Algo

DP :

Longest common subsequence (LCS)

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

sub : $\{1\}, \{2\}, \{3\}, \{4\}$

$\{1\}, \{2\}, \{2, 3\}$ .....

$\{1, 2\}, \{1, 3\}, \{1, 4\}, \{2, 3\}$ .....

$\{1, 2, 4\}, \{1, 2, 3\}$ ....

maintain onden sequence

$\{1, 4, 3\}$ → হবে না

step:

$lcs(\{\underline{1}, 2, 3, 4\}, \{\underline{1}, 2, 3\})$

$\Rightarrow 1 + lcs(\{\underline{2}, 3, 4\}, \{\underline{2}, 3\})$

$\Rightarrow 2 + lcs(\{\underline{3}, 4\}, \{\underline{3}\})$

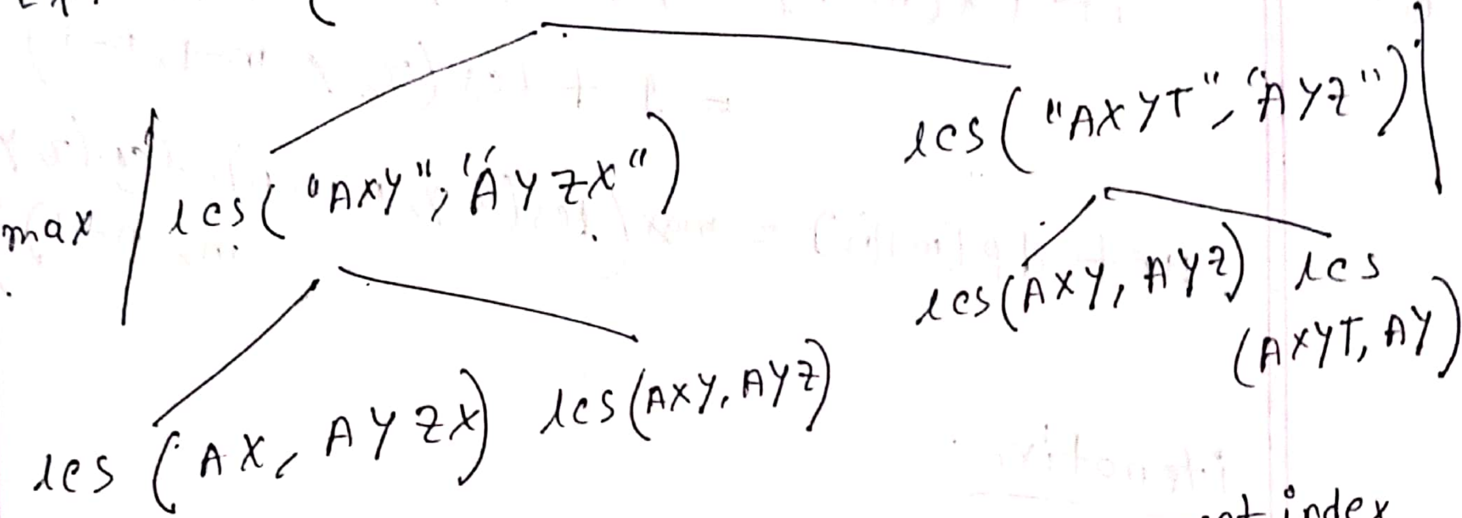
→ Sont
→ Fond
→ DP

Algo TT
8 dec

=) 3 + lcs({y}, { })

=) 3 + 0 ∴ length = 3

Ex: lcs ("AXYT", "AYZX")



lcs (x, y, m, n) { m → x string ~~size~~ current index
n → y " "

code:

```
if (m = 0
 || n = 0) return 0;
if (x[m] == y[n]) return 1 + lcs (x, y, m-1,
                                  n-1)
return max{lcs (x, y, m-1, n),
           lcs (x, y, m, n-1)} ; }
```

Time : O(2^n)

2D vecton [m] [n]

~~Top down~~ Bottom-up
(recursive)

Time:
O(m*n)

improvised code: if (m=0 || n=0) ret 0;

```
if (dp[m][n] != -1)
        return dp[m][n];
if (x[m-1] = y[n-1]) return dp[m][n]
        = 1 + lcs(x, y, m-1, n-1)
ret dp[m][n] = max(lcs(x, y, m-1, n), lcs(x, y,
                                          m, n-1));
```

iterative:

A	0	0	0	0	0
C	0	0	0	0	1
A	0	1	1	1	2
D	0	1	1	2	2
B	0	1	2	2	2