**Problem**

You are given 2 arrays A and B, each of the size N. Each element of these arrays is either a positive integer or −1. The total number of −1′s that can appear over these 2 arrays are ≥1 and ≤2.

Now, you need to find the number of ways in which we can replace each −1 with a non-negative integer, such that the sum of both of these arrays is equal.

****Input format****

* First line: An integer N
* Second line: N space-separated integers, where the of these denotes A[i]
* Third line: N space-separated integers, where the ith of these denotes B[i]

****Output format****

If there exists a finite number X, then print it. If the answer is not a finite integer, then print '****Infinite****'.

****Constraints****

1≤N≤105

−1≤A[i],B[i]≤109

The −1′s may spread out among both arrays, and their quantity is between 1 and 2 (both inclusive)

**Solution :**

import java.util.Scanner;

public class EqualSumArrays {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

int n = scanner.nextInt();

int[] arr1 = new int[n];

int[] arr2 = new int[n];

int sum1 = 0, sum2 = 0, countNeg1Arr1 = 0, countNeg1Arr2 = 0;

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

arr1[i] = scanner.nextInt();

if (arr1[i] == -1) countNeg1Arr1++;

else sum1 += arr1[i];

}

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

arr2[i] = scanner.nextInt();

if (arr2[i] == -1) countNeg1Arr2++;

else sum2 += arr2[i];

}

int totalNeg1 = countNeg1Arr1 + countNeg1Arr2;

int diff = Math.abs(sum1 - sum2);

if (totalNeg1 == 0) {

System.out.println(sum1 == sum2 ? 1 : 0);

} else if (totalNeg1 == 1) {

if (diff <= 9) {

System.out.println(1);

} else {

System.out.println(0);

}

} else {

if ((diff % totalNeg1 == 0) && (diff / totalNeg1 <= 9)) {

System.out.println(1);

} else if (diff <= totalNeg1 \* 9) {

System.out.println("Infinite");

} else {

System.out.println(0);

}

}

}

}