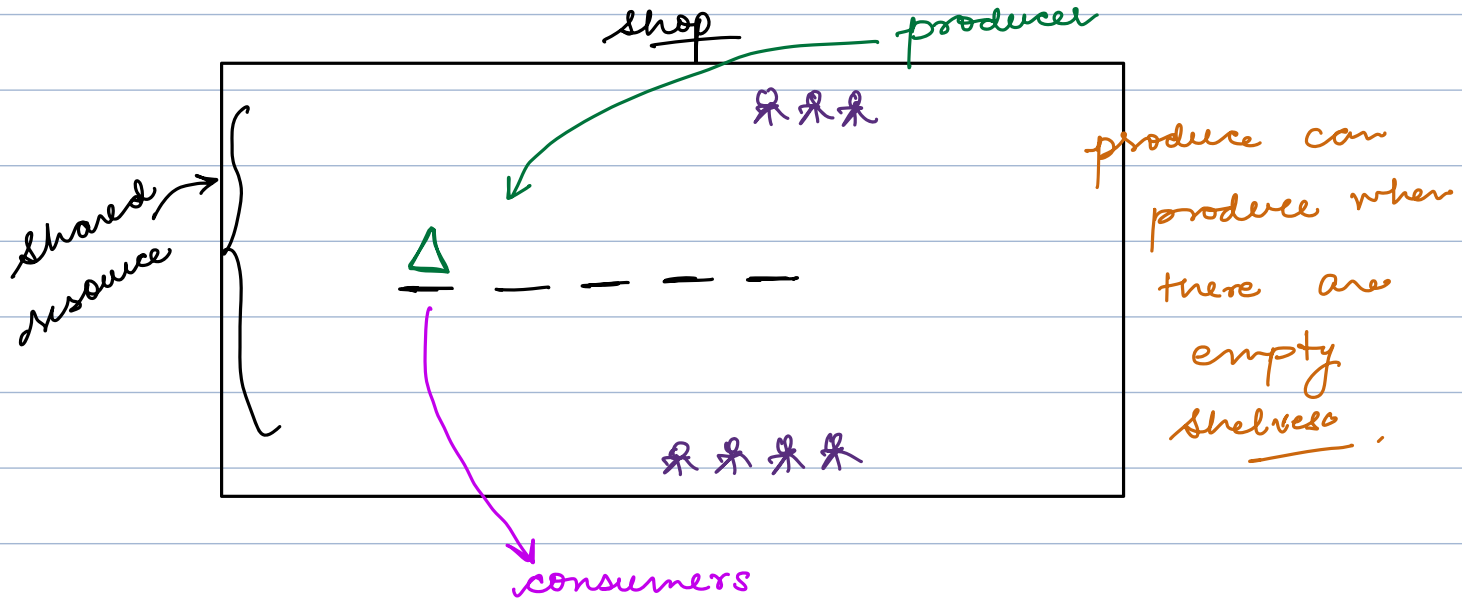
Agenda

- Producer Consumer
- Semaphores.
- Solve 2 leetcode problems using semaphore.

Producer Consumer

↓

Is restricting # 1 thread in CS best idea ?



↓

No item to consume.

{ Producer → Store
Consumer

8 produce

0

→ if (size < maxSize) 5

add item

8

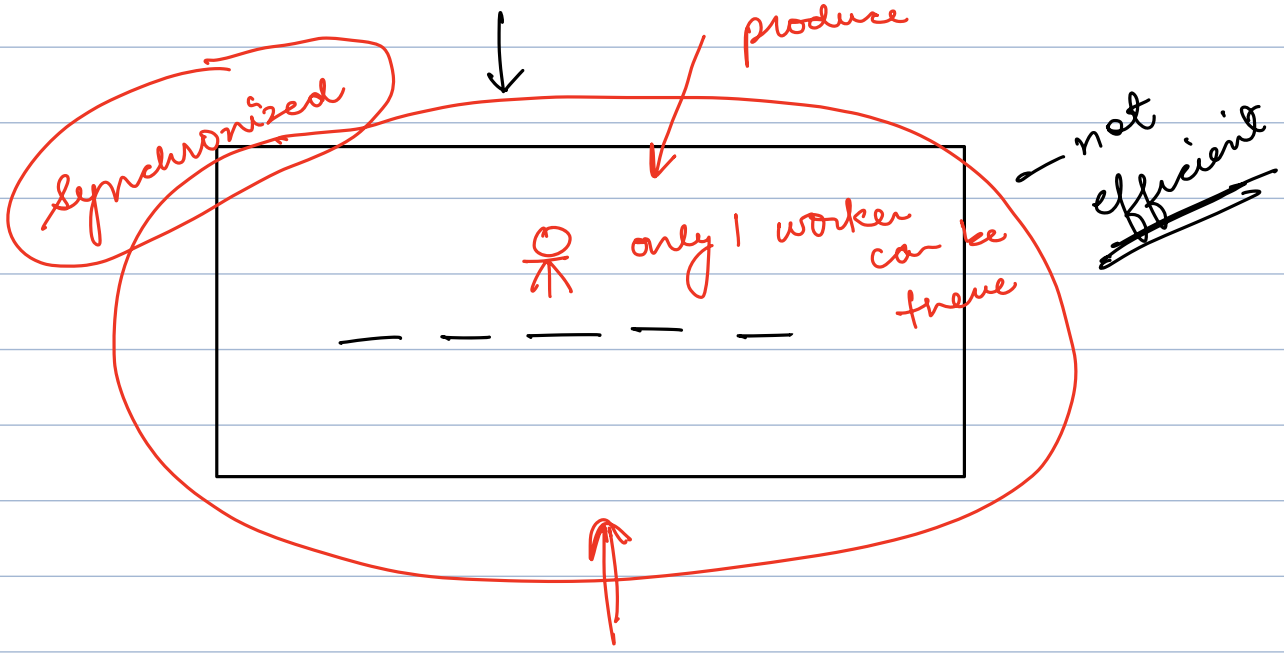
20 cons.

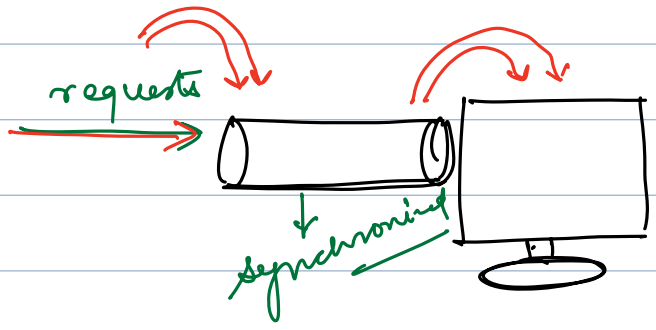
1 ↓ 20 8

if (size > 0)

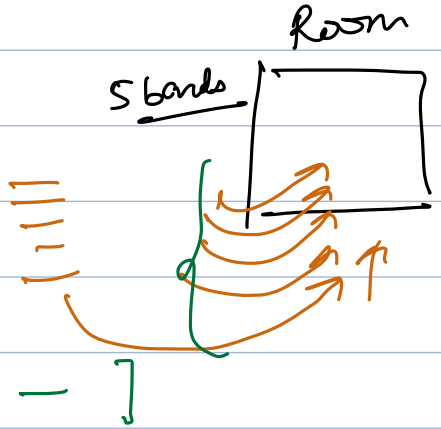
consume.

-1 out of index bounds.





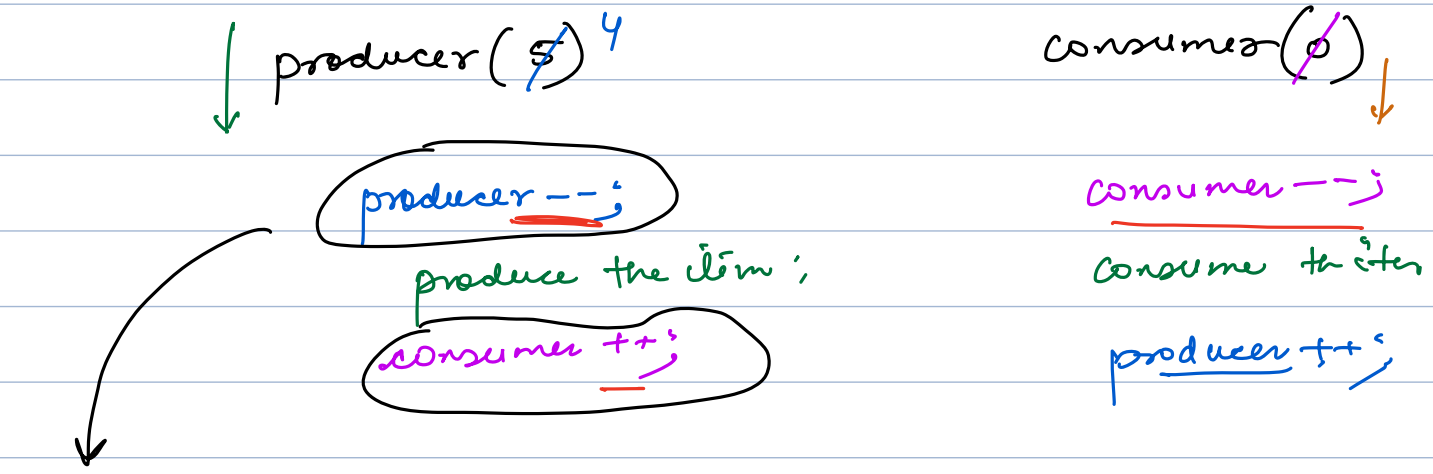
allow more than one thread in CS within limits.



Locks + count

① # consumers + # producers <= 5 inside the CS

② producer → consumer
signal



Semaphores

mutex + count

more than 1 thread in CS.

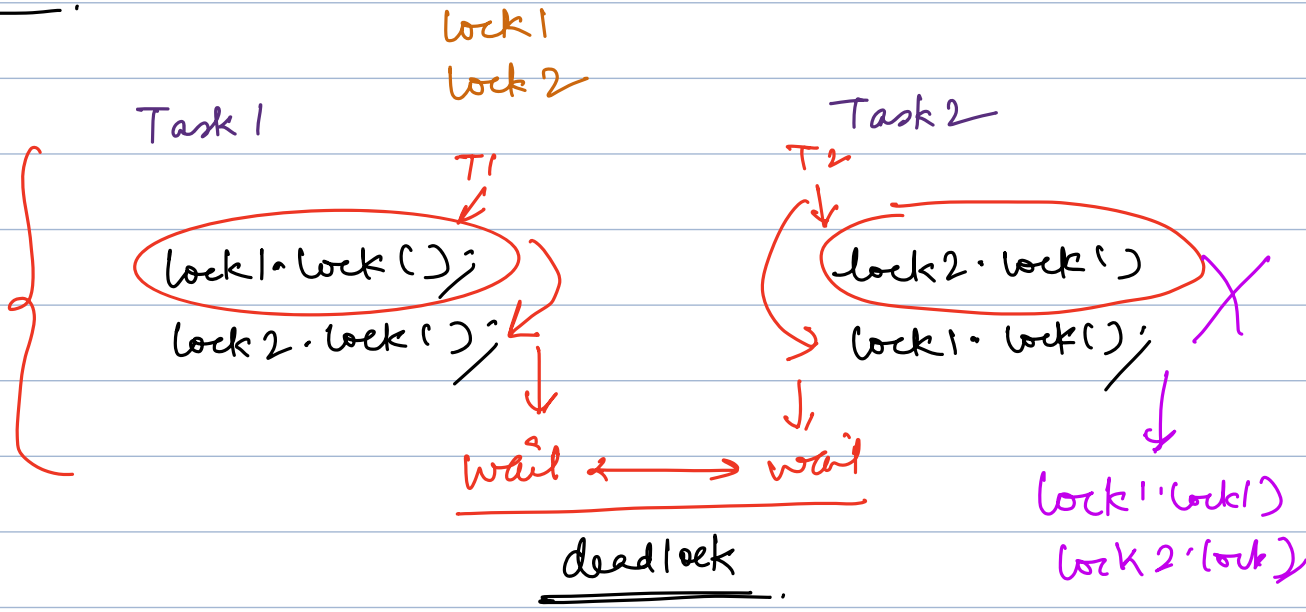
```
Semaphore s = new Semaphore(5);

s.acquire()
        if curr-val of counter > 0
            curr_val = -1
            allow task
        else wait.

s.release()
        count++;
```

10:36 pm

Deadlock



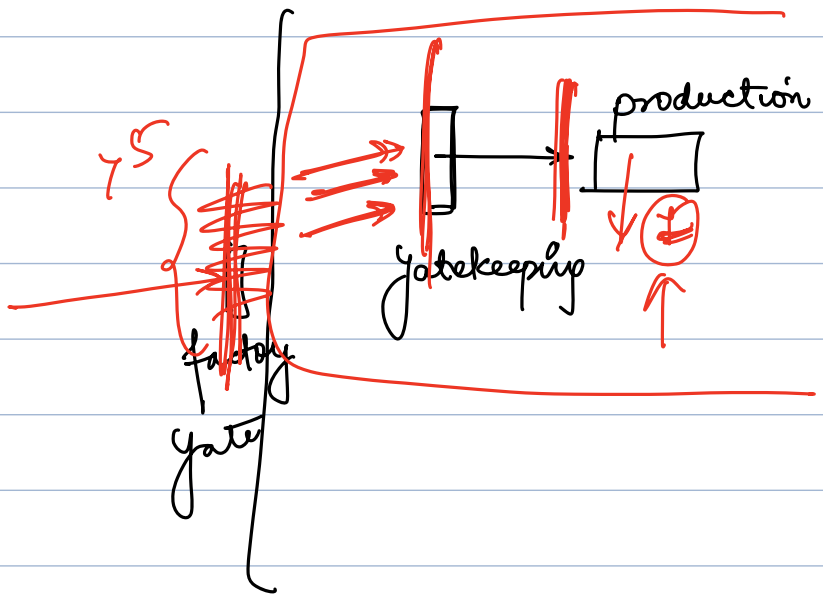
① take locks in some defined order.

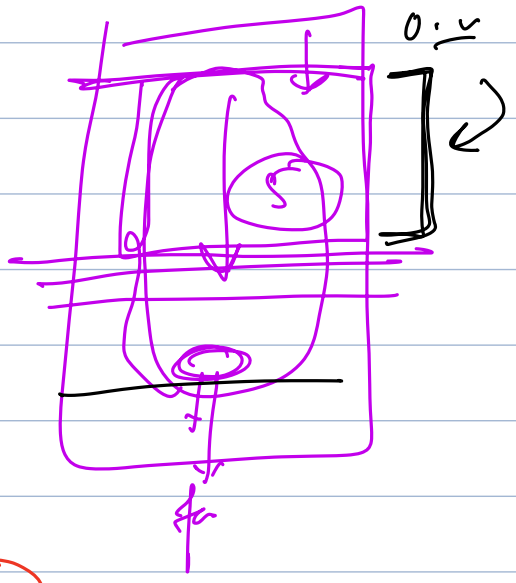
② lock1.tryLock(1, sec) timeouts

{ collection : concurrentHashMap
Atomic Datatypes.

[Generics + Collections + Streams API + Exception]

solve — Lab session — 2hrs — Assi.





CAS

critical valu