URI Online Judge | 1276

**Letter Range**

By TopCoder\* https://urionlinejudge.r.worldssl.net/gallery/images/flags/us.gif USA

**Timelimit: 1**

A letter range is a set of alphabetically consecutive letters taken from the lowercase alphabetic characters 'a' through 'z'. The lowest and highest letters of the range, separated by a colon (the character ':'), are used to represent a letter range. For example, the range "a:c" represents the consecutive letters 'a', 'b', and 'c'. (quotes are not part of the range). The range "w:z" represents the letters 'w', 'x', 'y', and 'z'. The range "m:m" respresents the single letter 'm'.

**Input**

Each line of input is one test case. Every test case contains one string, that can be empty and which letters may not be in alphabetical order, of lowercase letters (a-z) and spaces.

The string will contain between 0 and 50 characters, inclusive.

**Output**

For each test case in the input, your program must print the letter ranges ordered alphabetically by the low value of each range. Letter ranges contained in the result must represent the largest possible sequences of increasing consecutive letters found in the input text. Ignore space characters and duplicate letters contained in the input.

If the string is empty just print a new line.

For example, the string "fb xee ac" has three letter ranges, "a:c" (the letters a, b, and c), "e:f" (the letters e and f) and "x:x" (the letter x). Please refer to the examples.

| **Sample Input** | **Sample Output** |
| --- | --- |
| aha xyzzy quick brown fox jumps over the lazy dog fbxeac   x   b z a x c y bdfhjlnprtvxz  az def | a:a, h:h x:z a:z a:c, e:f, x:x x:x a:c, x:z b:b, d:d, f:f, h:h, j:j, l:l, n:n, p:p, r:r, t:t, v:v, x:x, z:z  a:a, d:f, z:z |

<https://www.urionlinejudge.com.br/judge/es/problems/view/1276>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication2

{

class Program

{

static void Main(string[] args)

{

while (true)

{

string s = Console.ReadLine();

if (s == null)

{

break;

}

if (s == string.Empty )

{

Console.WriteLine();

continue;

}

SortedSet<char> hs = new SortedSet<char>();

int i;

for (i = 0; i < s.Length; i++)

{

if (s[i] != ' ')

hs.Add(s[i]);

}

string ordenado = "";

foreach (char ch in hs)

{

ordenado += ch;

}

// Console.WriteLine(ordenado);

List<char[]> lista = new List<char[]>();

i = 1;

char ini = ' ', fin = ' ';

while (i < ordenado.Length)

{

ini = ordenado[i - 1];

while (i < ordenado.Length && ordenado[i - 1] + 1 == ordenado[i])

{

i++;

}

fin = ordenado[i - 1];

//Console.Write(ini + ":" + fin + ", ");

lista.Add(new char[] { ini, fin });

i++;

}

if (i - 1 >= 0 && i - 1 < ordenado.Length &&

fin != ordenado[i - 1])

{

//Console.WriteLine(ordenado[i-1] + ":" + ordenado[i-1]);

lista.Add(new char[] { ordenado[i - 1], ordenado[i - 1] });

}

// Console.WriteLine(i + " " + ordenado.Length);

for (i = 0; i < lista.Count - 1; i++)

{

Console.Write(lista[i][0] + ":" + lista[i][1] + ", ");

}

Console.Write(lista[i][0] + ":" + lista[i][1]);

Console.WriteLine();

}

Console.ReadLine();

}

}

}