LeetCode Explore Day 11 Problems Interview New Contest Discuss Store September LeetCoding Challenge 2021 Premium

Description Solution Discuss (484) Submissions Java Autocomplete

676. Implement Magic Dictionary

Medium 831 168 Add to List Share

Design a data structure that is initialized with a list of **different** words. Provided a string, you should determine if you can change exactly one character in this string to match any word in the data structure.

Implement the `MagicDictionary` class:

- `MagicDictionary()` Initializes the object.
- `void buildDict(String[] dictionary)` Sets the data structure with an array of distinct strings `dictionary`.
- `bool search(String searchWord)` Returns `true` if you can change **exactly one character** in `searchWord` to match any string in the data structure, otherwise returns `false`.

Example 1:

```
Input
["MagicDictionary", "buildDict", "search", "search", "search",
"search"]
[[], [["hello", "leetcode"]], ["hello"], ["hhllo"], ["hell"],
["leetcoded"]]
Output
[null, null, false, true, false, false]

Explanation
MagicDictionary magicDictionary = new MagicDictionary();
magicDictionary.buildDict(["hello", "leetcode"]);
magicDictionary.search("hello"); // return False
magicDictionary.search("hhllo"); // We can change the second 'h'
to 'e' to match "hello" so we return True
magicDictionary.search("hell"); // return False
magicDictionary.search("leetcoded"); // return False
```

Constraints:

- `1 <= dictionary.length <= 100`
- `1 <= dictionary[i].length <= 100`
- `dictionary[i]` consists of only lower-case English letters.
- All the strings in `dictionary` are **distinct**.
- `1 <= searchWord.length <= 100`
- `searchWord` consists of only lower-case English letters.
- `buildDict` will be called only once before `search`.
- At most `100` calls will be made to `search`.

Accepted 54,723 Submissions 97,730

Seen this question in a real interview before? Yes No

Companies

Related Topics

Similar Questions

```java
class MagicDictionary {

    /** Initialize your data structure here. */
    public MagicDictionary() {

    }

    public void buildDict(String[] dictionary) {

    }

    public boolean search(String searchWord) {

    }
}

/**
 * Your MagicDictionary object will be instantiated and called as such:
 * MagicDictionary obj = new MagicDictionary();
 * obj.buildDict(dictionary);
 * boolean param_2 = obj.search(searchWord);
 */
```

Problems Pick One Prev 17/39 Next Console Contribute Run Code Submit