Description | Solution | Discuss (334) | Submissions

Java | Autocomplete

1055. Shortest Way to Form String

Medium 744 43 Add to List Share

A **subsequence** of a string is a new string that is formed from the original string by deleting some (can be none) of the characters without disturbing the relative positions of the remaining characters. (i.e., `"ace"` is a subsequence of `"abcde"` while `"aec"` is not).

Given two strings `source` and `target`, return *the minimum number of* ***subsequences*** *of* `source` *such that their concatenation equals* `target`. If the task is impossible, return `-1`.

Example 1:

```
Input: source = "abc", target = "abcbc"
Output: 2
Explanation: The target "abcbc" can be formed by "abc" and "bc", which are
subsequences of source "abc".
```

Example 2:

```
Input: source = "abc", target = "acdbc"
Output: -1
Explanation: The target string cannot be constructed from the subsequences of
source string due to the character "d" in target string.
```

Example 3:

```
Input: source = "xyz", target = "xzyxz"
Output: 3
Explanation: The target string can be constructed as follows "xz" + "y" + "xz".
```

Constraints:

- `1 <= source.length, target.length <= 1000`
- `source` and `target` consist of lowercase English letters.

Accepted 51,870 | Submissions 89,594

Seen this question in a real interview before? Yes No

Companies

0 ~ 6 months 6 months ~ 1 year 1 year ~ 2 years

Google 3

Related Topics

String Dynamic Programming Greedy

Similar Questions

Is Subsequence	Easy
Number of Matching Subsequences	Medium

Hide Hint 1

Which conditions have to been met in order to be impossible to form the target string?

Hide Hint 2

If there exists a character in the target string which doesn't exist in the source string then it will be impossible to form the target string.

Hide Hint 3

Assuming we are in the case which is possible to form the target string, how can we assure the minimum number of used subsequences of source?

Hide Hint 4

For each used subsequence try to match the leftmost character of the current subsequence with the leftmost character of the target string, if they match then erase both character otherwise erase just the subsequence character whenever the current subsequence gets empty, reset it to a new copy of subsequence and increment the count, do this until the target sequence gets empty. Finally return the count.

```java
class Solution {
    public int shortestWay(String source, String target) {

    }
}
```