Dynamic Programming (4)

By: Aminul Islam

Steps in Dynamic Programming

Steps in Dynamic Programming

- Characterize the structure of an optimal solution.
- Recursively define the value of an optimal solution.
- Compute the value of an optimal solution, typically in a bottom-up fashion.
- Construct an optimal solution from computed values.

The longest common subsequence (LCS) problem

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.
For example, given a sequence “abcdefg”

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.
For example, given a sequence “abcdefg”
Some of its subsequences are:

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.
For example, given a sequence “abcdefg”
Some of its subsequences are:
“abc”

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.
For example, given a sequence “abcdefg”
Some of its subsequences are:
“abc”, “abg”

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.
For example, given a sequence “abcdefg”
Some of its subsequences are:
“abc”, “abg”, “bdf”

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.
For example, given a sequence “abcdefg”
Some of its subsequences are:
“abc”, “abg”, “bdf”, “aeg”

The longest common subsequence (LCS) problem

LCS Problem Statement: Given two sequences, the task is to find the longest subsequence (and its length) present in both of them. A subsequence is a sequence that appears in the same relative order, but not necessarily contiguous.
For example, given a sequence “abcdefg”
Some of its subsequences are:
“abc”, “abg”, “bdf”, “aeg”, “acefg”

Applications of the LCS problem

Applications of the LCS problem

- A classic computer science problem

Applications of the LCS problem

- A classic computer science problem
- The basis of data comparison programs such as the “diff” utility

Applications of the LCS problem

- A classic computer science problem
- The basis of data comparison programs such as the “diff” utility
- It is also widely used by many revision control systems such as “Git”

Applications of the LCS problem

- A classic computer science problem
- The basis of data comparison programs such as the “diff” utility
- It is also widely used by many revision control systems such as “Git”
- Has many applications in bioinformatics

Algorithm: Recursive solution to LCS Problem

Algorithm: Recursive solution to LCS Problem

```
int LCS( char *X, char *Y, int m, int n )
{
    if (m == 0 || n == 0)
      return 0;
    if (X[m] == Y[n])
      return 1 + LCS(X, Y, m-1, n-1);
    else
      return max(LCS(X, Y, m, n-1), LCS(X, Y, m-1, n));
}
```

Algorithm: Recursive solution to LCS Problem

```
int LCS( char *X, char *Y, int m, int n )
{
    if (m == 0 || n == 0)
      return 0;
    if (X[m] == Y[n])
      return 1 + LCS(X, Y, m-1, n-1);
    else
      return max(LCS(X, Y, m, n-1), LCS(X, Y, m-1, n));
}
```

Time complexity, $T(n) =$

Algorithm: Recursive solution to LCS Problem

```
int LCS( char *X, char *Y, int m, int n )
{
    if (m == 0 || n == 0)
      return 0;
    if (X[m] == Y[n])
      return 1 + LCS(X, Y, m-1, n-1);
    else
      return max(LCS(X, Y, m, n-1), LCS(X, Y, m-1, n));
}
```

Time complexity, $T(n) = \mathrm{T(n\text{-}1)+T(n\text{-}1)+1}$

Algorithm: Recursive solution to LCS Problem

```
int LCS( char *X, char *Y, int m, int n )
{
    if (m == 0 || n == 0)
       return 0;
    if (X[m] == Y[n])
       return 1 + LCS(X, Y, m-1, n-1);
    else
       return max(LCS(X, Y, m, n-1), LCS(X, Y, m-1, n));
}
```

Time complexity, $T(n) = \mathrm{T(n\text{-}1)+T(n\text{-}1)+1}$
$T(n) \in O(2^n)$

Algorithm: DP solution to LCS Problem

Algorithm: DP solution to LCS Problem

```
int LCS( char *X, char *Y, int m, int n )
{
    int L[m+1, n+1];
    int i, j;
    for (i=0; i<=m; i++)
    {
       for (j=0; j<=n; j++)
       {
          if (i == 0 || j == 0)
             L[i, j] = 0;
          else if (X[i] == Y[j])
             L[i, j] = L[i-1, j-1] + 1;
          else
             L[i, j] = max(L[i-1, j], L[i, j-1]);
       }
    }
    return L[m, n];
}
```

Example of LCS Problem(DP sol.)

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0						
A 1						
C 2						
B 3						
D 4						
E 5						
A 6						

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1						
C 2						
B 3						
D 4						
E 5						
A 6						

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0					
C 2	0					
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1				
C 2	0					
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1			
C 2	0					
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1		
C 2	0					
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	
C 2	0					
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0					
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1				
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1			
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2		
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0					
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1				
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2			
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2		
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0					
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1				
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2			
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2		
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0					
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1				
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2			
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2		
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0					

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1				

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2			

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2		

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

- length of LCS =

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

- length of LCS = 4

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

- length of LCS = 4
- LCS =

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
if (X[i] == Y[j])
  L[i, j] = L[i-1, j-1] + 1;
else
  L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

- length of LCS = 4
- LCS = A

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

- length of LCS = 4
- LCS = DA

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
        if (X[i] == Y[j])
          L[i, j] = L[i-1, j-1] + 1;
        else
          L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

- length of LCS = 4
- LCS = B DA

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
if (X[i] == Y[j])
  L[i, j] = L[i-1, j-1] + 1;
else
  L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

- length of LCS = 4
- LCS = A B DA

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
if (X[i] == Y[j])
   L[i, j] = L[i-1, j-1] + 1;
else
   L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

- length of LCS = 4
- LCS = A B DA
- Is "ACDA" another optimal solution?

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
if (X[i] == Y[j])
  L[i, j] = L[i-1, j-1] + 1;
else
  L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

- length of LCS = 4
- LCS = A B DA
- Is "ACDA" another optimal solution?
- Time Complexity =

Example of LCS Problem(DP sol.)

X	A	C	B	D	E	A
	1	2	3	4	5	6

Y	A	B	C	D	A
	1	2	3	4	5

```
if (X[i] == Y[j])
  L[i, j] = L[i-1, j-1] + 1;
else
  L[i, j] = max(L[i-1, j], L[i,
j-1]);
```

l[i,j]	j=0	1 A	2 B	3 C	4 D	5 A
i = 0	0	0	0	0	0	0
A 1	0	1	1	1	1	1
C 2	0	1	1	2	2	2
B 3	0	1	2	2	2	2
D 4	0	1	2	2	3	3
E 5	0	1	2	2	3	3
A 6	0	1	2	2	3	4

- length of LCS = 4
- LCS = A B DA
- Is "ACDA" another optimal solution?
- Time Complexity $= O(mn)$

How to find the sequence from the previous table?

How to find the sequence from the previous table?

- Start from the lower-right corner cell.

How to find the sequence from the previous table?

- Start from the lower-right corner cell.
- If the cell directly above or directly to the left contains a value equal to the value in the current cell, then move to that cell (if both are equal to the current one, then chose either one).

How to find the sequence from the previous table?

- Start from the lower-right corner cell.
- If the cell directly above or directly to the left contains a value equal to the value in the current cell, then move to that cell (if both are equal to the current one, then chose either one).
- If both such cells have values less than the value in the current cell, then output the character that is in the current cell and move diagonally up-left cell.

How to find the sequence from the previous table?

- Start from the lower-right corner cell.
- If the cell directly above or directly to the left contains a value equal to the value in the current cell, then move to that cell (if both are equal to the current one, then chose either one).
- If both such cells have values less than the value in the current cell, then output the character that is in the current cell and move diagonally up-left cell.
- Stop when in top left cell

How to find the sequence from the previous table?

- Start from the lower-right corner cell.
- If the cell directly above or directly to the left contains a value equal to the value in the current cell, then move to that cell (if both are equal to the current one, then chose either one).
- If both such cells have values less than the value in the current cell, then output the character that is in the current cell and move diagonally up-left cell.
- Stop when in top left cell

This gives you the characters in reverse order.