758. Bold Words In String ^C

Jan. 6, 2018 | 7.2K views

Previous Next
 Next
 Average Rating: 3.38 (8 votes)

6 🖸 🛅

Given a set of keywords words and a string S, make all appearances of all keywords in S bold. Any letters between
b> and tags become bold.

The returned string should use the least number of tags possible, and of course the tags should form a valid combination.

For example, given that words = ["ab", "bc"] and S = "aabcd", we should return "aabcd". Note that returning "aab<c/b>cd" would use more tags, so it is incorrect.

Note:

- 1. words has length in range [0, 50].
- 2. words[i] has length in range [1, 10].
- s has length in range [0, 500].
 All characters in words[i] and s are lowercase letters.

Approach #1: Brute Force [Accepted]

Intuition

Let's try to learn which letters end up bold, since the resulting answer will just be the canonical one - we put bold tags around each group of bold letters.

To do this, we'll check for all occurrences of each word and mark the corresponding letters bold.

Algorithm

Let's work on first setting mask[i] = true if and only if the i -th letter is bold. For each starting position i in s, for each word, if s[i] starts with word, we'll set the appropriate letters bold.

Now armed with the correct mask, let's try to output the answer. A letter in position i is the first bold letter of the group if mask[i] && ($i == 0 \mid | \cdot \mid mask[i-1]$), and is the last bold letter if mask[i] && ($i == N-1 \mid | \cdot \mid mask[i+1]$). Alternatively, we could use $i = 1 \mid mask[i+1]$ in Python.

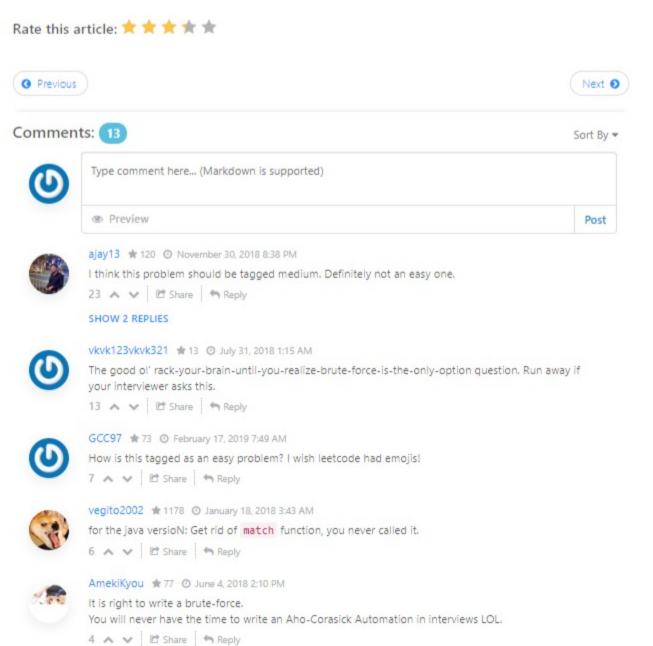
Once we know which letters are the first and last bold letters of a group, we know where to put the "" and "" tags.

```
Copy
Java Python
1 class Solution(object):
      def boldWords(self, words, S):
         N = len(S)
          mask = [False] * N
         for i in xrange(N):
             prefix = S[i:]
             for word in words:
                if prefix.startswith(word):
                    for j in xrange(i, min(i+len(word), N)):
10
                          mask[j] = True
11
12
          ans = []
         for incl, grp in itertools.groupby(zip(S, mask), lambda z: z[1]):
13
           if incl: ans.append("<b>")
            ans.append("".join(z[0] for z in grp))
if incl: ans.append("</b>")
15
16
         return "".join(ans)
```

Complexity Analysis

- Time Complexity: $O(N \sum w_i)$, where N is the length of s and w_i is the sum of w.
- ullet Space Complexity: O(N).

Analysis written by: @awice.



sevenhe716 ★ 70 ② March 2, 2019 3:24 PM optimized python solution using trie tree a

optimized python solution using trie tree and merge intervals:

trie tree is used to speed up string match (faster than find or startwith in large query request).
 Using merge intervals instead of mask to reduce Time and Space Complexity, both from O(n) to O(m), m represets interval numbers after merged.

salamanderrex ★ 32 ② September 17, 2018 3:50 AM
for python

for j in xrange(i, min(i+len(word), N)):
 mask[j] = True
Read More

2 A V Et Share Reply

oddguy ★2 ② December 27, 2018 9:48 PM
In fact you don't have to mask EVERY letter to

SHOW 1 REPLY

leetcodefan # 1503 @ May 3, 2019 11:38 AM



0 A V | E Share | Reply

calvinchankf ★ 2515 ② April 20, 2019 12:05 PM

I spent so much time on finding out a better approach and u now tell me that u actually accept the brute approach ⓐ

0 ✔ 比 Share Reply

