

## OS-III

### Thread Synchronisation

①

Producer - Consumer problem

- Semaphores

- Base + mutex

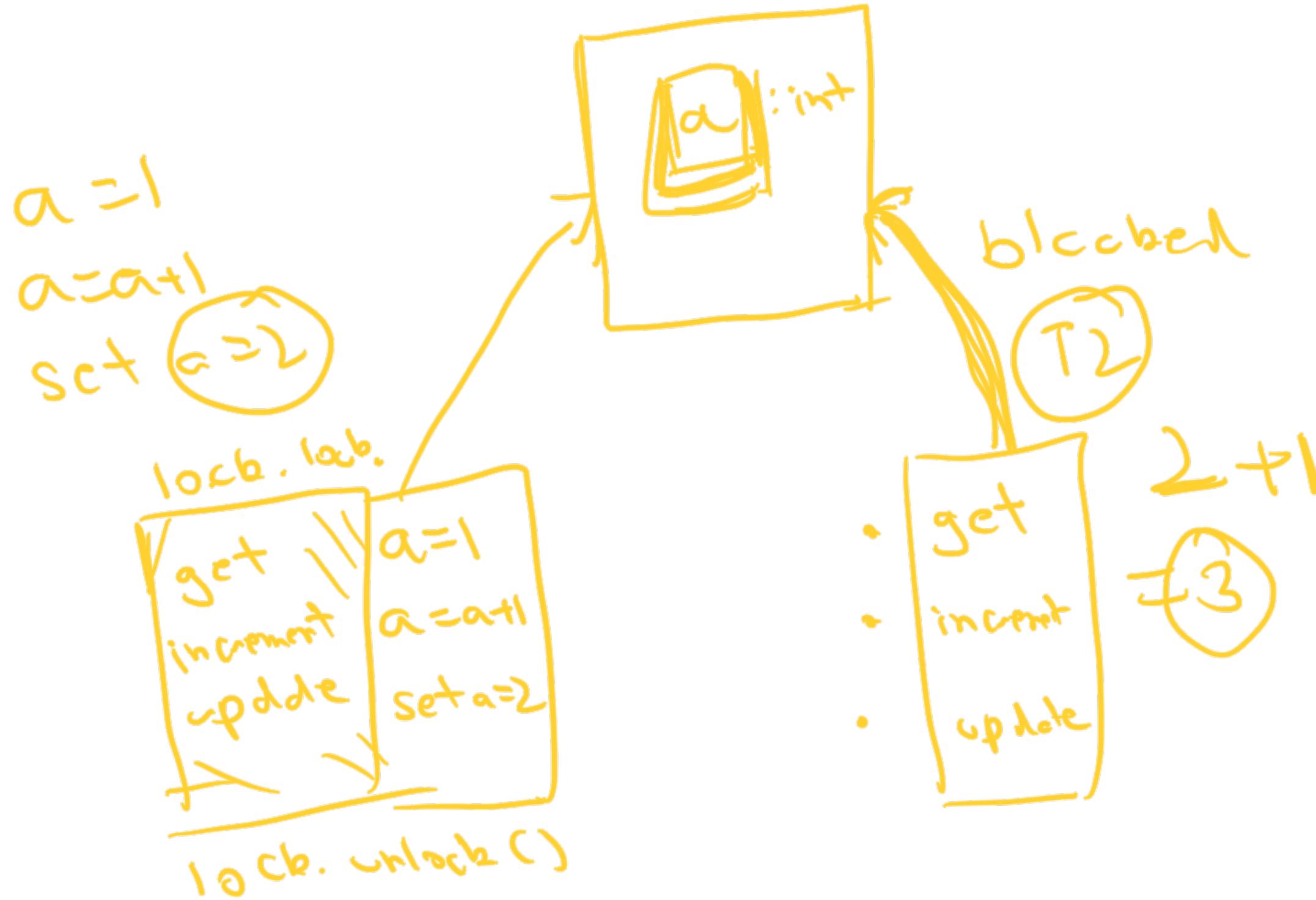
②

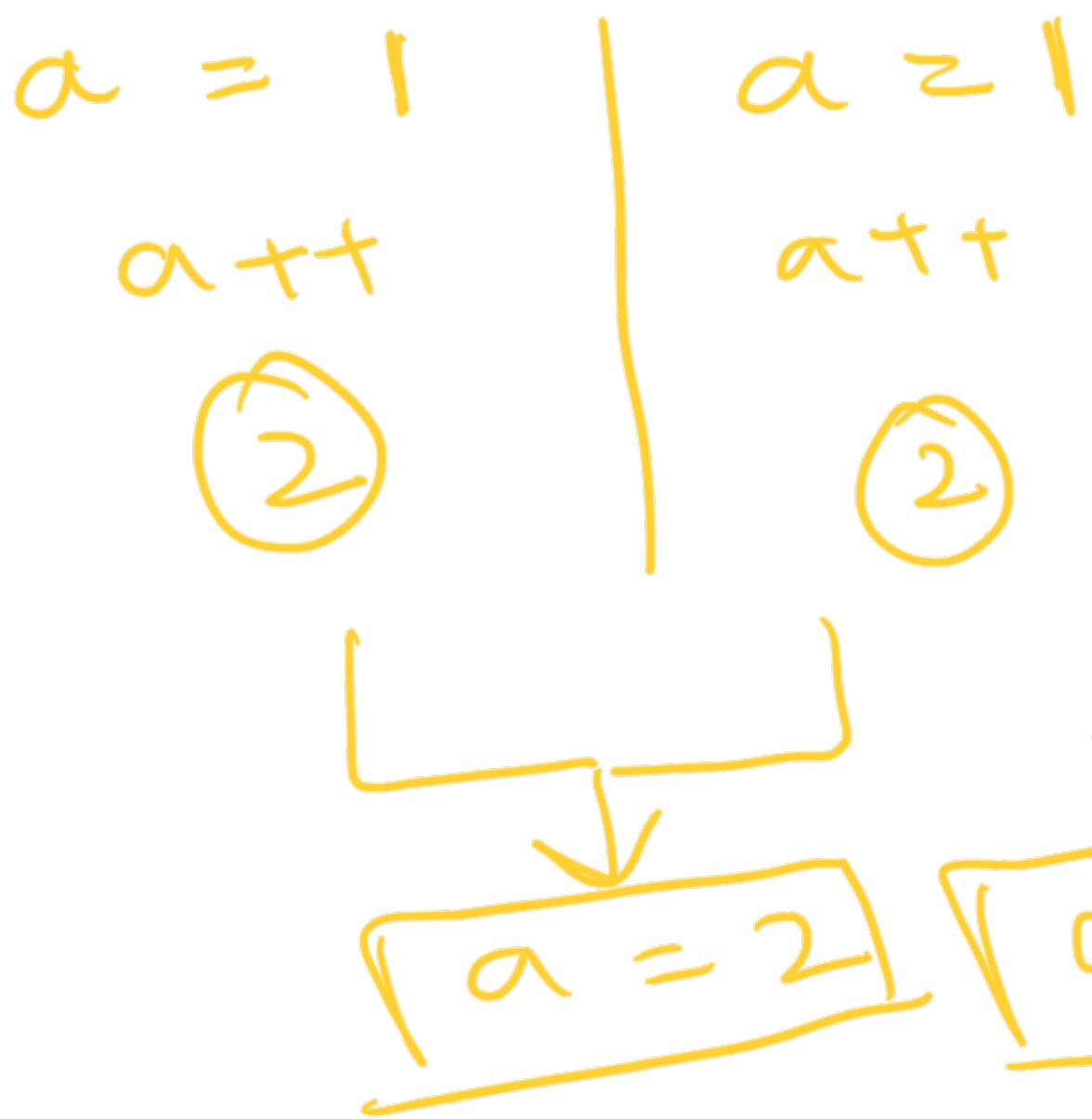
Concurrent DS

- Atomic Integer

- AS atomic integer

## - Concurrent Hash maps





## Thread synchronization

- lock - mutex
- critical section

- Only one thread can access the critical section



## Producers - Consumers

- Digestive

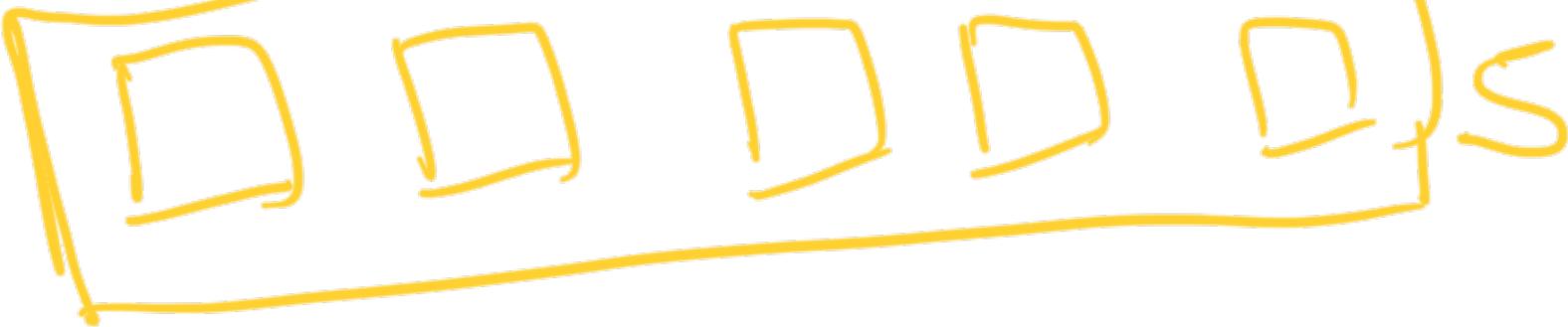




TC on sum  $\epsilon_{ij}$

can only pick up items  
if they are produced

- Bounded-buffer problem



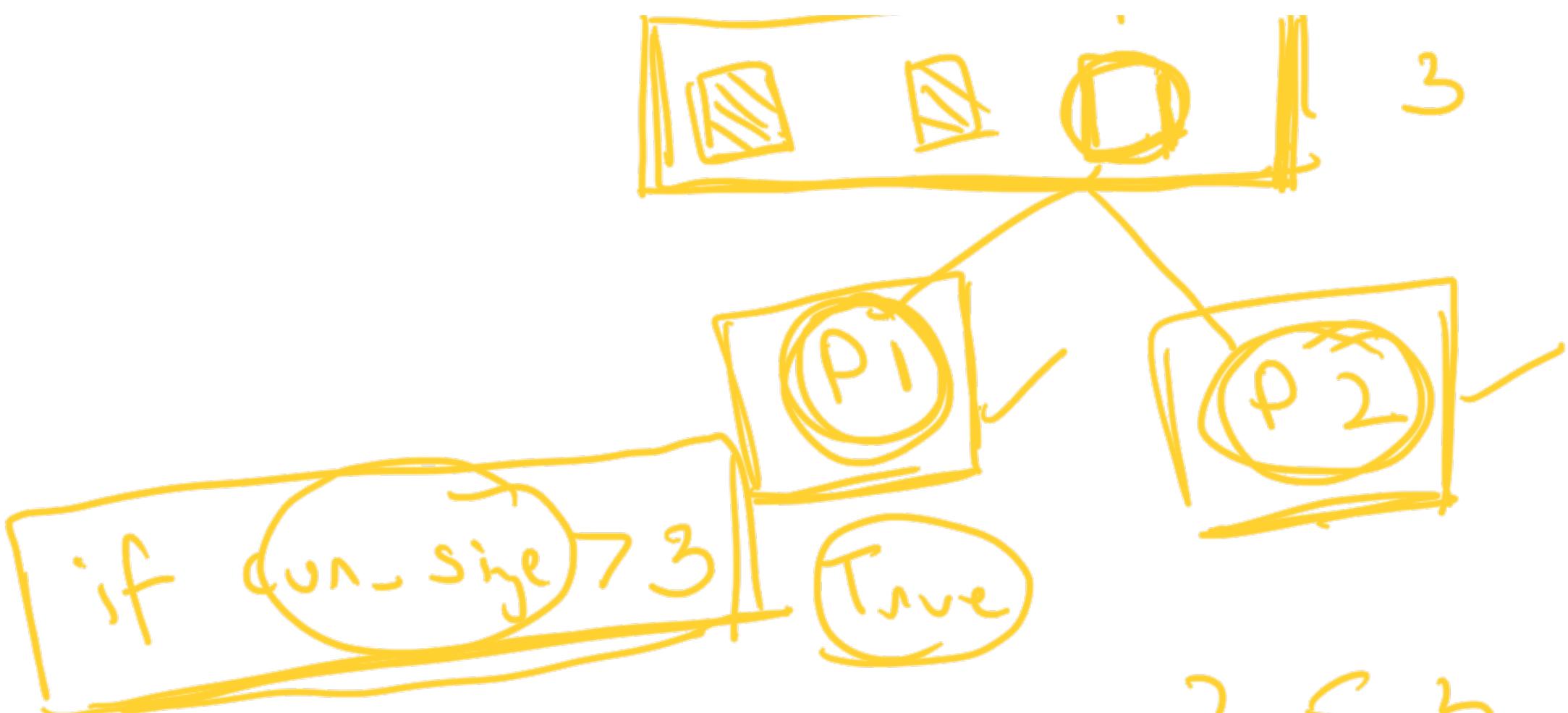
Consumer

```
if cur_size > 0:  
    => get item  
    cur_size - 1
```

Producer

```
if cur_size  
    < buffer  
    Create item  
    cur_size + 1
```





ne hit top  
add to the queue

$2 < 3$

P2  
P1

P1

$\text{cur} = 2$

$\text{if } 2 < 3$

P2

$\text{cur} = 2$

$\text{push}$

$\text{cur} = 3 + 1$

$$\text{curr} = 2 + 1^1 \dots = \underline{\underline{2}}$$

2nd



P 1

$$\text{curr\_size} = 2$$

if 2 < 3

push

P 2

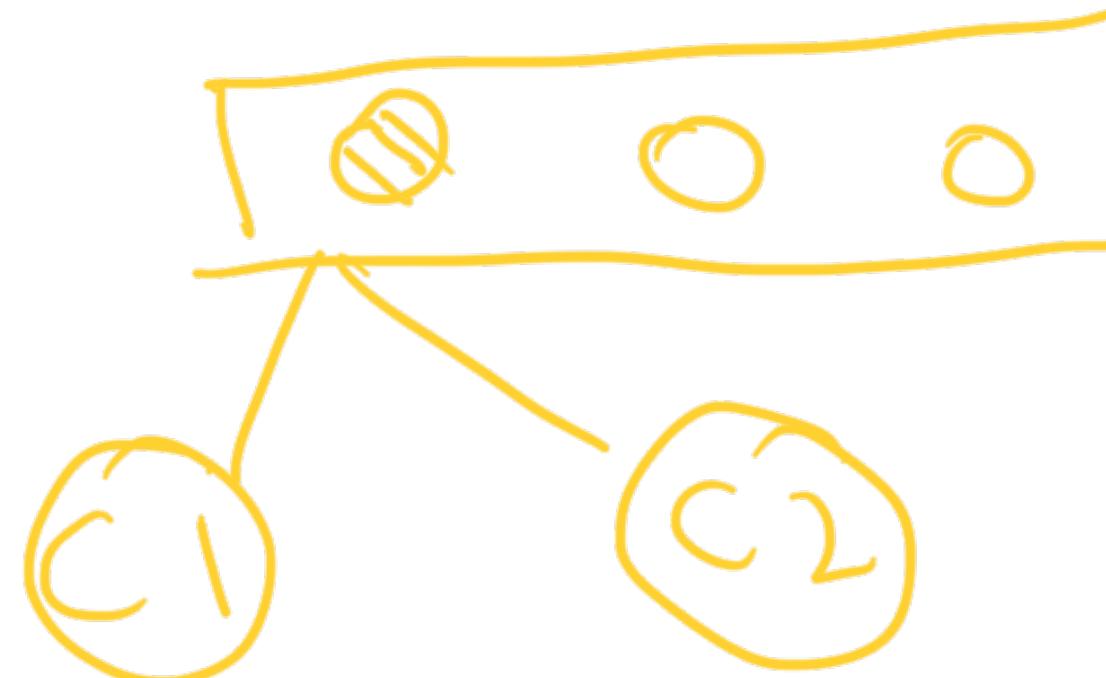
$$\text{curr\_size} = 2$$

if 2 < 3

push

$$2 + 2 \neq 4$$

Consumers



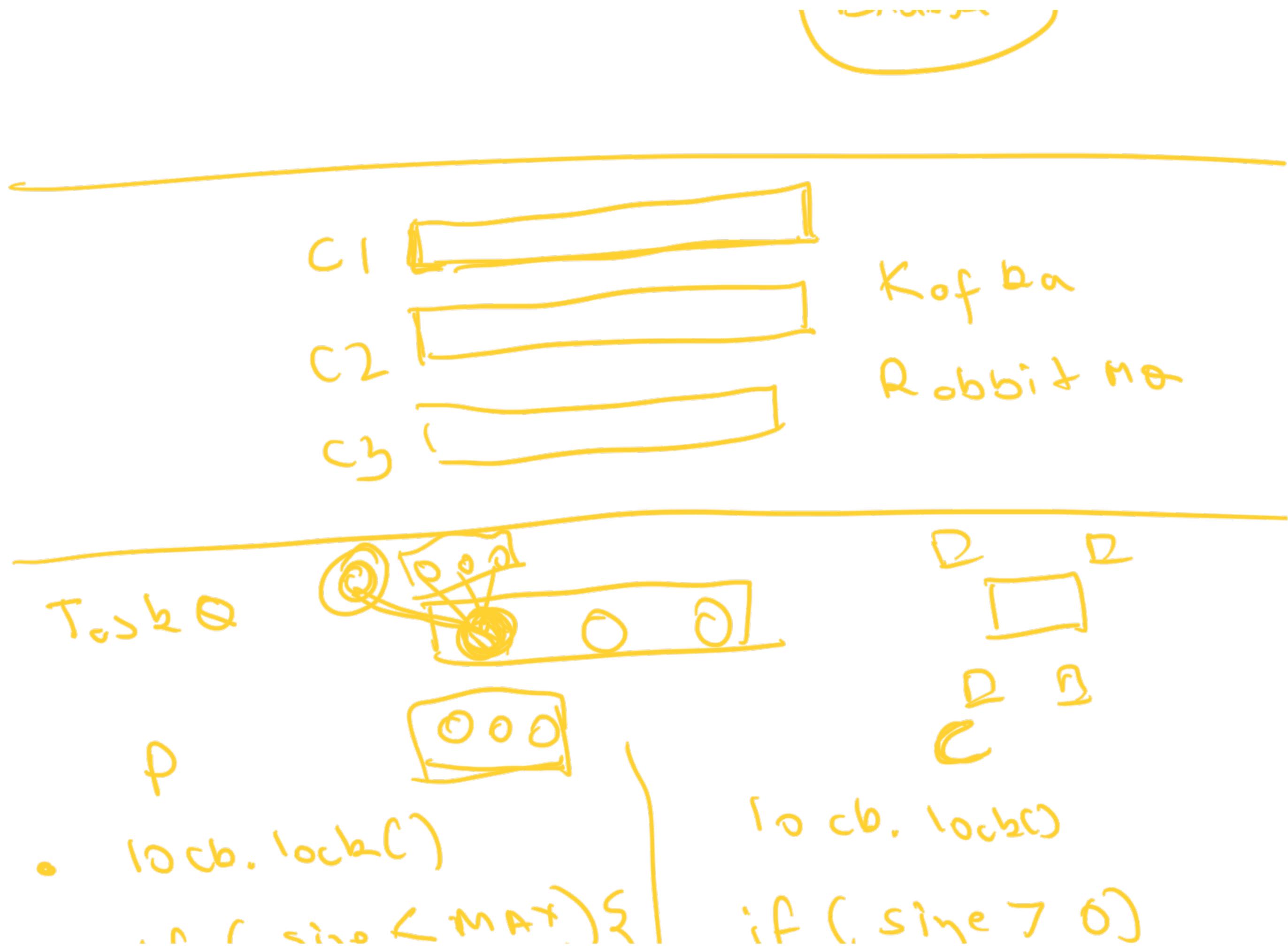
if  $\text{con\_size} > 0 \Rightarrow T$

C1.remove()

if  $\text{con\_size} > 0 \Rightarrow \pi$

C2.remove()

$F_{\text{cancel}}$



if C is

Add to the v

↳

lock, unlock()

Remove

lock, unlock()

→ Synchronisation problem ✓

→ | Sequential

Mutex in Java

= synchronized

= Lock - Reentrant lock

1

PC  $\rho_1$  ob1pm

2

M utpm

$$5:56 = 6:01$$

$$10:26 = 10:51$$

$$\angle 11:50$$

PC problem  $\rightarrow$  20 | exceptions

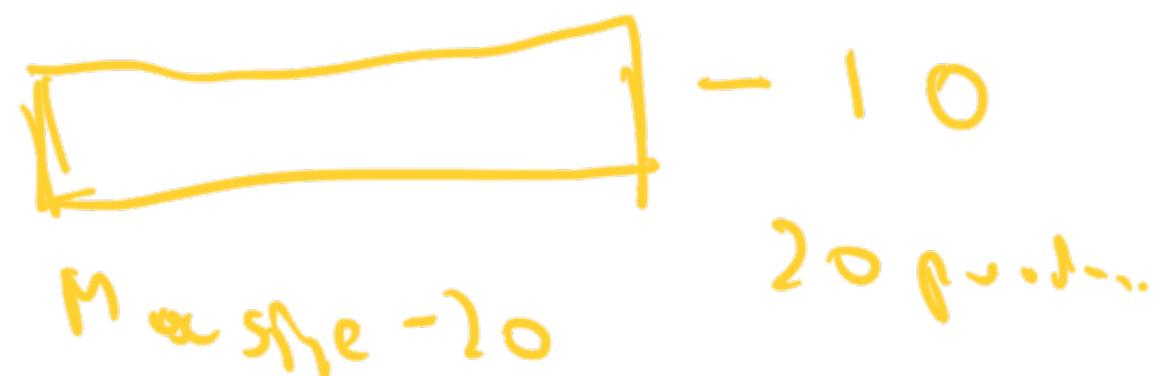
$\rightarrow$  sync. issues

$\rightarrow$  Thread sync

$\rightarrow$  Mutex - synchronized

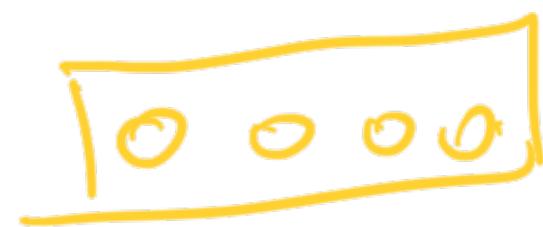
$\rightarrow$  sequential proc

Ideal



1. Process  $\rightarrow$  Max size

$\rightarrow 0$



Dynamic - on the bus

20 max size  
- 0 . . . 11

of size no. of  
producers should change  
0 free slots

② Consumers  $\rightarrow 0 - 20$

$\rightarrow \#$  of filled slots

$P \Rightarrow \#$  of empty slots

$C \Rightarrow \#$  of filled slots

①

what do we need?

Maintain these sets extending the CS

# SEMAPHORES

## Semaphores

→ ways to synchronize multiple threads

Semaphore (10)

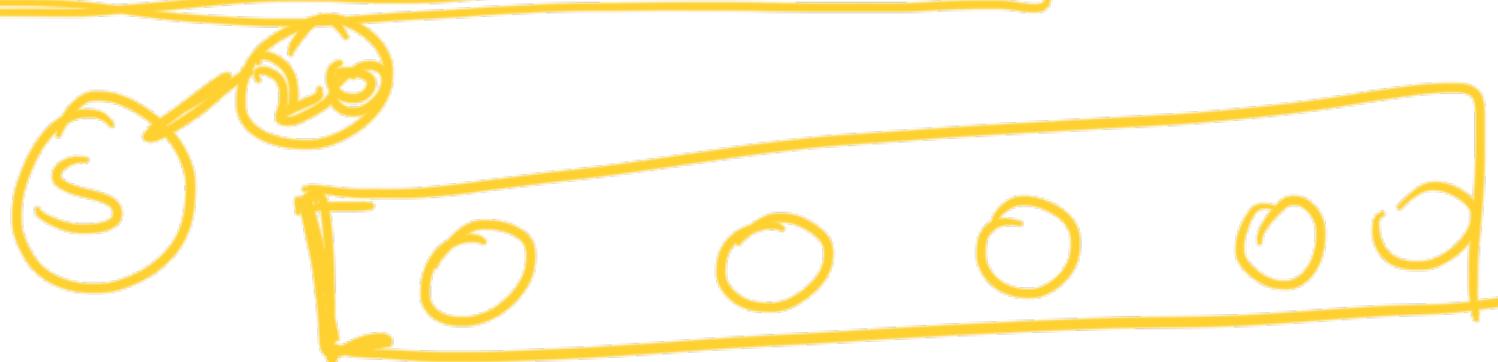
How many threads  
semaphore controls

Semaphore (0) = mutex

→ semaphore . acquire()

→ semaphore . release()

Semaphore (Semaphore)  $\rightarrow$  b processes can be active at the same time





if produce 1

→ signal to the C

Pro duce

C → Pro duce

Con sum er

→ Con sume

→ signal to cursor | → send signal  
to produce.

P [Semaphore]

while (True) {

- ① a acquire semaphore
- ② create unit & add to v
- ③ signal  
on sumen.

C [Semaphore]

while (True) {

- ① active
- ② con sume
- ③ send a signal  
to produc.

Semaphore (0)

Semaphore 20

① Semaphore (19)

$0 \rightarrow 1$

$\hookrightarrow$  Semaphore Consum. (1)



$\Rightarrow$  decrease our count

$\Rightarrow$  increase consume count

How many instances can run  
at a moment?

$S_p(20)$  - 20 max threads

Can be stretched