**Doubly Strings**

Jan2022-Medium-Q2-Doubly Strings <Dynamic Programming> <Nachiket Kanore>

**1. Topic** Dynamic Programming

**2. Difficulty Level :** Medium

**3. Question / Problem Statement**

A string S is called a *square* if it has **even length** and it’s **first half and last half are same**.

For example, the strings "", aabaab" and "xxxx" are squares, but "x", "aabb" and "xxyyxx" are not.   
  
You are given a String **s**. Find the longest square string that can be found from **s** by erasing some (possibly none, possibly all) of its characters.

Which is same as the longest square that occurs in **s** as a subsequence. Find the maximum length of any such square.   
  
Note: “” empty string is also square. Hence, answer >= 0 always exists.

**Note**

**s** will contain between 1 and 50 characters, inclusive.

Each character in s will be a lowercase English letter ('a'-'z').

**Function Description**  
In the provided code snippet, implement the provided **doublyStrings(...)** method using the variables to print the pattern for each line from 1 to i. You can write your code in the space below the phrase **“WRITE YOUR LOGIC HERE”**.   
  
There will be multiple test cases running so the Input and Output should match exactly as provided.  
The base Output variable **result** is set to a default value of **-404** which can be modified. Additionally, you can add or remove these output variables.

**Input Format**

Single line containing string **s**

**Sample Input**

*singing*

**Constraints**

1 <= len(s) <= 50

‘a’ <= s[i] <= ‘z’  
  
**Output Format**  
Integer answer  
  
**Sample Output**

6

**Explanation**  
***Required subsequence is “inging” which consists of two equal halves “ing” + “ing”***

**8. Solution Steps:**

[Basic Steps: Pseudo algorithm and Pseudo flow that will allow us to write code in most of the Languages.]

1. Break string **s** into two parts in all possible ways
2. There are O(N) such ways
3. For each such way, we have 2 strings to find first half of the **square** from the left part and second half of the **square** from the right part
4. Best square will be longest common subsequence between them
5. Answer will be maximum of 2 \* LCS(A, B) over all parts (A, B) of string s,

such that A + B = s

**9. Running Solution in C, C++ or Java**

[Running code that will run accurately on our System or Profoundly used IDE.]

#include <bits/stdc++.h>

using namespace std;

int LCS(string X, string Y) {

if (Y.length() > X.length())

swap(X, Y);

int m = X.length(), n = Y.length();

vector<vector<int>> c(2, vector<int>(n + 1, 0));

int i, j;

for (i = 1; i <= m; i++) {

for (j = 1; j <= n; j++) {

if (X[i - 1] == Y[j - 1])

c[1][j] = c[0][j - 1] + 1;

else

c[1][j] = max(c[1][j - 1], c[0][j]);

}

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

c[0][j] = c[1][j];

}

return (c[1][n]);

}

int doublyStrings(string s) {

string st = s;

int ans = 0;

for (int cp = 1; cp < st.size(); cp++) {

string st1, st2;

for (int i = 0; i < st.size(); i++) {

if (i < cp)

st1 += st[i];

else

st2 += st[i];

}

ans = max(ans, LCS(st1, st2));

}

return ans \* 2;

}

int main() {

cout << doublyStrings("singing") << '\n';

}

Input:

singing

Output:

6

**10. Test Cases [ Qty: 12 ]**

[ Test Cases: Minimum 12 which are composed of Easy, Complex, Negative and Boundary Value Cases (BVC). We should be able to insert these in Our System. Realistic Values ]

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case No** | **Input** | **Output** | **Score** |
| 1 | frankfurt | 4 | 0 |
| 2 | single | 0 | 0 |
| 3 | singing | 6 | 1 |
| 4 | aababbababbabbbbabbabb | 18 | 1 |
| 5 | x | 0 | 1 |
| 6 | aaaabbababcbbabbabbbcabcbabcabcb | 24 | 1 |
| 7 | aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa | 50 | 1 |
| 8 | aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaabaaaaaaaaaaaa | 48 | 1 |
| 9 | fjkjsakljflkjakljjfiwoqjfioqwfoiqwjfiojq | 16 | 1 |
| 10 | aadgpoachigi | 4 | 1 |
| 11 | anjcubddgdgkegaabjmafkheilekfeda | 10 | 1 |
| 12 | fjkomincdlqklcaeblckidiadiaynndedeaboffgfbdi | 14 | 1 |

**Format of Coding Question**

**Nomenclature: Question ID:** Jun2019-Easy-Q1-Multiple Graphs <specimen name>

**1. Topic** and relation of the question to some theory (This is what we will be introducing from now on.)

**2. Difficulty Level :** (Easy, Medium, High) as per the depth of Problem statement and solution difficulty.

**3. Question / Problem Statement**

[ Here the Question should be a short and sweet story that can sufficiently explain problem to be solved. This should not be a clumsy story. It should be simply understandable. Candidate should be able to Identify Topic by reading ]

**Note (if any):**

**4. Function Description:**

[ Here we provide the function name in **camelCase(...).** Here we define what the function will return or print. ]

**5. Input Format:**

[ Here the Candidate will be explained what is acceptable in the Input and how one should accept Input ]

**6. Sample Input:**

[ Here the Candidate will be provided with a running Sample Input ]

**Constraints (if any):**

**7. Output Format:**

[ Here the Candidate will be explained what is need to be as Output and how one should present Output ]

**8. Sample Output:**

[ Here the Candidate will be provided with a running Sample Output ]

**9. Explanation** of Sample Input and Output:

[ Here the Candidate will be guided and explained with how Sample Input and Output has arrived to what is shown. ]

**10. Solution Steps:**

[Basic Steps: Pseudo algorithm and Pseudo flow that will allow us to write code in most of the Languages.]

**11. Running Solution in C, C++ or Java**

[Running code that will run accurately on our System or Profoundly used IDE.]

**12. Test Cases [ Qty: 12 ]**

[ Test Cases: Minimum 12 which are composed of Easy, Complex, Negative and Boundary Value Cases (BVC). We should be able to insert these in Our System. Realistic Values ]

**13. Code stubs (if any):**  
[ Each code stub will have name as per function description ]

**Requirements:**

1. Come up with Better and easier Question Format. Less Story, hinting to closeness of answer. Story should be established with least or no mathematics. Story cannot be more than 2 sentences.

2. Coding Question should become a fun element yet should be able to judge deep serious intent of programming capacity of the Candidate and provide apparatus (How we can better calculate Candidate's Coding and Programming Skills... Use of Data Structures and ability to write optimum solution.)

4. Question should be EEOC and any discriminatory policy compliant:

eg.

a. Cricket is not common globally... Questions cannot be related to unfamiliar subject and topics.

b. Harmful Words: Bomb, Drugs... so on... Are not allowed.

c. Male and female OR Female and Male... Superiority and Inferiority Discrimination not to be a part of this.

5. Inspiration: Questions can be like:

***Easy***

Write a code to find Fibonacci series until 5th number. (It should be as simple as these)

***Situational Difficult Questions:***

Write a Code that can find 5 star rated songs from a set of Playlists which are repeated in each Playlist.

***New and Advanced Difficulty Questions.***

Best topics:

1. Algorithms:

<https://www.geeksforgeeks.org/fundamentals-of-algorithms/>

2. Advanced Data Structures:

[https://www.geeksforgeeks.org/data-structures/#AdvancedDataStructure](https://www.geeksforgeeks.org/data-structures/" \l "AdvancedDataStructure)

3. Latest Trends:

a. Find data from Spatial Matrix.

b. Solvable Analytics.

c. Solvable Big Data.

d. Solvable Smart Algos.

4. Data Science:

<https://www.testdome.com/tests/data-science-test/65>

<https://www.geeksforgeeks.org/tag/data-science/>