: General Q - Priority Q
deQ

Struktur Data

linier : Stack and Queue , Array

static &
Dynamic

Vector
=
dequeue + Fitur tambahan

non linier : Graf, Tree

static

x = Arr [2] → x = 3

1 2 3 4 5
0 1 2

Dynamic

Queue

1 2 3 4 5
1 2 3 4
1 2 3

antrian gak boleh nyelak

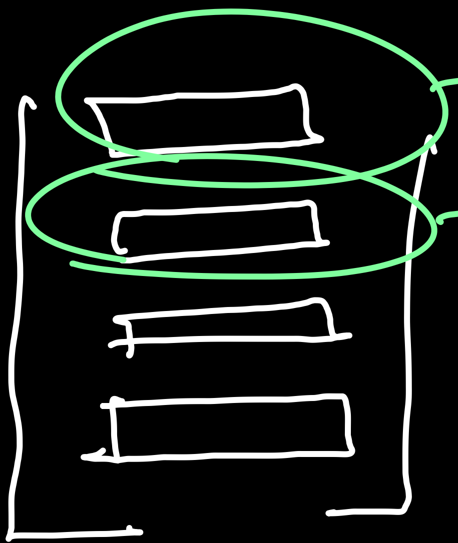
Static Array → Indexing

$A[i]$

$A[r][c]$

Dynamic Array → Melakukan langkah dinamis

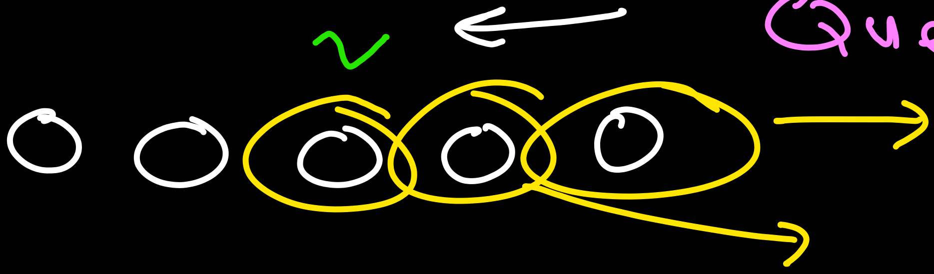
Stack



harus dikeluarin

Array = $A[5]$

Queue



* Stack (Tumpukan)

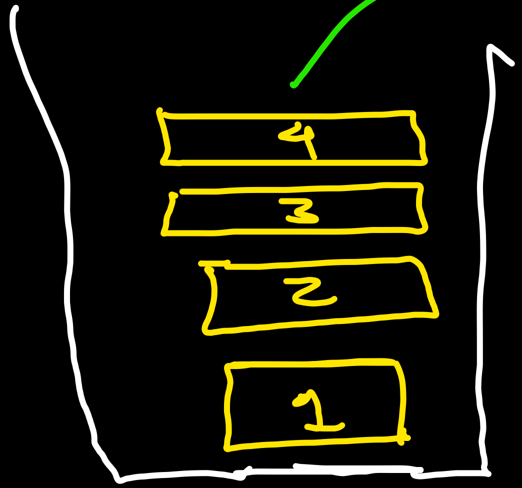
→

4 First out
4 Last in

→ LIFO

Last in First out

first out



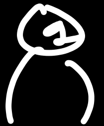
Pop() → dari paling baru

Push()

* Queue

antrian
wota
zee

→



Pertama dateng
Pertama ketemu
zee
→ FIFO

First in
First out

Keluar Pertama
Pertama pulang

FIFO

General Queue harus berurutan

1 2 3 4 5

(1,2,3,4,5)

DeQueue

1 2 3 4 5

5 keluar

2 keluar

1 2 3 4 5

1 3 4

push()

insert()

erase(pos)

hapus dari depan ← pop ( )

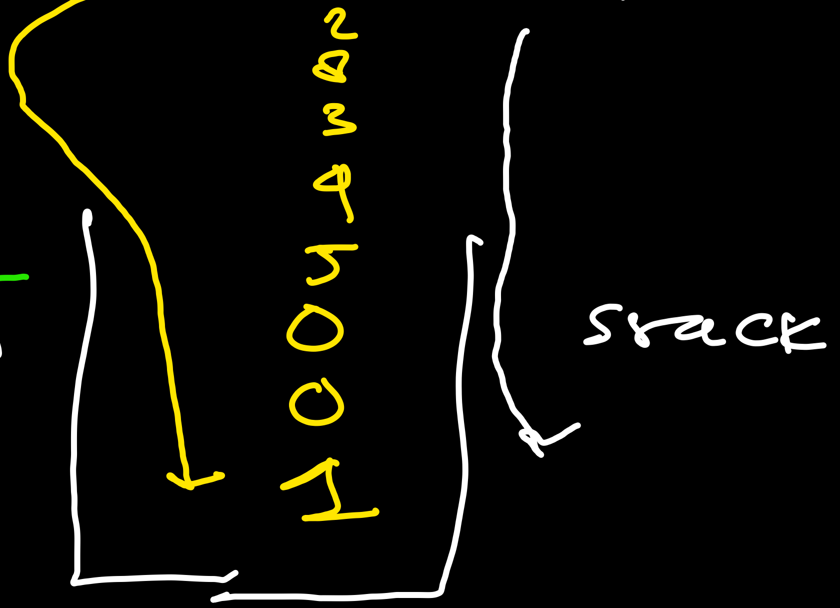
P. Queue

Q : 1 2 3 4 5

PQ : 5 4 3 2 1

Kita push dari array paling depan

array 1 0 0 5 4 3 8 2



F

array = [1,0,0,5,4,3,8,2]
stack = []

L

stack = [2,8,3,4,5,0,0,1]

Last in - First out

arr = 1 0 0 5 4 3 8 2

stack.push (arr)

arr .pop ()

Pop stack → hapus plg blkg

Q pop → hapus plg depan

arr = 1 0 0 5 4 3 8 2

Q = ~~1~~ 0 0 5 4 3 8 2

Q.pop () →

2
8
3
4
5
0
0
1

2 8 3 4

5 0 0 1

Q : ~~1~~ 0 0 5 4 3 8 2

Q.pop()

~~0~~ 0 5 4 3 8 2

Q.pop()

0 5 4 3 8 2

arr.pop()

1 0 0 5 4 3 8

arr.push(9)

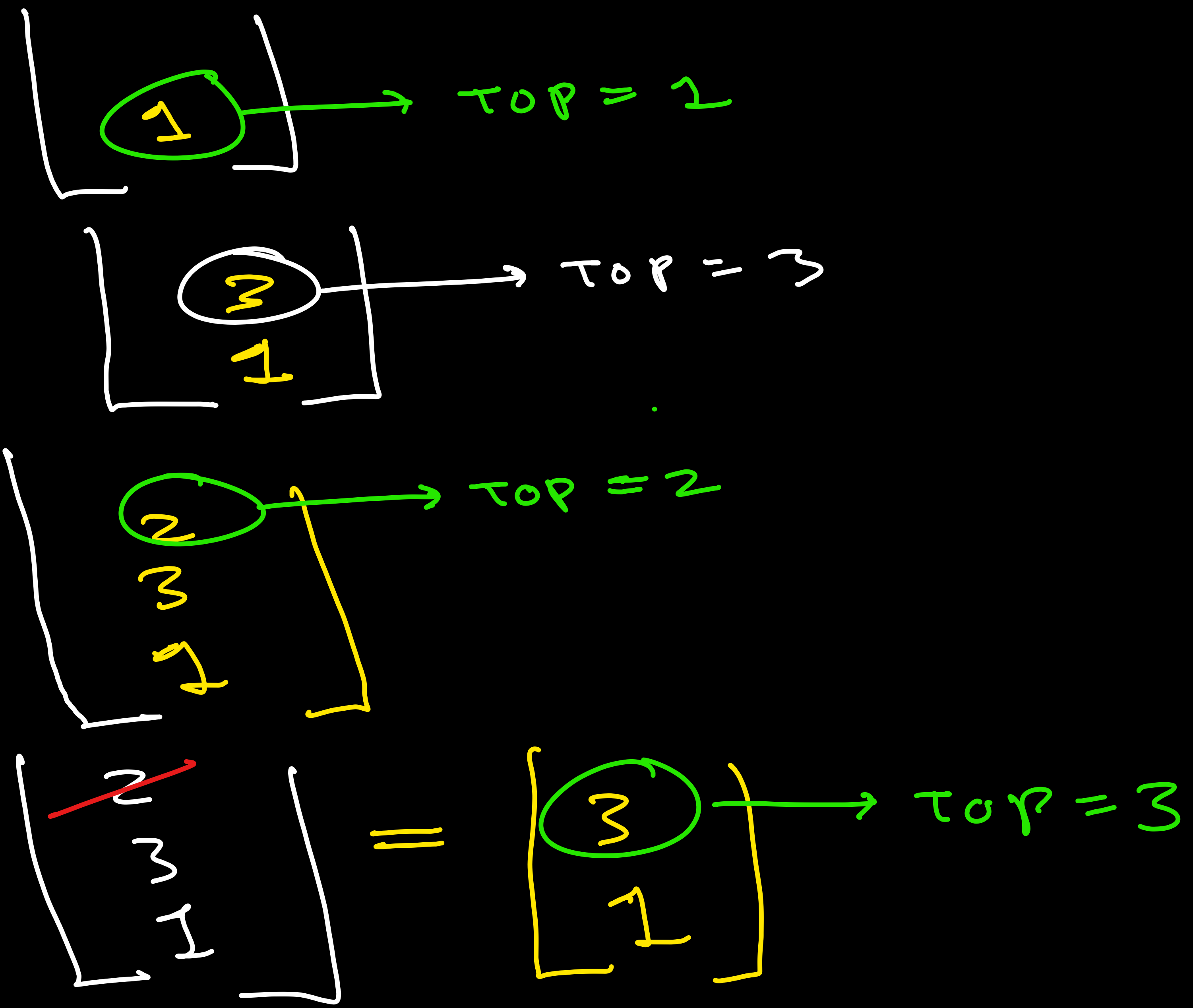
1 0 0 5 4 3 8 9

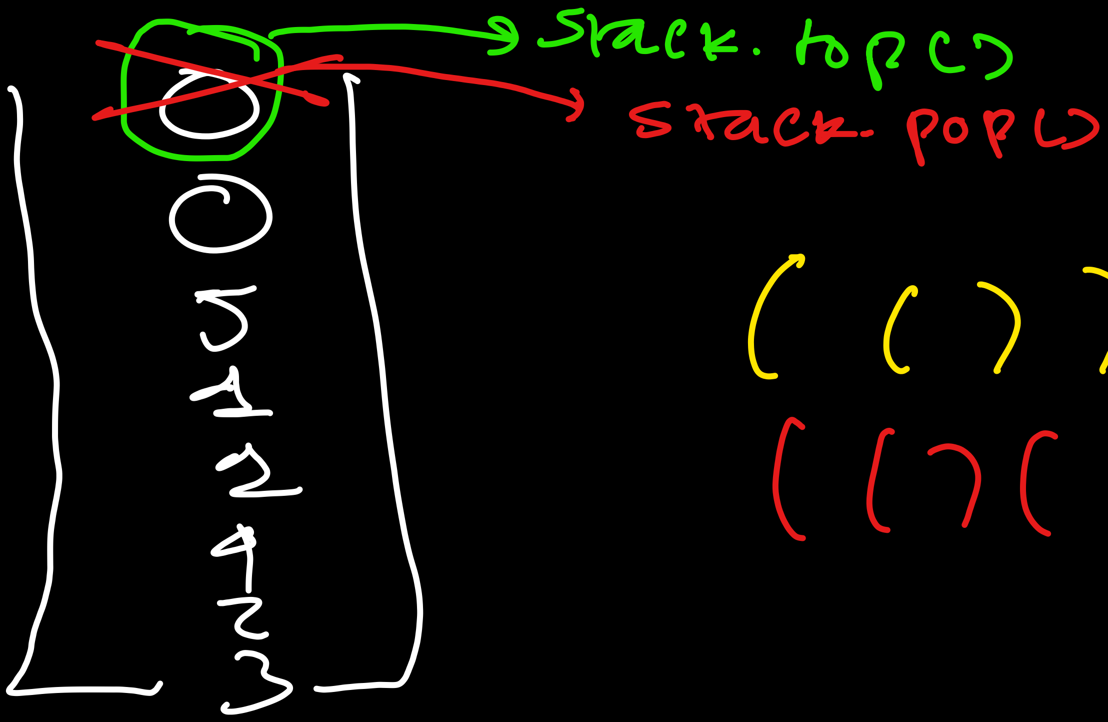
tumpuk.push(1) →

tumpuk.push(3) →

tumpuk(2) →

tumpuk.pop() →

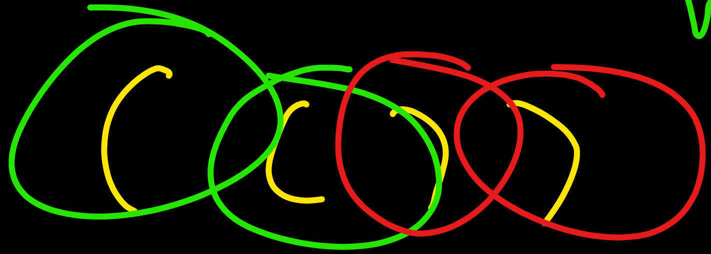


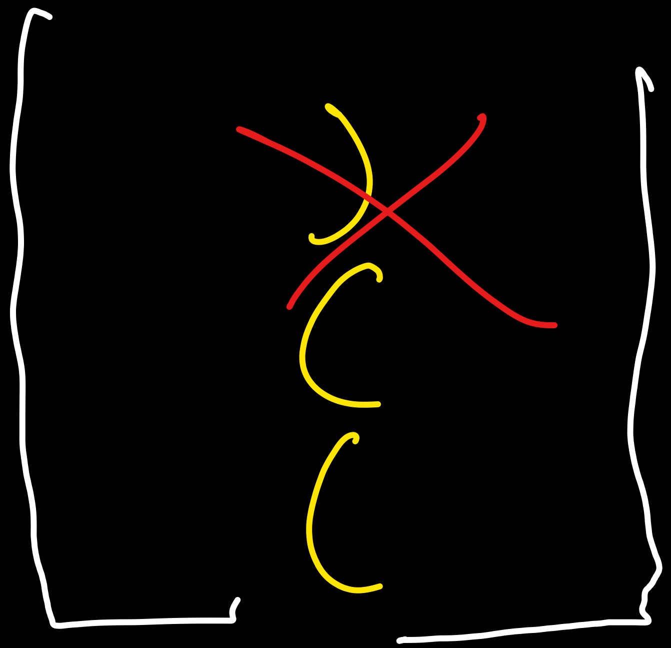


(( )) = valid

(( ) (= invalid

valid



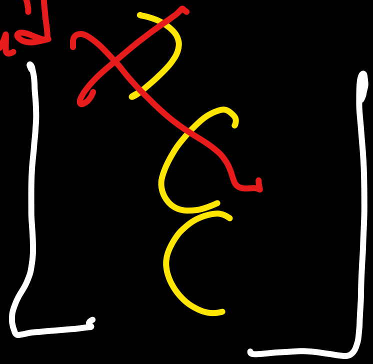


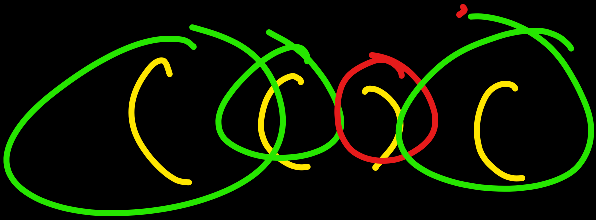
JIka di akhir stack itu kosong = valid

Jika di akhir tidak kosong = invalid

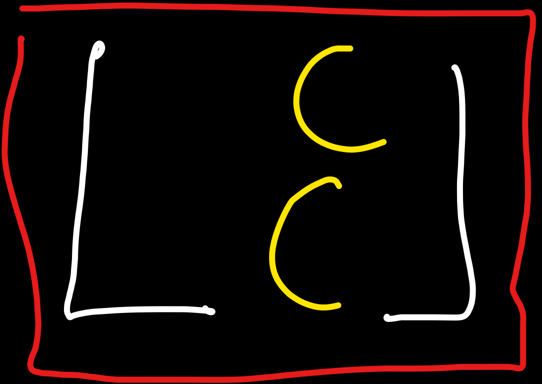
- Kalau ketemu tanda kurung buka masukin ke stack
- Kalau ketemu tanda kurung tutup hapus elemennya

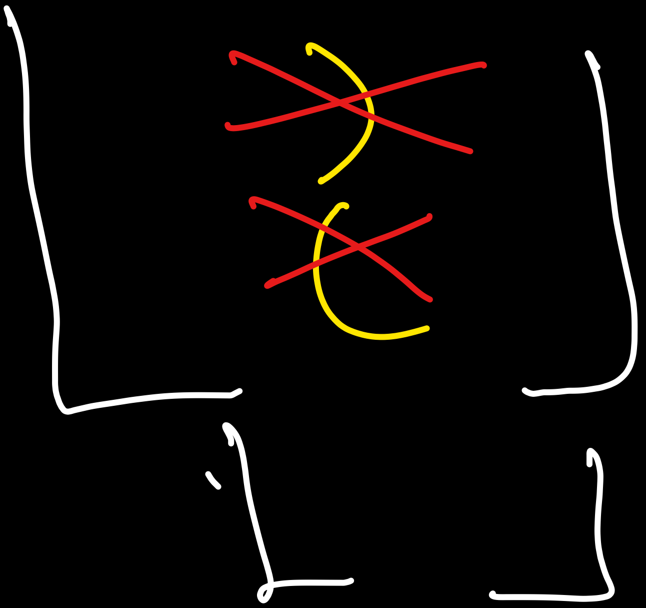
invalid





Menyisakan tanda kurung buka






200

OSNP

- Time limit : 2 s
- Memory : 256 Mb
- C / C++

$O(N) \rightarrow$ N kali operasi

$N = 10^8 \rightarrow$ 1 detik

$N = 10^{16} \rightarrow$ 2 detik

T

$N_i \to \text{nil}$

$f(n) = n \times f(n-1)$

$f(0) = 1$

T

N_1

N_2

N_3

$\ldots$

N_n

$f(n) = n \times f(n-1) \rightarrow$ Kompleksitas $O(N)$

$f(n) = 1 \times f(n-1)$

$= 1 \times (n-1) \times f(n-2)$

$= 1 \times (n-1) \times (n-2) \times f(n-3) \times \ldots$

$f(5) \rightarrow 5 \times f(4) \rightarrow 4 \times f(3) \rightarrow 3 \times f(2)$

1 2 3 4

$2 \times f(1)$ 5

$1 \times f(0)$

$\rightarrow 1$

$f(5) = 5$ operasi / iterasi

$f(6) = 6$ operasi / iterasi

$F(n)$ = kompleksitas $O(N)$

T

Setiap kasus uji ke-$i \rightarrow O(N)$

$O(N)$ 1

$O(N)$ 2

$O(N)$ 3

— – - - -

$O(N)$ T

$\underline{O(N \times T)}$

$10^8 = 1s$

$N_{max} = 10^3$

$T_{max} = 10^7$

$O(10^3 \times 10^7) = 10^{10}$

$\underline{\underline{TLE}} \leftarrow \underline{=> 1s}$

5

1 → 1! = 1
2 → 2! = 2 × 1
3 → 3! = 3 × 2 × 1
4 → 4! = 4 × 3 × 2 × 1
5 → 5! = 5 × 4 × 3 × 2 × 1

$\}\ O(N^2)$

5

5 → 5! = 5 × 4 × 3 × 2 × 1
5 → 5! = 5 × 4 × 3 × 2 × 1
5 → 5! = 5 × 4 × 3 × 2 × 1
5 → . . . .
5 → . . . .

$\}\ O(NT)$

5

1 → $1! = 1$

2 → $2! = 2 \cdot 1! = 2$

3 → $3! = 3 \cdot 2! = 3 \cdot 2 = 6$

4 → $4! = 4 \cdot 3! = 4 \cdot 6 = 24$

5 → $5! = 5 \cdot 4! = 5 \cdot 24 = 120$

Ingat

$1! = 1$, $2! = 2$,

$3! = 6$

5

5 → $5 \times 4 \times 3 \times 2 \times 1 = 120$

5 → $5!$ $= 120$

5 → $= 120$

5 → $= 120$

5 → $= 120$

Ingat

$5! = 120$

Ingat $\longrightarrow$ Memoization $\rightarrow$ $\leq$ 1s

$N_{max} = 10^3$ $T_{max} = 10^7$

CP $\rightarrow$ competitive programming

Jawaban Tepat & Efisien

$$f(n) = f(n-2) + f(n-2)$$

const:

$$\underline{O(2^N)}$$

* Without Memo

f(100)

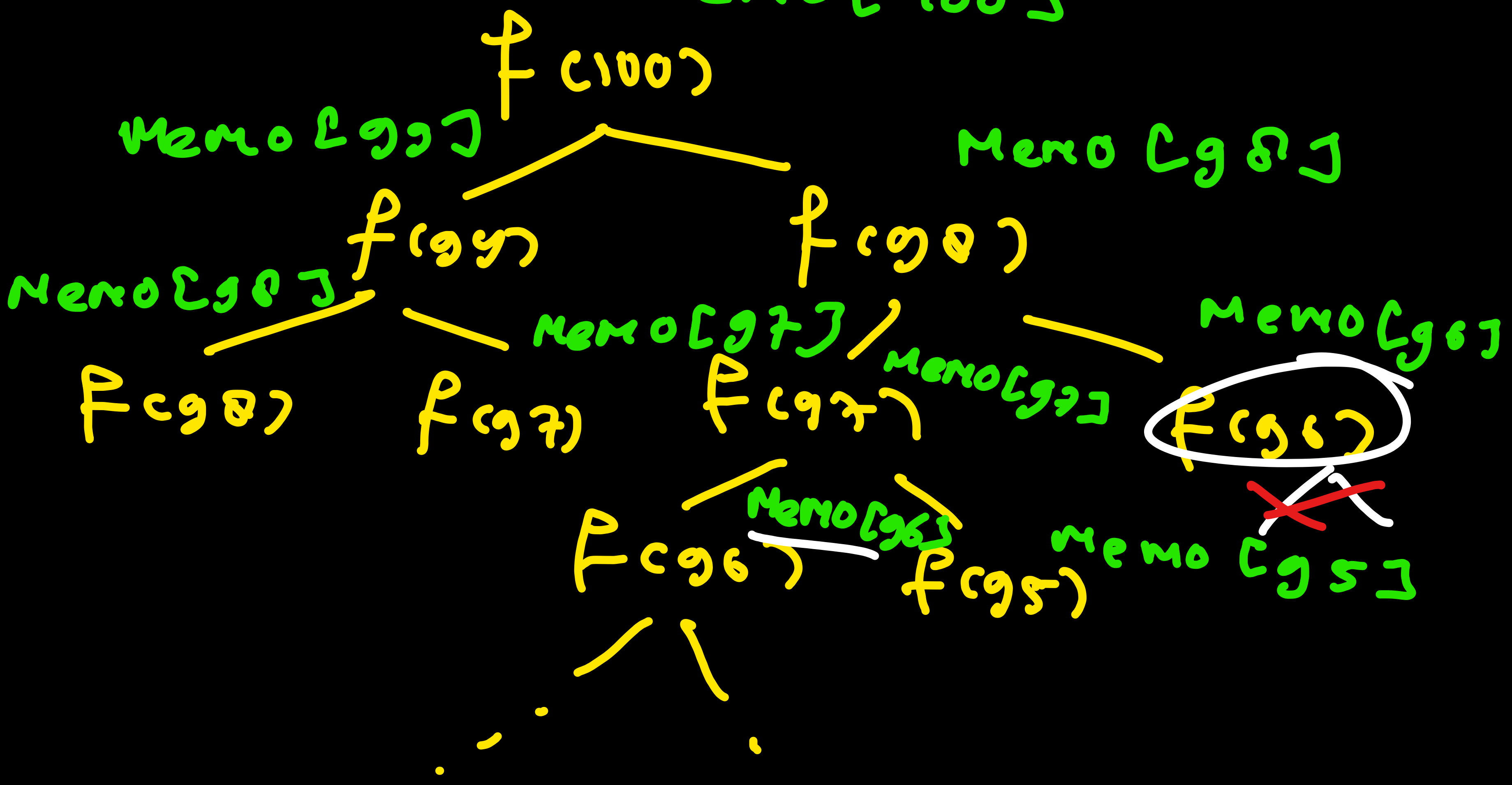
f(99) f(98)

f(98) f(97) f(97) f(96)

f(97) f(96) f(96) f(95) . . .

Cabangnya Banyak Bgt → 2^{100}

* With Memo



Best complexity : $O(\log_2 N)$

with Memo : $O(\log N) \rightarrow$ Dp

without memo : $O(2^N) \rightarrow$ Dnc

gk pake memo

Pake memo

$N = 100 \rightarrow 2^{100}$

$= 2^{100}$ operasi

${}_2\log 100$

≈ 7 operasi

Dp bisa pakai Dnc

Dnc belum tentu Dp

2 den'x = 10^{16}

M x N

$(1,1) \longrightarrow (x,y)$

$x = 2,\ y = 3$

Rows 2 colom 3

start

	0	1	2	3
2	2	3	2	3
2+4	4	5	2	4
2+4+7	7	6	1	3

2+4+7+6

→ 2+4+7+6+1+3

Top down

$dp(x, y)$ = Total item max, yg bisa didapat dari $(1,1) \longrightarrow (x, y)$

$dp(x, y)$

0	1	2	3
2	3	2	3
4	5	2	4
7	6	1	3

sedean mana jalan ke kiri / ke atas

Jalan ke-kiri : $(x-1, y)$

Jalan ke-atas : $(x, y-1)$

Menentukan langkah mana yang bisa kita dapatkan sehingga total angka yang diperoleh mendapatkan sebesar mungkin. Ke Kiri atau ke atas?

Max ((x - 1, y), (x, y - 1))

$$dp(x, y) = \max(dp(x-1, y), dp(x, y-1)) + item[x-1][y-1]$$

· koordinat

baris, kolom (1, N), (1, M)

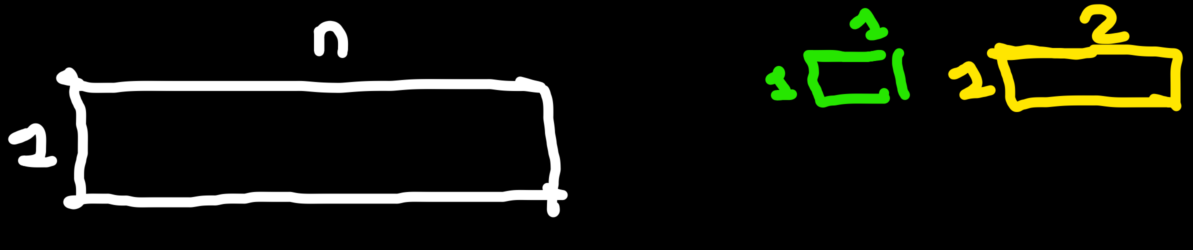
$dp(x,y) = max(dp(x - 1, y), dp(x, y -1)) + item[x][y]$

$dp(1,1) = item[0][0]$

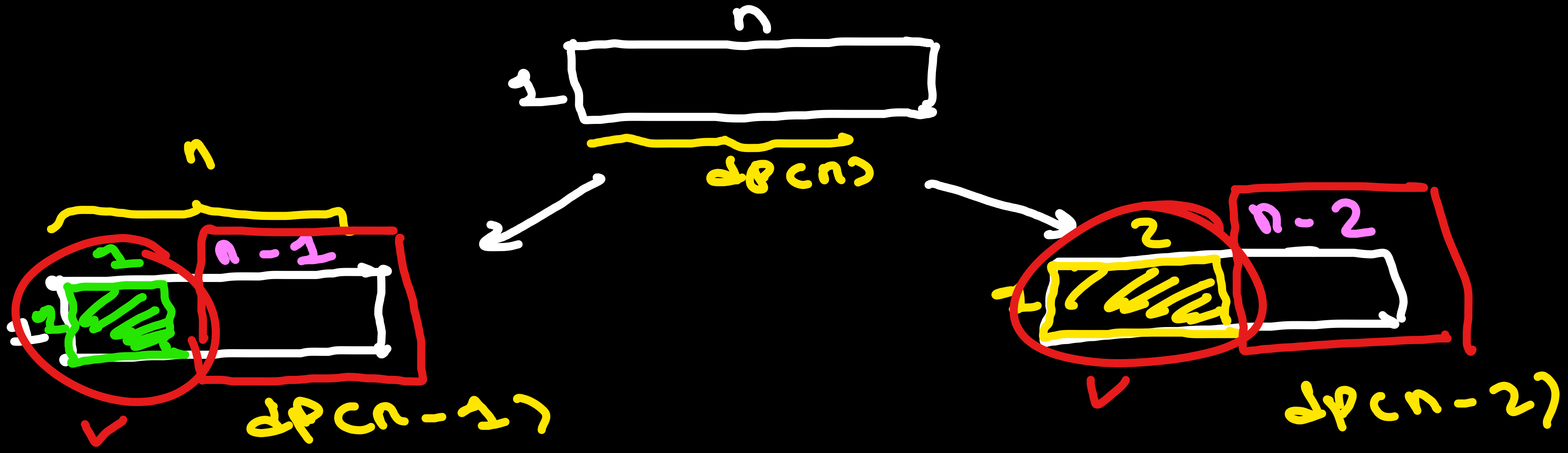
$dp(1,2) = item[0][0] + item[0][1]$

$dp(2,1) = item[0][0] + item[1][0]$

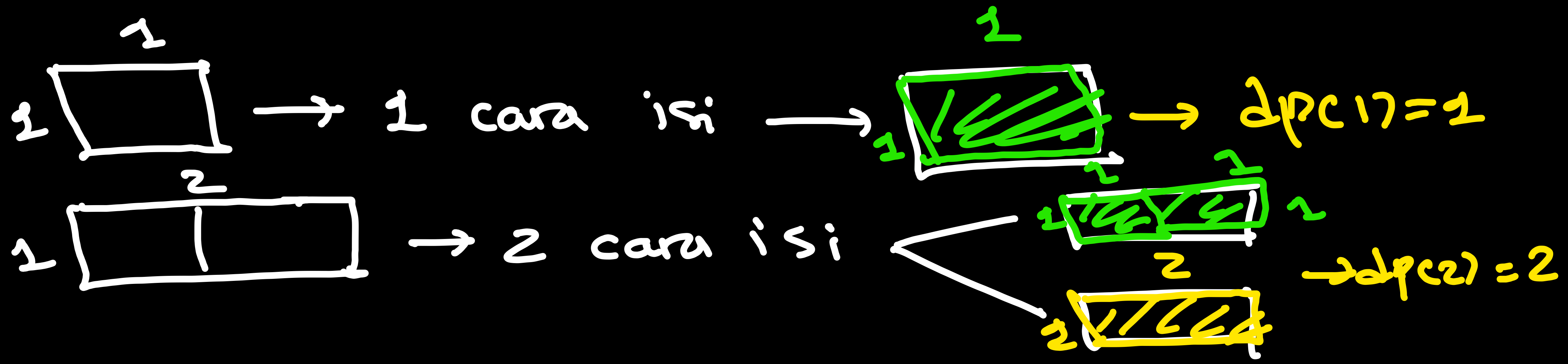
Diberikan lantai berukuran 1 x N, berapa banyak cara memasang ubin berukuran 1 x 1 dan 1 x 2 dengan syarat. Pemasangan tidak boleh tumpang tindih dan semua lantai harus terisi ubin.



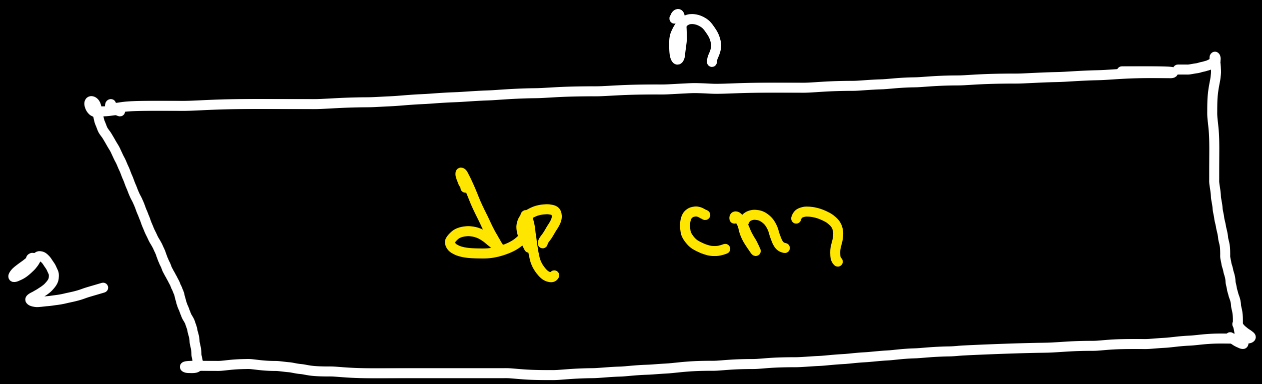
dp(n) = menyatakan banyaknya cara mengisi lantai berukuran 1 x N

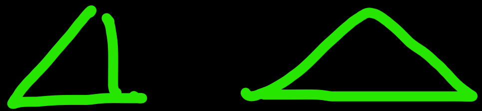


$$dp(n) = dp(n-1) + dp(n-2)$$



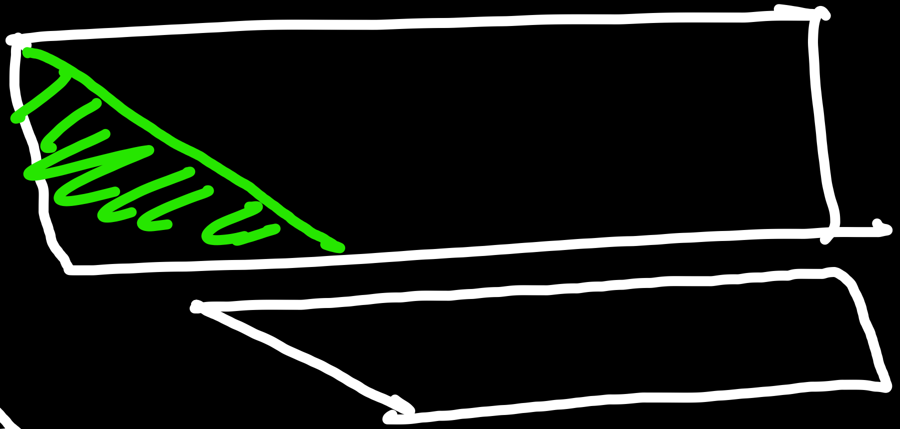
Subsoal 1 $(N = 1)$



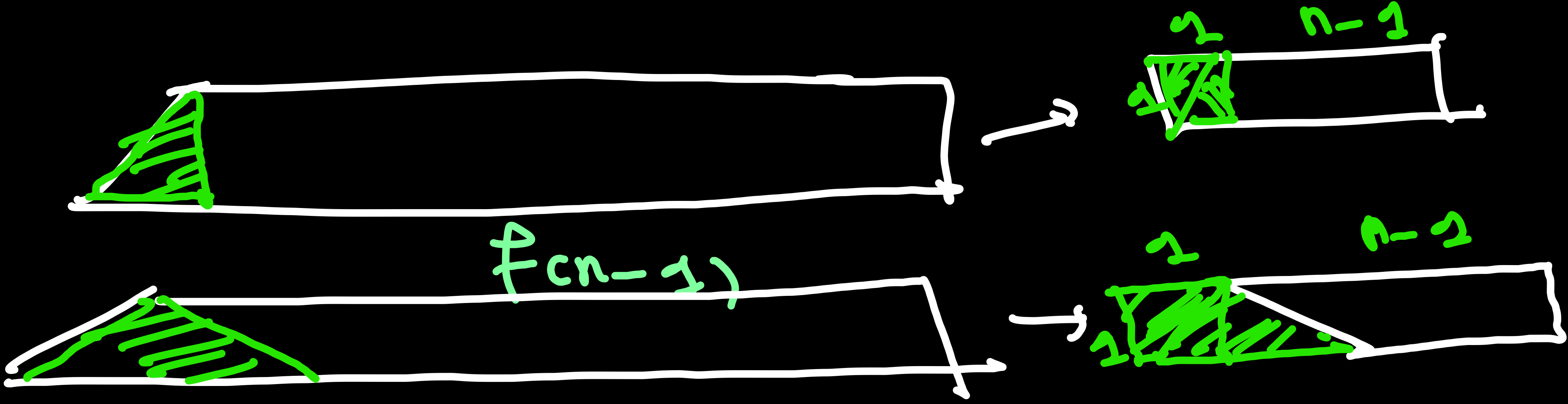


$d(n) = 2g(n)$

$g(n)$



$g(n)$



$g(n-1)$

$$dp(n) = 2g(n), \quad g(n) = dp(n-1) + g(n-1)$$

$$dp(n-1) = 2g(n-1)$$

$$3dp(n-1) = 6g(n-1)$$

$$3dp(n-1) = 2g(n)$$

$$g(n) = 2g(n-1) + g(n-1)$$

$$g(n) = 3g(n-1)$$

$$6g(n-1) = 2g(n)$$

$$dp(n) = 3\,dp(n-1)$$

$$\underline{dp(1) = 2}$$