Unique Word Abbreviation

An abbreviation of a word follows the form <first letter><number><last letter>. Below are some examples of word abbreviations:

```
a) it                      --> it    (no abbreviation)

     1
b) d|o|g                   --> d1g

               1    1  1
     1---5----0----5--8
c) i|nternationalizatio|n  --> i18n

               1
     1---5----0
d) l|ocalizatio|n          --> l10n
```

Assume you have a dictionary and given a word, find whether its abbreviation is unique in the dictionary. A word's abbreviation is unique if no *other* word from the dictionary has the same abbreviation.

Example:

```
Given dictionary = [ "deer", "door", "cake", "card" ]

isUnique("dear") ->
false

isUnique("cart") ->
true

isUnique("cane") ->
false

isUnique("make") ->
true
```

Solution 1

```
public class ValidWordAbbr {
    HashMap<String, String> map;
    public ValidWordAbbr(String[] dictionary) {
        map = new HashMap<String, String>();
        for(String str:dictionary){
            String key = getKey(str);
            // If there is more than one string belong to the same key
            // then the key will be invalid, we set the value to ""
            if(map.containsKey(key)){
                if(!map.get(key).equals(str)){
                    map.put(key, "");
                }
            }
            else{
                map.put(key, str);
            }
        }
    }

    public boolean isUnique(String word) {
        return !map.containsKey(getKey(word))||map.get(getKey(word)).equals(word)
;
    }

    String getKey(String str){
        if(str.length()<=2) return str;
        return str.charAt(0)+Integer.toString(str.length()-2)+str.charAt(str.leng
th()-1);
    }
}
```

written by skylinebaby original link here

Solution 2

To check for unique abbreviation, we maintain a mapping from a specific abbreviation to all words which have the abbreviation. Then we just need to check no other words have the same abbreviation as the given word.

The code is as follows.

```
class ValidWordAbbr {
public:
    ValidWordAbbr(vector<string> &dictionary) {
        for (string& d : dictionary) {
            int n = d.length();
            string abbr = d[0] + to_string(n) + d[n - 1];
            mp[abbr].insert(d);
        }
    }

    bool isUnique(string word) {
        int n = word.length();
        string abbr = word[0] + to_string(n) + word[n - 1];
        return mp[abbr].count(word) == mp[abbr].size();
    }
private:
    unordered_map<string, unordered_set<string>> mp;
};


// Your ValidWordAbbr object will be instantiated and called as such:
// ValidWordAbbr vwa(dictionary);
// vwa.isUnique("hello");
// vwa.isUnique("anotherWord");
```

written by jianchao.li.fighter original link here

Solution 3

```
public class ValidWordAbbr {

Map<String, String> map = new HashMap<>();

public ValidWordAbbr(String[] dictionary) {
    for (String dic : dictionary) {
        String key = getKey(dic);
        if (map.containsKey(key)) {
            map.put(key, "");
        } else {
            map.put(key, dic);
        }
    }
}

public boolean isUnique(String word) {
    String key = getKey(word);
    return !map.containsKey(key) || map.get(key).equals(word);
}

private String getKey(String word) {
    String key = word.charAt(0) + Integer.toString(word.length() - 2) + word.char
At(word.length() - 1);
    return key;
}
```

}

written by shuatidaxia original link here