ans = 0;
        std::unordered_map<int, int> map;
        map[0] = 1;
        int sum = 0;

        for (int i = 0; i < arr.size(); i++) {
            sum += arr[i];
            int rsum = sum - target;
            if (map.find(rsum) != map.end()) {
                ans += map[rsum];
            }
            map[sum]++;
        }
        return ans;
    }
};

int main() {
    vector<int> arr = {3, 9, -2, 4, 1, -7, 2, 6,
-5, 8, -3, -7, 6, 2, 1};
    int k = 5;
    cout << SubarraySumEqualsK::sol(arr,
k) << std::endl;
    return 0;
}
```

Example Input

```
vector<int> arr = {3, 9, -2, 4, 1, -7, 2, 6,
-5, 8, -3, -7, 6, 2, 1};
int k = 5;
```

Expected Output: 5

Step-by-Step Dry Run

Step	i	arr[i]	sum (Prefix Sum)	rsum = sum - k	map[rsum] (if exists)	ans (count of subarrays)	map[sum] (updated)
1	0	3	3	-2	0	0	{0:1, 3:1}
2	1	9	12	7	0	0	{0:1, 3:1, 12:1}
3	2	-2	10	5	0	0	{0:1, 3:1, 12:1, 10:1}
4	3	4	14	9	0	0	{0:1, 3:1, 12:1, 10:1, 14:1}
5	4	1	15	10	✓1	1	{0:1, 3:1, 12:1, 10:1, 14:1, 15:1}
6	5	-7	8	3	✓1	2	{0:1, 3:1, 12:1, 10:1, 14:1, 15:1, 8:1}
7	6	2	10	5	0	2	{0:1, 3:1, 12:1, 10:2, 14:1, 15:1, 8:1}
8	7	6	16	11	0	2	{0:1,

							3:1, 12:1, 10:2, 14:1, 15:1, 8:1, 16:1}
9	8	-5	11	6	0	2	{0:1, 3:1, 12:1, 10:2, 14:1, 15:1, 8:1, 16:1, 11:1}
10	9	8	19	14	✓1	3	{0:1, 3:1, 12:1, 10:2, 14:1, 15:1, 8:1, 16:1, 11:1, 19:1}
11	10	-3	16	11	✓1	4	{0:1, 3:1, 12:1, 10:2, 14:1, 15:1, 8:1, 16:2, 11:1, 19:1}
12	11	-7	9	4	0	4	{0:1, 3:1, 12:1, 10:2, 14:1, 15:1, 8:1, 16:2, 11:1, 19:1, 9:1}
13	12	6	15	10	✓2	6	{0:1, 3:1, 12:1, 10:2, 14:1, 15:2, 8:1, 16:2, 11:1, 19:1,

							9:1}
14	13	2	17	12	✓1	7	{0:1, 3:1, 12:1, 10:2, 14:1, 15:2, 8:1, 16:2, 11:1, 19:1, 9:1, 17:1}
15	14	1	18	13	0	7	{0:1, 3:1, 12:1, 10:2, 14:1, 15:2, 8:1, 16:2, 11:1, 19:1, 9:1, 17:1, 18:1}

Final Output

✓ **Output:** 7

Output:-
7