320. Generalized Abbreviation

Medium 646 216 Add to List Share

A word's **generalized abbreviation** can be constructed by taking any number of **non-overlapping** and **non-adjacent** substrings and replacing them with their respective lengths.

- For example, `"abcde"` can be abbreviated into:
  - `"a3e"` (`"bcd"` turned into `"3"`)
  - `"1bcd1"` (`"a"` and `"e"` both turned into `"1"`)
  - `"5"` (`"abcde"` turned into `"5"`)
  - `"abcde"` (no substrings replaced)
- However, these abbreviations are **invalid**:
  - `"23"` (`"ab"` turned into `"2"` and `"cde"` turned into `"3"`) is invalid as the substrings chosen are adjacent.
  - `"22de"` (`"ab"` turned into `"2"` and `"bc"` turned into `"2"`) is invalid as the substring chosen overlap.

Given a string `word` , return *a list of all the possible* ***generalized abbreviations*** *of* `word` . Return the answer in **any order**.

Example 1:

```
Input: word = "word"
Output:
["4","3d","2r1","2rd","1o2","1o1d","1or1","1ord","w3","w2d","w1r1","w1rd","wo2","wo1d","wor1","wor
```

Example 2:

```
Input: word = "a"
Output: ["1","a"]
```

Constraints:

- `1 <= word.length <= 15`
- `word` consists of only lowercase English letters.

Accepted 64,197 Submissions 111,825

Seen this question in a real interview before? Yes No

Companies

Related Topics

Similar Questions

C++ Autocomplete

```cpp
class Solution {
public:
    bool isdigit(char now)
    {
        return (now>='0' and now<='9');
    }
    void allPossibleAns(string word,string str,vector<string> &res,int index)
    {
        if(index == word.length())
        {
            res.push_back(str);
            return;
        }

        allPossibleAns(word,str+word[index],res,index+1);


        if(str.size() == 0 || !(isdigit(str[str.size()-1])) )
        {
            allPossibleAns(word,str+'1',res,index+1);
        }
        else
        {
            if(str[str.size()-1] == '9')
            {
                str.pop_back();
                str += "10";
            }
            else
                str[str.size()-1] += 1;

            allPossibleAns(word,str,res,index+1);
        }


    }
    vector<string> generateAbbreviating word) {
```

NEW

Your previous code was restored from your loc

Testcase Run Code Result Debugger

Accepted Runtime: 20 ms

Your input "frommeraqzasder"

Output ["fromme Diff

Expected ["15","14r","13e1","

Problems Pick One Prev 18/150 Next Run Code Submit