Strings: Making Anagrams

by amititkgp

Check out the resources on the page's right side to learn more about strings. The video tutorial is by Gayle Laakmann McDowell, author of the best-selling interview book Cracking the Coding Interview.

Alice is taking a cryptography class and finding *anagrams* to be very useful. We consider two strings to be anagrams of each other if the first string's letters can be rearranged to form the second string. In other words, both strings must contain the same exact letters in the same exact frequency For example, `bacdc` and `dcbac` are anagrams, but `bacdc` and `dcbad` are not.

Alice decides on an encryption scheme involving two large strings where encryption is dependent on the minimum number of character deletions required to make the two strings anagrams. Can you help her find this number?

Given two strings, a and b, that may or may not be of the same length, determine the minimum number of character deletions required to make a and b anagrams. Any characters can be deleted from either of the strings.

This challenge is also available in the following translations:

- Chinese
- Russian

Input Format

The first line contains a single string, a.
The second line contains a single string, b.

Constraints

- $1 \leq |a|, |b| \leq 10^4$
- It is guaranteed that a and b consist of lowercase English alphabetic letters (i.e., a through z).

Output Format

Print a single integer denoting the number of characters you must delete to make the two strings anagrams of each other.

Sample Input

```
cde
abc
```

Sample Output

```
4
```

Explanation

We delete the following characters from our two strings to turn them into anagrams of each other:

1. Remove `d` and `e` from `cde` to get `c`.

2. Remove `a` and `b` from `abc` to get `c`.

We must delete **4** characters to make both strings anagrams, so we print **4** on a new line.

Submissions: 26029
Max Score: 25
Difficulty: Easy

Rate This Challenge:

Need Help?

2:09

Strings: An Overview

More

Current Buffer (saved locally, editable) Java 8

```
7  import java.io.*;
8  import java.util.*;
9  import java.text.*;
10 import java.math.*;
11 import java.util.regex.*;
12 public class Solution {
13
14     public static int numberNeeded(String a, String b) {
15
16         int[] freq = new int[26];
17         int result = 0;
18
19         //fill the freq of first string to freq[]
20         for (int i=0; i<a.length(); ++i) {
21             freq[a.charAt(i) - 'a']++;
22         }
23
24         //deduct the common chars from second string and first
25         for (int i=0; i<b.length(); ++i) {
26             freq[b.charAt(i) - 'a']--;
27         }
28
29         //remaining add up as they are to be deleted uncommon ones
30         for (int iter: freq)
31             result += Math.abs(iter);
32
33         return result;
34     }
35     public static void main(String[] args) {
36         Scanner in = new Scanner(System.in);
37         String a = in.next();
38         String b = in.next();
39
40         System.out.println(numberNeeded(a, b));
41     }
42 }
43
```

Line: 43 Col: 1

Upload Code as File Test against custom input Run Code Submit Code

Congrats, you solved this challenge!

Success
Download

✔ Test Case #0	✔ Test Case #1	✔ Test Case #2
✔ Test Case #3	✔ Test Case #4	✔ Test Case #5
✔ Test Case #6	✔ Test Case #7	✔ Test Case #8
✔ Test Case #9	✔ Test Case #10	✔ Test Case #11
✔ Test Case #12	✔ Test Case #13	✔ Test Case #14

Next Challenge