5917. Check if an Original String Exists Given Two Encoded Strings

My Submissions (/contest/weekly-contest-265/problems/check-if-an-original-string-exists-given-two-encoded-strings/submissions/)

Back to Contest (/contest/weekly-contest-265/)

User Accepted:	0
User Tried:	0
Total Accepted:	0
Total Submissions:	0
Difficulty:	Hard

An original string, consisting of lowercase English letters, can be encoded by the following steps:

- Arbitrarily **split** it into a **sequence** of some number of **non-empty** substrings.
- Arbitrarily choose some elements (possibly none) of the sequence, and **replace** each with **its length** (as a numeric string).
- **Concatenate** the sequence as the encoded string.

For example, **one way** to encode an original string `"abcdefghijklmnop"` might be:

- Split it as a sequence: `["ab", "cdefghijklmn", "o", "p"]`.
- Choose the second and third elements to be replaced by their lengths, respectively. The sequence becomes `["ab", "12", "1", "p"]`.
- Concatenate the elements of the sequence to get the encoded string: `"ab121p"`.

Given two encoded strings `s1` and `s2`, consisting of lowercase English letters and digits `1-9` (inclusive), return `true` *if there exists an original string that could be encoded as* ***both*** `s1` *and* `s2`. *Otherwise, return* `false`.

Note: The test cases are generated such that the number of consecutive digits in `s1` and `s2` does not exceed `3`.

Example 1:

```
Input: s1 = "internationalization", s2 = "i18n"
Output: true
Explanation: It is possible that "internationalization" was the original string.
- "internationalization"
  -> Split:       ["internationalization"]
  -> Do not replace any element
  -> Concatenate:  "internationalization", which is s1.
- "internationalization"
  -> Split:       ["i", "nternationalizatio", "n"]
  -> Replace:     ["i", "18",                 "n"]
  -> Concatenate:  "i18n", which is s2
```

Example 2:

```
Input: s1 = "l123e", s2 = "44"
Output: true
Explanation: It is possible that "leetcode" was the original string.
- "leetcode"
  -> Split:      ["l", "e", "et", "cod", "e"]
  -> Replace:    ["l", "1", "2",  "3",   "e"]
  -> Concatenate: "l123e", which is s1.
- "leetcode"
  -> Split:      ["leet", "code"]
  -> Replace:    ["4",    "4"]
  -> Concatenate: "44", which is s2.
```

Example 3:

```
Input: s1 = "a5b", s2 = "c5b"
Output: false
Explanation: It is impossible.
- The original string encoded as s1 must start with the letter 'a'.
- The original string encoded as s2 must start with the letter 'c'.
```

Example 4:

```
Input: s1 = "112s", s2 = "g841"
Output: true
Explanation: It is possible that "gaaaaaaaaaaaa" was the original string
- "gaaaaaaaaaaaa"
  -> Split:      ["g", "aaaaaaaaaaa", "s"]
  -> Replace:    ["1", "12",          "s"]
  -> Concatenate: "112s", which is s1.
- "gaaaaaaaaaaaa"
  -> Split:      ["g", "aaaaaaaa", "aaaa", "s"]
  -> Replace:    ["g", "8",        "4",    "1"]
  -> Concatenate: "g841", which is s2.
```

Example 5:

```
Input: s1 = "ab", s2 = "a2"
Output: false
Explanation: It is impossible.
- The original string encoded as s1 has two letters.
- The original string encoded as s2 has three letters.
```

Constraints:

- `1 <= s1.length, s2.length <= 40`
- `s1` and `s2` consist of digits `1-9` (inclusive), and lowercase English letters only.
- The number of consecutive digits in `s1` and `s2` does not exceed `3`.

JavaScript

```
/**
 * @param {string} s1
 * @param {string} s2
 * @return {boolean}
 */
var possiblyEquals = function(s1, s2) {

};
```

Custom Testcase | Use Example Testcases

Run | Submit