**In First But Second**

Show Topic Tags   

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

Given two arrays, the task is that we 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 n and m elements in the array a[] and b[] respectively.  
  
**Output:**  
Print space separated numbers present in the first array but not in the second.  
   
**Constraints:**  
1<=T<=100  
1<=n,m<=1000  
1<=a[i],b[i]<=1000​  
  
**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

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

Contributor: Shashwat Jain

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

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package javaapplication250;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

import java.math.BigInteger;

import java.util.Arrays;

import java.util.HashMap;

import java.util.HashSet;

import java.util.LinkedHashSet;

/\*\*

\*

\* @author Administrador

\*/

public class JavaApplication250 {

/\*\*

\* @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) {

String[] input\_1 = br.readLine().trim().split(" ");

int n = Integer.parseInt(input\_1[0]);

int m = Integer.parseInt(input\_1[1]);

String[] input\_2 = br.readLine().trim().split(" ");

int[] a = new int[n];

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

a[i] = Integer.parseInt(input\_2[i]);

}

String[] input\_3 = br.readLine().trim().split(" ");

int[] b = new int[m];

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

b[i] = Integer.parseInt(input\_3[i]);

}

//Arrays.sort(a);

Arrays.sort(b);

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

if(Arrays.binarySearch(b, a[i]) < 0) {

System.out.print(a[i] + " ");

}

}

System.out.println();

}

}

}

---------EDITORIAL (HASHING)----------------

Find elements which are present in first array and not in second

Given two arrays, the task is that we find numbers which are present in first array, but not present in the second array.

Input : a[] = {1, 2, 3, 4, 5, 10};

b[] = {2, 3, 1, 0, 5};

Output : 4 10

4 and 10 are present in first array, but

not in second array.

Input : a[] = {4, 3, 5, 9, 11};

b[] = {4, 9, 3, 11, 10};

Output : 5

[**Recommended: Please solve it on “*PRACTICE* ” first, before moving on to the solution.**](http://practice.geeksforgeeks.org/problems/in-first-but-second/0)

**Method 1 (Simple)**  
A Naive Approach is to use two loops and check element which not present in second array.

|  |
| --- |
| // C++ simple program to find elements  // which are not present in second array  #include<bits/stdc++.h>  using namespace std;    // Function for finding elements which are there  // in a[] but not in b[].  void findMissing(int a[], int b[], int n, int m)  {      for (int i=0; i<n; i++)      {          int j;          for (j=0; j<m; j++)              if (a[i] == b[j])                  break;            if (j == m)              cout << a[i] << " ";      }  }    //Driven code  int main()  {      int a[] = { 1, 2, 6, 3, 4, 5 };      int b[] = { 2, 4, 3, 1, 0 };      int n = sizeof(a)/sizeof(a[0]);      int m = sizeof(b)/sizeof(b[1]);      findMissing(a, b, n, m);      return 0;  } |

Run on IDE

Output:

6 5

**Method 2 (Use Hashing)**  
In this method, we store all elements of second array in a hash table ([unordered\_set](http://www.geeksforgeeks.org/unorderd_set-stl-uses/)). One by one check all elements of first array and print all those elements which are not present in the hash table.

|  |
| --- |
| // C++ efficient program to find elements  // which are not present in second array  #include<bits/stdc++.h>  using namespace std;    // Function for finding elements which are there  // in a[] but not in b[].  void findMissing(int a[], int b[], int n, int m)  {      // Store all elements of second array in a hash      // table      unordered\_set <int> s;      for (int i = 0; i < m; i++)          s.insert(b[i]);        // Print all elements of first array that are      // not present in hash table      for (int i = 0;  i < n; i++)          if (s.find(a[i]) == s.end())              cout << a[i] << " ";  }    //Driven code  int main()  {      int a[] = { 1, 2, 6, 3, 4, 5 };      int b[] = { 2, 4, 3, 1, 0 };      int n = sizeof(a)/sizeof(a[0]);      int m = sizeof(b)/sizeof(b[1]);      findMissing(a, b, n, m);      return 0;  } |

Run on IDE

Output:

5 6

Time complexity : O(n)  
Auxiliary Space : O(n)

This article is contributed by **DANISH\_RAZA**. If you like GeeksforGeeks and would like to contribute, you can also write an article using [contribute.geeksforgeeks.org](http://www.contribute.geeksforgeeks.org/)or mail your article to contribute@geeksforgeeks.org. See your article appearing on the GeeksforGeeks main page and help other Geeks.

Please write comments if you find anything incorrect, or you want to share more information about the topic discussed above.