

# DP #2

---

김현정 Acka1357@gmail.com

# 포도주 시식

Problem: <https://www.acmicpc.net/problem/2156>

- N개의 잔에 포도주가 들어있다.
  - 각 잔에 들어있는 포도주의 양은 다르다.
  - 잔을 선택하면 그 잔의 포도주는 모두 마셔야 한다.
  - 마신 후에는 빈 잔을 원래 위치에 다시 놓는다.
  - 연속으로 놓여있는 3잔을 모두 마실 수는 없다.
- 
- 최대한로 마실 수 있는 포도주의 양

# 포도주 시식

Problem: <https://www.acmicpc.net/problem/2156>

- $i$ 번 잔을 마실때 필요한 정보
- $i$ 번 잔이 몇번째 잔인가?
- 1, 2잔째라면  $i$ 번 잔을 마실 수 있다.

# 포도주 시식

Problem: <https://www.acmicpc.net/problem/2156>

- $D[i][0]$ :  $i$ 번 잔을 마시지 않았을 때 최대로 마실 수 있는 포도주의 양
  - $i-1$ 번 잔을 마셨는지 아닌지는 중요하지 않다.
- $D[i][1]$ :  $i$ 번 잔이 연속 1잔째일 때 최대로 마실 수 있는 포도주의 양
  - $i-1$ 번 잔을 마시지 않고,  $i$ 번 잔을 마시는 경우
- $D[i][2]$ :  $i$ 번 잔이 연속 2잔째일 때 최대로 마실 수 있는 포도주의 양
  - $i-2$ 번 잔을 마시지 않고,  $i-1$ ,  $i$ 번 잔을 마시는 경우

# 포도주 시식

Problem: <https://www.acmicpc.net/problem/2156>

- $D[i][0]: \max(D[i - 1][0], D[i - 1][1], D[i - 1][2])$
- $D[i][1]: D[i - 1][0] + i$ 번 포도주의 양
- $D[i][2]: D[i - 1][1] + i$ 번 포도주의 양

# 포도주 시식

Problem: <https://www.acmicpc.net/problem/2156>

- 답:  $\max_{i=0 \text{ to } 2} D[N][i]$ 
  - N번 잔까지 (안)마신 모든 상태 중 최대값

# 포도주 시식

Problem: <https://www.acmicpc.net/problem/2156>

- C/C++:

<https://gist.github.com/Acka1357/9b65179b4b47c889b6013fff8663f665>

- JAVA:

<https://gist.github.com/Acka1357/194f2ad6611d8cdcbef657af2bbf2177>

# 스티커

Problem: <https://www.acmicpc.net/problem/9465>

- $2 \times N$ 으로 나열된 스티커가 있다.
  - 스티커 한 장을 떼면 그 스티커와 변을 공유하는 스티커는 모두 찢어진다. (왼쪽, 오른쪽, 위, 아래)
  - 각 스티커에 점수가 있을 때
  - 떼 수 있는 스티커 점수의 최대값
- 
- $1 \leq N \leq 100,000$



# 스티커

Problem: <https://www.acmicpc.net/problem/9465>

- $i$ 번 줄의 스티커는
- 윗쪽만 떼거나
- 아래쪽만 떼거나
- 둘 다 떼지 않는 경우로 나뉜다.

# 스티커

Problem: <https://www.acmicpc.net/problem/9465>

- $i$ 번 줄의 윗쪽 스티커를 떼기 위해서는
- $i-1$ 번 줄의 윗쪽 스티커를 떼지 않았어야 한다.
  
- $i$ 번 줄의 아래쪽 스티커를 떼기 위해서는
- $i-1$ 번 줄의 아래쪽 스티커를 떼지 않았어야 한다.

# 스티커

Problem: <https://www.acmicpc.net/problem/9465>

- $D[i][j]$ :  $i$ 번 줄의 스티커까지 떼다/안떼다를 결정했으며  
 $i$ 번 줄의 상태가  $j$ 일 때 스티커 점수의 최대값
  - $j = 0$ :  $i$ 번 줄의 스티커를 떼지 않았다.
  - $i = 1$ :  $i$ 번 줄의 윗쪽 스티커를 떼었다.
  - $i = 2$ :  $i$ 번 줄의 아래쪽 스티커를 떼었다.

# 스티커

Problem: <https://www.acmicpc.net/problem/9465>

- $D[i][0]: \max(D[i - 1][0], D[i - 1][1], D[i - 1][2])$
- $D[i][1]: \max(D[i - 1][0], D[i - 1][2]) + i$ 번 줄 윗쪽 스티커의 점수
- $D[i][2]: \max(D[i - 1][0], D[i - 1][1]) + i$ 번 줄 아래쪽 스티커의 점수

# 스티커

Problem: <https://www.acmicpc.net/problem/9465>

- 답:  $\max_{i=0 \text{ to } 2}(D[N][i])$ 
  - N번 줄까지 다 떼었을때 점수의 최대값
  - $== D[N + 1][0]$

# 스티커

Problem: <https://www.acmicpc.net/problem/9465>

- C/C++:  
<https://gist.github.com/Acka1357/8afb1d7818953980b593f09a0794553a>
- JAVA:  
<https://gist.github.com/Acka1357/1887fba7981ac177c849125d42bc50c1>

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- $N \times N$  게임판의 가장 왼쪽 위칸에서 가장 오른쪽 아래 칸으로 가야 한다.
- 각 칸에 적힌 수는 현재 칸에서 갈 수 있는 거리이다.
- 오른쪽이나 아래쪽 방향으로만 움직일 수 있다.
- 가장 왼쪽 위칸에서 가장 오른쪽 아래 칸으로 규칙에 맞게 이동할 수 있는 경로의 개수
- $1 \leq N \leq 100$

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- 오른쪽이나 아래쪽 방향으로만 움직일 수 있다.
- $d[i][j] \leftarrow d[i - x][j], d[i][j - x]$



# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- $D[i][j]$ :  $(i, j)$ 까지 올 수 있는 경로의 개수

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- $D[i][j]: \sum_{k=1}^{i-j} D[i-k][j] \text{ (if } A[i-k][j] == k)$   
 $+ \sum_{k=1}^{j-i} D[i][j-k] \text{ (if } A[i][j-k] == k)$
- $A[i][j]: (i, j)$ 에 적힌 숫자

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- $D[i][j]: \sum_{k=1}^{i-1} D[i-k][j] \text{ (if } A[i-k][j] == k)$   
 $+ \sum_{k=1}^{j-1} D[i][j-k] \text{ (if } A[i][j-k] == k)$

```
int get_count(int r, int c){
    if(0 <= d[r][c]) return d[r][c];

    for(int i = 1; i <= r; i++)
        if(a[r-i][c] == i){
            d[r][c] += get_count(r-i, c);
        }
}
```

```
for(int i = 1; i <= c; i++)
    if(a[r][c-i] == i){
        d[r][c] += get_count(r, c-i);
    }
return d[r][c];
}
```

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- $D[i + A[i][j]][j] \leftarrow D[i][j]$
- $D[i][j] + A[i][j] \leftarrow D[i][j]$

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- $D[i + A[i][j]][j] \leftarrow D[i][j]$
- $D[i][j] + A[i][j] \leftarrow D[i][j]$

```
for(int i = 0; i < N; i++)  
    for(int j = 0; j < N; j++){  
        if(a[i][j] == 0) continue;  
        if(i + a[i][j] < N) d[i + a[i][j]][j] += d[i][j];  
        if(j + a[i][j] < N) d[i][j + a[i][j]] += d[i][j];  
    }
```

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- 답:  $D[N - 1][N - 1]$ 
  - $(i, j)$ 까지 오는 경로의 수

# 점프

Problem: <https://www.acmicpc.net/problem/1890>

- C/C++:  
<https://gist.github.com/Acka1357/9f430fcb96c0774fac01659c8b3189b9>
- JAVA:  
<https://gist.github.com/Acka1357/1c4f77db413720ff12f646897ba5a089>

# Longest Common Subsequence

최장 공통 부분 수열



# Longest Common Subsequence

최장 공통 부분수열

- 수열
- {1, 3, 5, 4, 2, 6}

# Longest Common Subsequence

최장 공통 부분수열

- 부분수열
  - 기존 수열의 일부 항을 원래 순서대로 나열해 얻을 수 있는 모든 수열
- {1, 3, 5, 4, 2, 6}
- {1, 3, 5, 4}
- {3, 5, 4, 2}
- {1, 5, 6}
- {3, 2}
- {5}

# Longest Common Subsequence

최장 공통 부분수열

- 최장 공통 부분 수열
  - 주어진 수열 모두의 부분 수열이 되는 수열 중 가장 긴 수열

# Longest Common Subsequence

최장 공통 부분수열

- $A = \{1, 3, 5, 4, 2, 6\}$
- $B = \{1, 2, 3, 4, 5, 6\}$

# Longest Common Subsequence

최장 공통 부분수열

- $A = \{1, 3, 5, 4, 2, 6\}$
- $B = \{1, 2, 3, 4, 5, 6\}$
- $LCS(A, B): \{1, 3, 4, 6\}$  or  $\{1, 3, 5, 6\}$
- $LCS(A, B)$ 길이: 4

# Longest Common Subsequence

최장 공통 부분수열

- 최장 공통 부분수열의 길이를 먼저 구해보자.

# Longest Common Subsequence

최장 공통 부분수열

- $A = A[1, L_A]$
- $B = B[1, L_B]$
- $lcs[i][j] = \text{LCS}(A[1, i], B[1, j])$ 의 길이

# Longest Common Subsequence

최장 공통 부분수열

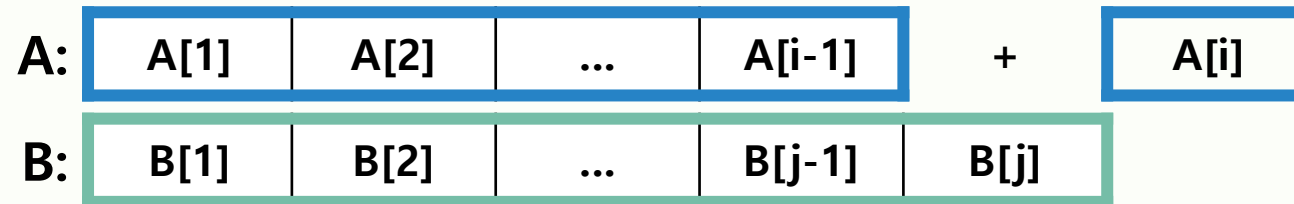
- $LCS(A[1, i], B[1, j])$ 는 다음과 같이 나눌 수 있다.
- 1.  $LCS(A[1, i - 1] + A[i], B[1, j])$
- 2.  $LCS(A[1, i], B[1, j - 1] + B[j])$
- 3.  $LCS(A[1, i - 1] + A[i], B[1, j - 1] + B[j])$



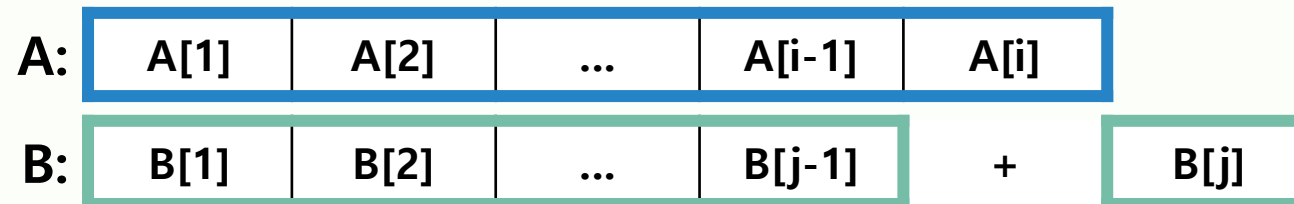
# Longest Common Subsequence

최장 공통 부분수열

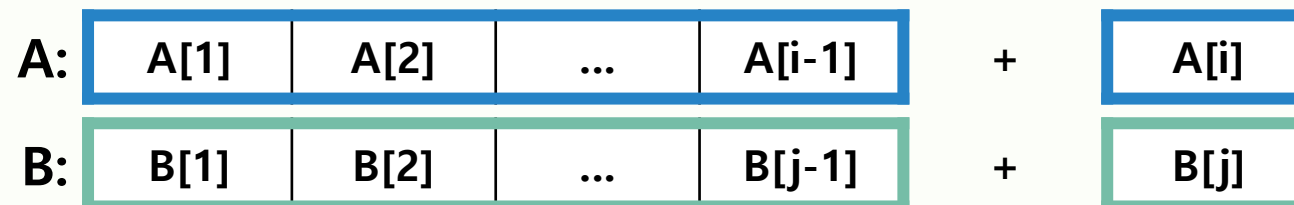
- 1.  $\text{LCS}(A[1, i-1] + A[i], B[1, j])$



- 2.  $\text{LCS}(A[1, i], B[1, j-1] + B[j])$



- 3.  $\text{LCS}(A[1, i-1] + A[i], B[1, j-1] + B[j])$



# Longest Common Subsequence

최장 공통 부분수열

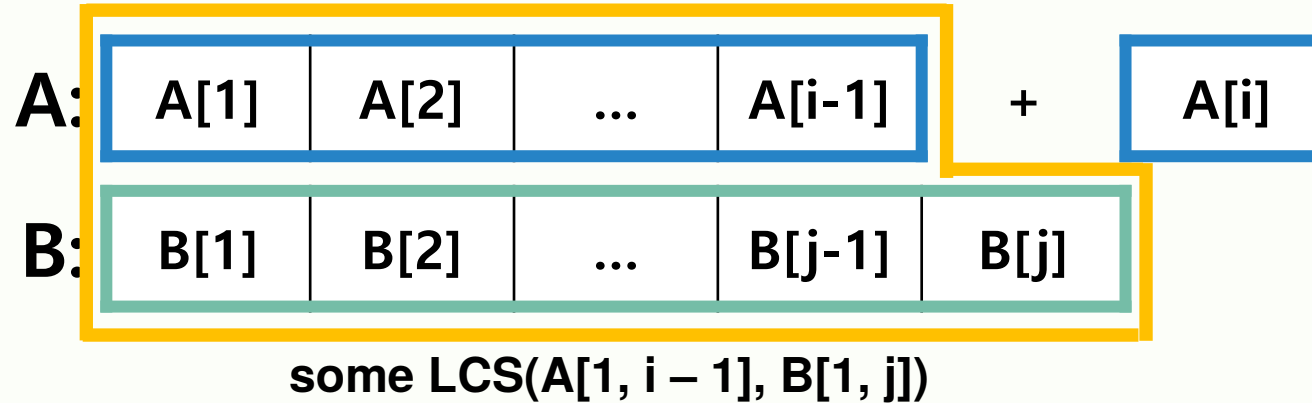
- $\text{LCS}(A[1, i - 1] + A[i], B[1, j])$



# Longest Common Subsequence

최장 공통 부분수열

- $\text{LCS}(A[1, i - 1] + A[i], B[1, j])$



# Longest Common Subsequence

최장 공통 부분수열

- $\text{LCS}(A[1, i - 1] + A[i], B[1, j])$



- $\text{lcs}[i][j] = \text{lcs}[i - 1][j]$

# Longest Common Subsequence

최장 공통 부분수열

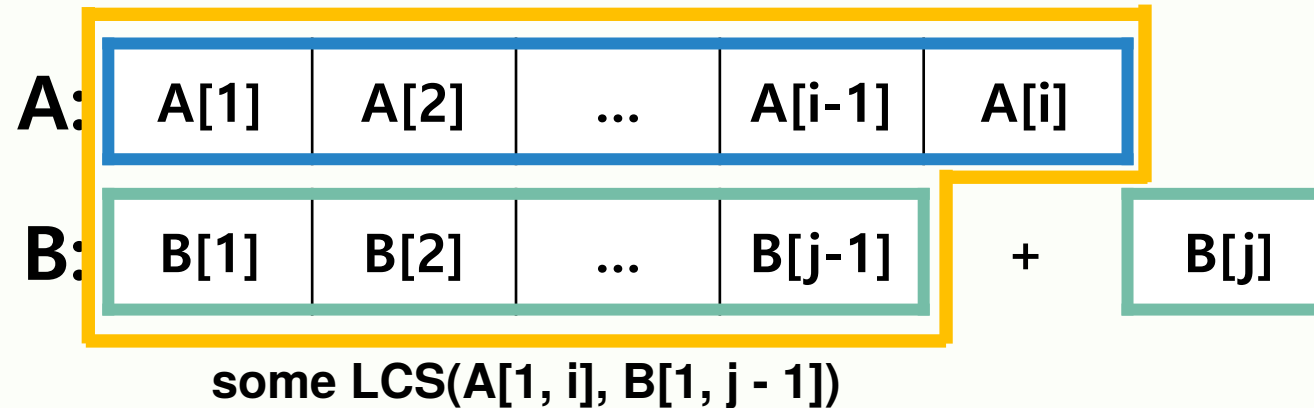
- $\text{LCS}(A[1, i], B[1, j - 1] + B[j])$



# Longest Common Subsequence

최장 공통 부분수열

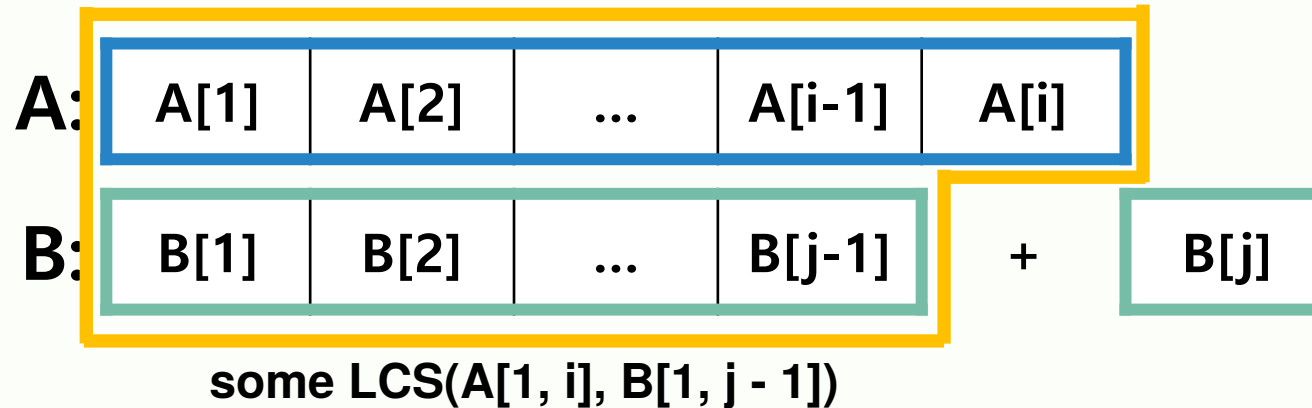
- $\text{LCS}(A[1, i], B[1, j - 1] + B[j])$



# Longest Common Subsequence

최장 공통 부분수열

- $\text{LCS}(A[1, i], B[1, j - 1] + B[j])$



- $\text{lcs}[i][j] = \text{lcs}[i][j - 1]$

# Longest Common Subsequence

최장 공통 부분수열

- $\text{LCS}(A[1, i - 1] + A[i], B[1, j - 1] + B[j])$

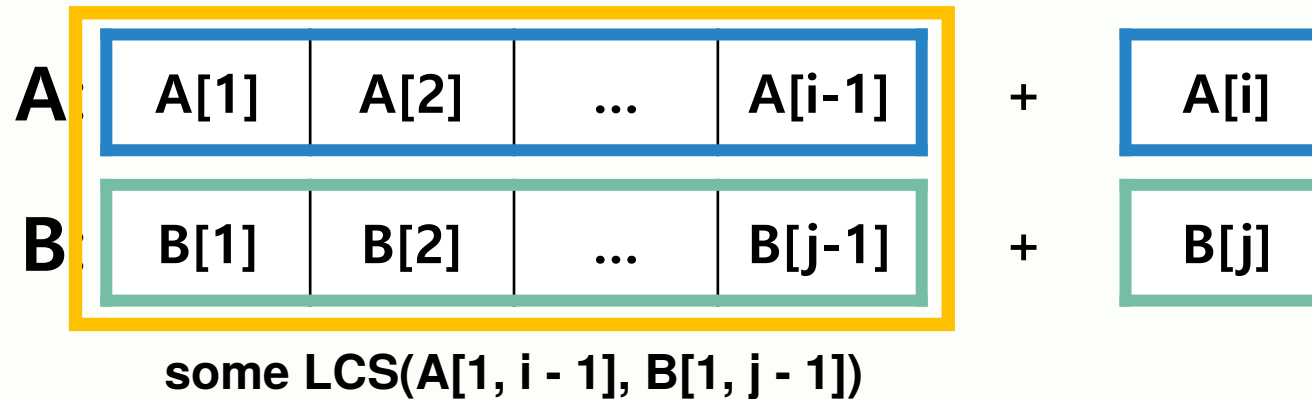




# Longest Common Subsequence

최장 공통 부분수열

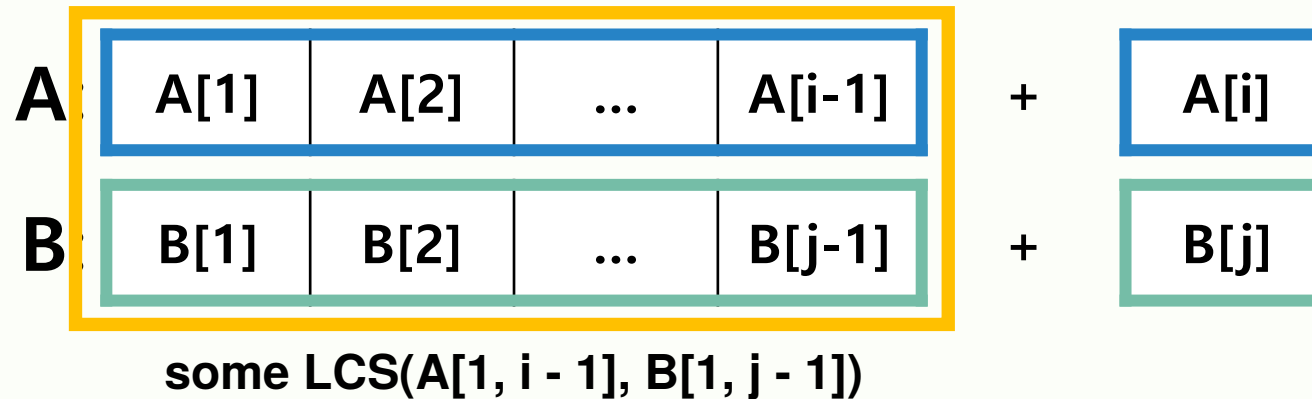
- $\text{LCS}(A[1, i - 1] + A[i], B[1, j - 1] + B[j])$



# Longest Common Subsequence

최장 공통 부분수열

- $\text{LCS}(A[1, i - 1] + A[i], B[1, j - 1] + B[j])$



- $A[i] == B[j]: \text{lcs}[i][j] = \text{lcs}[i - 1][j - 1] + 1$
- **else:**  $\text{lcs}[i][j] = \text{lcs}[i - 1][j - 1]$

# Longest Common Subsequence

최장 공통 부분수열

- $lcs[i][j] = lcs[i - 1][j - 1] + 1$  (if  $A[i] == B[j]$ )
- $lcs[i][j] = \max(lcs[i - 1][j], lcs[i][j - 1])$  (if  $A[i] != B[j]$ )

# Longest Common Subsequence

최장 공통 부분수열

- $lcs[i][j] = lcs[i - 1][j - 1] + 1$  (if  $A[i] == B[j]$ )
- $lcs[i][j] = \max(lcs[i - 1][j], lcs[i][j - 1])$  (if  $A[i] != B[j]$ )
- $lcs[i][0] = lcs[0][j] = 0$

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0							
1	1							
3	2							
5	3							
4	4							
2	5							
6	6							

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0						
3	2	0						
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0						
3	2	0						
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1					
3	2	0						
5	3	0						
4	4	0						
2	5	0						
6	6	0						



# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1					
3	2	0						
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1				
3	2	0						
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0						
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0						
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1					
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1				
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1				
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2			
5	3	0						
4	4	0						
2	5	0						
6	6	0						



# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0						
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2		
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence


최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2		
4	4	0						
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	
4	4	0						
2	5	0						
6	6	0						



# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2			
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2			
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2	3		
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2	3		
2	5	0						
6	6	0						



# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2	3		
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2	3	3	
2	5	0						
6	6	0						

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2	3	3	3
2	5	0	1	2	2	3	3	3
6	6	0	1	2	2	3	3	4

# Longest Common Subsequence

최장 공통 부분수열

	B[ ]		1	2	3	4	5	6
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
1	1	0	1	1	1	1	1	1
3	2	0	1	1	2	2	2	2
5	3	0	1	1	2	2	3	3
4	4	0	1	1	2	3	3	3
2	5	0	1	2	2	3	3	3
6	6	0	1	2	2	3	3	4

# Longest Common Subsequence

최장 공통 부분수열

- $D[i][j]$ :  $A[1, i]$ 와  $B[1, j]$ 의 최장 공통 부분수열의 길이

# Longest Common Subsequence

최장 공통 부분수열

- 답:  $D[L_A][L_B]$ 
  - 수열 A, B의 최장 공통 부분수열의 길이

# LCS

---

Problem: <https://www.acmicpc.net/problem/9251>

- 문자열 A와 B의 LCS의 길이

# LCS

Problem: <https://www.acmicpc.net/problem/9251>

- LCS는 수열뿐 아니라 순서가 있는 모든 자료에 대해 동일한 개념으로 존재한다.



# LCS

---

Problem: <https://www.acmicpc.net/problem/9251>

- C/C++:  
<https://gist.github.com/Acka1357/8896f8efe6cc2ad26a5019c38ade135d>
- Java:  
<https://gist.github.com/Acka1357/973137d33a2b691fd7e31c639e4b8cc0>

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 문자열 A와 B의 LCS의 길이와
- 해당 길이의 LCS를 찾는 문제

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 예제의 DP 테이블을 살펴봅시다.

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- $D[6][6]$ : 4

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- $D[6][6]$ : 4

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- K와 P가 다르므로 이 값은 D[5][6]에서 왔음을 알 수 있다.

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[5][6]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	3

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[5][6]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	3



# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[5][6]$ 는?

	B[ ]		C	A	P	C	A	K
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	3

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- A[5]와 B[6]이 'K'로 같으므로, D[4][5]에서 왔음을 알 수 있다.

	B[ ]		C	A	P	C	A	K
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	3

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 또한 이 'K'는 최장증가부분수열에 속한다는 것을 알 수 있다.

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	3

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[4][5]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[4][5]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[3][5]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[3][5]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- A[3]과 B[5]가 'A'로 같으므로, 이 'A'는 쓰였으며, D[2][4]에서 왔다.

	B[ ]		C	A	P	C	A	K
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4



# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[2][4]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[2][4]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- A[2]와 B[4]가 'C'로 같으므로, 이 C는 쓰였고 D[1][3]에서 왔다.

	B[ ]		C	A	P	C	A	K
A[ ]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[1][3]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[1][3]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 그렇다면  $D[1][2]$ 는?

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- A[1]과 B[2]가 'A'로 같으므로, 이 'A'는 쓰였고 D[0][1]에서 왔다.

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- 이후 값이 0이되므로 더 볼 필요가 없다.

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4



# LCS 2

Problem: <https://www.acmicpc.net/problem/9251>

- $\text{LCS}(A, B)$ : "ACAK"

	B[]		C	A	P	C	A	K
A[]		0	1	2	3	4	5	6
	0	0	0	0	0	0	0	0
A	1	0	0	1	1	1	1	1
C	2	0	1	1	1	2	2	2
A	3	0	1	2	2	2	3	3
Y	4	0	1	2	2	2	3	3
K	5	0	1	2	2	2	3	4
P	6	0	1	2	3	3	3	4

# LCS 2

---

Problem: <https://www.acmicpc.net/problem/9251>

- C/C++:  
<https://gist.github.com/Acka1357/30434b025175a8bb78929a943d2171b8>
- Java:  
<https://gist.github.com/Acka1357/6d8b92ae6f395b90fdd00298c8a9076e>

# LCS 3

Problem: <https://www.acmicpc.net/problem/1958>

- 두 문자열의 LCS가 아닌
- 세 문자열의 LCS를 구하는 문제

# LCS 3

Problem: <https://www.acmicpc.net/problem/1958>

- 두 문자열의 LCS를 구할때와 마찬가지로 경우를 나눌 수 있다.

# LCS 3

Problem: <https://www.acmicpc.net/problem/1958>

- 두 문자열의 LCS를 구할때와 마찬가지로 경우를 나눌 수 있다.
- $LCS(A[1, i], B[1, j], C[1, k])$ 
  - $LCS(A[1, i - 1] + A[i], B[1, j], C[1, k])$
  - $LCS(A[1, i], B[1, j - 1] + B[j], C[1, k])$
  - $LCS(A[1, i], B[1, j], C[1, k - 1] + C[k])$
  - $LCS(A[1, i - 1] + A[i], B[1, j - 1] + B[j], C[1, k])$
  - $LCS(A[1, i], B[1, j - 1] + B[j], C[1, k - 1] + C[k])$
  - $LCS(A[1, i - 1] + A[i], B[1, j], C[1, k - 1] + C[k])$
  - $LCS(A[1, i - 1] + A[i], B[1, j - 1] + B[j], C[1, k - 1] + C[k])$

# LCS 3

Problem: <https://www.acmicpc.net/problem/1958>

- $\text{if}(\text{A}[i] == \text{B}[j] \ \&\& \ \text{B}[j] == \text{C}[k])$
- $\text{lcs}[i][j][k] = \text{lcs}[i - 1][j - 1][k - 1] + 1$
- else
- $\text{lcs}[i][j][k] = \max(\text{lcs}[i - 1][j][k], \text{lcs}[i][j - 1][k], \text{lcs}[i][j][k - 1],$   
 $\text{lcs}[i - 1][j - 1][k], \text{lcs}[i][j - 1][k - 1], \text{lcs}[i - 1][j][k - 1])$

# LCS 3

Problem: <https://www.acmicpc.net/problem/1958>

- C/C++:  
<https://gist.github.com/Acka1357/129abe5142db9575a9ac56f4daa61bf7>
- Java:  
<https://gist.github.com/Acka1357/8f5a413693b46fc4a5d8de2640dcea2d>