**[Next Permutation](https://leetcode.com/problems/next-permutation/)**

**class WordDictionary {**

**public Map<Integer , List<String>> map;**

**/\*\* Initialize your data structure here. \*/**

**public WordDictionary() {**

**map = new HashMap<>();**

**}**

**/\*\* Adds a word into the data structure. \*/**

**public void addWord(String word) {**

**if(!map.containsKey(word.length())) {**

**map.put(word.length() , new ArrayList<>());**

**}**

**map.get(word.length()).add(word);**

**}**

**/\*\* Returns if the word is in the data structure. A word could contain the dot character '.' to represent any one letter. \*/**

**public boolean search(String word) {**

**if(!map.containsKey(word.length())) {**

**return false;**

**}**

**List<String> dict = map.get(word.length());**

**for(int i = 0 ; i < dict.size() ; i++) {**

**if(isSame(dict.get(i) , word)) {**

**return true;**

**}**

**}**

**return false;**

**}**

**public boolean isSame(String dict , String word) {**

**for(int i = 0 ; i < word.length() ; i++) {**

**if(dict.charAt(i) != word.charAt(i) && word.charAt(i) != '.') {**

**return false;**

**}**

**}**

**return true;**

**}**

**}**

**Time Complexity :**

**Search** : O(n \* l), no of words added, l is average length of word added

**Add** : O(1), constant time

Space Complexity : O(n \* l), no of words added, l is average length of word added