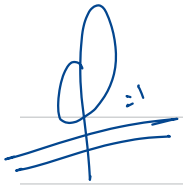
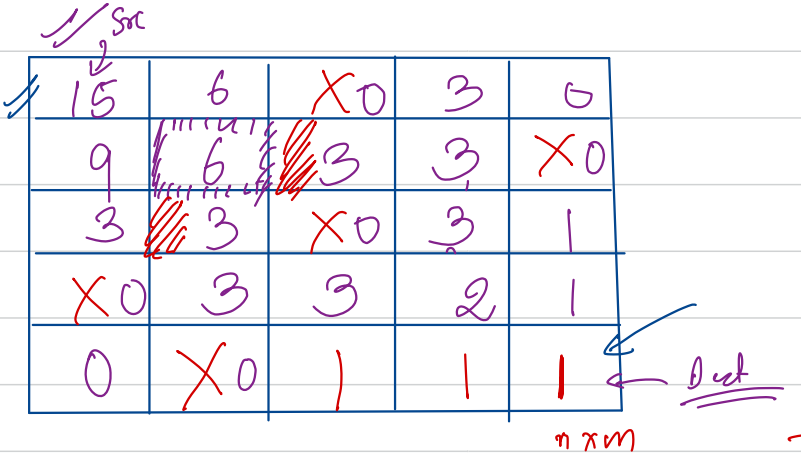
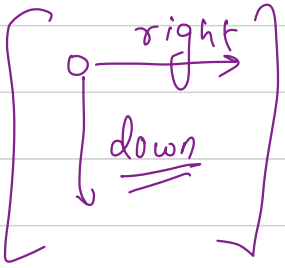




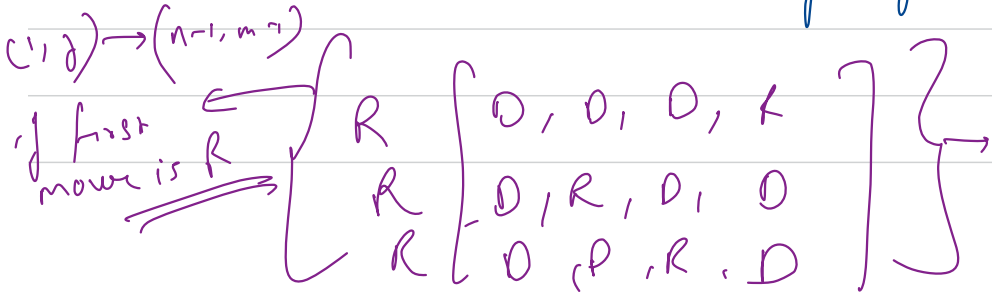


if src == dest

↳ # of ways → 1

$T(n) \rightarrow O(mn)$

$SC \rightarrow O(n)$



$f(i, j, n-1, m-1)$ = $f(i, j+1, n-1, m-1)$ + $f(i+1, j, n-1, m-1)$

TC → (Time reqd for one state × # uniq states)

func" returns # of ways to reach $(n-1, m-1)$ from (i, j) using right/down as the moves

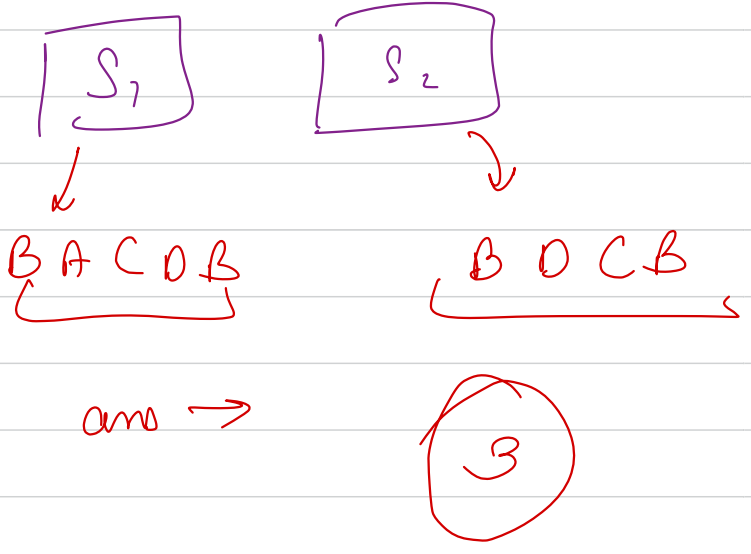
if (grid[i][j] is blocked)
return 0;

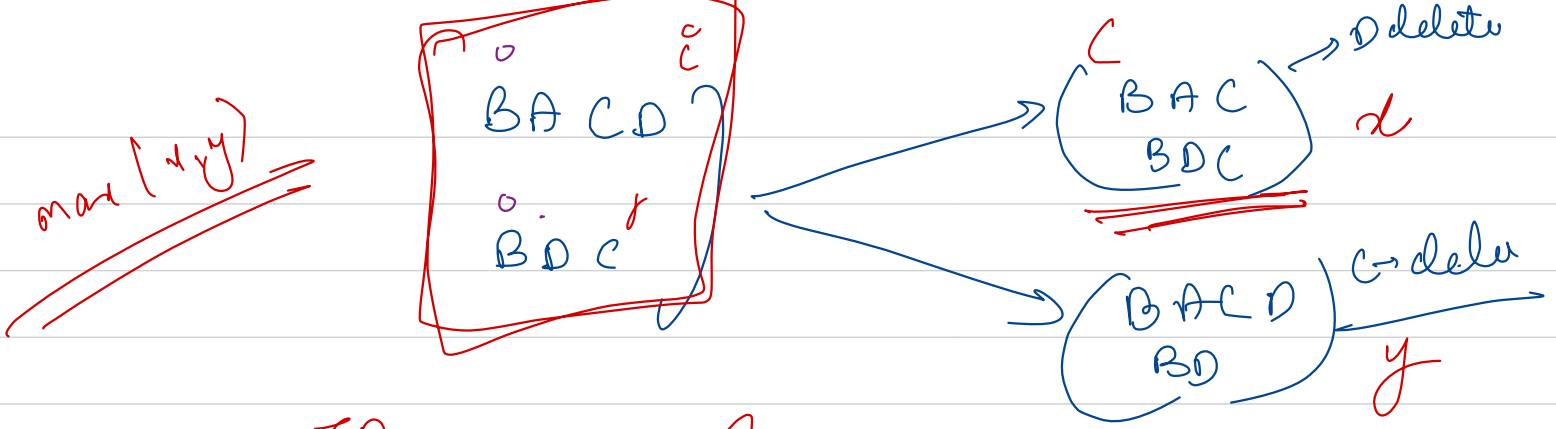
DP → $O(mn)$

$f(i, j) = f(i+1, j) + f(i, j+1)$

maximum → 2D dp

Q. LCS You are given 2 strings, & you need to find the longest common Subsequence.

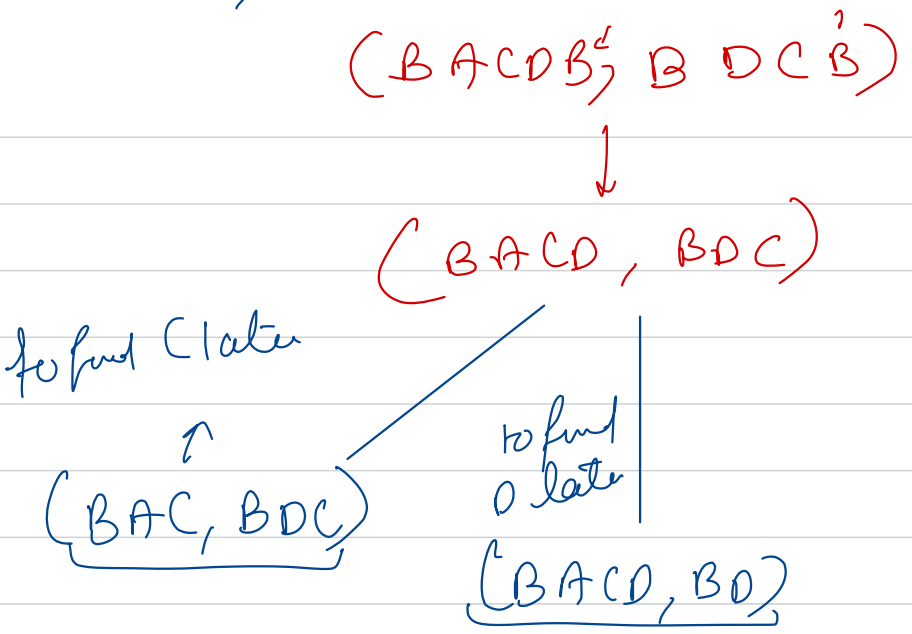




TD

$f(i,j)$ = func^n to calc LCS of $S_1[0,i]$ and $S_2[0,j]$

$$f(i,j) = \begin{cases} 1 + f(i-1, j-1) & \text{if } S_1[i] == S_2[j] \\ \max\left(f(i-1,j),\ f(i,j-1)\right) & \text{else} \end{cases}$$


(BACDB', BDCB')
(BACD, BDC)
to find C later
(BAC, BDC)
to find D later
(BACD, BD)

TD

$f(i,j)$ =

$1 + f(i-1, j-1)$ if $s_1[i] == s_2[j]$

$\max\left(f(i-1,j),\ f(i,j-1)\right)$ else

↓

func^n to calc LCS of $S_1[0,i]$ and $S_2[0,j]$

BACDB
BDCB

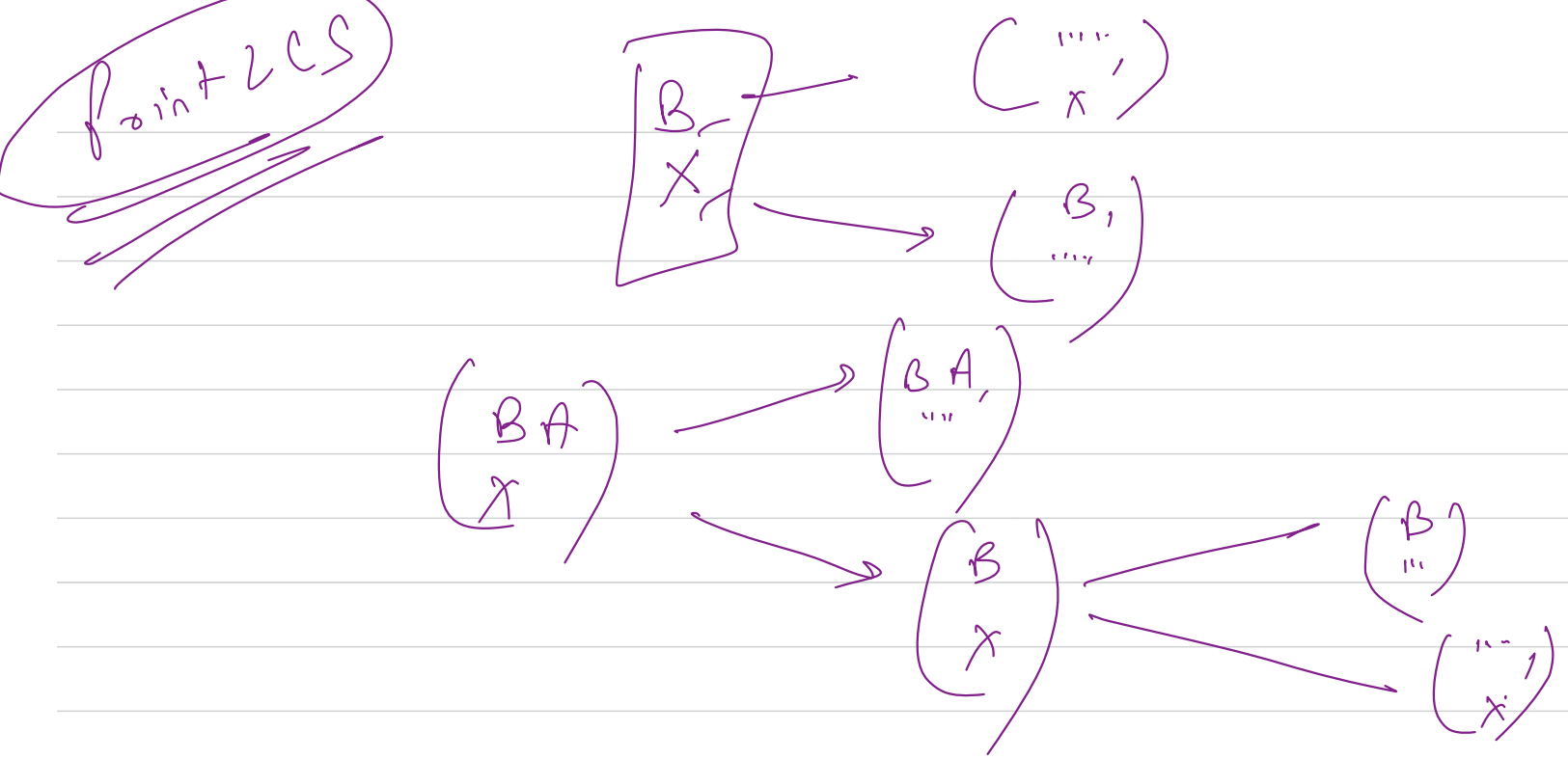
		""	B	D	C	B
		0	1	2	3	4
	"" 0	0	0	0	0	0
	B 1	0	1	1	1	1
	A 2	0	1	1	1	1
→	C 3	0	1	1	2	2
→	D 4	0	1	2	2	2
	B 5	0	1	2	2	3

$(m+1) * (n+1)$

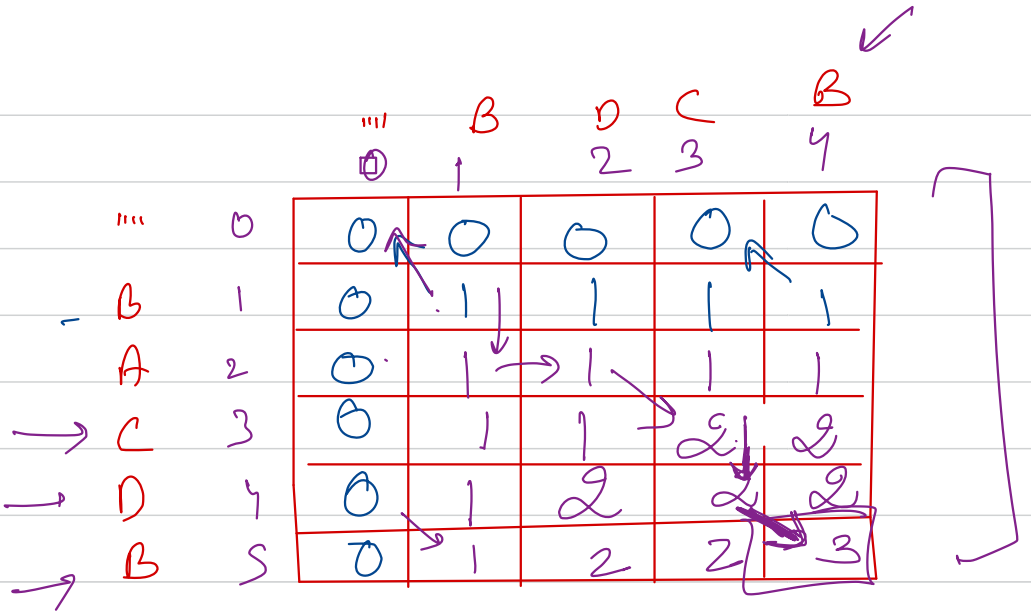
TC → $O(mn)$

SC → (m)

dp[m][n] → final ans



dp(i,j) → LCS of $S_1[0, i]$ and $S_2[0, j]$



O(nm)

BCB

Start from bottom right

ans = "B"

ans = "C"

ans = B

Q. Given 2 strings, find the largest common Substring.

B D C B

B A C B B

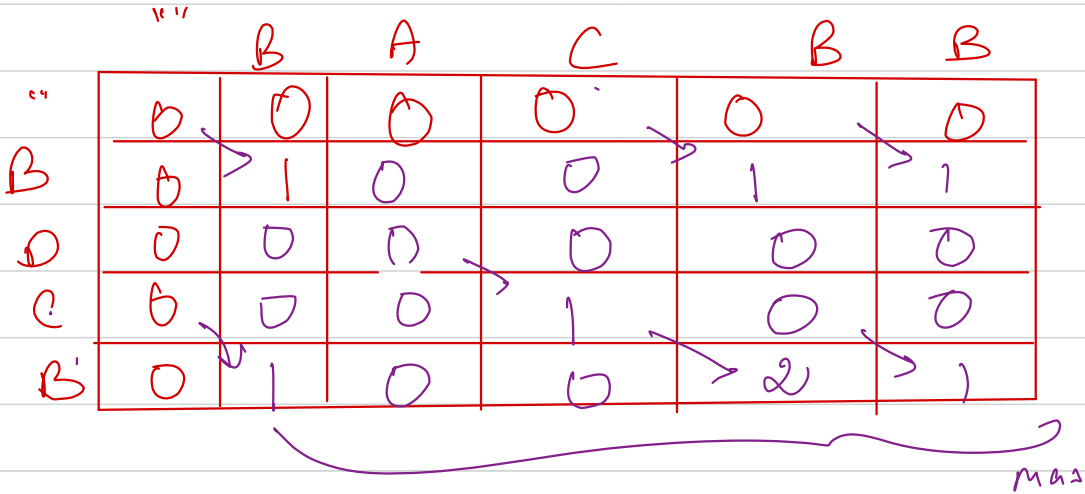
CB → 2

$$f(i,j) = 1 + f(i-1, j-1) \quad \text{if } S_1[i] == S_2[j]$$

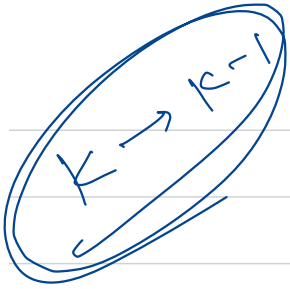
$$= 0 \quad \text{else}$$

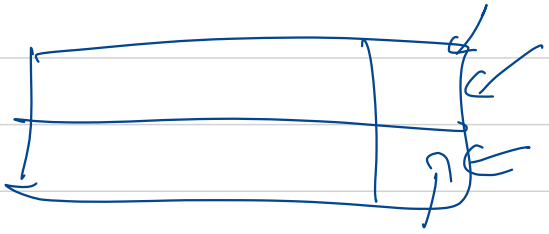
$$\max_{S_1[0,i],\ S_2[0,j]}$$

2D DP



$$f(i,j) = \begin{cases} 1 + f(i-1, j-1) & \text{if } s_1[i] == s_2[j] \\ 0 & \text{else} \end{cases}$$





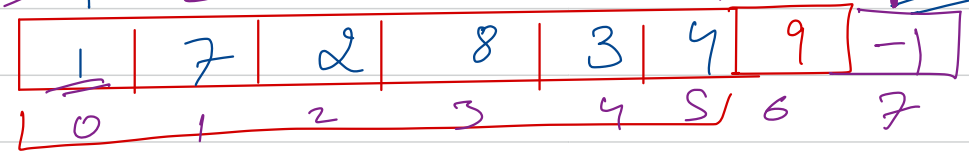
$$f(i,j,k) = 1 + f(i-1, j-1, k) \quad \text{if } s_1[i] == s_2[j]$$

$$\max\left(f(i-1,j,k),\ f(i,j-1,k),\ 1+f(i-1,j-1,k)\right)$$

$k \geq 0$

Q: Given an array find the longest inc subsequence

1, 7 dp → 1 2 2 3 3 4 5 LIS



{1, 2, 3, 4} → {1, 2, 3} + {4}

mark

$i \to f(i)$

lis, ending at i^{th} index element.

$v_j < v_i$

$dp[i] = 1 + dp[j]$

If is ending at index 1, it will be empty

from the index < 1