链滴

Regular Expression Matching

作者：yudake

原文链接：https://ld246.com/article/1519520682529

来源网站：链滴

题目描述

Implement regular expression matching with support for '.' and '*'.

'.' Matches any single character.
'*' Matches zero or more of the preceding element.

The matching should cover the **entire** input string (not partial).

The function prototype should be:
bool isMatch(const char *s, const char *p)

Some examples:
isMatch("aa","a") → false
isMatch("aa","aa") → true
isMatch("aaa","aa") → false
isMatch("aa", "a*") → true
isMatch("aa", ".*") → true
isMatch("ab", ".*") → true
isMatch("aab", "c*a*b") → true

解题思路

和牛客网的正则表达式匹配是一道题。

这里看到LeetCode上有利用动态规划的方法，主要思想是利用一个二维数组memo，保存已经对比的字符串i和匹配串j位，减小了时间复杂度。

- 如果memo[i][j]不为空，说明字符串i位与匹配串j位已经匹配过，直接返回匹配结果就可以；
- 如果匹配串已经匹配完：
  - 如果字符串匹配完，返回true；
  - 否则返回false；
- 正常匹配会有以下情况：
  - 如果匹配串下一个字符不为‘*’：
    - 如果字符串i与匹配串j不相等，则memo[i][j]=false；
    - 若相等，继续下一位匹配；
  - 如果下一位字符为‘*’：
    - 匹配串后移两位（相当于匹配串本位出现次数为0），或者在i与j相等情况下字符串后移一位相当于匹配串本位出现一至多次），任意一种匹配成功都可以返回true。

附在这里。

代码

```
enum Result {
    TRUE, FALSE
```

```
}

class Solution {
    Result[][] memo;

    public boolean isMatch(String text, String pattern) {
        memo = new Result[text.length() + 1][pattern.length() + 1];
        return dp(0, 0, text, pattern);
    }

    public boolean dp(int i, int j, String text, String pattern) {
        if (memo[i][j] != null) {
            return memo[i][j] == Result.TRUE;
        }
        boolean ans;
        if (j == pattern.length()){
            ans = i == text.length();
        } else {
            boolean first_match = (i < text.length() &&
                                   (pattern.charAt(j) == text.charAt(i) ||
                                    pattern.charAt(j) == '.'));

            if (j + 1 < pattern.length() && pattern.charAt(j+1) == '*'){
                ans = (dp(i, j+2, text, pattern) ||
                       first_match && dp(i+1, j, text, pattern));
            } else {
                ans = first_match && dp(i+1, j+1, text, pattern);
            }
        }
        memo[i][j] = ans ? Result.TRUE : Result.FALSE;
        return ans;
    }
}
```

原文链接：Regular Expression Matching