

- Urutin nilai kartunya

Jumlah \rightarrow Kairan
semua

Max pref sum * ...

Max

$$\underbrace{\text{pref sum}}_{\text{...}} \left[\underbrace{l - \dots - r}_{\text{...}} \right] * \text{...}$$

$\max(1 - 2 * \dots)$

(-) X
(+) X
(-)

(+)
(-)
(-)

1 5 g 7

5
-3 -2 -4 -1 5

$$\text{Pref sum} = (1+5+g) = 15$$

$$15 \times 7 = 105$$

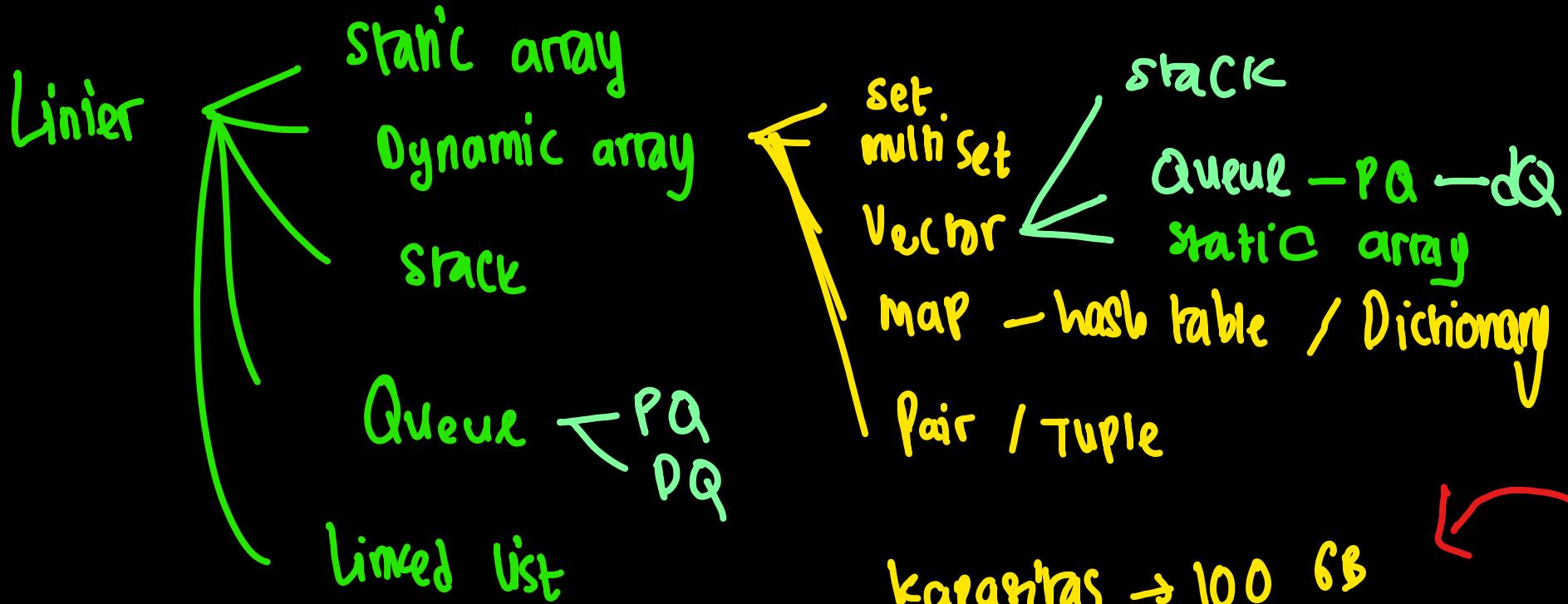
$$(-3 -2 -1) * -4 + 5$$

$$(1+5+7) * g = 13 * g$$
$$= =$$

Tipe data

primitif : int, string, ...
non primitif : — struktur data

unier
non unier



Kapasitas $\rightarrow 100 \text{ GB}$

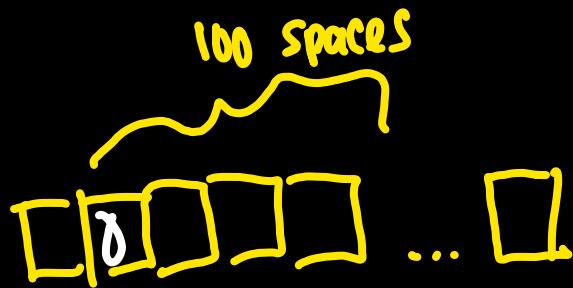
$\text{int arr}[100 \text{ GB}] =$
 $\text{arr}[1] = 0$

~~$\text{int } x = 5 \rightarrow \text{MLE}$~~

* Array

int arr[100] →

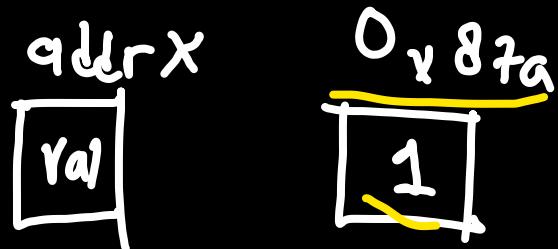
arr[1] = 0



Functional Programming → Pointer

Pointer

int x = 1 →

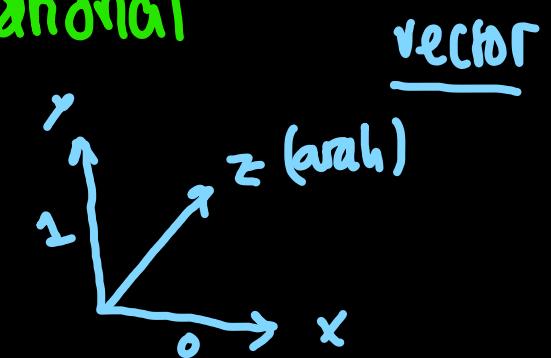
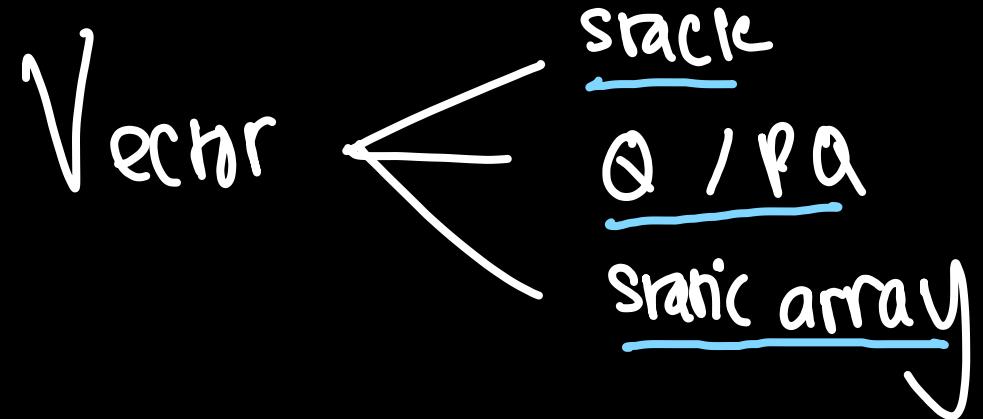


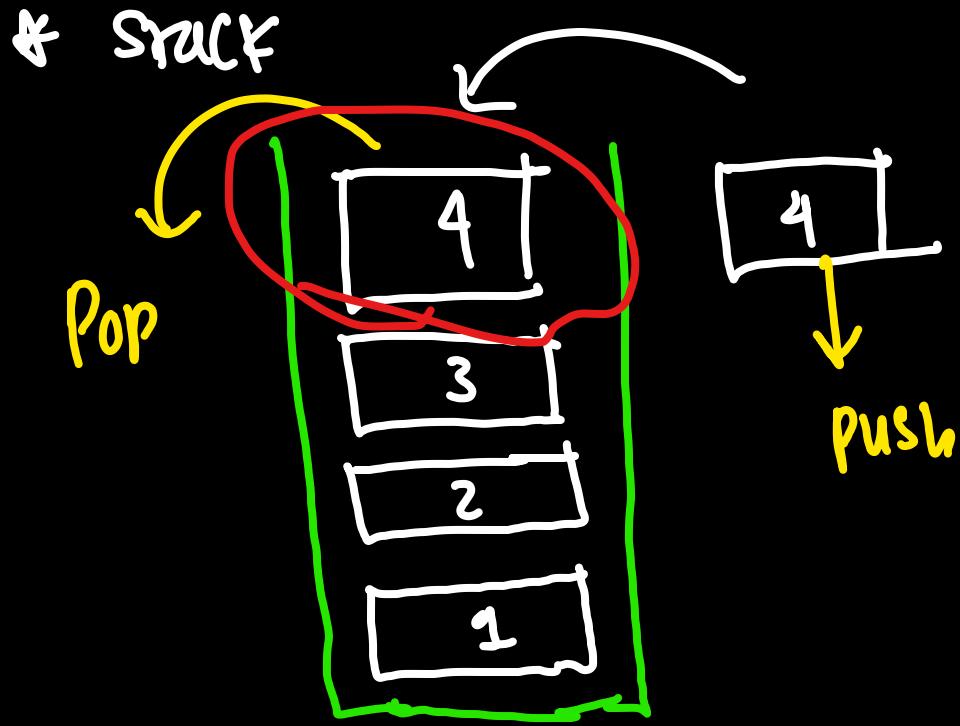
* static vs Dynamic

- Malloc

- Dynamic < operable action y push, pop, insert, ..

tahap / langkah komputasional sistematis





array : [1, 1, 2, 3]
 First in last in

LIFO (Last in - First out)

- * DFS (Depth First Search)
- * certain algorithms

stack : [3, 2, 1, 1]
 first out last out

* Queue (general)



orang Pertama masuk = Pusing depan
orang terakhir masuk = Pusing belakang

FIFO (First in - First out)

belakang → depan
depan → ~~belakang~~

~~Sembbarang Tempat~~

* BFS (Breadth - First - Search)

array = [1, 1, 2, 3] → Q : [1, 1, 2, 3]

$$1 \leq n \leq 2^{23}$$

$n = \dots$

$$1 \leq n \leq 10^{23}, \quad n \neq 15 \rightarrow \text{edge case}$$

* Priority Queue \rightarrow top: max

$$\text{arr: } [1, 5, 3, 2, 0]$$

$$\xrightarrow{\text{PQ}} \text{PQ : } [0, 1, 2, 3, 5]$$

+ Binary - heap (Tree)
 \hookrightarrow Graph \rightarrow • Shortest Path
+ min cost
Tree

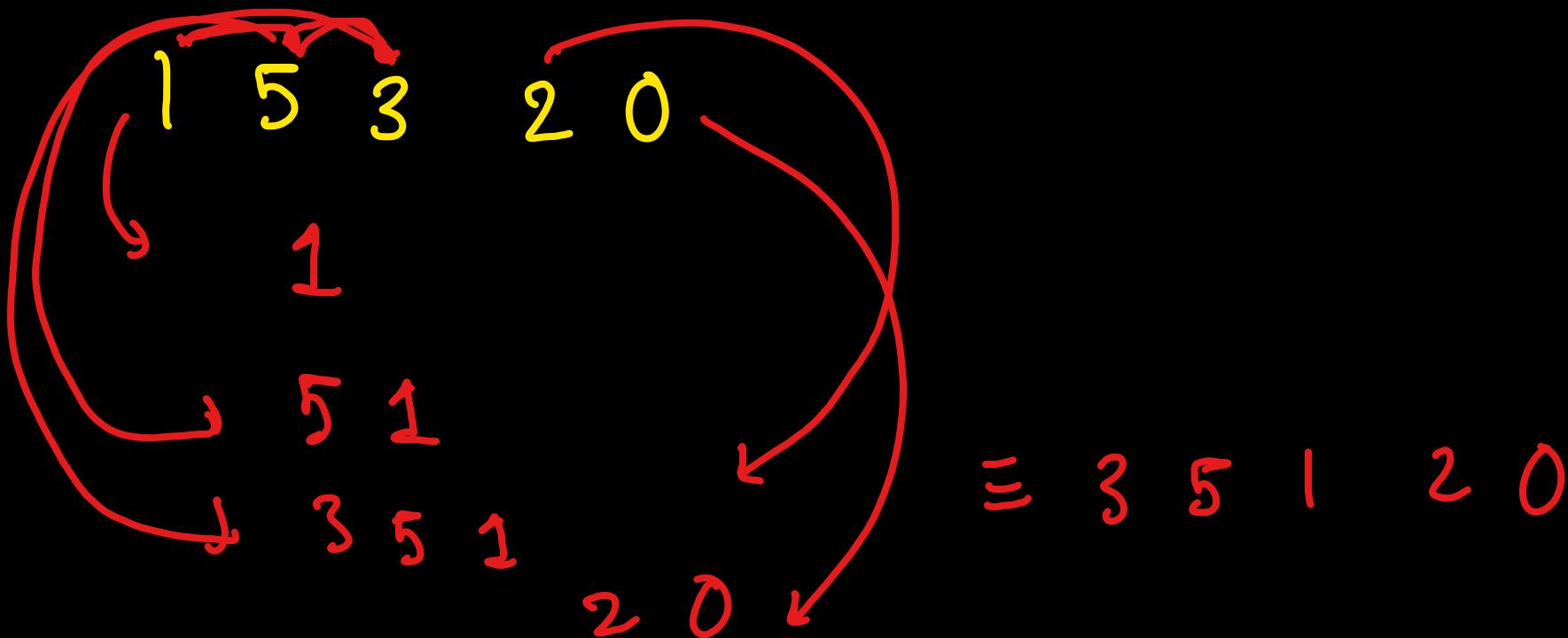
$$\xrightarrow{\text{PQ}} \text{PQ : } [5, 3, 2, 1, 0]$$

(min-heap)
(max-heap)

* Dynamic Queue

1 2 2 5

1 2 2 3 5



da location Memory

0x1a 0x2a 0x3a 0x4a 0x5a

1 2 3 4 5

iterasi $it = dq.begin()$

Tempatkan di idx ke-2

$it++$ = Sebelah Kanan 0x1a

$it + 2$ = 2 lokasi sebelah Kanan 0x1a

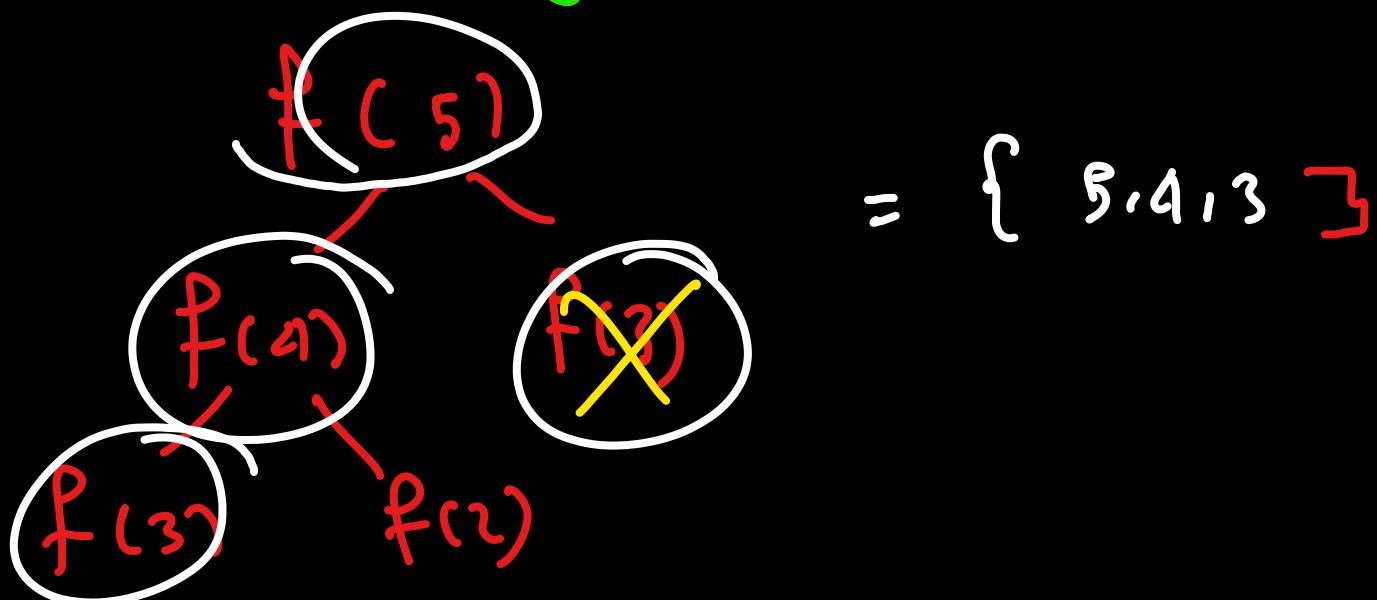
* Set (himpunan)

→ gak boleh ada duplikat

$$\text{arr} = \{ 1, 1, 2, 3 \} \xrightarrow{\text{set}} \text{set} = \{ 1, 2, 3 \}$$

→ subset → gaboleh bernang

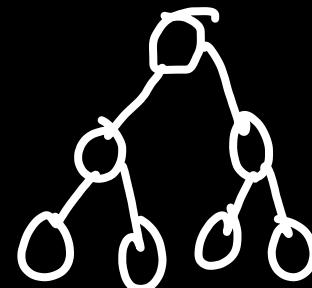
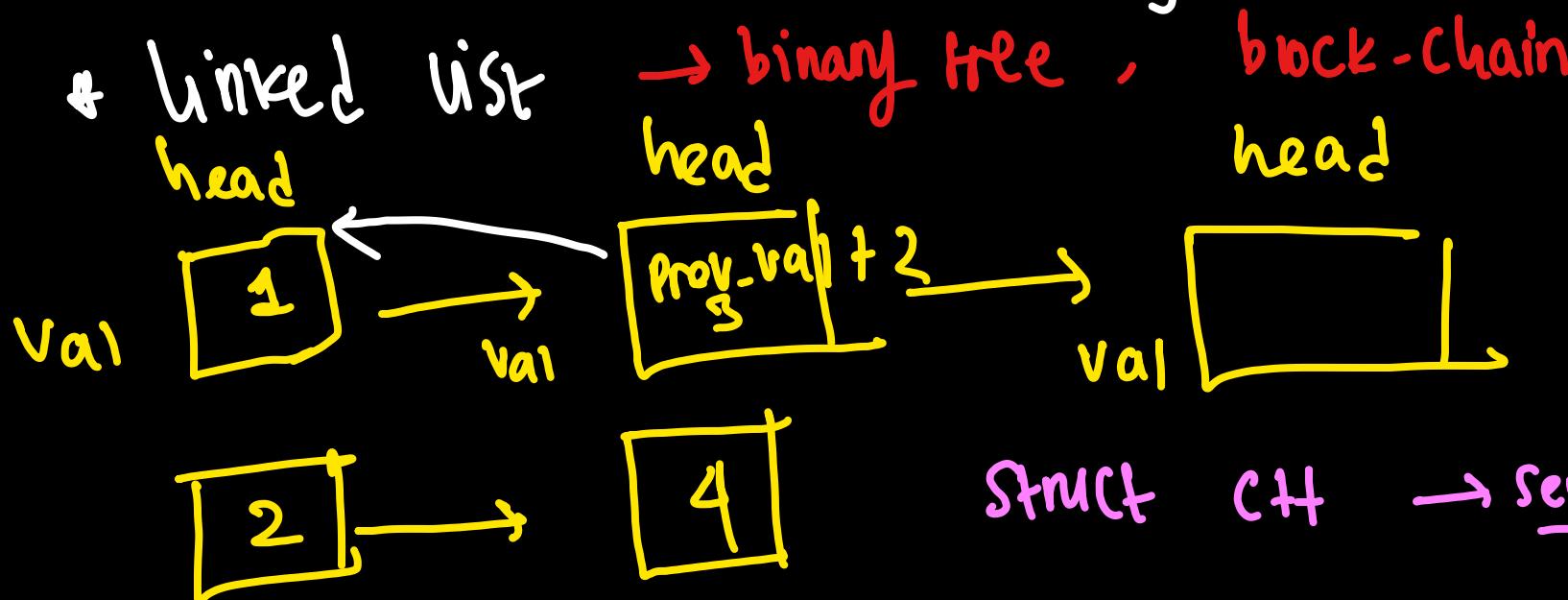
* pruning Tree



* Multi Set

Set (set \hookrightarrow) = $\{ \underline{\{1,2\}}, \underline{\{1,2\}} \}$

multiset = $\{ \{1,2\} \}$



Memo [1000001] array

$$N = 10 \rightarrow 10$$

Memo [1] = 1
Memo [2] = 1
Memo [3] = 2 } 3 space located

* Map \rightarrow hash table

`arr[9] = "C"`

hash table → map (dictionary)

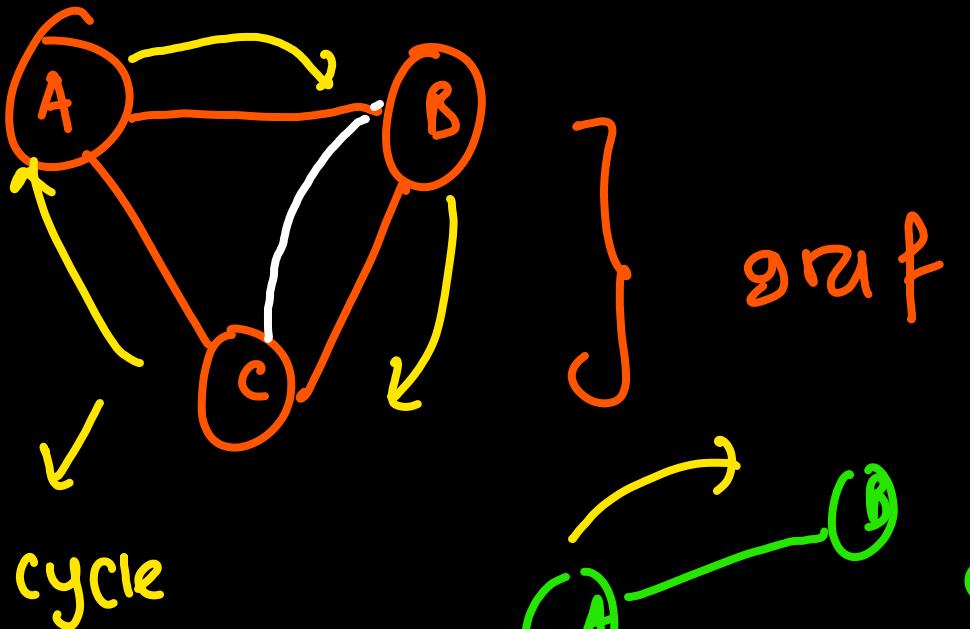
`Map <string, int> umur;`

key	value
Bintang	16
Abdan	18
Revan	15

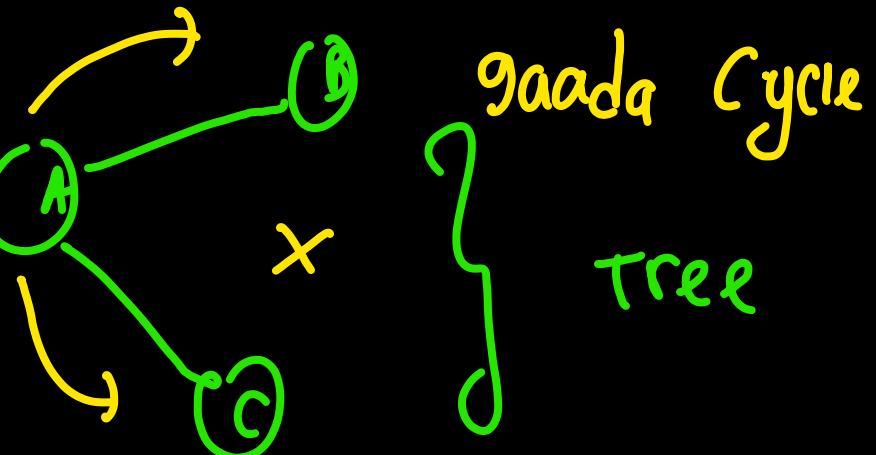
`umur["Bintang"] = 16`

`umur["Abdan"] = 18`

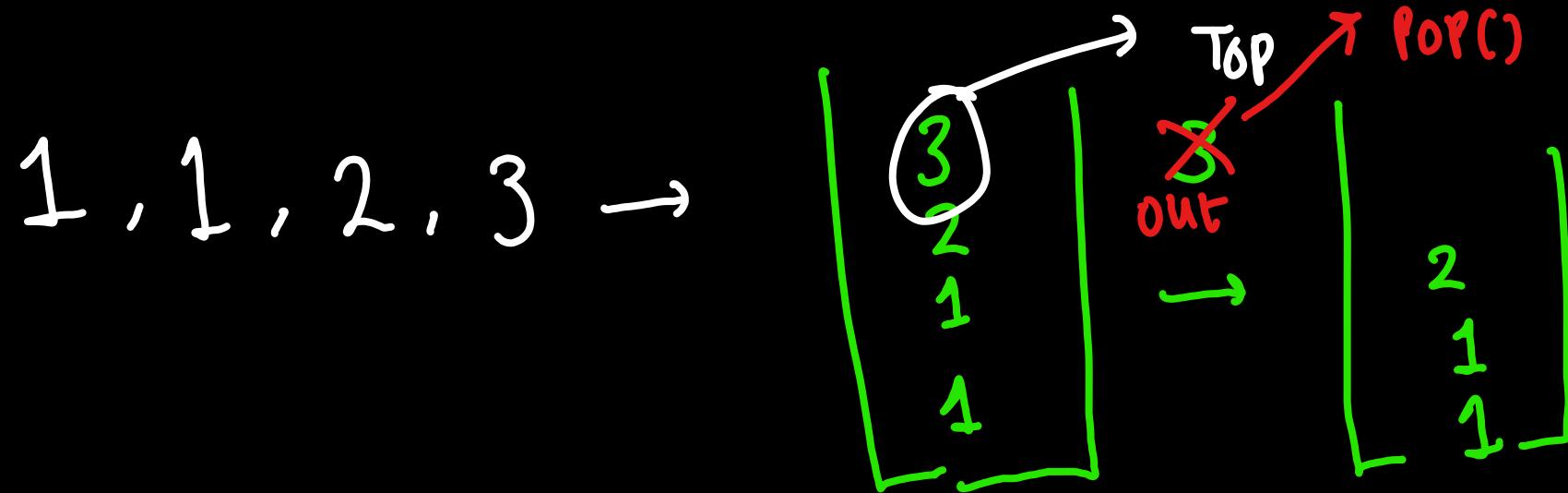
+ enumerate key, value



$\text{adj}[a] = \{B\}$
 $\text{adj}[B] = \{a\}$
 $\text{adj}[c] = \{a, b\}$

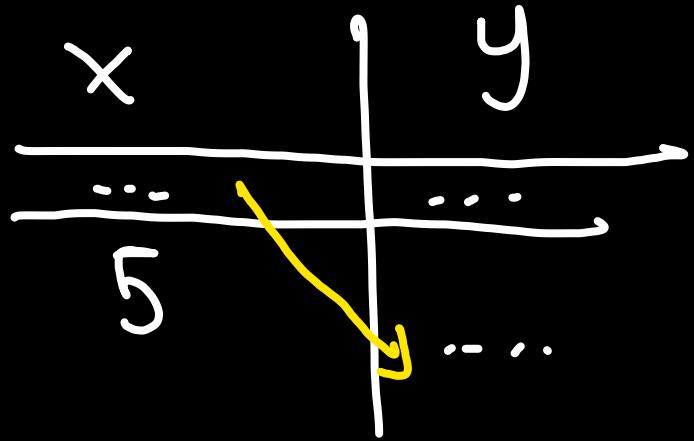


$\text{Can} = x \rightarrow \text{Search}$
 $\text{Can} \leq x \rightarrow \text{lower-bound}$
 $\text{Can} \geq x \rightarrow \text{upper-bound}$

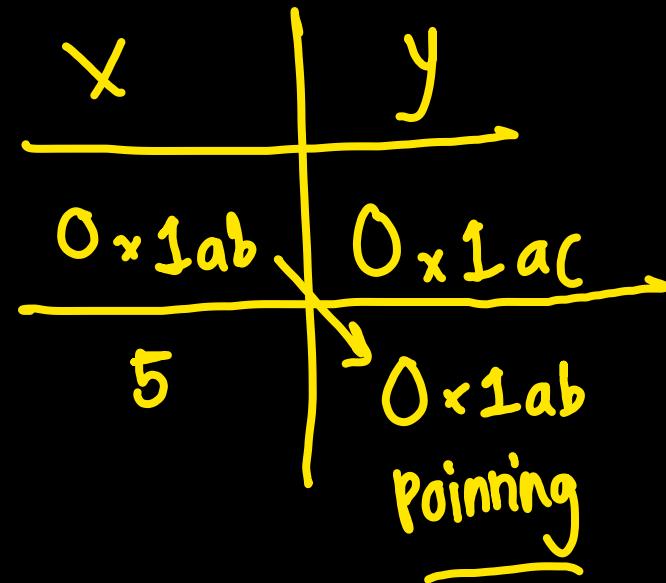


`int x = 5
int y = x`] addr 0x1ab 0x1ac) → pass by value
 Val 5 | 5

Pass by Reference → y berasah, x bersahabah → y navigator x,
 y referring to x



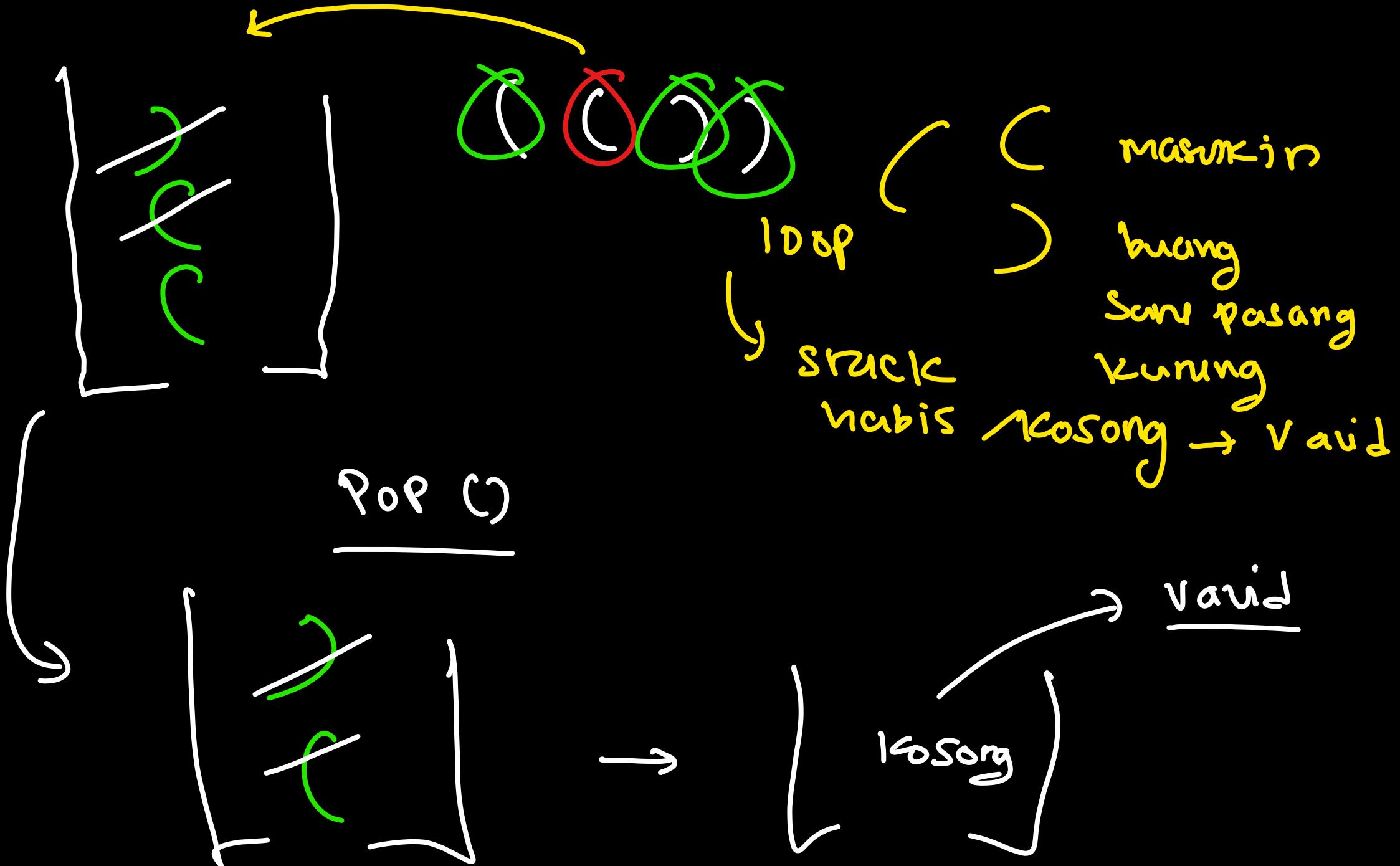
Val:



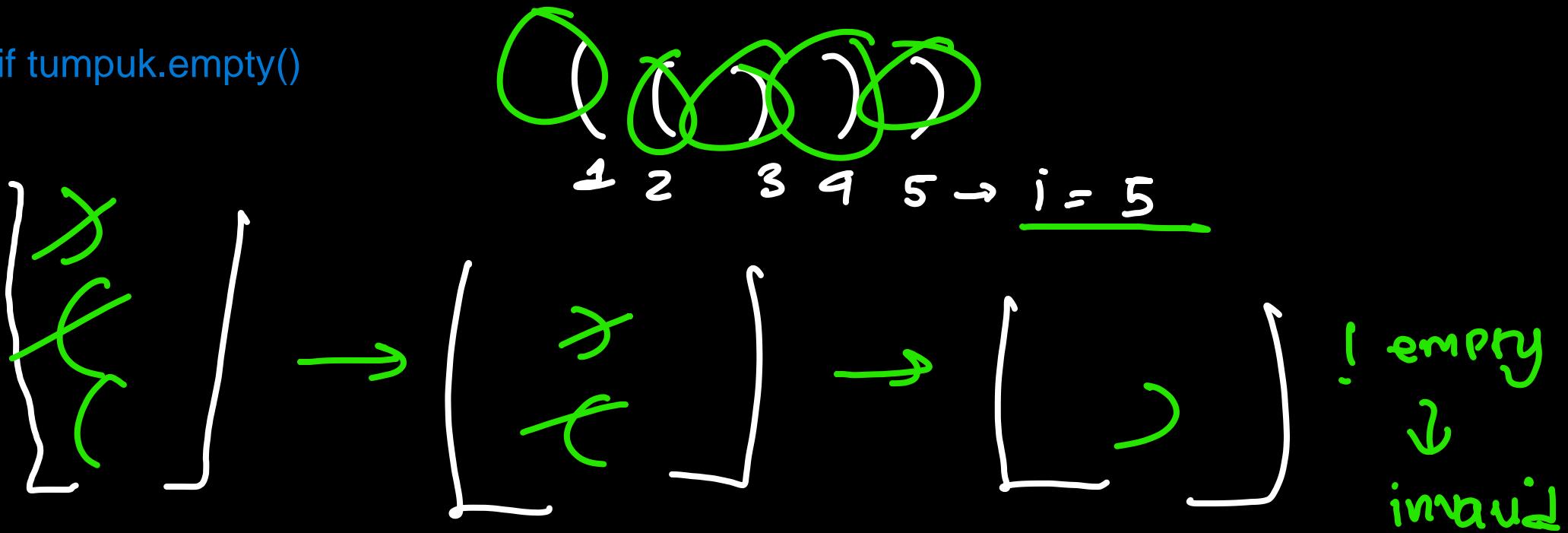
Parenthesis \rightarrow Tanda Kunung

~~(())~~ \rightarrow Valid ?

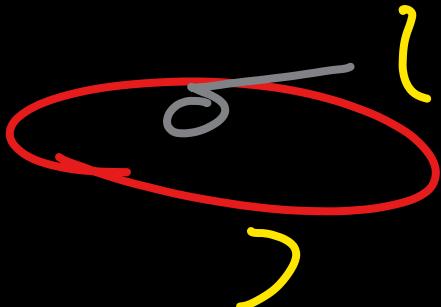
~~(()))~~ \rightarrow Tidak valid



if tumpuk.empty()



If stack empty and iterasi = size str
vaid



))

awal

- * $\geq \rightarrow$ invalid
- * jumlah buka ! = jumlah tutup
- * ())) C \rightarrow
- * ())) ((() cek rule stack

akhirnya

C \rightarrow invalid

SPPPSSSP

int x

const int x

Var, const

↳

global, local scope

