Question Explanation : (PRE REQUESTIES)

Given an array of size N , check whether we can divide the array into two equal parts which has same sum.

If so , return true else return false.

Observation :

* If the sum is not divisible by 2 , then return false

Real Question :

Given an array of N integers , check whether the given array is split into 3 equal paritions.return the number of ways to divide the array , other wise -1.

Step 1 : First calculate the sum of array , if and only if , it is divisible by 3 then only proceed else return -1.

Step 2 : If the sum is 3Y , then each part is Y.

Step 3 : Store the all frequencies in the hashmap , along with the count and if the sum is 2Y coun the occurrence of Y.

Step 4 : return the ans;

import java.util.HashMap;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int n = scanner.nextInt(); // Read the value of n

int[] b = new int[n]; // Initialize an array b of size n

for (int i = 0; i < n; i++) {

b[i] = scanner.nextInt();

}

int tot\_sum = 0;

int c = 0;

// Calculate the total sum of the elements in b

for (int i = 0; i < n; i++) {

tot\_sum += b[i];

}

// Check if the total sum is divisible by 3

if (tot\_sum % 3 == 0) {

HashMap<Integer, Integer> prefixSumCount = new HashMap<>(); // HashMap to store frequency of prefix sums

int prefix\_sum = 0;

int y = tot\_sum / 3;

for (int i = 0; i < n - 1; i++) {

prefix\_sum += b[i]; // Sum of first i elements

// If prefix\_sum equals 2\*y, check how many times y has occurred so far

if (prefix\_sum == 2 \* y) {

c += prefixSumCount.getOrDefault(y, 0); // Add the frequency of prefix\_sum / 2 (which is y) to count

}

// Increment the frequency of the current prefix\_sum

prefixSumCount.put(prefix\_sum, prefixSumCount.getOrDefault(prefix\_sum, 0) + 1);

}

}

System.out.println(c); // Output the value of c

scanner.close();

}

}