

$$F = [0, 1]$$

fibonacci(n):

si $n=0$

resp.

0

si $n=1$

resp.

1

si $n > 1$

resp.

fibonacci(n-1) + fibonacci(n-2)

recursion directa

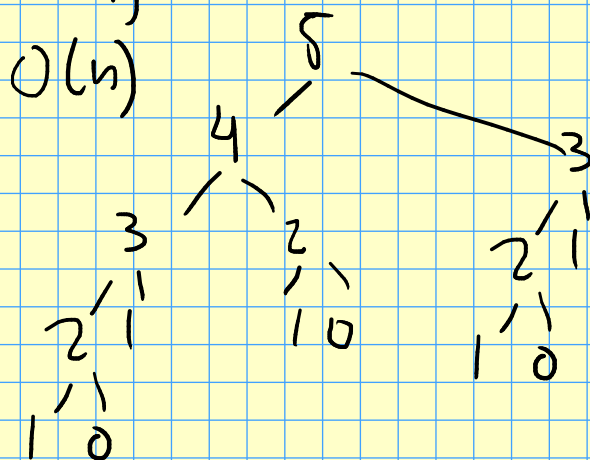
$$O(4^n)$$

$$O(1)$$

recursivo

$$O(n)$$

$$O(n)$$



fib(10)

fib(9)

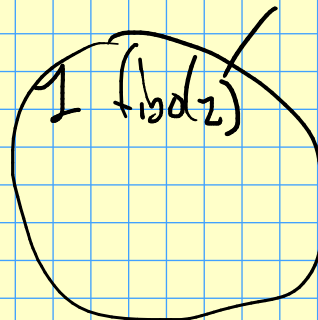
fib(8)

fib(7)

fib(6)

fib(5) 1

fib(3)



fib(1)

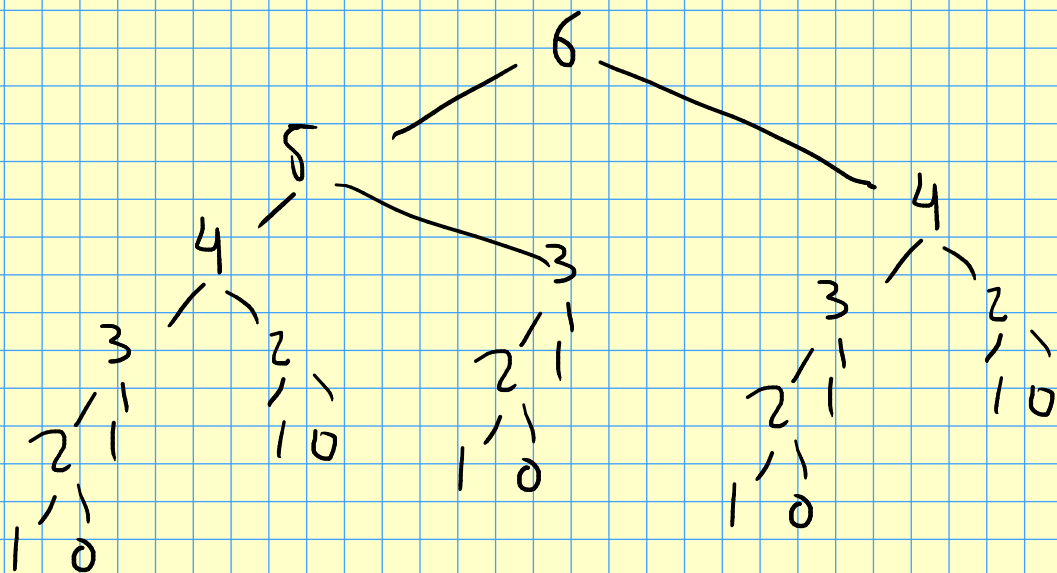
1

fib(4)

fib(2)

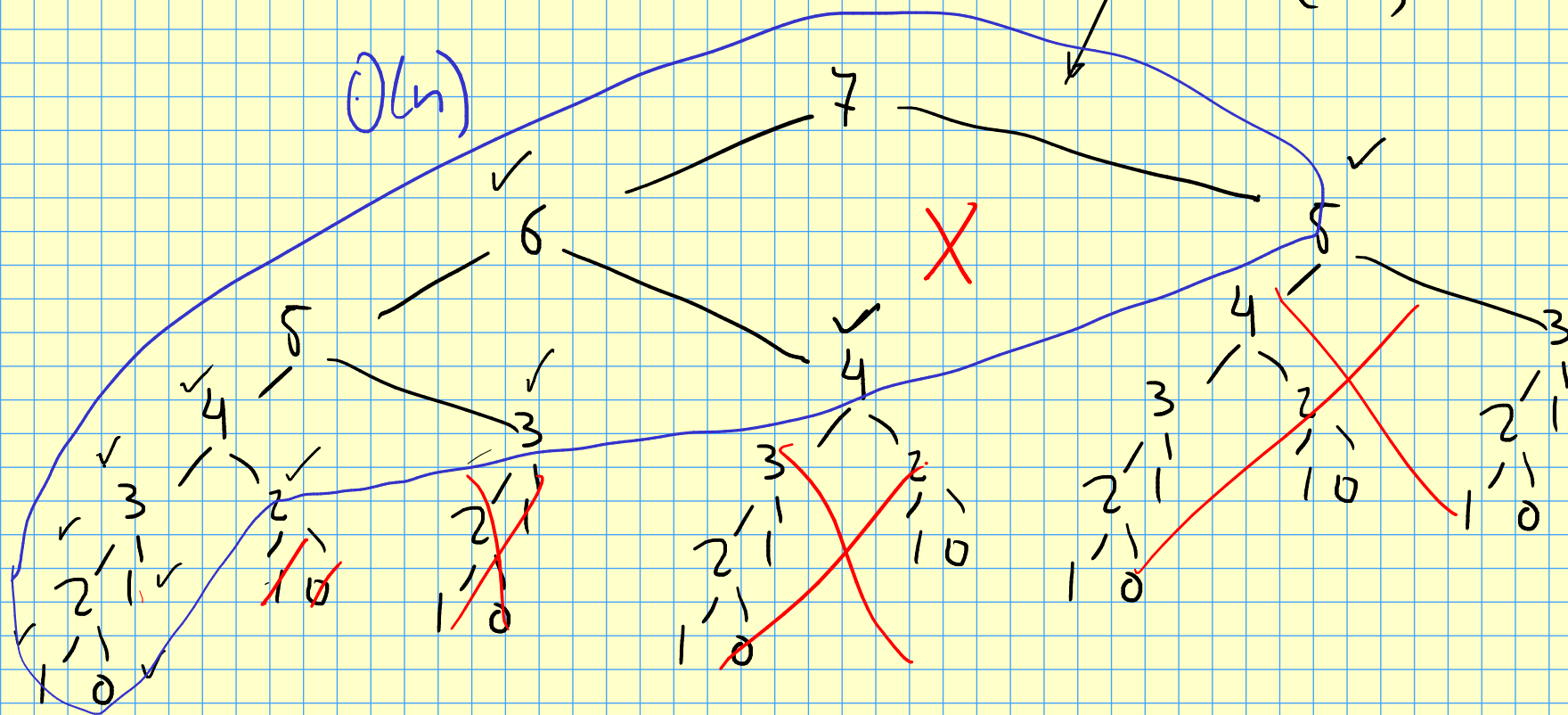
fib(1)

fib(0)

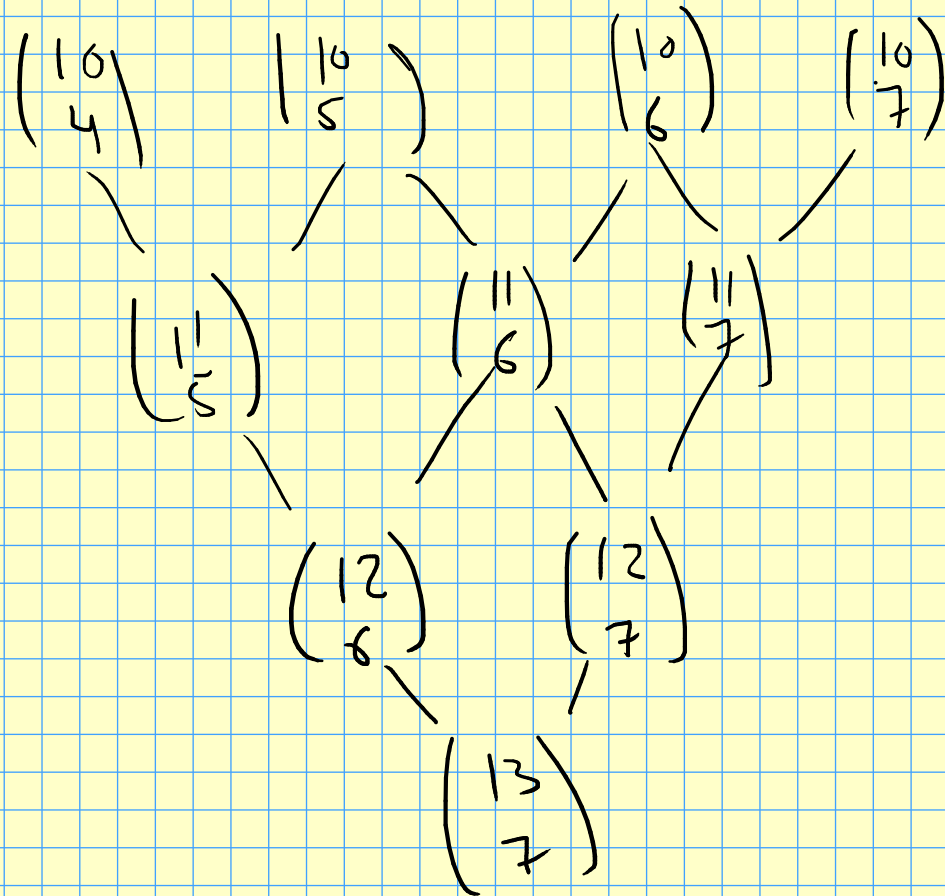


$$\Theta(4^n)$$

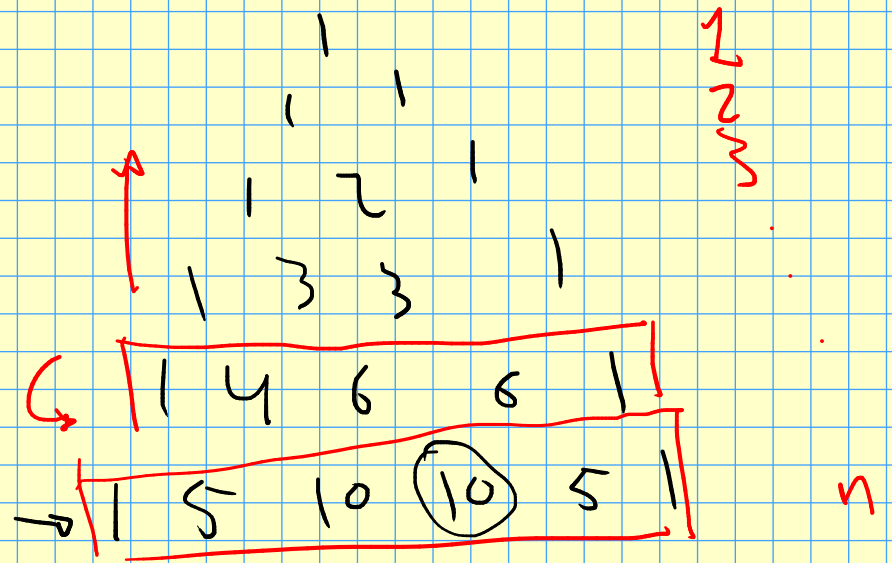
$$\Theta(2^n)$$



Mucho mejor idea: cada que busquemos un nuevo valor, lo almacenamos.
Si lo necesitamos volver a usar, lo tenemos de memoria.



$L = [$
 $\quad [1],$
 $\quad [1, 2],$
 $\quad [1, 2, 1],$
 $\quad [1, 3, 3, 1],$
 $\quad [1, 4, 6, 4, 1]]$



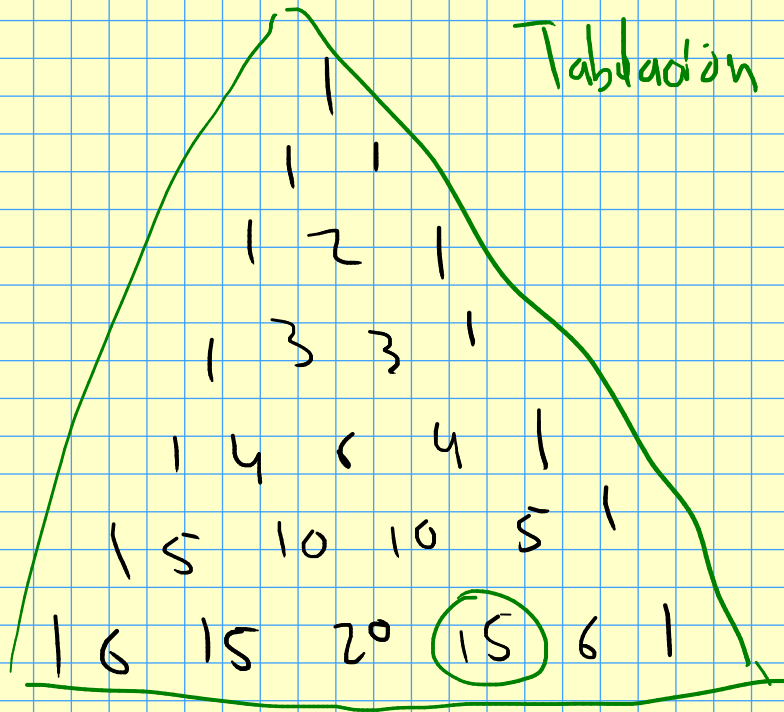
$$1 + \dots + n = O(n^2)$$

$\binom{n}{k} \rightarrow$

- hacer tabla hasta nivel n .
- reportar $L[n][k]$

espacio $O(n)$

Tabulation



Memorization

