codestudio
POWERED BY CODING NINJAS

Practice ▾   Guided Paths 🔥   Interview Prep ▾   Challenges ▾   New   Knowledge Centre ▾   New   Co

Is SubSequence

Posted: 18 Nov, 2020
Difficulty: Easy

🏢

PROBLEM STATEMENT   Try Problem

You have been given two strings 'STR1' and 'STR2'.

Your task is to find if 'STR1' is a subsequence of 'STR2'.

A subsequence of a string is a new string that can be derived from the original string by deleting some characters (can be none) without changing the relative ordering of other characters.

Example:

```
'ACE' is a subsequence of 'ABCDE' because 'ACE' can be formed by deleting 'B' and 'D' without changing
the relative order of characters. 'ADB' is not a subsequence of 'ABCDE' because we can get 'ABD' from
'ABCDE' but not 'ADB' and in 'ADB' relative order of 'B' and 'D' are different from original strings.
```

Note:

```
1.Strings 'STR1' and 'STR2' consists only of English uppercases.

2.Length of string 'STR2' will always be greater than or equal to the length of string 'STR1'.
```

Example:

For example, the given 'STR1' is 'BAE' and 'STR2' is 'ABADE'.
String 'STR1' is a subsequence of string 'STR2' because 'BAE' can be formed by deleting 'A' and 'D' from 'ABADE' and the relative ordering of the characters of the string 'ABADE' persists.

Input Format:

The first line of input contains an integer 'T' denoting the number of test cases.
The next '2*T' lines represent the 'T' test cases.

The first line of each test case contains the string 'STR1' on a separate line denoting the subsequence that we need to find in 'STR2' and 'N' is the length of 'STR1'.

The second line of each test case contains the string 'STR2' on a separate line denoting the string in which we need to find the subsequence and 'M' is the length of 'STR2'.

Output Format

For each test case, print a string 'True' if 'STR1' is a subsequence of 'STR2' otherwise print 'False'.

Note:

You are not required to print the output explicitly, it has already been taken care of. Just implement the function.

Constraints:

1 <= T <= 50
1 <= N, M <= 10^4

Where N and M denote the lengths of STR1 and STR2respectively.

Time limit: 1 second

✅ APPROACH 1

Let's understand this approach with an example.

Consider 'STR1' is 'BAE' and 'STR2' is 'ABADE'.
Start with first 'B' in 'STR1' and find 'B' in 'STR2' if found then start with 'A' and find 'A' in 'STR2' if 'A' exist in 'STR2' then check the position of 'A' must be after the last find character 'B' and repeat the same step for all characters of 'STR1'.

- We use two loops, outer loop to iterate through 'STR2' and inner loop to iterate through 'STR1'.

- Consider 'N' is the number of characters in the given strings 'STR1' and 'M' is the number of characters in the given string 'STR2'.

- Keep two variables 'PREV' with an initial value of '-1' to store the last character index in 'STR2' and 'COUNT' variable with an initial value of '0' to store matched characters of 'STR1' in 'STR2'.

- The outer loop runs from 0 to 'N-1' for 'i'

- For each character of 'STR1', we check if it is present in 'STR2' or not.

- For checking, we iterate the inner loop 'j' from '0' to 'M-1' and check if 'STR1[i]' is equal to 'STR2[j]'.

- Check 'j' greater than 'PREV', then update 'PREV' with 'j' and increase 'COUNT' by '1'.

- Then check for next 'i'

- Finally, if 'COUNT' is equal to 'N' then return 'true' else return 'false'.

Try Problem

✅ APPROACH 2

Basic idea is to use a queue in which we insert all the characters of 'STR1' and then check whether the top character is

characters that are present in the queue.

- In this, we use the queue (first in first out) algorithm and call 'QUE' to our queue.

- First, we insert all characters of 'STR1' into the queue 'QUE'.

- For each character in 'STR2' we check if it is the same as the top character of 'QUE'.

- If it is the same then we pop the top character of 'QUE' and check for the next top character and repeat the same step.

- In the end, if 'QUE' is empty then return 'True' else return 'False' because if 'QUE' is empty then all characters of 'QUE' and 'STR1' are present in 'STR2' in the same order.

Try Problem

✅ APPROACH 3   ⌃

Let's understand this approach with an example.

Consider 'STR1' is 'BAE' and 'STR2' is 'ABADE'
Iterate 'STR2' and check for characters of 'STR1' if any point time character of 'STR1' and 'STR2' are the same then check for the next character of 'STR1'.

- This is the optimized solution of approach1.

- Consider 'N' is the number of characters in the given strings 'STR1' and 'M' is the number of characters in the given string 'STR2'.

- Keep one variable 'IDX' with initial value '0' to point to the matched character of 'STR1' and 'STR2'.

- We use a loop 'i' from '0' to 'M-1' and iterate 'STR2'.

- If any point of time 'STR2[i]' is equal to 'STR1[idx]' then increase 'IDX' by '1'

Practice ▾   Guided Paths 🔥   Interview Prep ▾   Challenges ▾   New   Knowledge Centre ▾   New   Cc

- If 'IDX' is less than 'N' then return 'false' because some characters of 'STR1' are missing in 'STR2'.

Try Problem

SIMILAR PROBLEMS

Min Stack

Posted: 22 Jan, 2022
Difficulty: Easy

Raju and his string

Posted: 9 Feb, 2022
Difficulty: Easy

Check Palindrome

Posted: 23 Feb, 2022
Difficulty: Easy

Most Frequent Element

Posted: 25 Feb, 2022
Difficulty: Easy

Reverse String

Categories: Coding courses for beginners | Web Development Courses | Data Science & Machine Learning Courses | Competitive Programming Course | Android App Development Courses | Courses for interview preparation

Popular Courses: C++ Foundation with Data Structures | Java Foundation with Data Structures | Python Foundation with Data Structures | Competitive Programming | Full Stack Web Development

Career Tracks: Ninja Competitive Programmer Track | Ninja Android Developer Career Track | Ninja Web Developer Career Track - NodeJS & ReactJs | Ninja Web Developer Career Track - NodeJS | Ninja Data Scientist Career Track | Ninja Machine Learning Engineer Career Track

Interested in Coding Ninjas Flagship Courses?   Click Here

CODING NINJAS

About Us

Press

Privacy Policy

Terms & Conditions

Bug Bounty

PRODUCTS

Interview Problems

Interview Experiences

Interview Bundle

Guided Paths

Library

COMMUNITY

CodeStudio

Blog

Events

Campus Ninjas

FOLLOW US ON

▶