# **Count common characters in two strings**

Given two strings **s1** and **s2** consisting of lowercase English alphabets, the task is to count all the pairs of indices **(i, j)** from the given strings such that **s1[i] = s2[j]** and all the indices are distinct i.e. if **s1[i]** pairs with some **s2[j]** then these two characters will not be paired with any other character.

**Example**

***Input:*** *s1 = “abcd”, s2 = “aad”*

***Output:*** *2*

*(s1[0], s2[0]) and (s1[3], s2[2]) are the only valid pairs.*

*(s1[0], s2[1]) is not includes because s1[0] has already been paired with s2[0]*

***Input:*** *s1 = “geeksforgeeks”, s2 = “platformforgeeks”*

***Output:*** *8*

**Approach:** Count the frequencies of all the characters from both the strings. Now, for every character if the frequency of this character in string **s1** is **freq1** and in string **s2** is **freq2** then total valid pairs with this character will be **min(freq1, freq2)**. The sum of this value for all the characters is the required answer.

// Java implementation of the approach

import java.util.\*;

class GFG

{

// Function to return the count of

// valid indices pairs

static int countPairs(String s1, int n1,

String s2, int n2)

{

// To store the frequencies of characters

// of string s1 and s2

int []freq1 = new int[26];

int []freq2 = new int[26];

Arrays.fill(freq1, 0);

Arrays.fill(freq2, 0);

// To store the count of valid pairs

int i, count = 0;

// Update the frequencies of

// the characters of string s1

for (i = 0; i < n1; i++)

freq1[s1.charAt(i) - 'a']++;

// Update the frequencies of

// the characters of string s2

for (i = 0; i < n2; i++)

freq2[s2.charAt(i) - 'a']++;

// Find the count of valid pairs

for (i = 0; i < 26; i++)

count += (Math.min(freq1[i], freq2[i]));

return count;

}

// Driver code

public static void main(String args[])

{

String s1 = "geeksforgeeks", s2 = "platformforgeeks";

int n1 = s1.length(), n2 = s2.length();

System.out.println(countPairs(s1, n1, s2, n2));

}

}