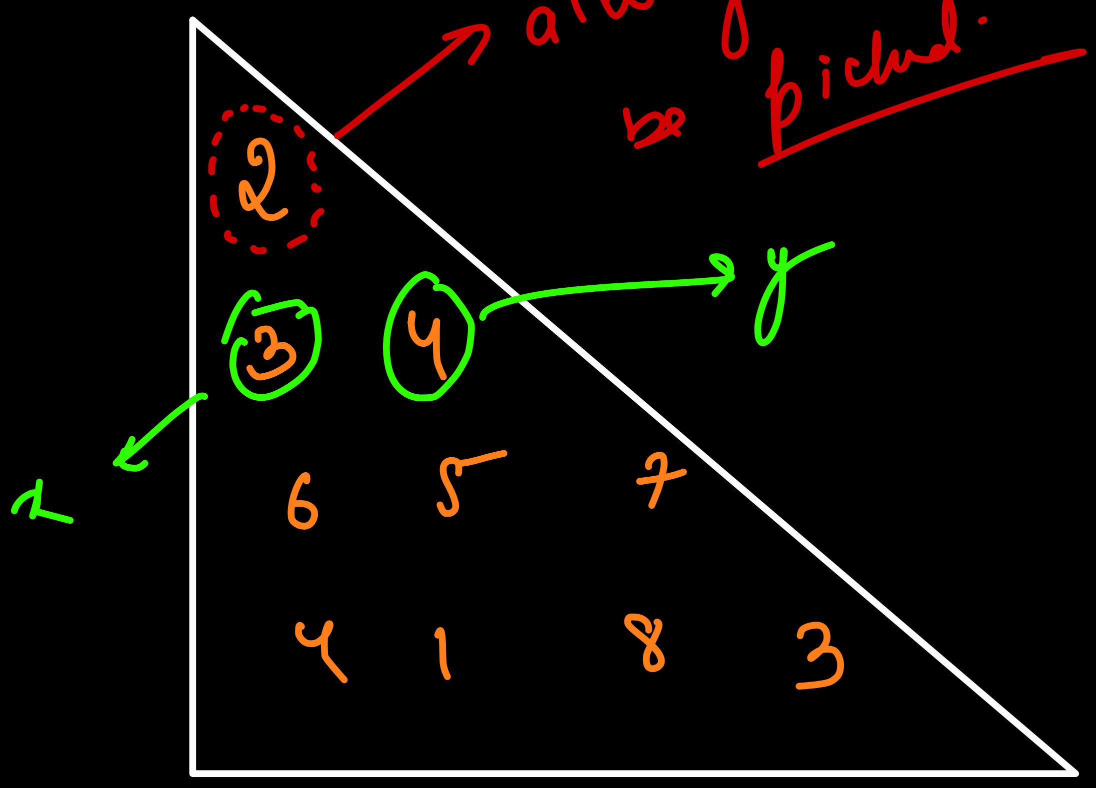


always need to be picked.

row, i

$\downarrow$

row+1, i

$\rightarrow$ row+1, i+1

min path sum

current row — i^{th} index

$\downarrow$ next row — i^{th} index

$\rightarrow$ $(i+1)^{th}$ index

from every row we pick one element

x → min path sum from ③

y → min path sum from ④

ans → $2 + \min(x, y)$

$$f(row, col) = mat[row][col] + \min \begin{cases} f(row+1, col) \\ f(row+1, col+1) \end{cases}$$

↓

function returns the min path sum if we start from (row, col)

Base Case

if (row == lastRow)

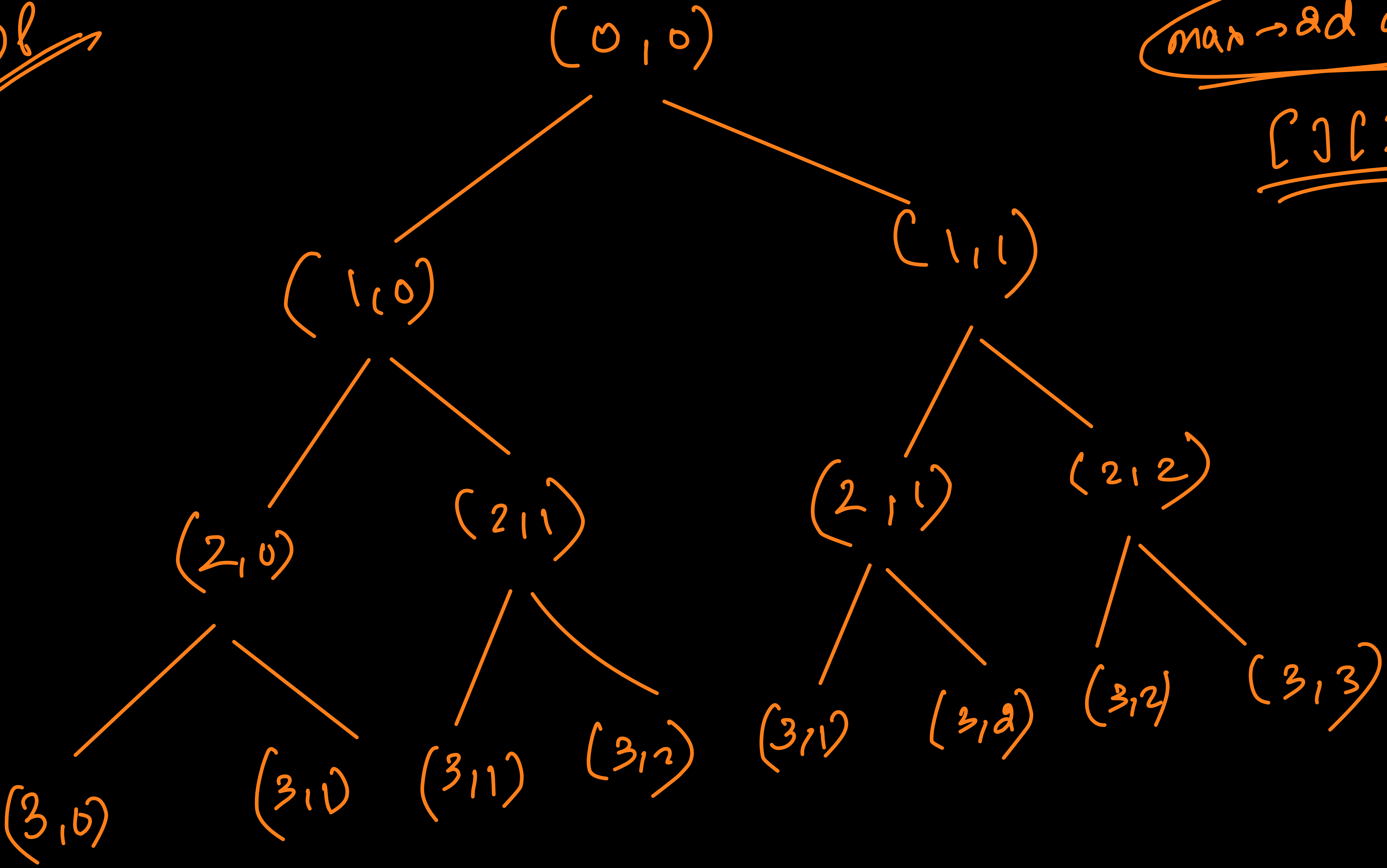
return mat[row][col]

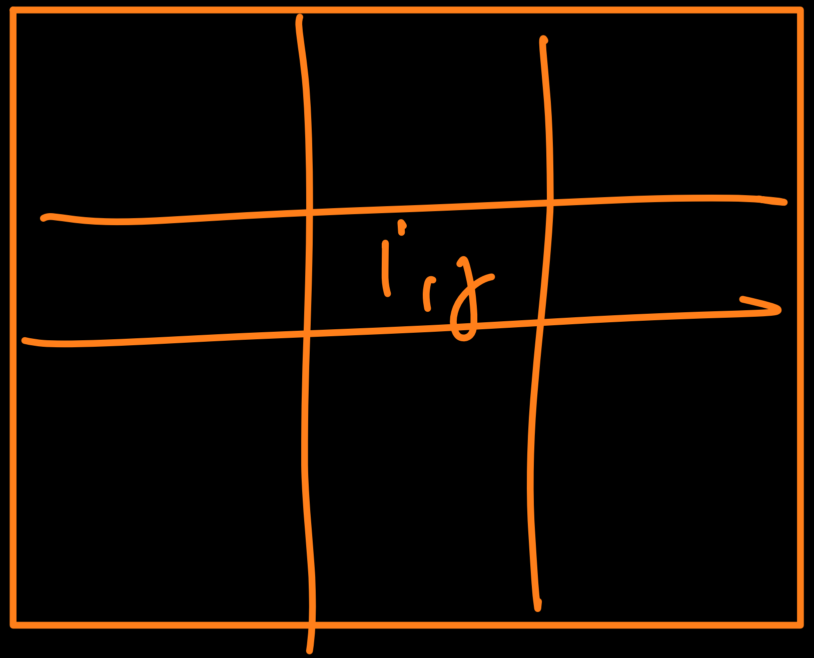
ans → $f(0,0)$

DP

max → 2d dp

[ ] [ ]



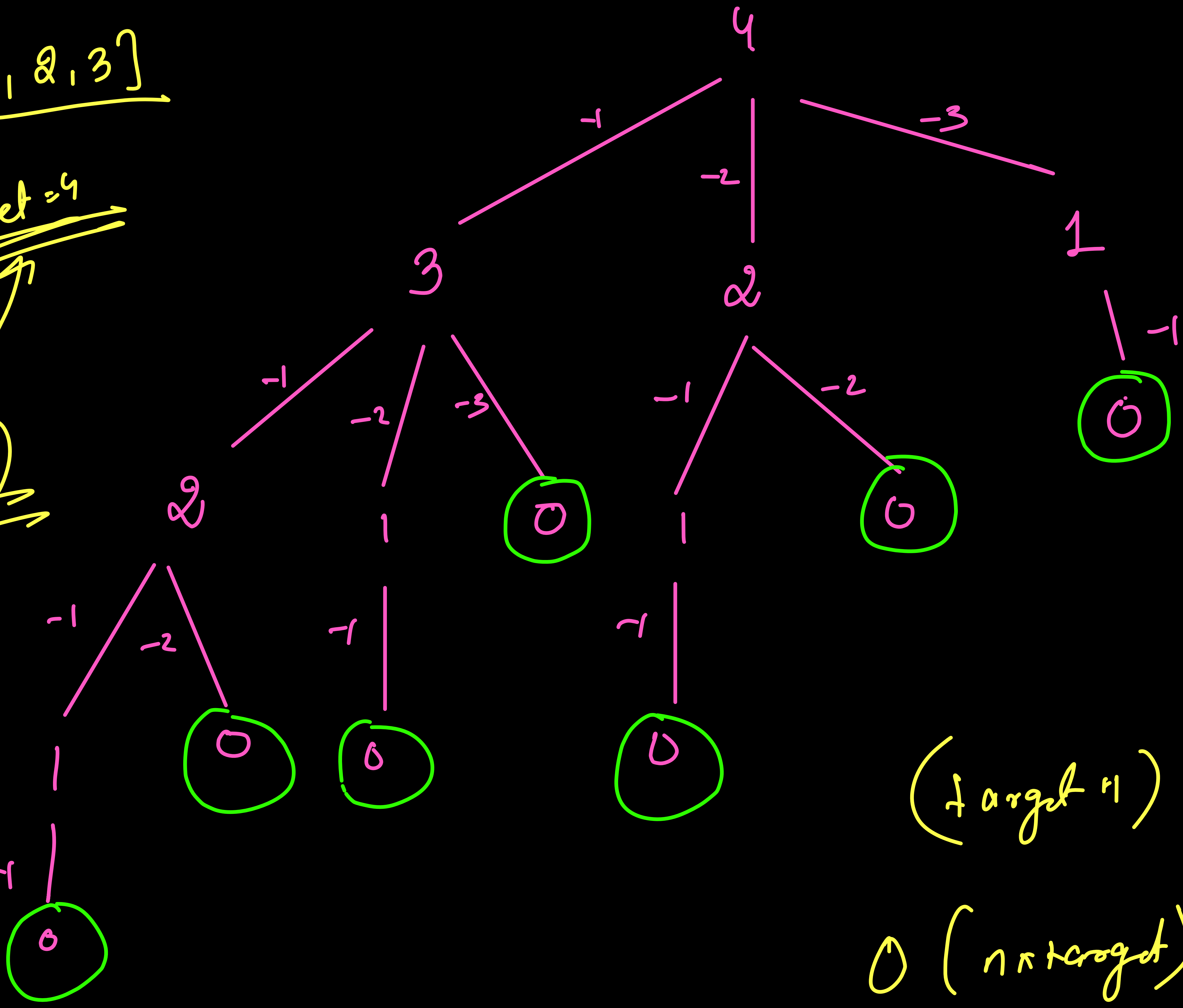


$dp[i][j] \rightarrow \underline{\underline{f(i,j)}}$

[1, 2, 3]

target = 4

f(4)



(target+1) × n

O(n×target)

$$f(\text{target}) = f(\text{target} - arr[0]) + f(\text{target} - arr[1]) + \vdots\ f(\text{target} - arr[n-1])$$

↓

returns no. of ways to prepare combinations for the given target

$$f(\text{target}) = \sum f(\text{target} - \text{arr}[k])$$

1d

$$k \in [0, n-1] \text{ and } \text{target} - \text{arr}[k] \geq 0$$

Base Case

$$(\text{target} == 0) \longrightarrow \text{ans} \to 1$$

Subsequen → continuity doesn't matter but Relative order matters

[1, 2, 3, 4]

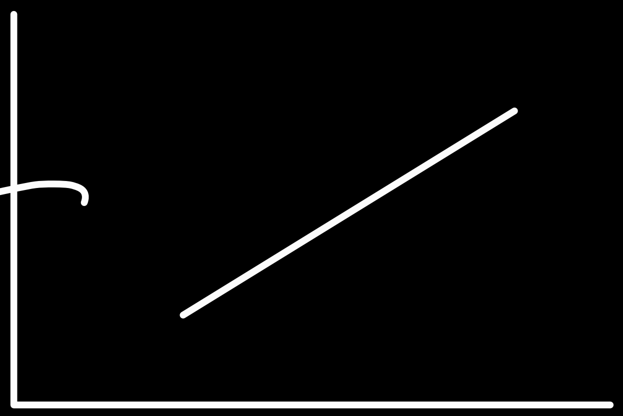
(1, 4) (3, 4)

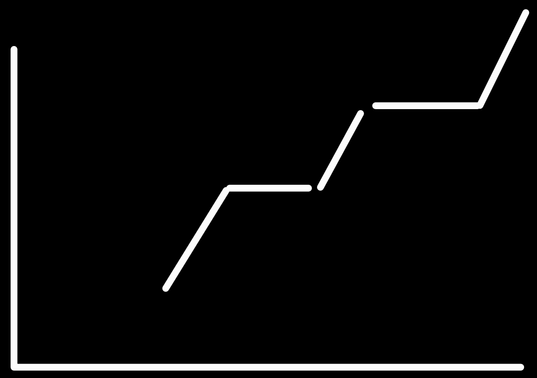
(1, 3) (2, 3, 4)

But (4, 1) ✗

LIS

Strictly inc





[1 , 3 , 5 , 4 , 7]

Longest Inc subseq

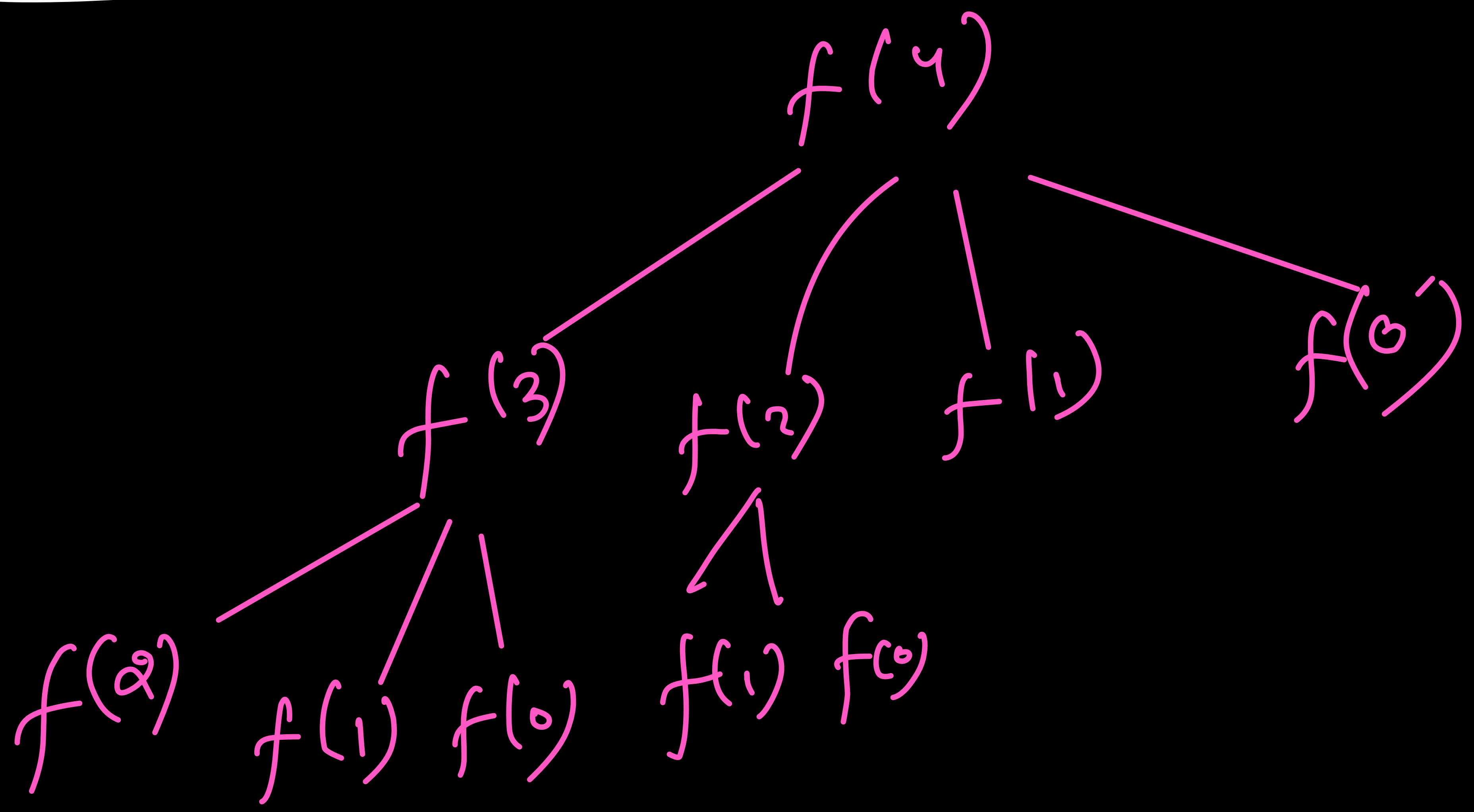
We figure out length of the l.i.s ending at index i.

Brute force → To generate all possible subsequences

$(1, 2, 3)_n \longrightarrow \boxed{2^n}$

(1) (3) (1, 3) (1, 2, 3)

(2) (1, 2) (2, 3) (...)



j j   i

[1, 3, 5, 4, 7]

0 1 2 3 4

lis → count

0	1	2	3	4
1	2	3	3	4

→ lis

0	1	2	3	4	5
	1	1	2	2	

→ count

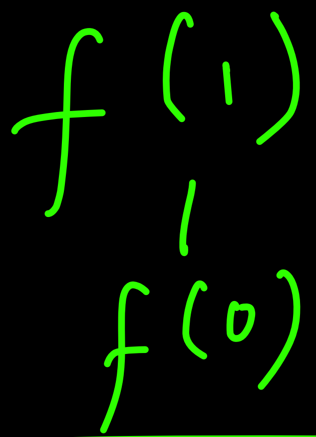
$$f(i) = \max(1 + f(j))$$

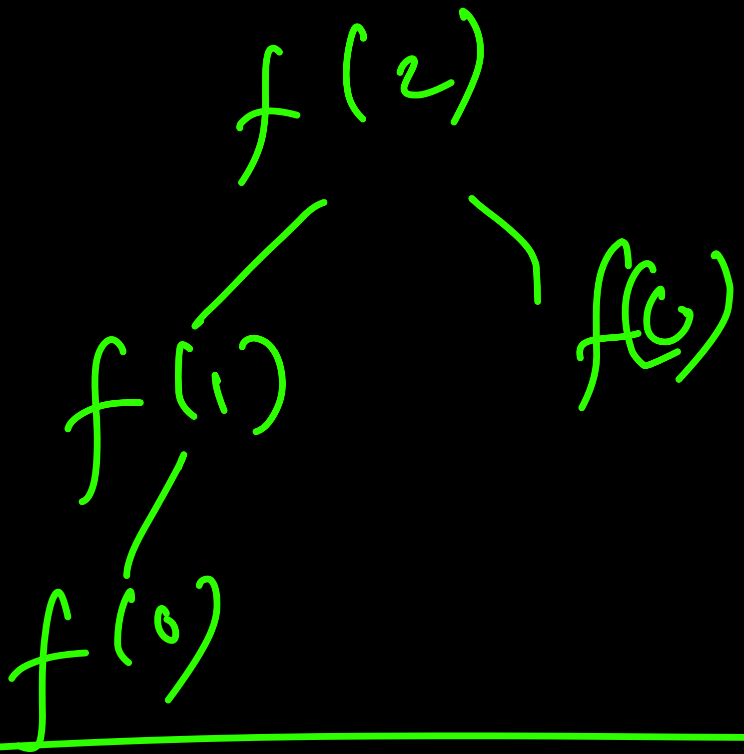
length of lis ending at i

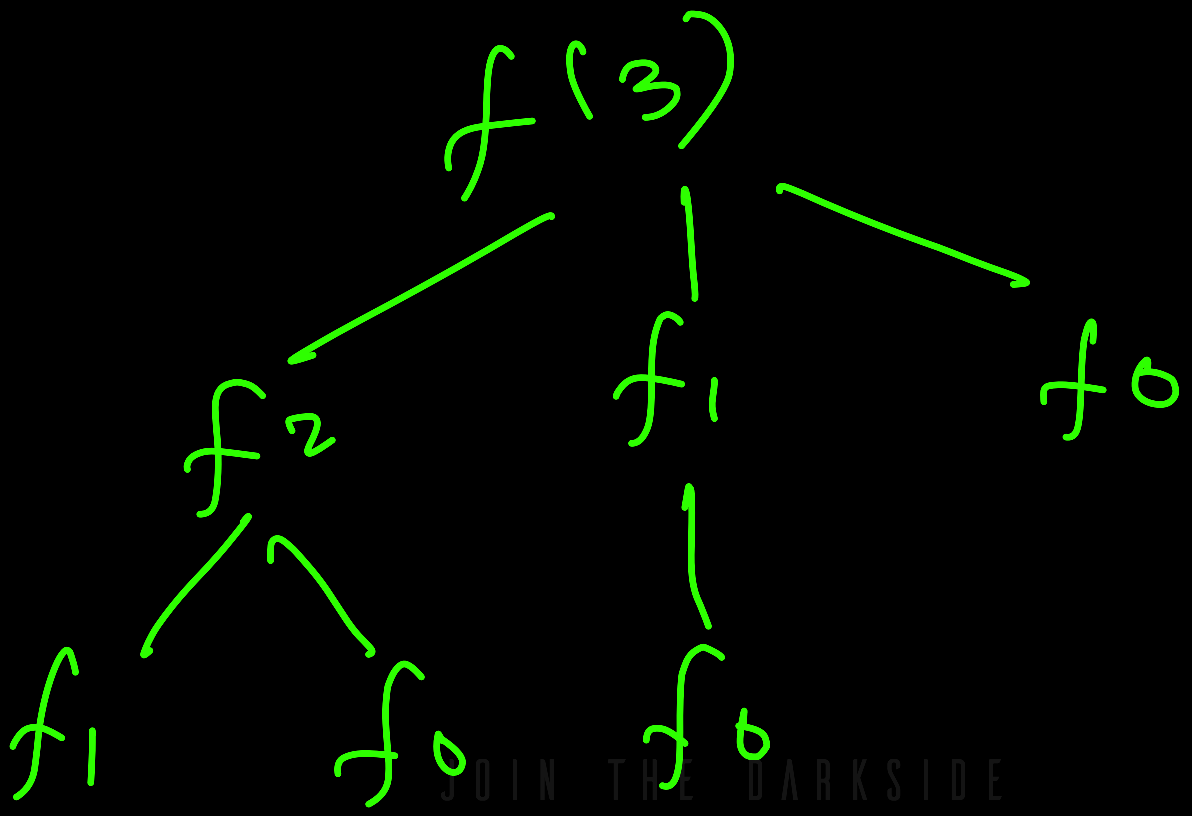
$\forall j \in [0, i-1]$ and $arr[j] < arr[i]$

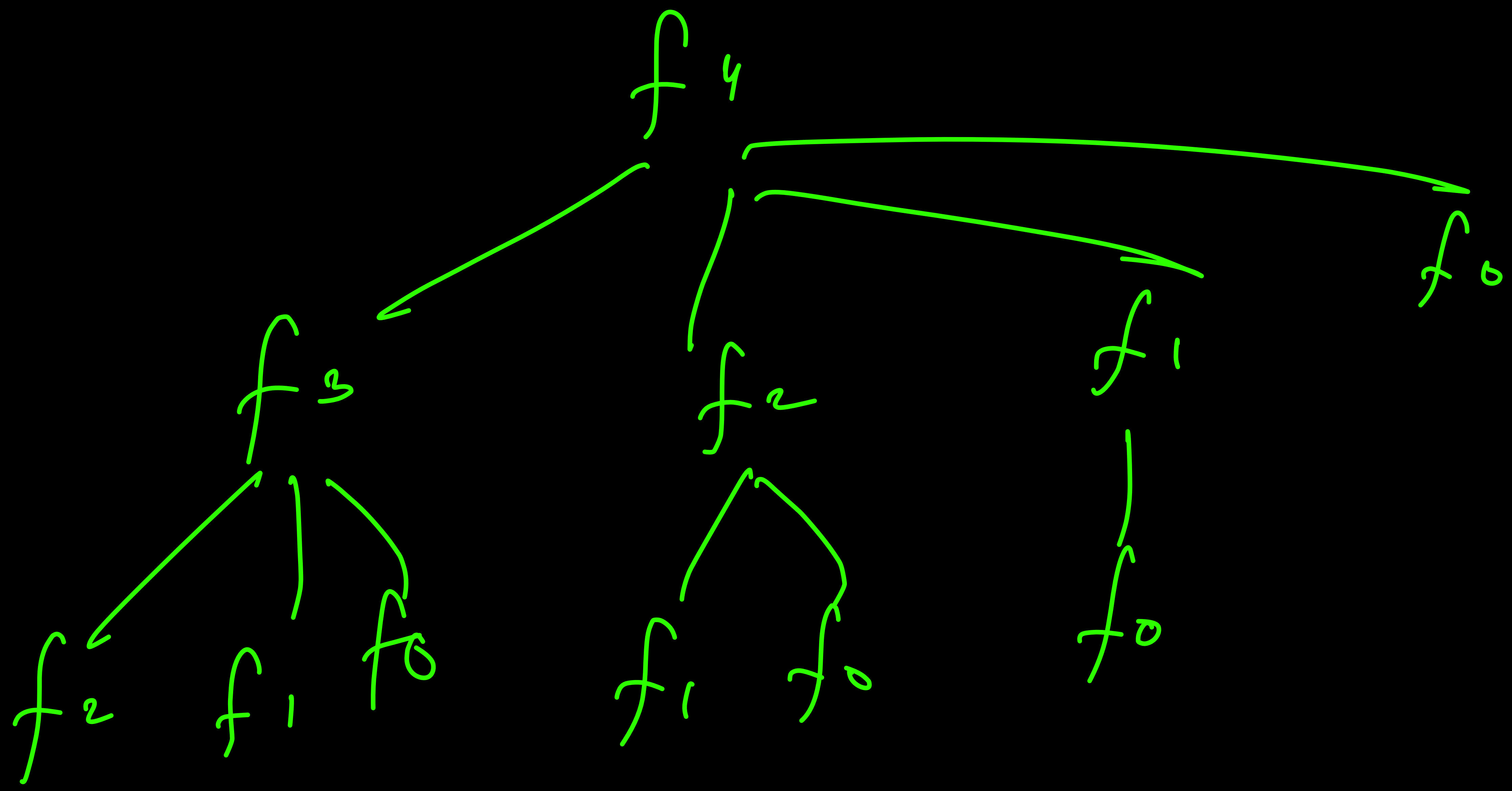
count(i) → no of lis of length i

ans → $\max(f(i)) \quad \forall i \in [0, n-1]$


f(1)
f(0)


f(2)
f(1)
f(0)
f(0)


f(3)
f2
f1
f0
f1
f0
f0


f4
f3
f2
f1
f0
f2
f1
f0
f1
f0
f0