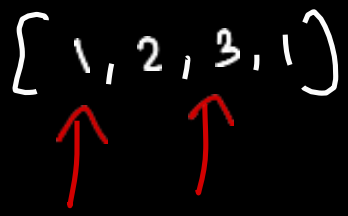
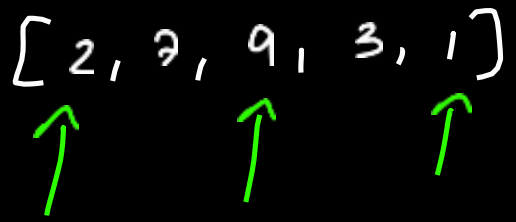
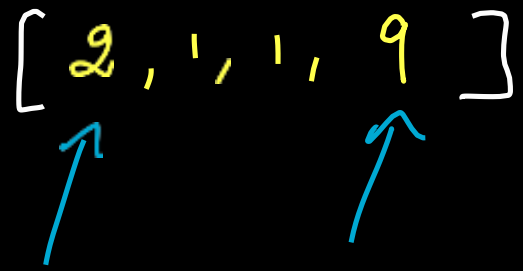
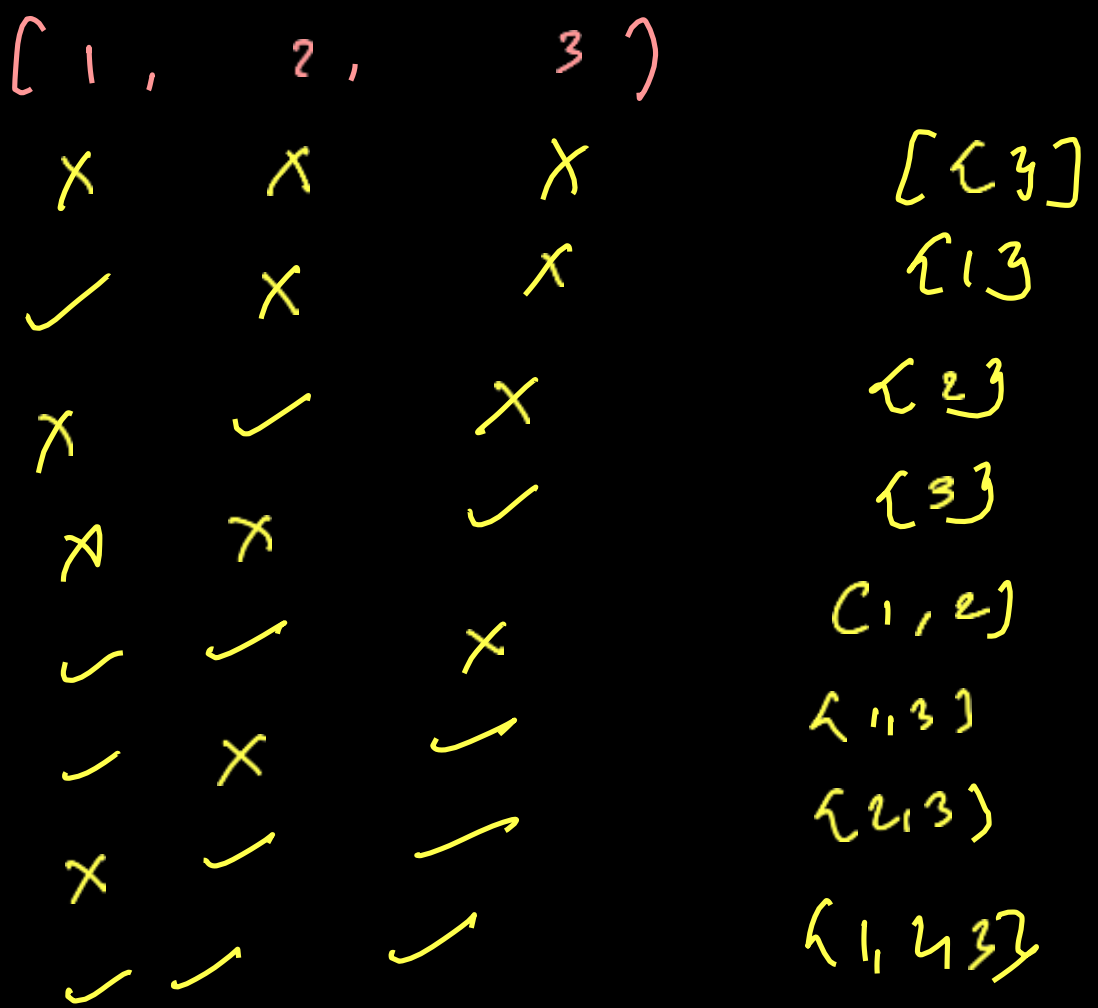
PW SKILLS

Dynamic Programming 1 (Part 2)





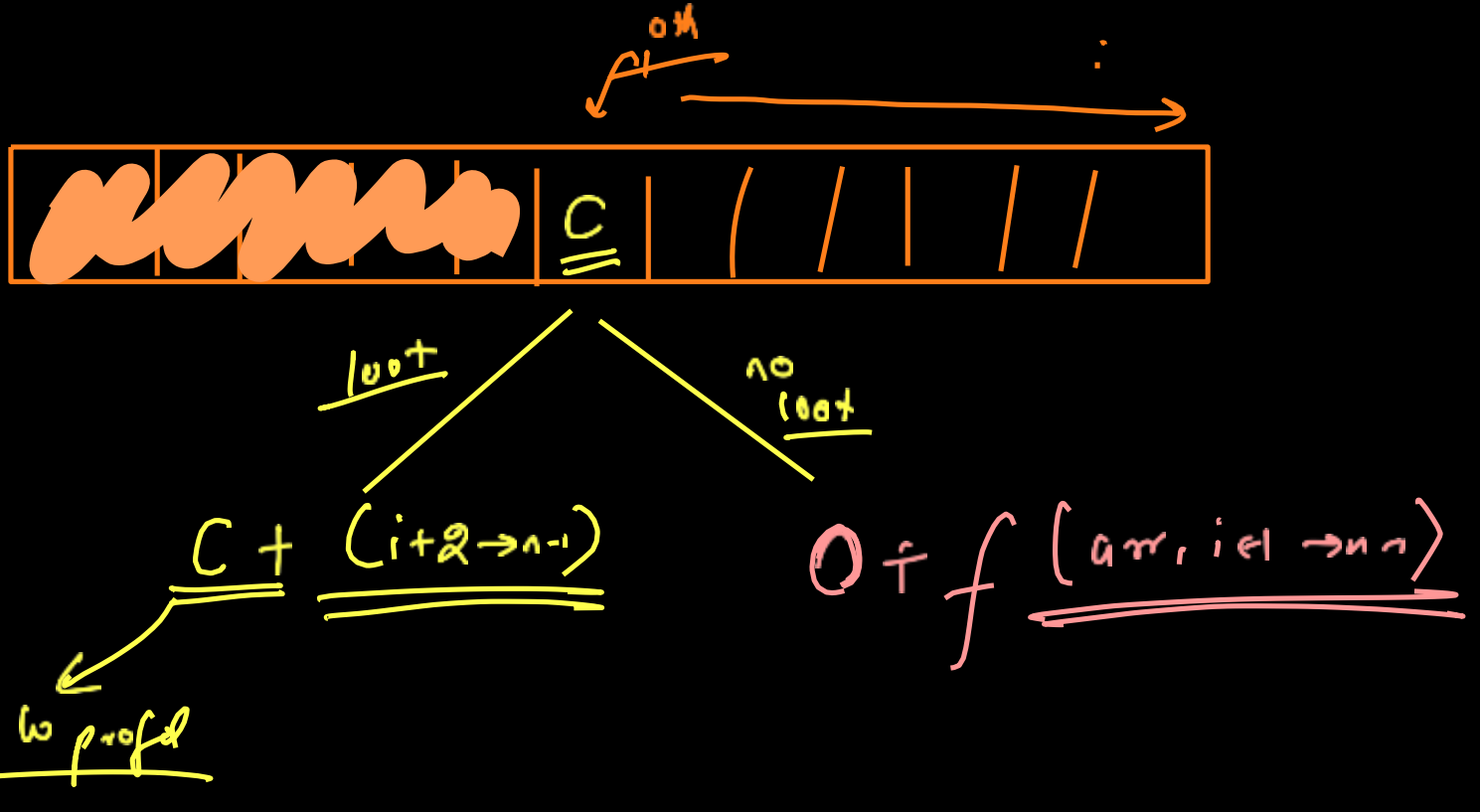




→ we have a lot of ways to decide -

[1, 2, 3] → 2^3

Brute force → try all possibilities of that



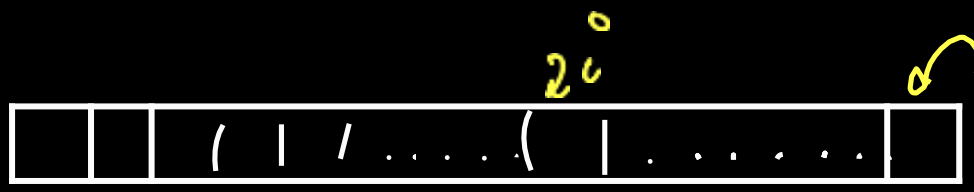
$f(arr, i) = \max \begin{cases} arr[i] + f(arr, i+2) \\ 0 + f(arr, i+1) \end{cases}$

you decide to rob the ith house

you don't rob the ith house

This recursive func^n returns max profit by looting houses from index i to n-1.

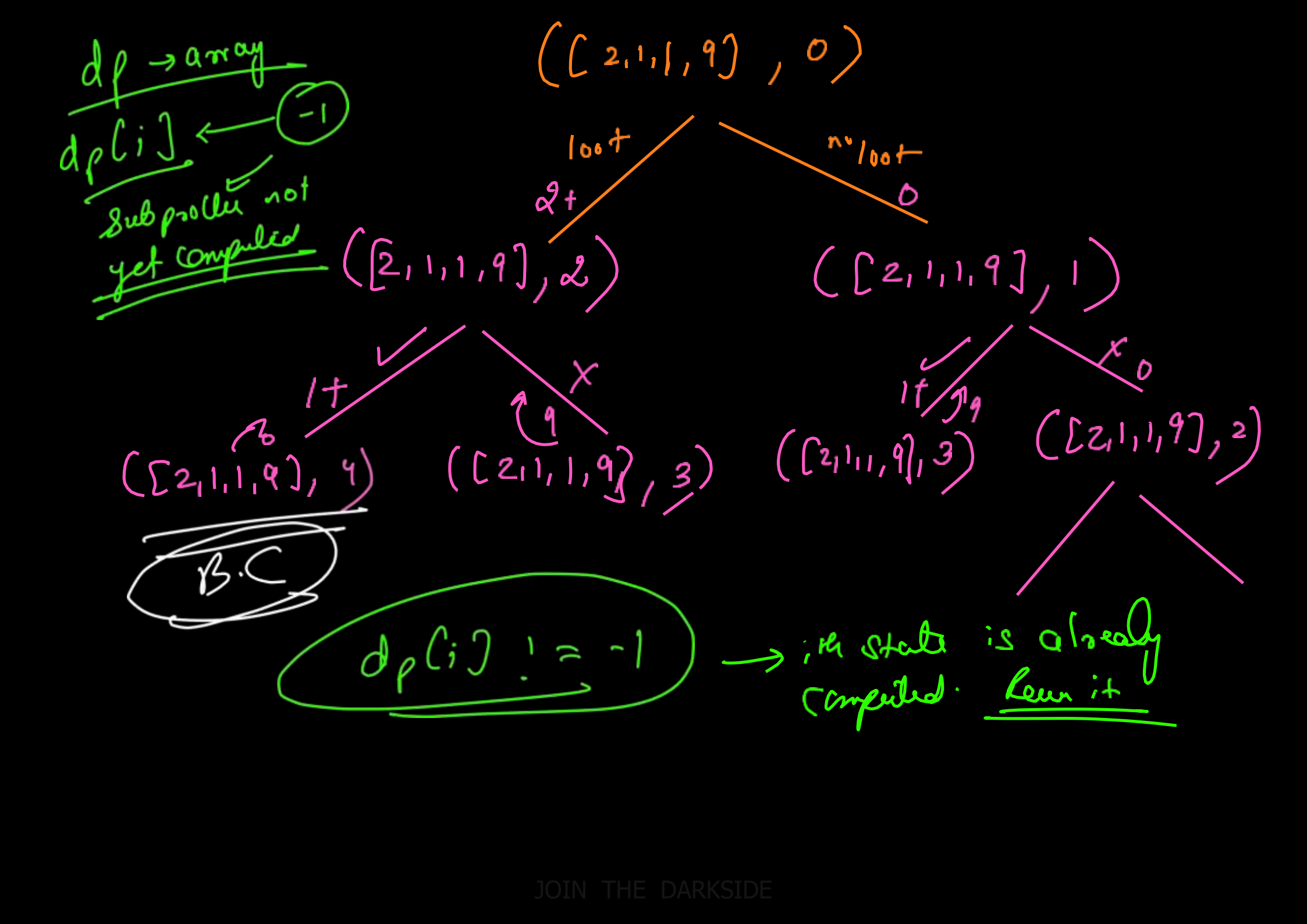
Such that no 2 adjacent houses are looted.



final ans → $f(arr, 0)$ ← (call from main)

BC

```
if (i == n-1)
    ↳ steal the
    return arr[i];
if (i == n-2)
    max(arr[i], arr[i+1]);
```



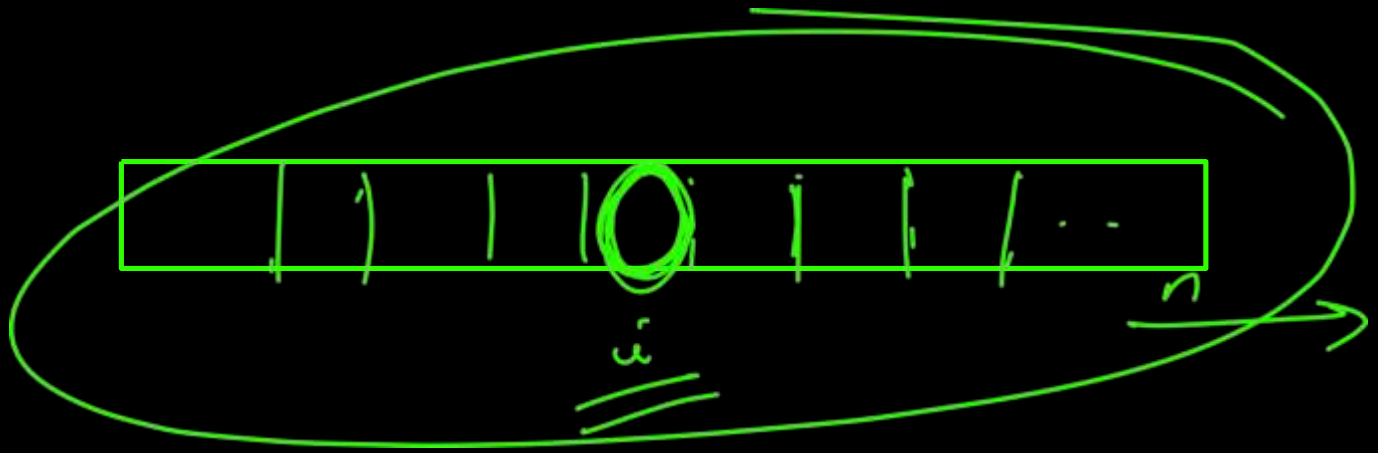
State of the dp → a set of all the parameters using which we can identify a subproblem uniquely.

How many unique subproblem could be there ??

↳ no. of subproblems depend on i. $[0, n-1]$

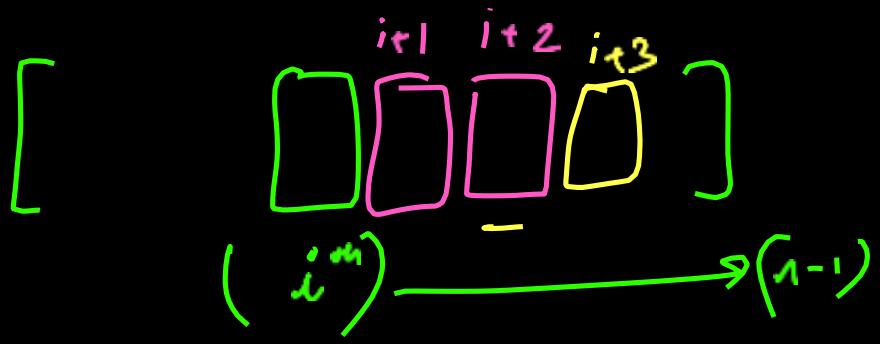
↳ Total n unique subproblems

1 variable → 1 Dimensional dp → 1d array



Bottom Up

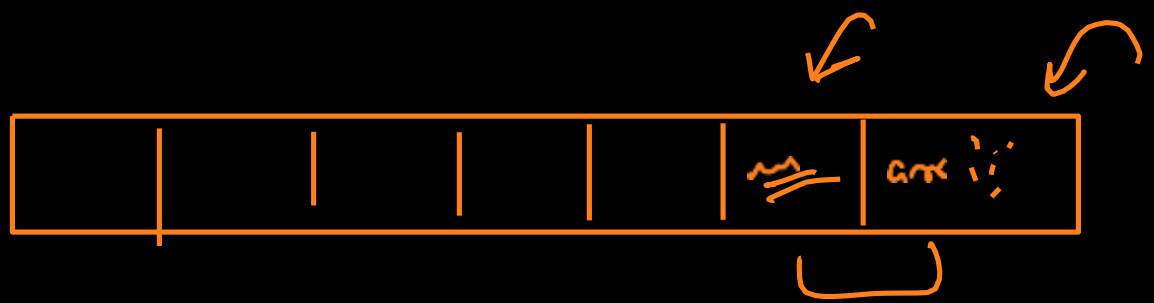
iterative



to calculate ans of any ith state, we need to find ans of all the state $\geq i$

$$f(arr, i) = \max \begin{cases} f(arr, i+1) \\ arr[i] + f(arr, i+2) \end{cases}$$

$$dp(i) = \max \begin{cases} dp(i+1) \\ arr[i] + dp(i+2) \end{cases}$$



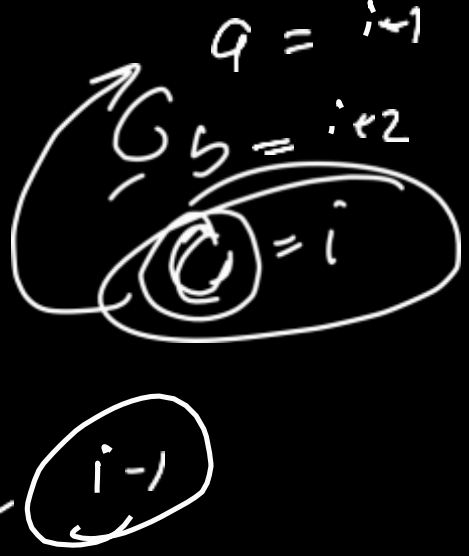
[2, 7, 9, 3, 1]

$f(arr, i)$

profit in $[0, i]$

12	10	10	3	1
i			$n-2$	$n-1$

$$dp(i) = \max \begin{cases} dp(i+1), \\ dp(i+2) + arr[i] \end{cases}$$



final ans → $dp(0)$

$$f(arr, i) = \max \begin{pmatrix} arr[i] + f(arr, i-2) \\ f(arr, i-1) \end{pmatrix}$$

if $(i == 0)$ return $arr[0]$

if $(i == 1)$ $\max(\_\_)$

Q → Given a no. n, you can perform any of the following ops on it some no. of times.

(1) → Reduce n to n-1

(2) → if n is divisible by 2 to make it n/2

(3) → if n is divisible by 3 make it n/3

find out in how many minimum steps you can reduce n to 1.

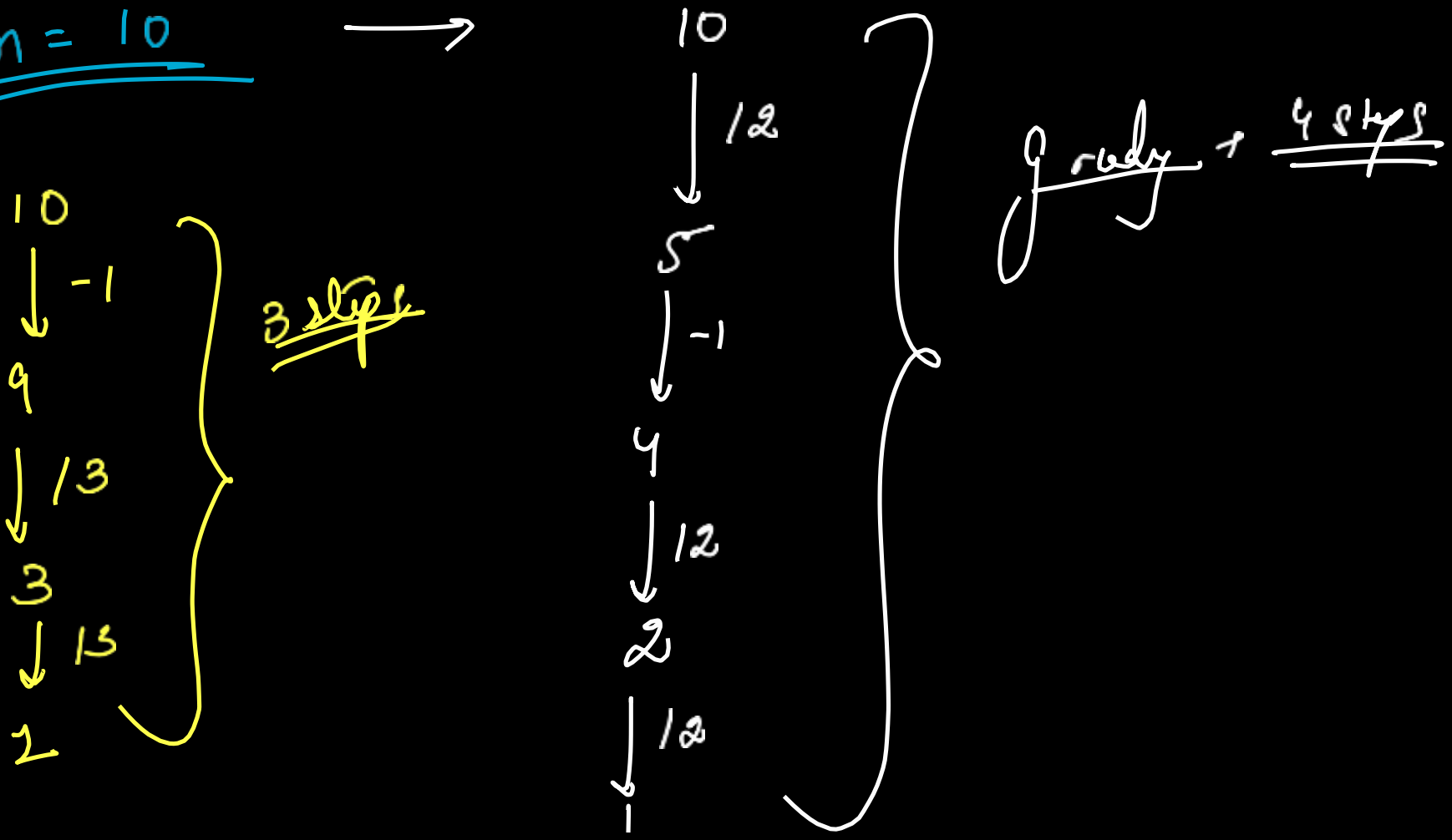
Ex → n = 7

ans → 3

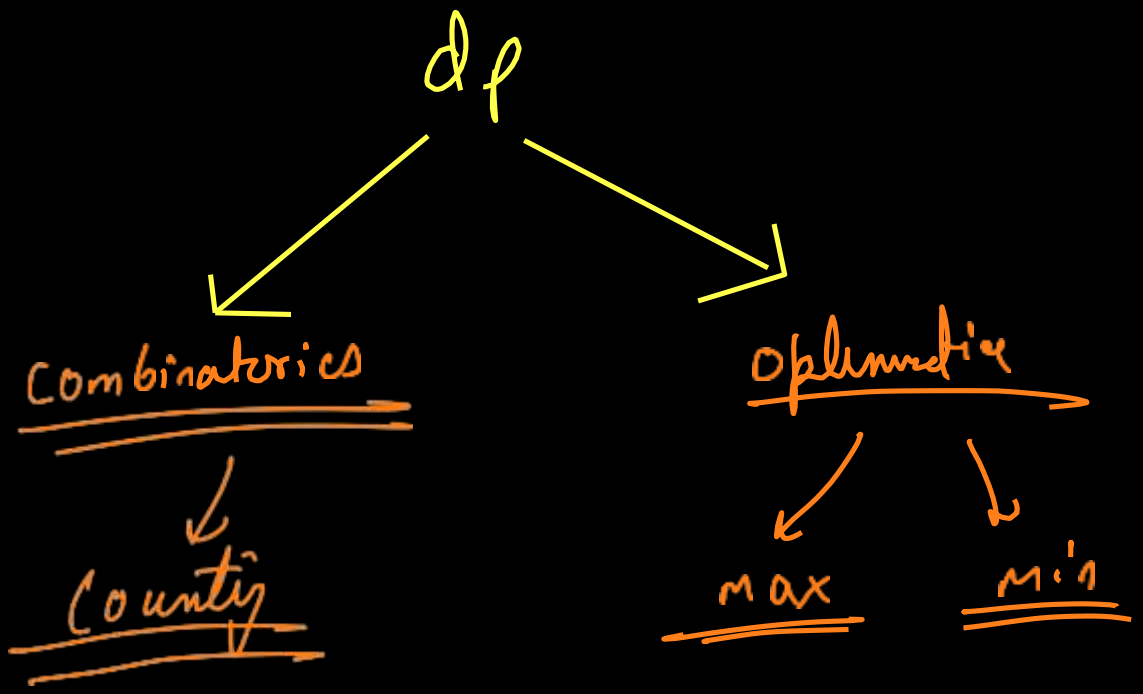
7
↓ -1
6
↓ /2
3
↓ /3
1

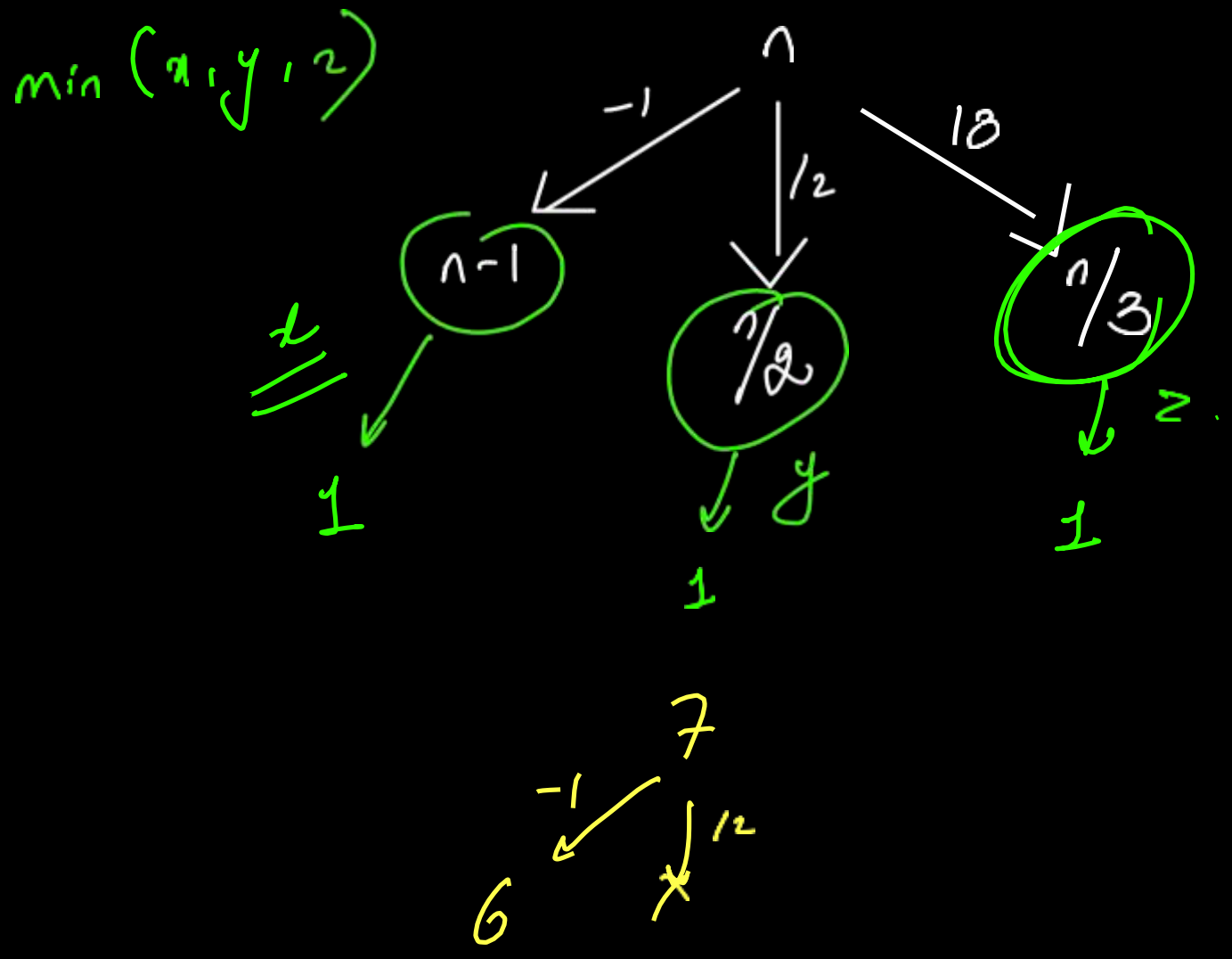
greedy will give wrong ans.

n = 10 →



↳ dp → optimization technique





$f(n)$ = 1 + min { $f(n-1)$, $(n\%2==0)\ ?\ f(n/2) : \infty$, $(n\%3==0)\ ?\ f(n/3) : \infty$ }

)

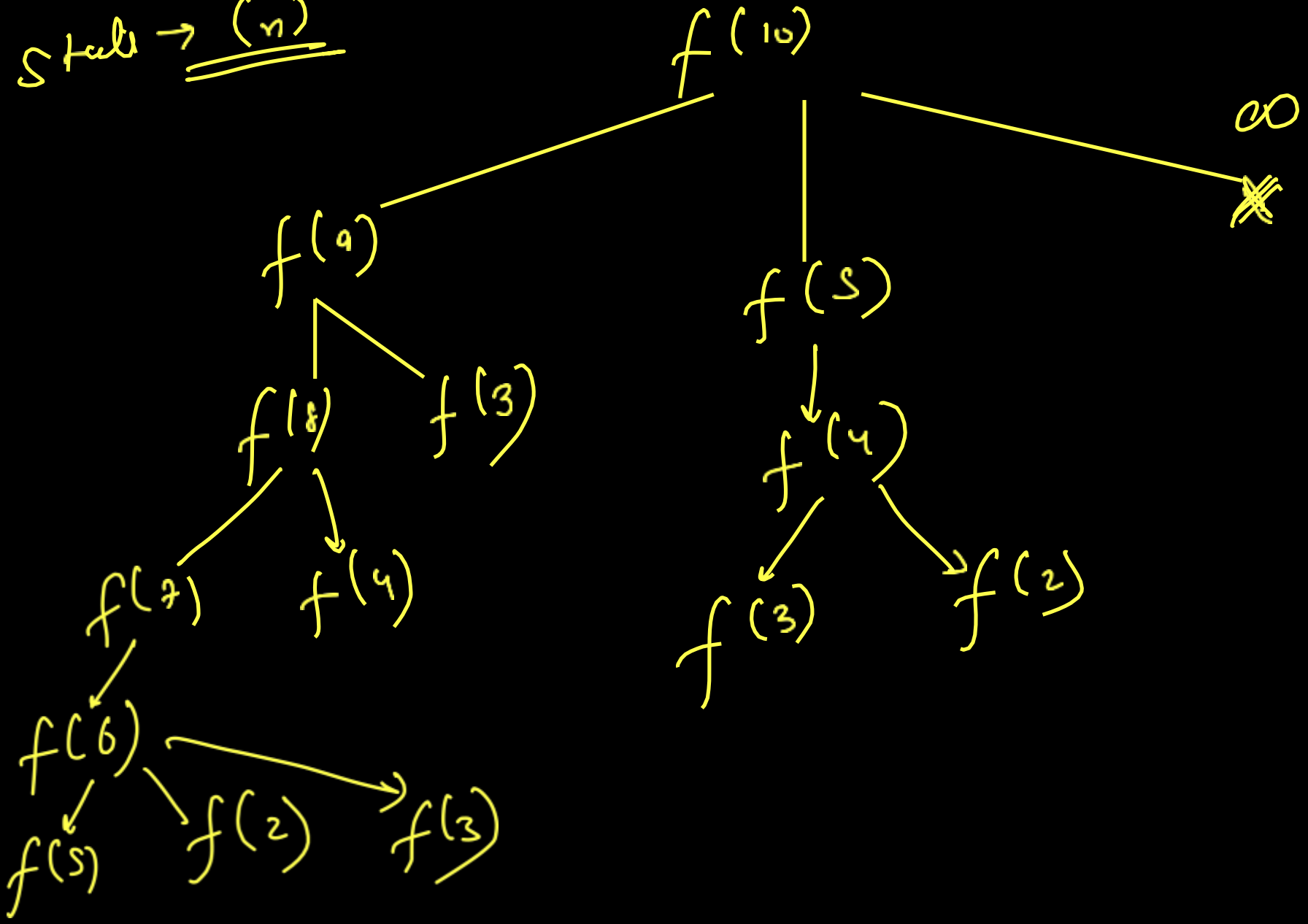
min steps to reduce n to 1.

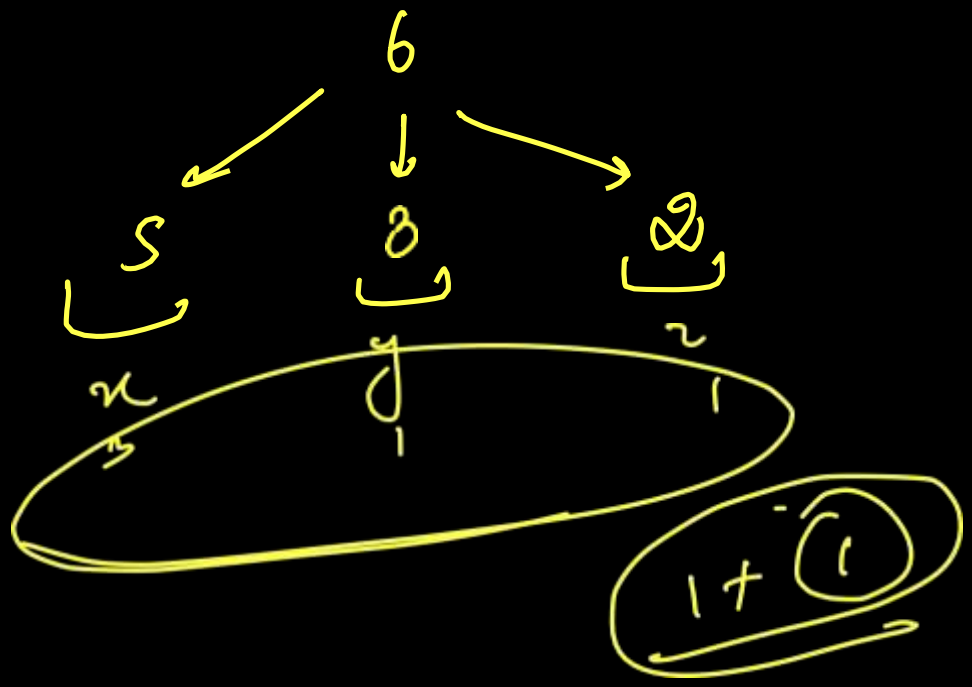
Base Case → $n==1 \rightarrow 0$

$n==2 \rightarrow 1$

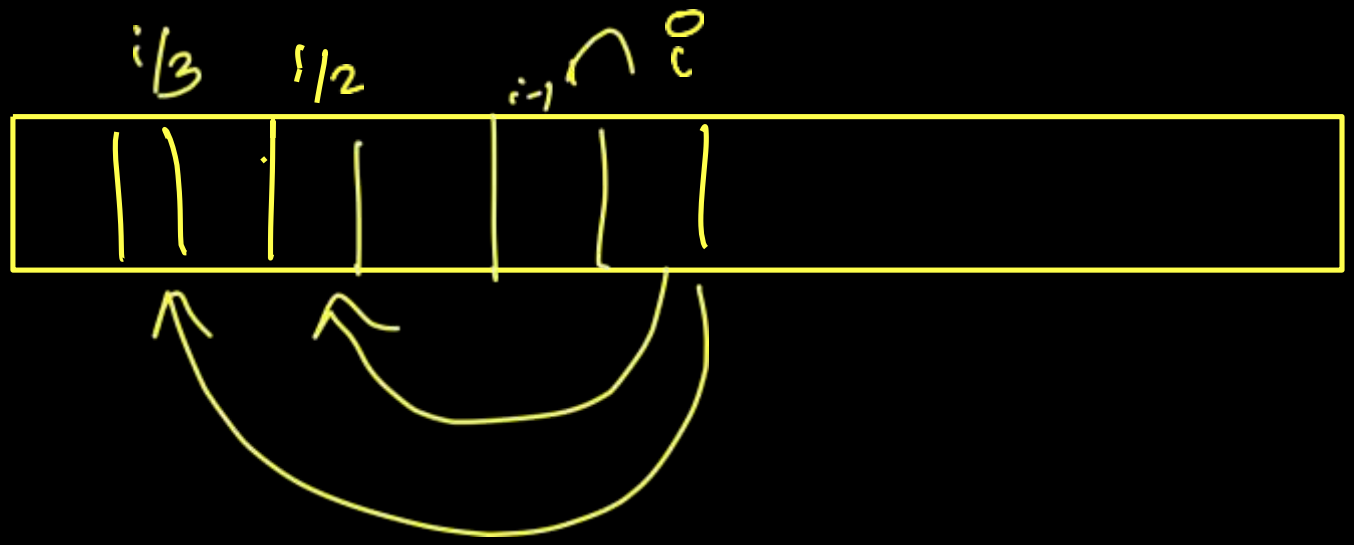
$n==3 \rightarrow 1$

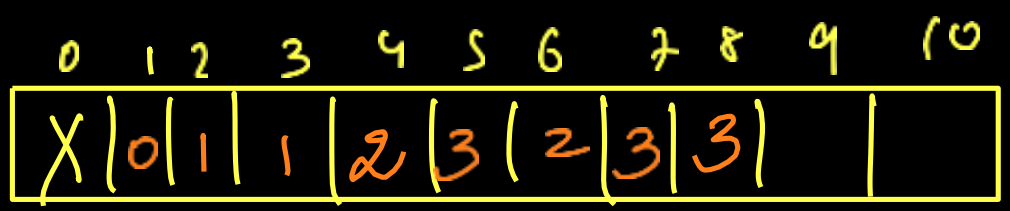
Stack → O(n)





$$dp[i] = \min\left(dp[i-1],\ (i\%2==0)\ ?\ dp[i/2] : \infty,\ (i\%3==0)\ ?\ dp[i/3] : \infty\right)$$





10

n+1

4

PW SKILLS

THANK YOU