**Check if two arrays are equal or not**

Show Topic Tags   

[Goldman Sachs](http://practice.geeksforgeeks.org/company/Goldman%20Sachs/)

Given two arrays of equal length, the task is to find if given arrays are equal or not. Two arrays are said to be equal if both of them contain same set of elements, arrangements (or permutation) of elements may be different though.

**Note :** If there are repetitions, then counts of repeated elements must also be same for two array to be equal.

Examples:

Input : A[] = {1, 2, 5, 4, 0};

B[] = {2, 4, 5, 0, 1};

Output : 1

Input : A[] = {1, 2, 5};

B[] = {2, 4, 15};

Output : 0

**Input:**  
The first line of input contains an integer T denoting the no of test cases. Then T test cases follow.  Each test case contains an integer N denoting the size of the array. Then in the next two lines are N space separated values of the array of arrays A[] and B[].  
  
**Output:**  
For each test case in a new line print 1 if the arrays are equal else print 0.  
  
**Constraints:**  
1<=T<=100  
1<=N<=100  
1<=A[],B[]<=1000  
  
**Example:  
Input:**  
2  
5  
1 2 5 4 0  
2 4 5 0 1  
3  
1 2 5  
2 4 15  
**Output:**  
1  
0

\*\*For More Examples Use Expected Output\*\*

<http://practice.geeksforgeeks.org/problems/check-if-two-arrays-are-equal-or-not/0>

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication241;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.HashSet;

import java.util.Iterator;

/\*\*

\*

\* @author Administrador

\*/

public class JavaApplication241 {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) throws IOException {

// TODO code application logic here

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int t = Integer.parseInt(br.readLine());

while(t-- > 0) {

int n = Integer.parseInt(br.readLine());

String[] sa = br.readLine().trim().split(" ");

int[] a = new int[n];

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

a[i] = Integer.parseInt(sa[i]);

}

String[] sb = br.readLine().trim().split(" ");

int[] b= new int[n];

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

b[i] = Integer.parseInt(sb[i]);

}

Arrays.sort(a);

Arrays.sort(b);

int ans = 1;

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

if(a[i] != b[i]) {

ans = 0;

break;

}

}

System.out.println(ans);

}

}

}

---------con HASHING------------

import java.util.\*;

import java.lang.\*;

import java.io.\*;

class GFG {

public static void main(String[] args) throws IOException {

// TODO code application logic here

BufferedReader br = new BufferedReader(new InputStreamReader(System.in));

int t = Integer.parseInt(br.readLine());

while(t-- > 0) {

int n = Integer.parseInt(br.readLine());

String[] sa = br.readLine().trim().split(" ");

int[] a = new int[n];

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

a[i] = Integer.parseInt(sa[i]);

}

String[] sb = br.readLine().trim().split(" ");

int[] b= new int[n];

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

b[i] = Integer.parseInt(sb[i]);

}

HashMap<Integer, Integer> hma = new HashMap<Integer, Integer>();

for(int i =0; i<a.length; i++) {

if(hma.containsKey(a[i])) {

hma.put(a[i], hma.get(a[i])+ 1);

}else{

hma.put(a[i], 1);

}

}

HashMap<Integer, Integer> hmb = new HashMap<Integer, Integer>();

for(int i =0; i<b.length; i++) {

if(hmb.containsKey(b[i])) {

hmb.put(b[i], hmb.get(b[i])+ 1);

}else{

hmb.put(b[i], 1);

}

}

int ans = 1;

Iterator<Map.Entry<Integer, Integer>> it = hma.entrySet().iterator();

while (it.hasNext()) {

Map.Entry pair = (Map.Entry)it.next();

// System.out.println(pair.getKey() + " = " + pair.getValue());

if(!hmb.containsKey((Integer)pair.getKey())) {

ans =0;

}

if(hmb.containsKey((Integer)pair.getKey())) {

if(pair.getValue() != hmb.get((Integer)pair.getKey())) {

ans =0;

}

}

// it.remove(); // avoids a ConcurrentModificationException

}

Iterator<Map.Entry<Integer, Integer>> it2 = hmb.entrySet().iterator();

while (it2.hasNext()) {

Map.Entry pair = (Map.Entry)it2.next();

// System.out.println(pair.getKey() + " = " + pair.getValue());

if(!hma.containsKey((Integer)pair.getKey())) {

ans =0;

}

if(hma.containsKey((Integer)pair.getKey())) {

if(pair.getValue() != hmb.get((Integer)pair.getKey())) {

ans =0;

}

}

// it.remove(); // avoids a ConcurrentModificationException

}

System.out.println(ans);

}

}

}