You are given a string containing only lowercase English letters. Find its smallest substring that has only one occurence in the string. If there are multiple such strings return the lexicographically smallest one.

For example, for "cababc" the answer is "ba", and for "aaa" the answer is "aaa".

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

string containing only lowercase English letters

* **[output] string**

unique substring

<https://codefights.com/challenge/6b99hgaXfJXXhQY3m/main>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

public static int Count (string text, string pattern)

{

int cont = 0;

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

{

if (text.Substring(i, pattern.Length) == pattern)

{

cont++;

}

}

return cont;

}

static string uniqueSubstring(string s)

{

string menor = s;

List<string> lista = new List<string>();

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

{

for (int j = i; j < s.Length; j++)

{

string subs = s.Substring(i, j - i + 1);

// Console.WriteLine(subs);

int cont = Count(s, subs);

if (cont == 1)

{

if (subs.Length < menor.Length)

{

menor = subs;

lista = new List<string>();

lista.Add(subs);

}

else if(subs.Length == menor.Length)

{

lista.Add(subs);

}

}

}

}

lista.Sort();

return lista[0];

}

static void Main(string[] args)

{

//string s = "xyxyxyxyxyxyxyxyxyyxyxyxyxyxyxyxyxxxyxyxyxy";

//Console.WriteLine(Count(s, "xx"));

//uniqueSubstring(s);

string s = "cababc";

Console.WriteLine(uniqueSubstring(s));

Console.WriteLine(Count(s, "c"));

Console.ReadLine();

}

}

}

…………………………………………………………SOLUCION POR ALEX……………………………………………………………………………..

String uniqueSubstring(String s) {

int n = s.length(), i = -1, j;

SortedSet<String> b = new TreeSet<String>(), a = new TreeSet<String>();

while(++i < n){

for(j = i; j < n; j++){

String x = s.substring(j - i, j + 1);

if(b.contains(x)) a.remove(x);

else a.add(x);

b.add(x);

}

if(a.size() > 0) break;

}

return a.first();

}

S uniqueSubstring(S s) {

M m;

for (int i = 0, j, n = s.size(); i++ < n;) {

for (j = -1; j + i < n;)

++m[s.substr(++j, i)];

for (M::iterator a = m.begin(); a++ != m.end();)

if (a->second == 1)

return a->first;

}

}