**Shil and Palindrome**

Attempted by: **332**

/

Accuracy: **87%**

/

Maximum Score: **20**

/

6 Votes

Tag(s):

Basic Programming, Easy

**PROBLEM**

**EDITORIAL**

**MY SUBMISSIONS**

**ANALYTICS**

Shil is your new boss and he likes palindromes very much. Palindrome is a string that can be read the same way in either direction, from the left to the right and from the right to the left. (ex. madam , aabaa, racecar)

Given a string **S** , beautiful Palindrome is a lexicographical minimum palindrome that can be formed by rearranging all the characters of string **S**. In order to please your boss, find a beautiful Palindrome that can be formed with help of string **S**.

String **x** is lexicographically less than string **y**, if either **x** is a prefix of **y** (and **x** ≠ **y**), or there exists such **i** (1 ≤ **i** ≤ min(**|x|, |y|)**), that **xi < yi**, and for any **j** (1 ≤ **j** < **i**) **xj = yj**. Here **|a|** denotes the length of the string **a**. The lexicographic comparison of strings is implemented by operator < in modern programming languages​​.

**Input:**  
Only line of input contains string **S**. All the letters of this string will be in lower letters(**'a' - 'z'**).

**Output:**  
Output lexicographical minimum Palindrome that can be formed by rearranging all the letters of string **S**. If no such Palindrome exist for given input, print **-1**.

**Constraints:**   
1≤|S|≤100000

**SAMPLE INPUT**

aabcc

**SAMPLE OUTPUT**

acbca

**Explanation**

After rearranging all the characters , all the palindromes that can be formed are cabac and acbca. Out of these two lexicographical minimum one is acbca.

**Time Limit:**3.0 sec(s) for each input file.

**Memory Limit:**256 MB

**Source Limit:**1024 KB

**Marking Scheme:**Marks are awarded when all the testcases pass.

**Allowed Languages:**C, C++, Clojure, C#, D, Erlang, F#, Go, Groovy, Haskell, Java, Java 8, JavaScript(Rhino), JavaScript(Node.js), Lisp, Lisp (SBCL), Lua, Objective-C, OCaml, Octave, Pascal, Perl, PHP, Python, Python 3, R(RScript), Racket, Ruby, Rust, Scala, Scala 2.11.8, Swift, Visual Basic

<https://www.hackerearth.com/practice/basic-programming/implementation/basics-of-implementation/practice-problems/algorithm/shil-and-palindrome/>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

//string s = "aabcc";

//string s = "bbbbbbbaaaaaaaann";

//string s = "a";

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

for (char ch = 'a'; ch <= 'z'; ch++) frec[ch] = 0;

for (int i = 0; i < s.Length; i++) frec[s[i]]++;

char[] keys = frec.Keys.ToArray();

int[] values = frec.Values.ToArray();

int valuesImpares = values.Count(e => e % 2 != 0);

// Console.WriteLine(valuesImpares);

if (valuesImpares > 1)

{

Console.WriteLine("-1");

}

else

{

string izq = "", der = "", resto = "";

char unico = ' ';

for (int i = 0; i < keys.Length-1; i++)

{

if (values[i] % 2 == 0)

{

izq += new string(keys[i], values[i] / 2);

der = new string(keys[i], values[i] / 2) + der;

}

else

{

//resto += new string(keys[i], values[i]);

izq += new string(keys[i], (values[i] - 1) / 2);

der = new string(keys[i], (values[i] - 1) / 2) + der;

unico = keys[i];

}

}

if (valuesImpares == 1)

{

resto += new string(keys[keys.Length - 1], values[values.Length - 1] / 2) + unico + new string(keys[keys.Length - 1], values[values.Length - 1] / 2);

}

else if(valuesImpares ==0)

{

resto += new string(keys[keys.Length - 1], values[values.Length - 1] / 2) + new string(keys[keys.Length - 1], values[values.Length - 1] / 2);

}

if (s.Length == 1) Console.WriteLine(s);

else Console.WriteLine(izq + resto + der);

}

Console.ReadLine();

}

}

}

------------SOLUCION MAS LARGA--------------

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

//string s = "aabcc";

//string s = "bbbbbbbaaaaaaaann";

//string s = "a";

Dictionary<char, int> frec = s.ToCharArray().GroupBy(x => x)

.ToDictionary(x => x.Key, x => x.Count());

char[] keys = frec.Keys.ToArray();

int[] values = frec.Values.ToArray();

Array.Sort(keys, values);

//for (int i = 0; i < keys.Length; i++)

//{

// Console.WriteLine(keys[i] + " " + values[i]);

//}

int valuesImpares = 0;

for (int i = 0; i < values.Length; i++)

{

if (values[i] % 2 != 0)

{

valuesImpares++;

}

}

if (valuesImpares > 1)

{

Console.WriteLine("-1");

}

else

{

string izq = "", der = "", resto ="";

if (valuesImpares == 1)

{

char unico =' ';

for (int i = 0; i < keys.Length-1; i++)

{

if (values[i] % 2 == 0)

{

izq += new string(keys[i], values[i] / 2);

der = new string(keys[i], values[i] / 2) + der;

}

else

{

//resto += new string(keys[i], values[i]);

izq += new string(keys[i], (values[i] - 1) / 2);

der = new string(keys[i], (values[i] - 1) / 2) + der;

unico = keys[i];

}

}

resto += new string(keys[keys.Length - 1], values[values.Length - 1] / 2) + unico + new string(keys[keys.Length - 1], values[values.Length - 1] / 2);

}

else if(valuesImpares ==0)

{

for (int i = 0; i < keys.Length; i++)

{

if (values[i] % 2 == 0)

{

izq += new string(keys[i], values[i] / 2);

der = new string(keys[i], values[i] / 2) + der;

}

}

}

if (s.Length == 1) Console.WriteLine(s);

else Console.WriteLine(izq + resto+ der);

}

Console.ReadLine();

}

}

}