LeetCode\_290\_WordPattern\_词语模式\_Easy

# LeetCode\_290\_WordPattern\_词语模式\_Easy

## 题目介绍

/\*\*

\* Given a pattern and a string str, find if str follows the same pattern.

\*

\* Here follow means a full match, such that there is a bijection between a letter in pattern and a non-empty word in str.

\*

\* Example 1:

\* Input: pattern = "abba", str = "dog cat cat dog" Output: true

\* Example 2:

\* Input:pattern = "abba", str = "dog cat cat fish" Output: false

\* Example 3:

\* Input: pattern = "aaaa", str = "dog cat cat dog" Output: false

\* Example 4:

\* Input: pattern = "abba", str = "dog dog dog dog" Output: false

\* Notes:

\* You may assume pattern contains only lowercase letters,

\* and str contains lowercase letters separated by a single space.

## 思路分析

\* 思路分析：

\* 首先pattern字符作为key还是单词word作为key呢？

\* 利用pattern的每个字符作为hash键存在问题：

\* 如abba;dog dog dog dog; a与dog对应，而b与dog又对应，这种情况是错误的，

\* 在b与dog映射时，要检查出a与dog已经映射需要遍历一般值，比较麻烦。

\* 解决办法：直接用HashMap，将单词word作为key，字符作为value。

\* 然后还需要一个hash数组isUsed，存储字符是否已经使用过。

\* 单词数和pattern字符数不一致，直接返回false；

\* 首先判断单词是否已经在hashMap中，若存在，则获取其匹配的字符，

\* 判断是否与当前pattern中字符是否一致，若不一致，直接返回false；

\* 若hashMap中不存在该单词，则判断其pattern中对应字符是否已经使用过，

\* 若已经使用过，则直接返回false，否则，设置为true，并添加到hashMap中。

## Java代码

public boolean **wordPattern**(String pattern, String str) {

if(pattern == null||pattern.length() == 0||str == null||str.length() == 0) return false;

String[] strs = str.split(" ");

if(strs.length != pattern.length()) return false;

char[] patternChars = pattern.toCharArray();

boolean[] isUsed = new boolean[26];//存储字符26个小写字符是否已经使用过

HashMap<String,Character> hashMap = new HashMap<String,Character>();

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

if(hashMap.containsKey(strs[i])){//已经存在该单词

if(hashMap.get(strs[i]) != patternChars[i]) return false;

}else{

if(isUsed[patternChars[i]-'a']){

return false;//已经使用过

}else{

isUsed[patternChars[i]-'a']=true;

hashMap.put(strs[i],patternChars[i]);

}

}

}

return true;

}