411. Minimum Unique Word Abbreviation Premium

Hard | Topics | Companies

A string can be **abbreviated** by replacing any number of **non-adjacent** substrings with their lengths. For example, a string such as `"substitution"` could be abbreviated as (but not limited to):

- `"s10n"` (`"s ubstitutio n"`)
- `"sub4u4"` (`"sub stit u tion"`)
- `"12"` (`"substitution"`)
- `"su3i1u2on"` (`"su bst i t u ti on"`)
- `"substitution"` (no substrings replaced)

Note that `"s55n"` (`"s ubsti tutio n"`) is not a valid abbreviation of `"substitution"` because the replaced substrings are adjacent.

The **length** of an abbreviation is the number of letters that were not replaced plus the number of substrings that were replaced. For example, the abbreviation `"s10n"` has a length of `3` (`2` letters + `1` substring) and `"su3i1u2on"` has a length of `9` (`6` letters + `3` substrings).

Given a target string `target` and an array of strings `dictionary`, return *an **abbreviation** of `target` with the **shortest possible length** such that it is **not an abbreviation** of **any** string in `dictionary`. If there are multiple shortest abbreviations, return any of them.*

Example 1:

```
Input: target = "apple", dictionary = ["blade"]
Output: "a4"
Explanation: The shortest abbreviation of "apple" is "5", but this is also an abbreviation of "blade".
The next shortest abbreviations are "a4" and "4e". "4e" is an abbreviation of blade while "a4" is not.
Hence, return "a4".
```

Example 2:

```
Input: target = "apple", dictionary = ["blade","plain","amber"]
Output: "1p3"
Explanation: "5" is an abbreviation of both "apple" but also every word in the dictionary.
"a4" is an abbreviation of "apple" but also "amber".
"4e" is an abbreviation of "apple" but also "blade".
"1p3", "2p2", and "3l1" are the next shortest abbreviations of "apple".
Since none of them are abbreviations of words in the dictionary, returning any of them is correct.
```

Constraints:

- `m == target.length`
- `n == dictionary.length`
- `1 <= m <= 21`
- `0 <= n <= 1000`
- `1 <= dictionary[i].length <= 100`
- $\log_2(n) + m <= 21$ if `n > 0`
- `target` and `dictionary[i]` consist of lowercase English letters.
- `dictionary` does not contain `target`.

Seen this question in a real interview before? 1/5

Yes No

Accepted 15K | Submissions 37.5K | Acceptance Rate 39.9%

Topics

Array String Backtracking Bit Manipulation

Companies

0 - 6 months

Google 2

Similar Questions

Generalized Abbreviation	Medium
Valid Word Abbreviation	Easy
Word Abbreviation	Hard

Discussion (1)

Copyright © 2024 LeetCode All rights reserved