244. Shortest Word Distance II

Medium 691 199 Add to List Share

Design a data structure that will be initialized with a string array, and then it should answer queries of the shortest distance between two different strings from the array.

Implement the `WordDistance` class:

- `WordDistance(String[] wordsDict)` initializes the object with the strings array `wordsDict`.
- `int shortest(String word1, String word2)` returns the shortest distance between `word1` and `word2` in the array `wordsDict`.

Example 1:

```
Input
["WordDistance", "shortest", "shortest"]
[[["practice", "makes", "perfect", "coding", "makes"]], ["coding", "practice"], ["makes", "coding"]]
Output
[null, 3, 1]

Explanation
WordDistance wordDistance = new WordDistance(["practice", "makes", "perfect", "coding", "makes"]);
wordDistance.shortest("coding", "practice"); // return 3
wordDistance.shortest("makes", "coding");    // return 1
```

Constraints:

- `1 <= wordsDict.length <= 3 * 10^4`
- `1 <= wordsDict[i].length <= 10`
- `wordsDict[i]` consists of lowercase English letters.
- `word1` and `word2` are in `wordsDict`.
- `word1 != word2`
- At most `5000` calls will be made to `shortest`.

Accepted 106,854 | Submissions 185,595

Seen this question in a real interview before? Yes No

Companies

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

LinkedIn | 48 Amazon | 2

Related Topics

Array Hash Table Two Pointers String Design

Similar Questions

Merge Two Sorted Lists Easy

Shortest Word Distance Easy

Shortest Word Distance III Medium

Java Autocomplete

```java
class WordDistance {

    HashMap<String, ArrayList<Integer>> locations;

    public WordDistance(String[] words) {
        this.locations = new HashMap<String, ArrayList<Integer>>();

        // Prepare a mapping from a word to all it's locations (indices).
        for (int i = 0; i < words.length; i++) {
            ArrayList<Integer> loc = this.locations.getOrDefault(words[i], new ArrayList<Integer>());
            loc.add(i);
            this.locations.put(words[i], loc);
        }
    }

    public int shortest(String word1, String word2) {
        ArrayList<Integer> loc1, loc2;

        // Location lists for both the words
        // the indices will be in SORTED order by default
        loc1 = this.locations.get(word1);
        loc2 = this.locations.get(word2);

        int l1 = 0, l2 = 0, minDiff = Integer.MAX_VALUE;
        while (l1 < loc1.size() && l2 < loc2.size()) {
            minDiff = Math.min(minDiff, Math.abs(loc1.get(l1) - loc2.get(l2)));
            if (loc1.get(l1) < loc2.get(l2)) {
                l1++;
            } else {
                l2++;
            }
        }

        return minDiff;
    }
}
```