|  |  |
| --- | --- |
| **Count of Subarrays Having Sum Equal to K in C++** | |
| #include <iostream>  #include <unordered\_map>  #include <vector>  using namespace std;  int solution(vector<int>& arr, int target) {  int ans = 0;  unordered\_map<int, int> map;  map[0] = 1; // Initialize with sum 0 having count 1  int sum = 0;    for (int i = 0; i < arr.size(); i++) {  sum += arr[i];  if (map.find(sum - target) != map.end()) {  ans += map[sum - target];  }  map[sum]++;  }    return ans;  }  int main() {  vector<int> arr = {1, 1, 1};  int target = 2;  cout << solution(arr, target) << endl; // Output: 2  return 0;  } | **Dry Run for Input:**  vector<int> arr = {1, 1, 1};  int target = 2;  **Initial Values:**   * ans = 0 * map = {0: 1} (since map[0] = 1 initially) * sum = 0   **Iteration Breakdown:**   | **i** | **arr[i]** | **sum (cumulative sum)** | **sum - target** | **map[sum - target]** | **ans** | **map (updated)** | | --- | --- | --- | --- | --- | --- | --- | | 0 | 1 | 1 | 1 - 2 = -1 | Not found | 0 | {0: 1, 1: 1} | | 1 | 1 | 2 | 2 - 2 = 0 | map[0] = 1 (found) | 1 | {0: 1, 1: 1, 2: 1} | | 2 | 1 | 3 | 3 - 2 = 1 | map[1] = 1 (found) | 2 | {0: 1, 1: 2, 2: 1, 3: 1} |   **Explanation of each iteration:**   * **At i = 0**:   + arr[0] = 1   + sum = 1   + We check if sum - target = 1 - 2 = -1 is in map. It is **not**.   + We update the map with map[1]++, so map = {0: 1, 1: 1}. * **At i = 1**:   + arr[1] = 1   + sum = 2   + We check if sum - target = 2 - 2 = 0 is in map. It **is** (map[0] = 1), so we add 1 to ans (i.e., ans += 1).   + We update the map with map[2]++, so map = {0: 1, 1: 1, 2: 1}. * **At i = 2**:   + arr[2] = 1   + sum = 3   + We check if sum - target = 3 - 2 = 1 is in map. It **is** (map[1] = 1), so we add 1 to ans (i.e., ans += 1).   + We update the map with map[3]++, so map = {0: 1, 1: 2, 2: 1, 3: 1}.   **Final Output:**   * The total number of subarrays whose sum equals target = 2 is **2**. |
| Output: 2 | |