Determines if leftString is a permutation (rearrangement of characters) of therightString. If so, return true; otherwise returnfalse.

**Example**

For leftString = "abc" and rightString = "bca", the output should be  
IsPermutationOf(leftString, rightString) = true.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] string leftString**

The left string to be checked.

* **[input] string rightString**

The right string to be checked.

* **[output] boolean**

Returns true if leftString is a permutation of rightString (and, of course, vice versa) and false otherwise.

<https://codefights.com/challenge/SKpuysh7LNmHcRTax/main>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static bool IsPermutationOf(string leftString, string rightString)

{

if (leftString.Length != rightString.Length)

{

return false;

}

Dictionary<char, int> dl = new Dictionary<char, int>();

foreach (char ch in leftString)

{

if (dl.ContainsKey(ch))

{

dl[ch]++;

}

else

{

dl[ch] = 1;

}

}

Dictionary<char, int> dr = new Dictionary<char, int>();

foreach (char ch in rightString)

{

if (dr.ContainsKey(ch))

{

dr[ch]++;

}

else

{

dr[ch] = 1;

}

}

foreach (KeyValuePair<char, int> kvp in dl)

{

if (!dr.ContainsKey(kvp.Key))

{

return false;

}

else

{

if (kvp.Value != dr[kvp.Key])

{

return false;

}

}

}

return true;

}

static void Main(string[] args)

{

Console.WriteLine(IsPermutationOf("abc", "bcae"));

Console.ReadLine();

}

}

}