Member Count: 677,138 - September 4, 2014 [Get Time]

[TopCoder]

Hello, faijur | Logout

My TopCoder

Competitions

Overview

Copilot Opportunities

Design

Development

UI Development

QA and Maintenance

CloudSpokes

Algorithm

High School

The Digital Run

Submit & Review

TopCoder Networks

Events

Statistics

Tutorials

Forums

Surveys

My TopCoder

Help Center

About TopCoder

UML TOOL

Member Search:

Handle: Go

Advanced Search

Dashboard > TopCoder Competitions > ... > Algorithm Problem Set Analysis > SRM 618

Search

TopCoder Competitions

SRM 618

View | Attachments (18) | Info | Browse Space

Added by [[rng_58]] , last edited by vexorian on May 01, 2014 (view change)
Labels: (None) EDIT

Single Round Match 618

Thursday, April 24th, 2014

Archive
Match Overview
Discuss this match

Match summary

The Problems

WritingWords | LongWordsDiv2 | MovingRooksDiv2 | Family | LongWordsDiv1 | MovingRooksDiv1

LongWordsDiv2

Rate It | Discuss it

Used as: Division Two - Level Two:

Value	500
Submission Rate	623 / 744 (83.74%)
Success Rate	407 / 623 (65.33%)
High Score	**waled_top** for 499.87 points (0 mins 27 secs)
Average Score	366.09 (for 407 correct submissions)

All the bad subsequences

This is primarily an implementation problem. It is easy to tell if the string has consecutive equal characters. The input constraints guarantee it will use only upper case letters. The main issue is to make sure thee are never any subsequence of the kind XYXY.

Something useful: There are 26 upper case letters, this means that there are $26 \cdot 26 = 676$ pairs (X, Y) possible. We can just generate all 676 such pairs. For each of those pairs, we can generate string **XYXY**. We can check if this string **XYXY** is a subsequence of the input string. If this is true for any of the generated **XYXY** strings, then the string is disliked.

Check if a string is a subsequence of another string

This is a classical problem. For example, when checking if string **b** = "ABAB" is a substring of **a** = "XYAXBCCCATTBC" , we can do the following: Since "A" is the first letter of the subsequence, we find the earliest position i of **a** that has this letter and match it with "A", the remaining matchings need to use larger indexes. We proceed to find the earliest position j that contains a "B" after the position i, and so and so. If we are able to match all positions of the candidate subsequence, then it is a subsequence.

Code

```
// is string a a subsequence of string b?
bool subSequence(string a, string b)
{
    int i = 0;
    int j = 0;
    while ( (i < b.size()) && (j < a.size() ) ) {
        if (a[j] == b[i]) {
            j ++;
        }
        i ++;
    }
    return (j == a.size() );
}

string find(string word)
{
    // Any XYXY which is a subsequence?
    for (char X = 'A'; X <= 'Z'; X++) {
        for (char Y = 'A'; Y <= 'Z'; Y++) {
            string xyxy = string(1,X)+string(1,Y)+string(1,X)+string(1,Y);
            if (subSequence(xyxy, word)) {
                return "Dislikes";
            }
        }
    }
    // any two equal consecutive characters?
    for (int i = word.size() - 1; i > 0; i--) {
        if (word[i] == word[i-1]) {
            return "Dislikes";
        }
    }
    return "Likes";
}
```

```
import string

class LongWordsDiv2:
 def find(self, word):
    def isSubsequence(a, b):
        i = 0
        j = 0
        while i < len(b) and j < len(a):
            if a[j] == b[i]:
                j += 1
            i += 1
        return j == len(a)

    for i in range(len(word)-1):
        if word[i] == word[i+1]:
            return "Dislikes"

    for X in string.uppercase:
        for Y in string.uppercase:
            if isSubsequence(X+Y+X+Y, word):
                return "Dislikes"
```

Alternative solutions and additional comments.

<Place your comments here>

Next problem: MovingRooksDiv2

Author

By **vexorian**

TopCoder Member

Editorial feedback	Choose
I liked it.	✓
I didn't like it.	✓

Comments (Hide Comments)

In Java,
You do not need to convert between the character and its numeric representation.
Java will use the Unicode representation in case of arithmetic operations.
So, the following line will work also in Java:
sum += ch - 'A' + 1;

Posted by alhussain at May 04, 2014 17:57 | Reply To This

In the C++ solution of the "Family" problem,
<code>int color [50];</code> should be
<code>int color [100];</code>

as it mentioned in the problem statement,
<em>parent1 will contain between 1 and 100 elements, inclusive.</em> and
<em>For each i, the i-th element (0-based) of parent1 will be between -1 and i-1 inclusive</em>

Please correct me, If I misunderstood any point. Thanks!

Posted by tausiq at May 10, 2014 04:27 | Reply To This

Yes.

Posted by vexorian at May 10, 2014 08:58 | Reply To This

Add Comment

Home | About TopCoder | Press Room | Contact Us | Privacy | Terms
Developer Center | Corporate Services