# 动态规划识别 LCS 实验报告

## 一、 问题重述

- 【基本要求】用动态规划实现一个从两个字符串得到最大公共子序列，语言不限，提交代码和实验报告
- 【可选题】在第一题的基础上，如何进一步实现从两个字符串归纳出一个可以涵盖它们的正则表达式
  - 例如，S1=“吴亦凡尽然回复了李雪琴，我也是醉了\n”，S2=“她说我不照顾孩子，好吧，我也是醉了\n”，可以得到和这两个字符串匹配的正则表达式“\w+,我也是醉了\n”，遇到一个新字符串S3，可以判断是否用了相同的梗

## 二、 动态规划算法求解概述

最长公共子串（Longest Common Substring）与最长公共子序列（Longest Common Subsequence）的区别：子串要求在原字符串中是连续的，而子序列则只需保持相对顺序一致，并不要求连续。对于此题的要求，求子串更符合题意。

那么对于输入的 2 个字符串，根据其长度，构造对应的矩阵，来记录 2 个字符串每个字符的匹配情况。如果匹配，矩阵对应位置 DParray [i][j]的值为左上角的值加 1，LCS_len 记录最大公共子串的长度，endpos 用来记录最大公共字符串末尾位置的后一个位置。最终 LCStr 函数对于输入的两个字符串，返回它们的最大公共字符串，其开始位置以及其长度。

本题代码如下：

```
1.  str1 = input("Please input the 1st str: ")
2.  str2 = input("Please input the 2nd str: ")
3.
4.  def LCStr(s1, s2):
5.      DParray = [[0 for i in range(len(s2) + 1)] for j in range(len(s1) + 1)]
6.      LCS_len = 0
7.      endpos = 0
8.      for i in range(len(s1)):
9.          for j in range(len(s2)):
10.             if s1[i] == s2[j]:
11.                 DParray[i][j] = DParray[i - 1][j - 1] + 1
12.                 if DParray[i][j] > LCS_len:
13.                     LCS_len = DParray[i][j]
14.                     endpos = i + 1
15.     # 返回值依次为：最大子串，开始位置，最大子串长度
16.     return s1[endpos-LCS_len:endpos], endpos-LCS_len, LCS_len
17.
18. #“吴亦凡竟然回复了李雪琴，我也是醉了”，“她说我不照顾孩子，好吧，我也是醉了”
19. LCS, start, maxlen = LCStr(str1, str2)
```

```
if start == 0:
    if str1[start+maxlen-1] == "，" or str1[start+maxlen-1] == ",":
        print("The template being matched: ", LCS, "\w+")
    else:
        print("The template being matched: ", LCS + ",", "\w+")
else:
    if str1[start-1] == "，" or str1[start-1] == ",":
        print("The template being matched: ", "\w+,", LCS)
    else:
        print("The template being matched: ", "\w+", LCS)
str3 = input("Please input the 3rd str：")
if LCS.lstrip("，").rstrip("，") in str3:
    print("The 3rd string is in line with the template.")
else:
    print("The 3rd string is not in line with the template.")
```

在本题中进行了前后逗号和输出格式的相关判别，输出判定的固定模式（梗），然后在输入一段文本，判定是否包含上面得到的字符子串。

## 三、 结果展示

- Case 1:

```
Please input the 1st str：吴亦凡竟然回复了李雪琴，我也是醉了
Please input the 2nd str：她说我不照顾孩子，好吧，我也是醉了
The template being matched:  \w+ ，我也是醉了
Please input the 3rd str：他竟然连怎么洗衣服都不会，我也是醉了
The 3rd string is in line with the template.
```

- Case 2:

```
Please input the 1st str：早安，北京！
Please input the 2nd str：早安，打工人！
The template being matched:  早安， \w+
Please input the 3rd str：早安，上海。
The 3rd string is in line with the template.
```

- Case 3:

```
Please input the 1st str：罗本退役了，爷青结
Please input the 2nd str：离毕业只剩三年，爷青结
The template being matched:  \w+ ，爷青结
Please input the 3rd str：爷爷的青春结束了
The 3rd string is not in line with the template.
```