



Dynamic search results

by [ahmed_fathy_aly](#)

Problem

Submissions

Leaderboard

Discussions

To help empower people to experience the world, we need to make sure they can find what they're looking for when they search our site. And what people look for changes – they might look for a pool and then decide it's not important, or change how much they're willing to pay per night. They might add a filter...who knows? Can you help us make sure our search results are relevant?

The search results list consists of a number of unique positive integers (these are property IDs). Initially there's a list (L1) of length n , and we want to transform it to a second list (L2) of length m . There might be some properties which are on both lists (maybe in different positions) and other properties that are on one list but not on the other.

There are 3 possible moves to do the transformation:

- Add one element to any position in the list.
- Remove one element from the list.
- Move an existing element of the list from one position to another.

We need you to output the minimum number of transformations needed to go from L1 to L2.

Input Format

The first line has two integers n and m representing the sizes of the old and new lists respectively. The second line has n elements, the contents of the first list. The third line has m elements, the contents of the second list.

Constraints

- $0 \leq n, m \leq 200000$ where n and m are the sizes of the two given lists $I1$ and $I2$ respectively
- $0 \leq A_i \leq 100000000$ where A_i is an element in either one of the lists

Output Format

One number, the minimum number of transformations needed to transform $I1$ to $I2$

Sample Input 0

```
6 6
4 2 42 3 5 1
43 1 2 3 4 5
```

Sample Output 0

```
4
```

Explanation 0

- 4 2 42 3 5 1 (initial list).
- 4 1 2 42 3 5 (move element 1 to the position 9).
- 43 4 1 2 42 3 5 (add element 43 to the position 9).

- 43 4 1 2 3 5 (remove element 42 to the position 9).
- 43 1 2 3 4 5 (move element 4 to position 4).

Sample Input 1

```
5 4
1 2 3 4 5
5 4 3 2
```

Sample Output 1

```
4
```

Explanation 1

- 1 2 3 4 5 (initial list).
- 5 1 2 3 4 (move element 5 to position 0).
- 5 4 1 2 3 (move element 4 to position 1).
- 5 4 3 1 2 (move element 3 to position 2).
- 5 4 3 2 (remove element 1).

f t in

Contest ends in a day

Submissions: 100



Max Score: 100



Difficulty: Medium

Rate This Challenge:

☆☆☆☆☆

[More](#)

Current Buffer (saved locally, editable)  

Java 8  

```
1 import java.io.*;
2 import java.math.*;
3 import java.security.*;
4 import java.text.*;
5 import java.util.*;
6 import java.util.concurrent.*;
7 import java.util.function.*;
8 import java.util.regex.*;
9 import java.util.stream.*;
10 import static java.util.stream.Collectors.joining;
11 import static java.util.stream.Collectors.toList;
12
13 public class Solution {
14
15     // Complete the solve function below.
16     static int solve(List<Integer> list1, List<Integer> list2) {
17
18     }
19
20
21     public static void main(String[] args) throws IOException {
22         BufferedReader bufferedReader = new BufferedReader(new InputStreamReader(System.in));
23         BufferedWriter bufferedWriter = new BufferedWriter(new FileWriter(System.getenv("OUTPUT_PATH")));
24
25         String[] nm = bufferedReader.readLine().replaceAll("\\s+$", "").split(" ");
26
27         int n = Integer.parseInt(nm[0]);
28
29         int m = Integer.parseInt(nm[1]);
30
31         List<Integer> list1 = Stream.of(bufferedReader.readLine().replaceAll("\\s+$", "").split(" "))
32             .map(Integer::parseInt)
33             .collect(toList());
34
35         List<Integer> list2 = Stream.of(bufferedReader.readLine().replaceAll("\\s+$", "").split(" "))
36             .map(Integer::parseInt)
37             .collect(toList());
```