14주차 : LCS(Longest Common Subsequence)

LCS(Longest Common Subsequence)란, 말 그대로 **최장 공통 부분 수열을** 말한다.

풀어서 설명하면, 두 문자열에서 서로 공통되는 가장 긴 부분 문자열을 말한다. LCS 알고리즘 문제는 이 부분 문자열을 구하는 것이 핵심!!

- DP를 활용
  - 부분 문제 선정
  - 점화식 유도

<최장공통문자열/길이, 최장공통부분문자열/길이>

BASE.

1. 2차원 배열(`LCS`라 명명하겠음)을 활용하여 두 문자열(A, B)을 행(`i`), 열(`j`)에 매칭.

2. 편의상 `i, j`가 `0`일때는 모두 `0`을 넣어줘 마진값을 설정 이후 `i, j`가 `1`이상일 때부터 검사 시작

최장공통문자열(LCS) 점화식 및 풀이

```
if i == 0 or j == 0:
  LCS[i][j] = 0
elif string_A[i] == string_B[j]:
  LCS[i][j] = LCS[i - 1][j - 1] + 1
else:
  LCS[i][j] = 0
```

<2차원 배열을 활용한 최장공통문자열/길이 검사순서>

1. 문자열A, B를 한글자씩 비교
2. A, B가 다르면 `LCS[i][j] = 0`
3. A, B가 같으면 `LCS[i - 1][j - 1]` 값을 찾아 `+1` 합니다.
    a. 연속으로 이어지는 문자열을 찾는 것이기 때문 (그림판 설명)
4. 위 과정을 반복하여 만들어지는 배열의 가장 큰 수가 **최장공통문자열 길이**,
5. 문자열을 알고 싶다면 가장 큰 수의 인덱스부터 `i - 1, j - 1` 하면서 `result` 리스트에 담고 뒤집은 값이 **최장공통문자열**

최장공통부분문자열(LCS)의 길이 점화식 및 풀이

```
if i == 0 or j == 0:
  LCS[i][j] = 0
elif string_A[i] == string_B[j]:
  LCS[i][j] = LCS[i - 1][j - 1] + 1
else:
  LCS[i][j] = max(LCS[i - 1][j], LCS[i][j - 1])
```

<2차원 배열을 활용한 최장공통부분문자열 길이 검사순서>

1. 문자열A, B를 한글자씩 비교
2. A, B가 다르면 `LCS[i][j] = max(LCS[i - 1][j], LCS[i][j - 1])`
3. A, B가 같으면 `LCS[i - 1][j - 1]` 값을 찾아 `+1` 합니다.
4. 위 과정을 반복하여 만들어지는 배열의 가장큰 수가 **최장공통부분문자열 길이**,

<2차원 배열을 활용한 최장공통부분문자열 검사순서>

1. 문자열을 알고 싶다면 배열의 마지막 원소 위치부터 시작하여, `LCS[i - 1][j]`와 `LCS[i][j - 1]` 중 현재 값과 같은 값 탐색.

2-1. 만약 **같은 값이 있다면 해당 값으로 이동**합니다.

2-2. 만약 **같은 값이 없다면** `result` **배열에 해당 문자를 넣고** `LCS[i -1][j - 1]` **로 이동**합니다.

3. 2번 과정을 반복하다가 0으로 이동하게 되면 종료합니다.
4. `result` 배열의 역순이 **최장공통부분문자열** 입니다.

예제 풀이

- LCS (Gold5) : https://www.acmicpc.net/problem/9251

  ▼ 풀이

```
# 양승열
str1 = input()
str2 = input()
len_str1 = len(str1)
len_str2 = len(str2)

print('str1 : ', str1)
print('str2 : ', str2)
print('len_str1 : ', len_str1)
print('len_str2 : ', len_str2)

lcs_list = [[0] * (len_str2 + 1) for _ in range(len_str1 + 1)]

# 최장 공통 부분 문자열 길이 구하기
for i in range(1, len_str1 + 1):
    for j in range(1, len_str2 + 1):
        if str1[i-1] == str2[j-1]:
            lcs_list[i][j] = lcs_list[i-1][j-1] + 1
        else:
            lcs_list[i][j] = max(lcs_list[i][j-1], lcs_list[i-1][j])

print(lcs_list[-1][-1])
```

오늘의 문제

- LCS 3 (Gold3) : https://www.acmicpc.net/problem/1958

출처

- https://velog.io/@emplam27/알고리즘-그림으로-알아보는-LCS-알고리즘-Longest-Common-Substring와-Longest-Common-Subsequence