Que. : Unique Paths

(1) Why apply recursion?

⇒ Bcz we have to try all possible ways to get answer so ~~only recur~~ we use recursion.

(2) How to write Recurrance relection?

⇒ (i) Express in term of index here ~~to~~ is a 2D matrix so express in $(\underset{\uparrow \text{row}}{i}, \underset{\uparrow \text{col}}{j})$

(ii) Explore/Do all stuffs on that Grid.

(iii) If Question is count ways → sum up all the stuff

______ max → max(all the stuff)

______ min → min(all the stuff)

Take an example

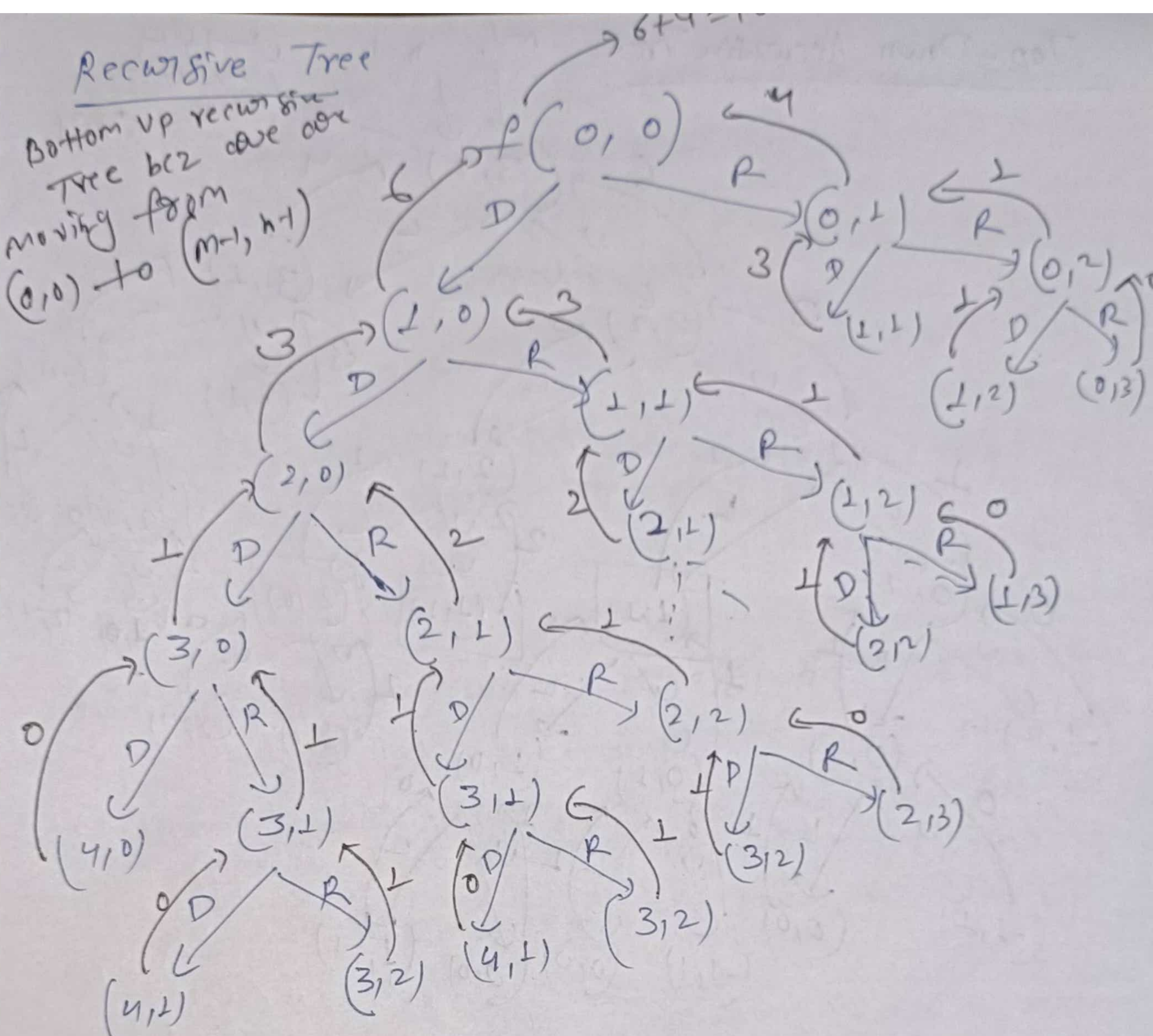
m=4, n=3

2D Grid / 2D Matrix

	0	1	2
0	Start (0,0)	(0,1)	(0,2)
1	(1,0)	(1,1)	(1,2)
2	(2,0)	(2,1)	(2,2)
3	(3,0)	(3,1)	Destination (3,2)

Recursive Tree

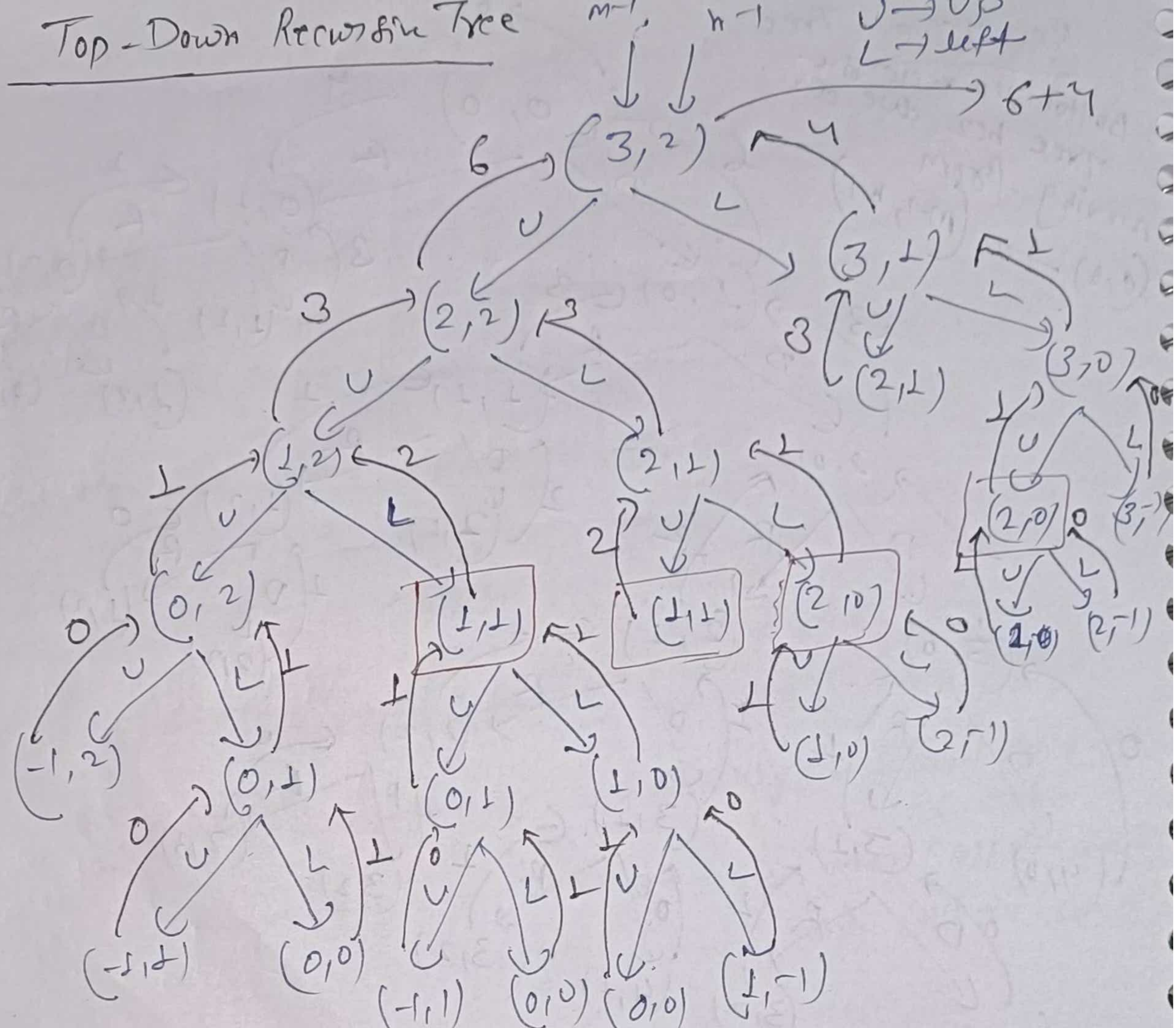
Bottom up recursive Tree bcz we are moving from (0,0) to (m-1, n-1)



(3,1) → return 1 that mean if we are at (3,1) grid & we want to reach (3,2) grid then their is only 1 unique path exist that's way (3,1) returning 1

f(x,y) → return P that mean from (x,y) grid to reach ((m-1), (n-1)) their are only P unique path exist.

Top-Down Recursion Tree



(0,2) → returning 1 that mean if we start from (0,0) & want to reat at (0,2) so their is only 1 unique path

Recursion tree में भी हम देख सकते हैं कि हम (0,0) से value return करते हुए ऊपर की तरफ जा रहे हैं।

Top-Down Approch :- it means first we are going Down till the base case & after that we computing result & ~~returnin~~ returning values & ~~go~~ go up wards.

In Recursive Tree we can saw that their are multiple overlaping subproblems so we can apply Dynamic programming.

① first Decide the Data structure.

So in order to apply DP first we have to use DP datastructure i.e. what we should use like 1D array, 2D array, map or something else.

⇒ यहाँ हम 1D array use नहीं कर सकते bcz 1D array तब use करते हैं जब only one ~~vari~~veraible change होता है क्योंकि उस 1 veraible की सभी valuse के corrosponding ans को हम 1D array में store कर सकते हैं।

But here are two veraibles m & n are changing simultaneously so we have to use 2D array so we can store the ans corresponding to (m, n) in this 2D array

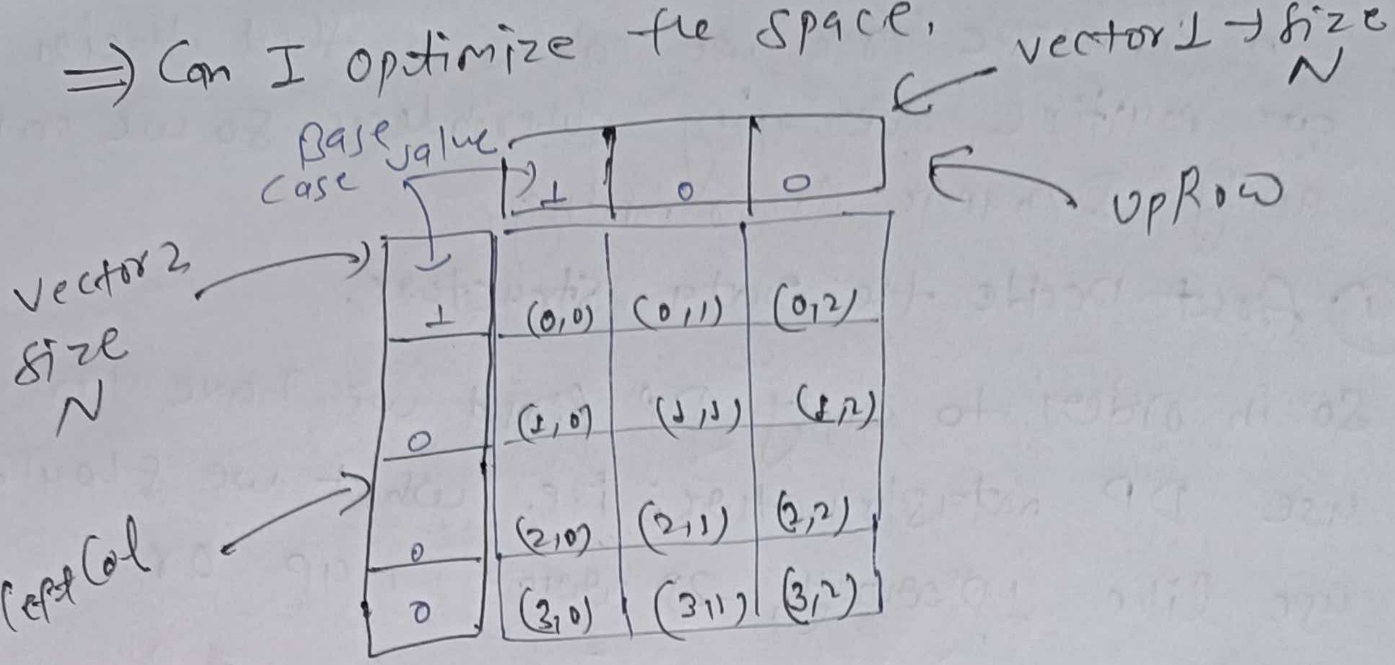
② Decide the size of the data structure.

→ 2D array → row
→ col

row → 0 to m & col = 0 to n

DP[m][n]

⇒ Can I optimize the space,



In tabulation method

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

In the most cases → $dp[i][j]$ Depend on → same column previous row $dp[i-1][j]$

" → same row previous col $dp[i][j-1]$

So Rather than creating a M x N DP we create two vector of size M & N

Suppose you want to find (i, j) grids unique path

```
        int down = upRow[j]; // dp[i-1][j]
        int Right = leftCol[i]; //dp[j][i-1]

   upRow[j] = leftCol[i] = down + Right
```