**In First But Second**

Submissions: [4297](https://practice.geeksforgeeks.org/problem_submissions.php?pid=2313)  Accuracy:

36.63%

   Difficulty: [Medium](https://practice.geeksforgeeks.org/Medium/0/0/)   Marks: 4

Associated Course(s): [Interview Preparation](https://practice.geeksforgeeks.org/courses/interview-preparation/)

Show Topic Tags   

[Accolite](https://practice.geeksforgeeks.org/company/Accolite/)

Given two arrays **A** and **B**of positive integers. Your task is to find numbers which are present in the first array, but not present in the second array.

**Input:**  
The first line of input contains an integer **T** denoting the number of test cases. Each test case contains space separated integers N and M which denotes the number of elements in the array A and B. Next two line contains space separated array elements.  
  
**Output:**  
Print space separated numbers present in the first array but not in the second.  
   
**Constraints:**  
1 <= T <= 100  
1 <= N, M <= 107  
1 <= Ai,Bi <= 1018

**Example:  
Input:**  
2  
6 5  
1 2 3 4 5 10  
2 3 1 0 5  
5 5  
4 3 5 9 11  
4 9 3 11 10  
  
**Output:**  
4 10   
5

**Explanation:**  
**Testcase 1:** 4 and 10 are present in first array while not in second array.

\*\* For More Input/Output Examples Use ['Expected Output'](https://practice.geeksforgeeks.org/problems/in-first-but-second/0#ExpectOP) option \*\*

Contributor: Shashwat Jain  
[Author: shashwat jain](https://auth.geeksforgeeks.org/user/shashwat%20jain/practice/)

<https://practice.geeksforgeeks.org/problems/in-first-but-second/0>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace ConsoleApp2

{

class Program

{

static void MostrarNoComunes(long[] A, long[] B, int N, int M)

{

HashSet<long> hashB = new HashSet<long>(B);

//Dictionary<long, int> hashA = new Dictionary<long, int>();

//for (int i = 0; i < N; i++)

//{

// if (hashA.ContainsKey(A[i])) hashA[A[i]]++;

// else hashA[A[i]] = 1;

//}

//Dictionary<long, int> hashB = new Dictionary<long, int>();

//for (int i = 0; i < M; i++)

//{

// if (hashB.ContainsKey(B[i])) hashB[B[i]]++;

// else hashB[B[i]] = 1;

//}

for (int i = 0; i < A.Length; i++)

{

if (!hashB.Contains(A[i]))

{

Console.Write(A[i] + " ");

}

}

Console.WriteLine();

}

static void Main(string[] args)

{

int t = int.Parse(Console.ReadLine());

while (t-- > 0)

{

string[] input = Console.ReadLine().Trim().Split(' ');

int N = int.Parse(input[0]);

int M = int.Parse(input[1]);

long[] A = Array.ConvertAll(Console.ReadLine().Trim().Split(' '), e => long.Parse(e));

long[] B = Array.ConvertAll(Console.ReadLine().Trim().Split(' '), e => long.Parse(e));

MostrarNoComunes(A, B, N, M);

}

Console.ReadLine();

}

}

}