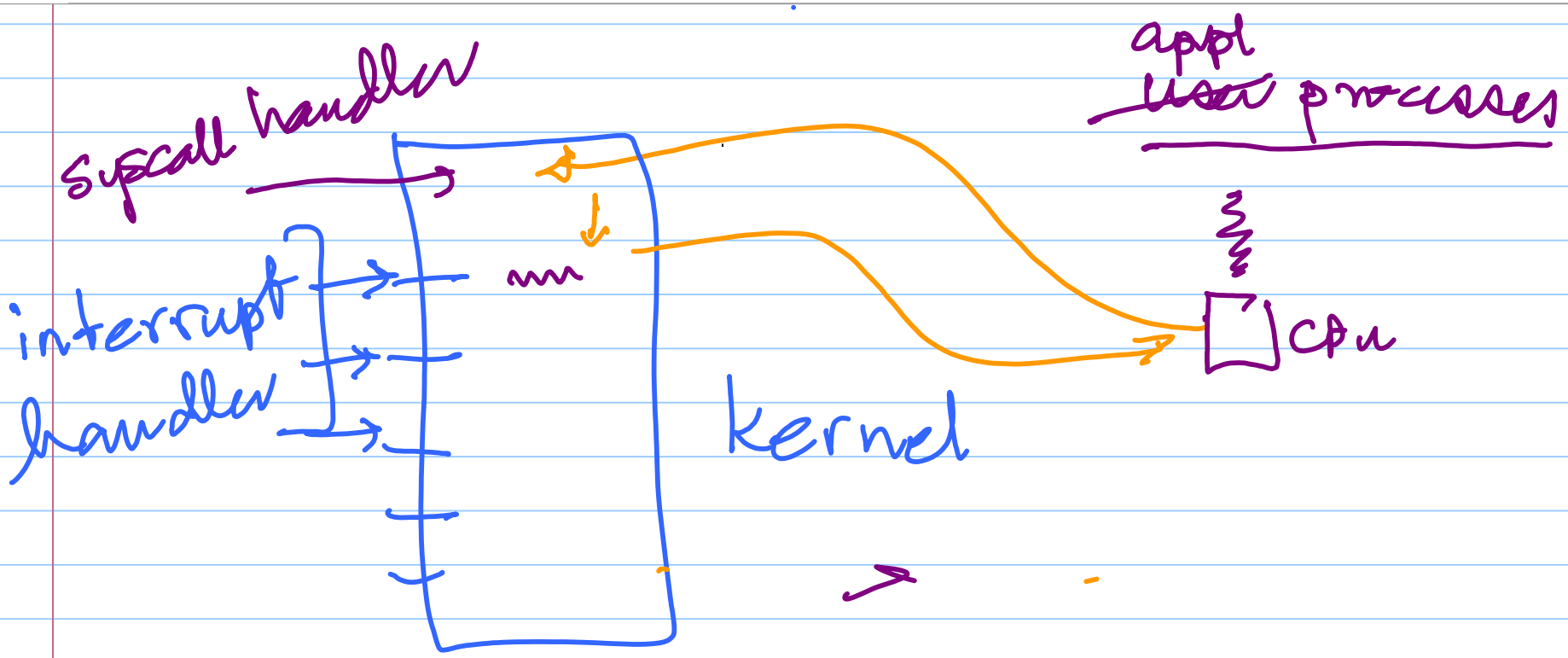
CSE 531: Distributed and Multiprocessor Operating Systems

Class: 08 Date: 9/21

Fall 2020

Note Title

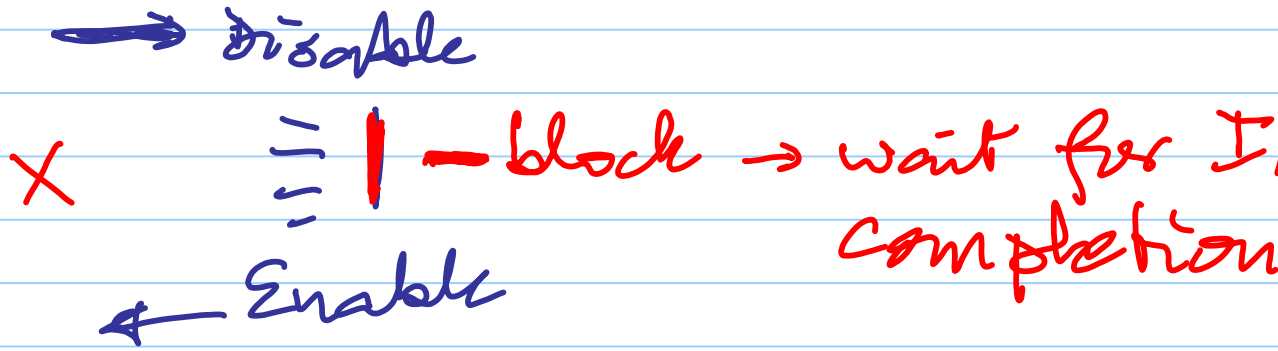


app
syscall
kernel
1
2
3

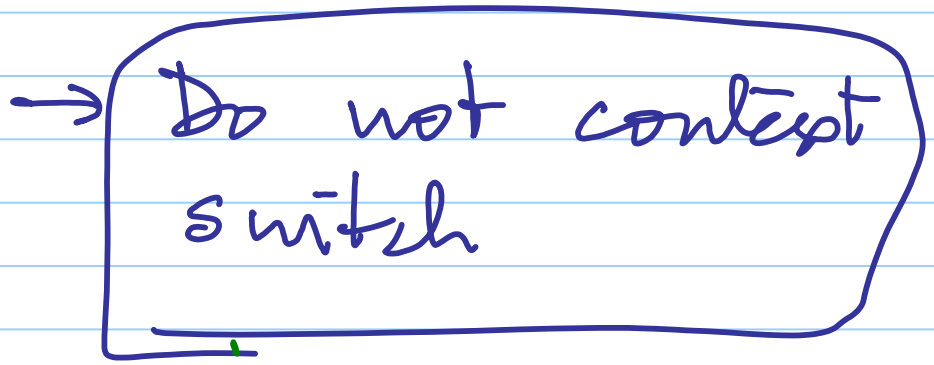
1

Single CPU kernel

→ Disable Int while kernel executes



Non preemptive kernel



→ kernel

→ flag ← do not context switch

⋮ → can be made to work

⋮

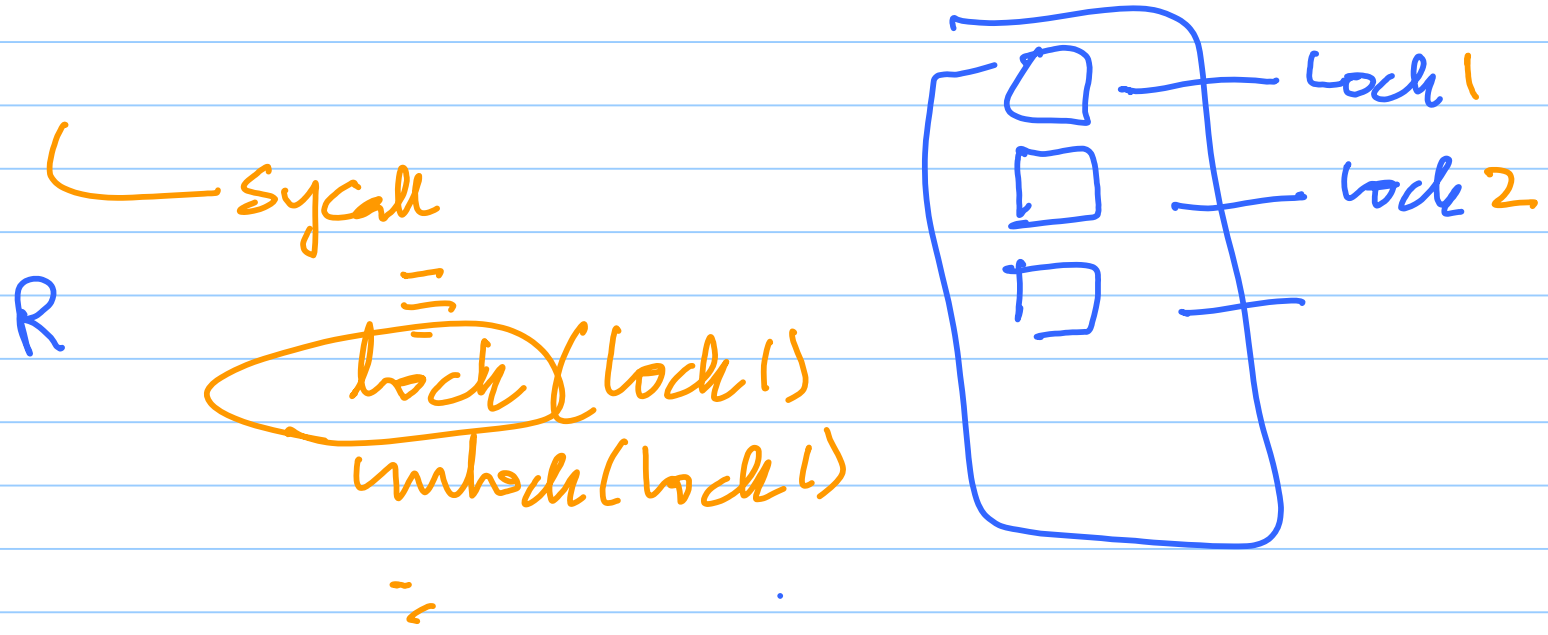
Biglock

→ lock (kernel lock)

• unlock

① preemptible kernel

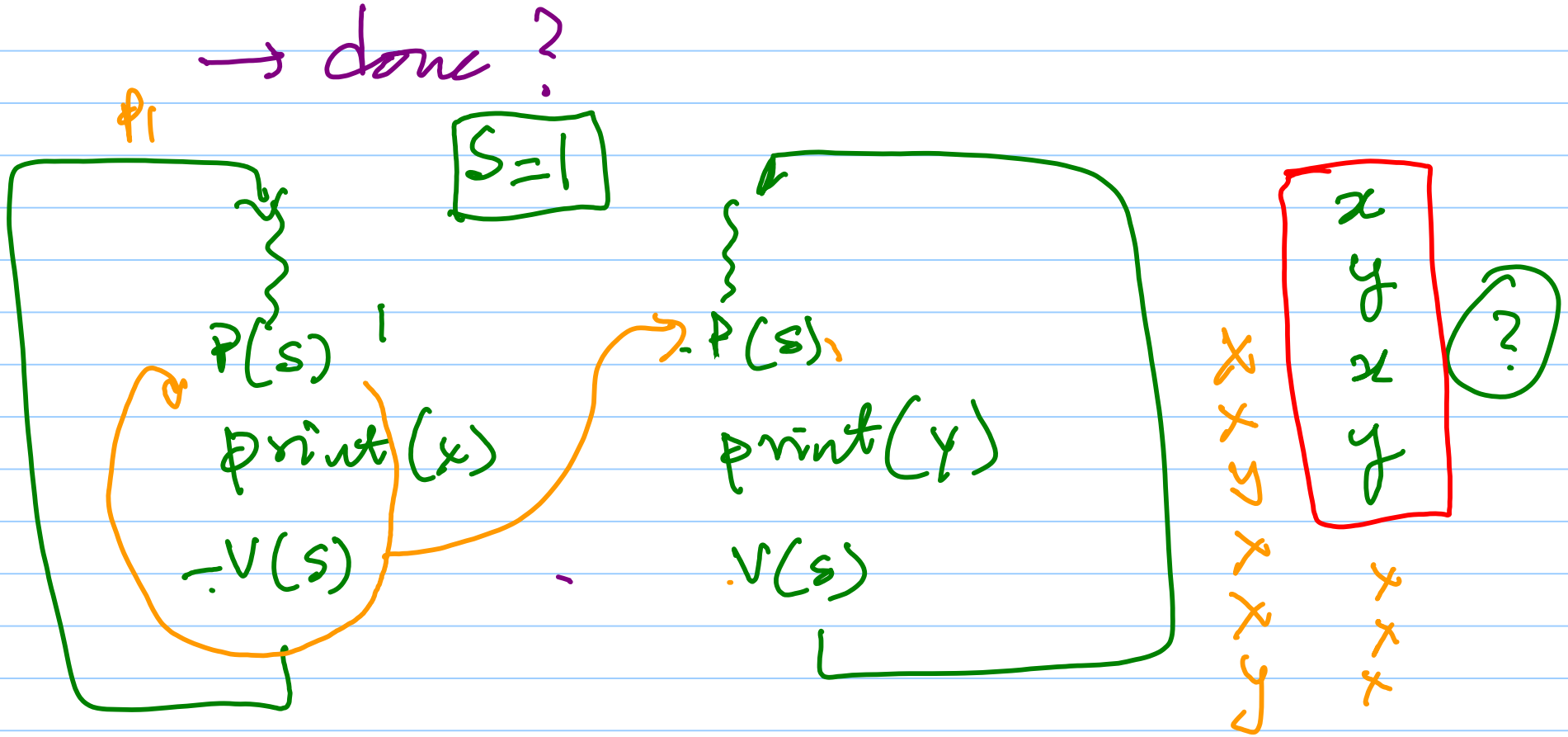
② fine grain locks



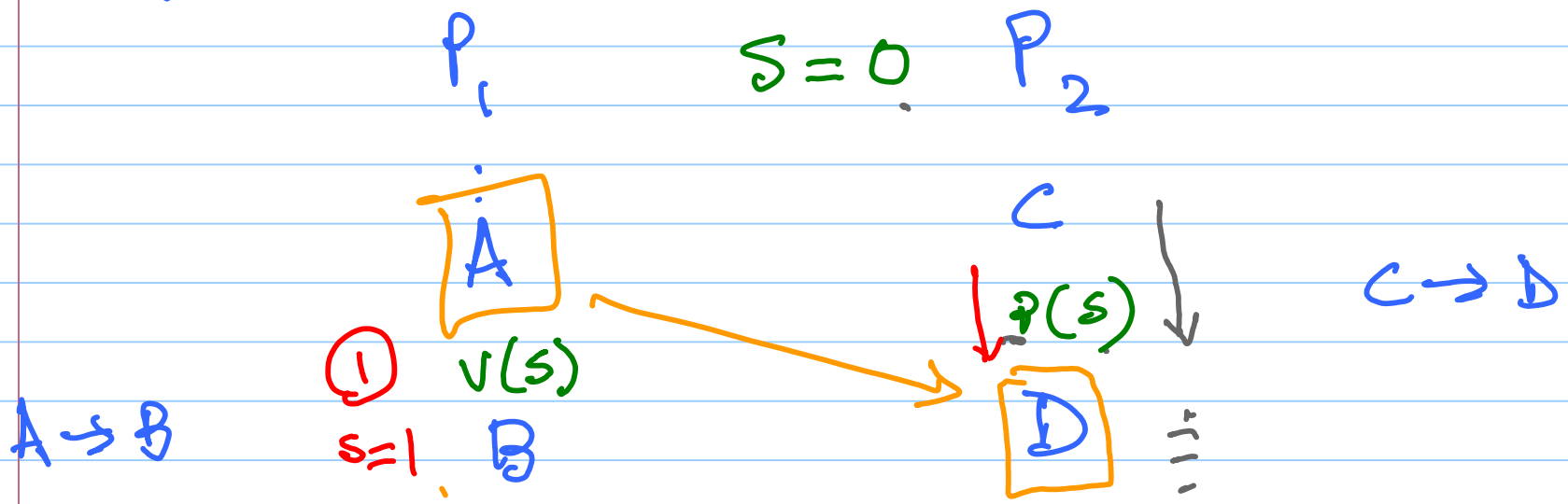
lock
spinlock
mutex lock
block()
longer CS
test + set
semaphore
small critical section

Semaphores as mutex locks

→ done ?

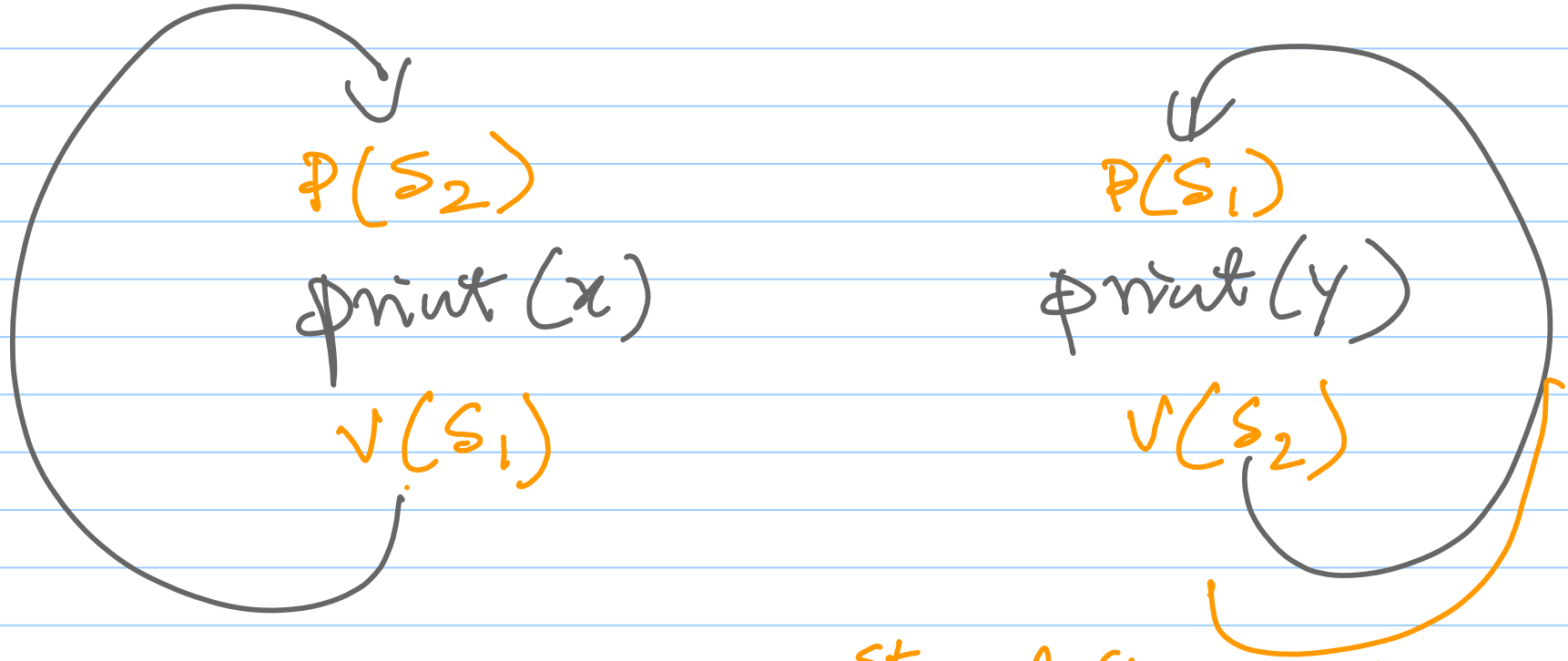


Synchronization



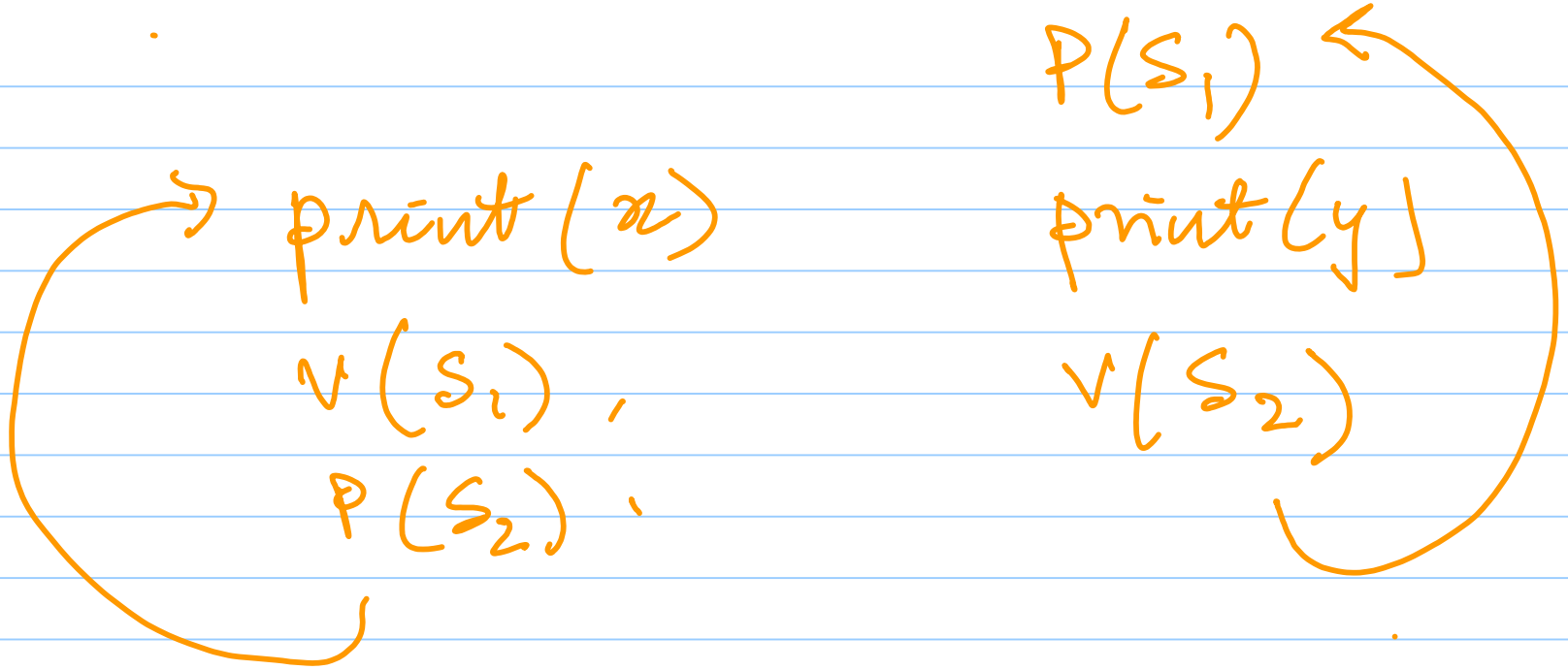
D can happen only After A is completed

$S_1 = 0$ $S_2 = 1$

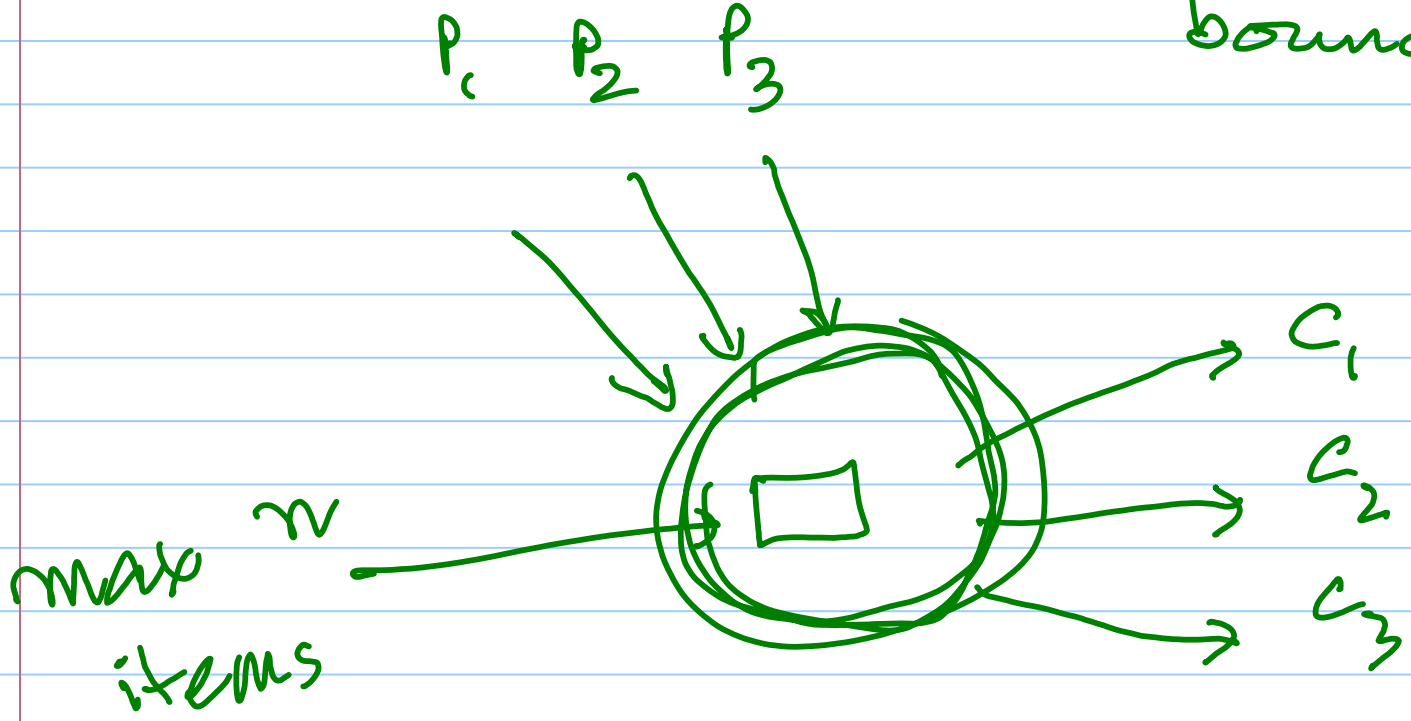


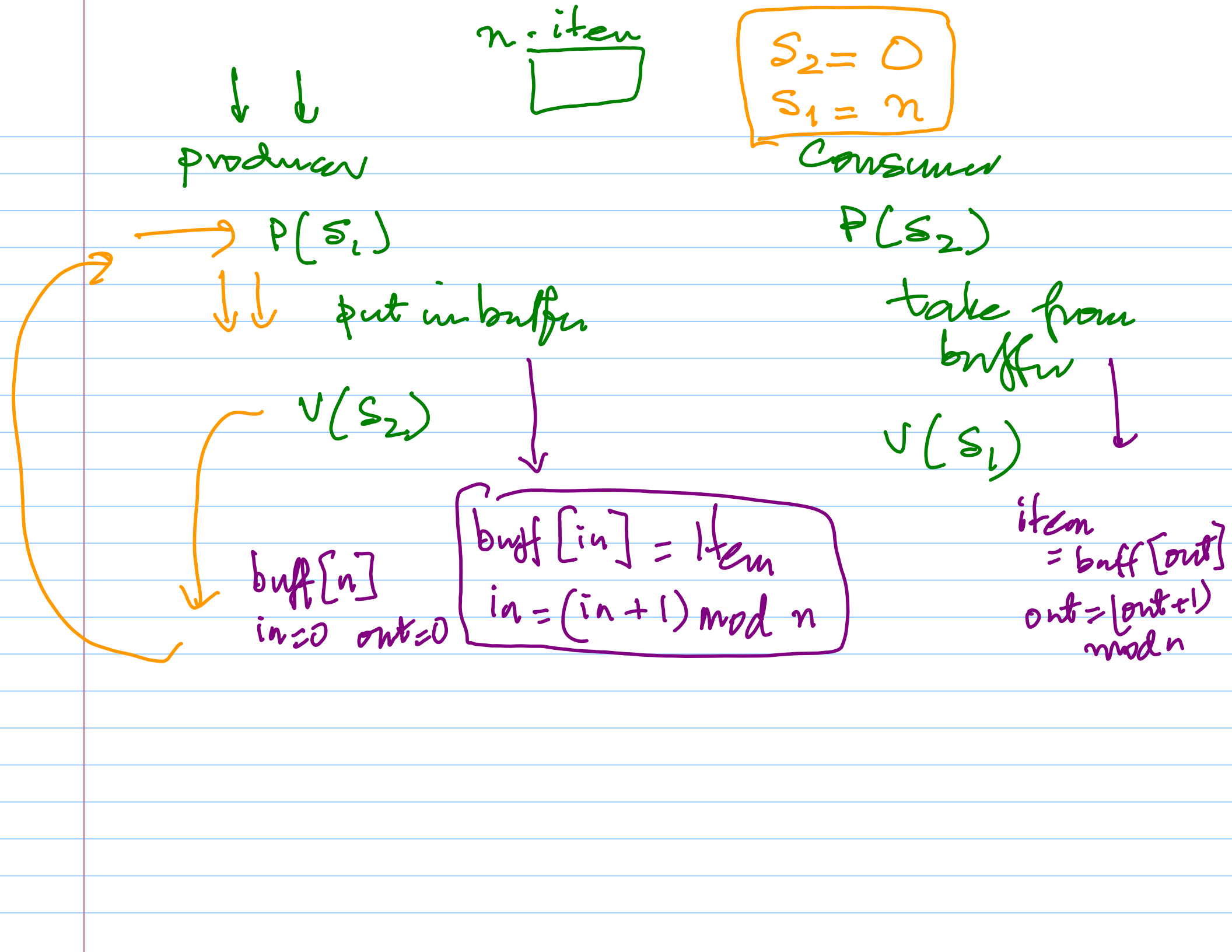
→ x will print 1st, followed by y, x, y, x

$S_1 = 0 \quad S_2 = 1$



producer - consumer (aka bounded buffer)

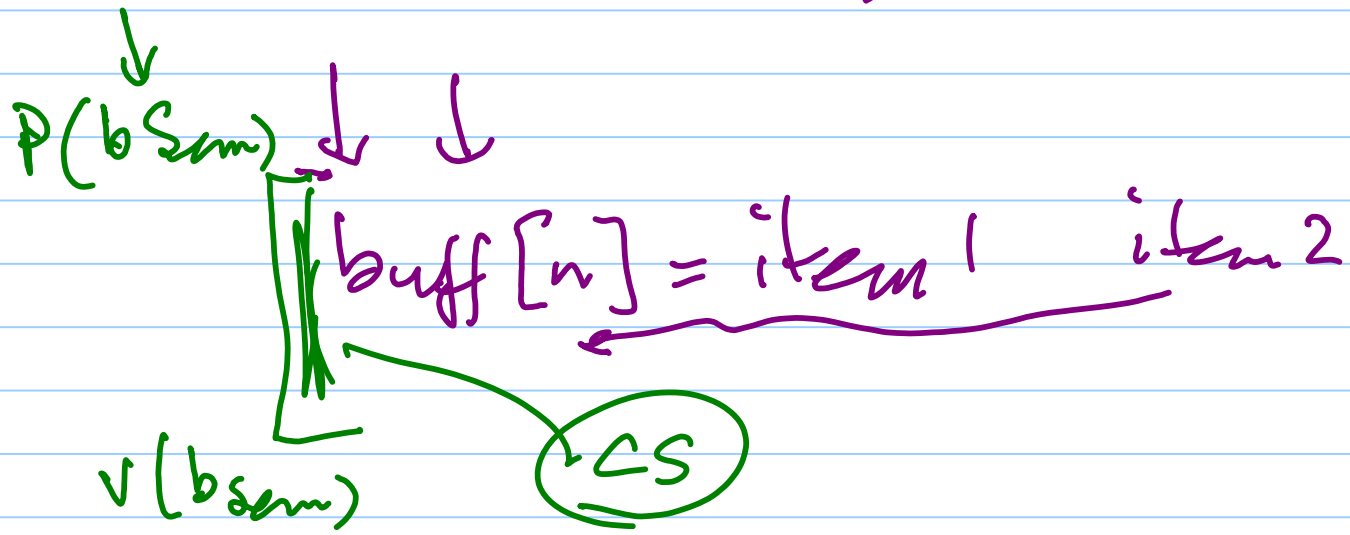


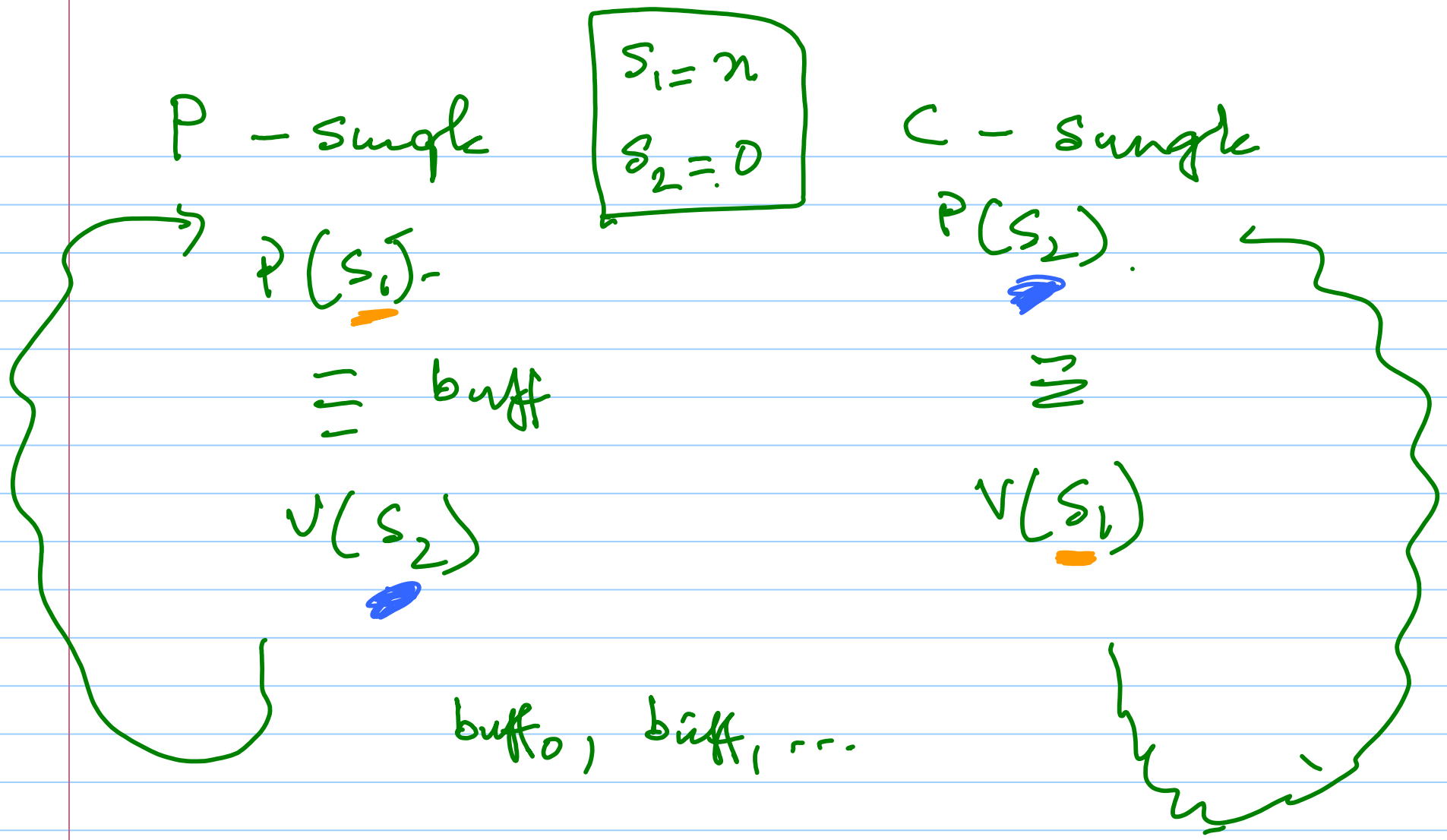

n · item
$S_2 = 0$
$S_1 = n$
producer
consumer
P(S_1)
put in buffer
V(S_2)
buff[n]
in=0 out=0
buff[in] = item
in = (in+1) mod n
P(S_2)
take from buffer
V(S_1)
item = buff[out]
out = (out+1) mod n

Prod Cons

1 prod, 1 cons → works

1 multiple prod/cons race condition




$S_1 = n$
$S_2 = 0$
P - single
P(S_1)
buff
V(S_2)
C - single
P(S_2)
V(S_1)
buff$_0$, buff$_1$ ...

- checking values of Semaphores ?

P(s) → NO !

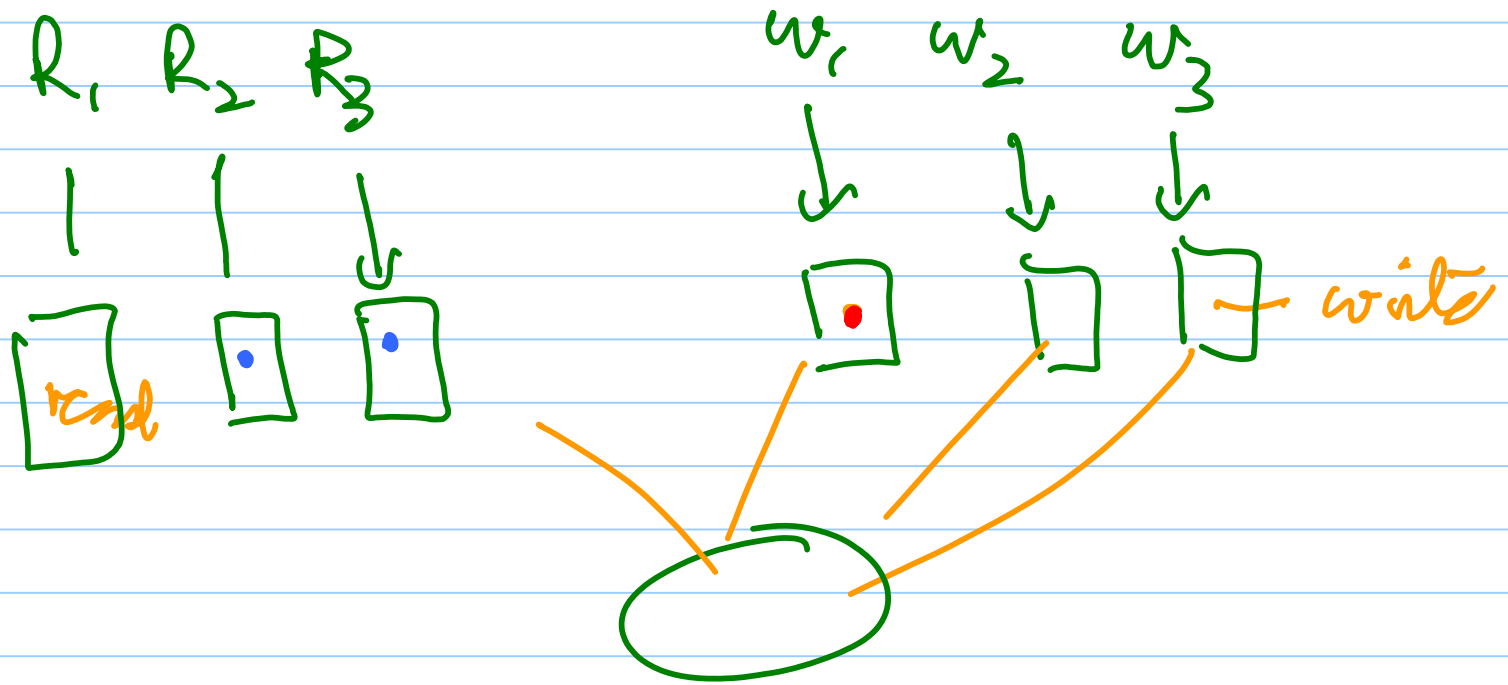
Proof

$P(S_i)$

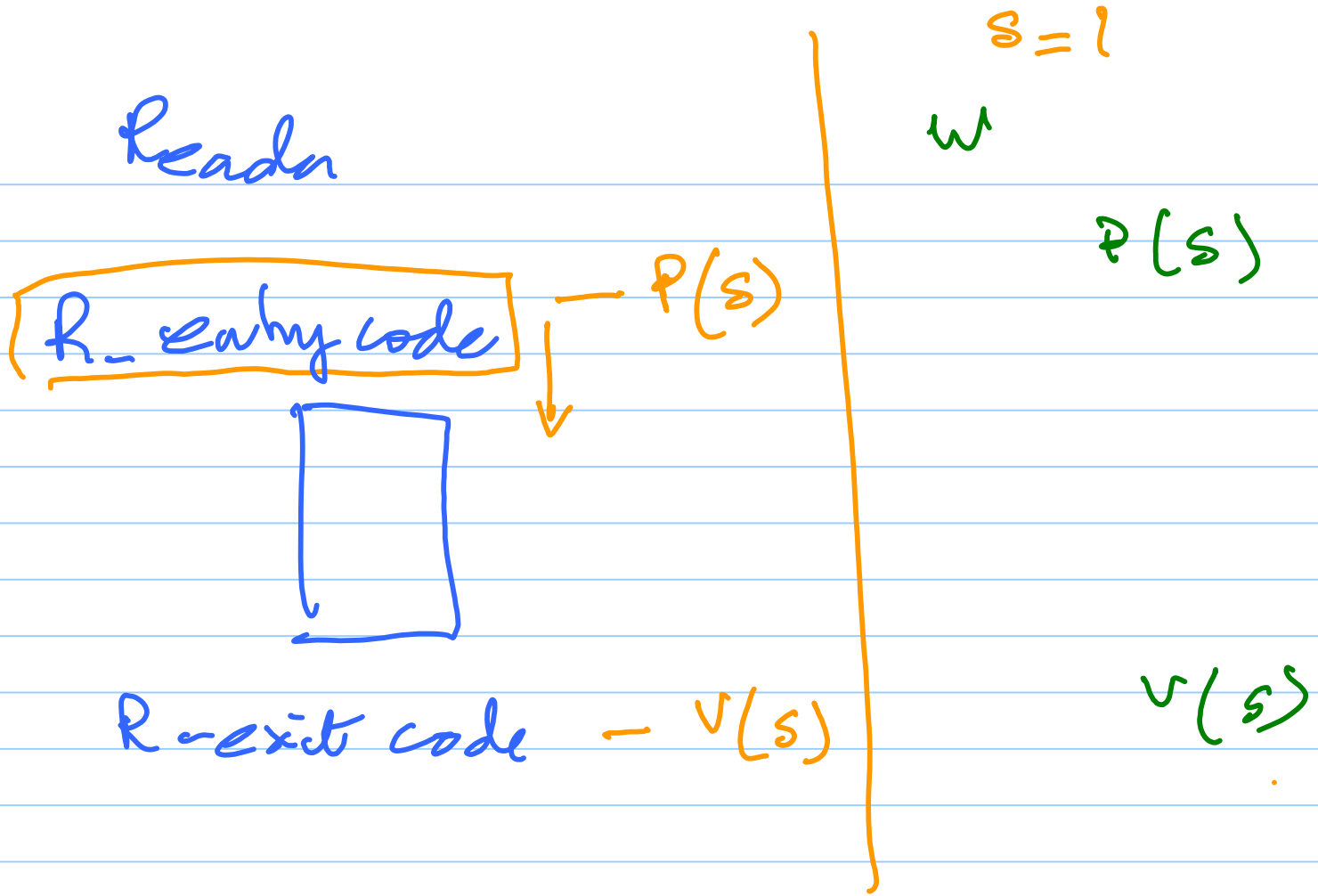
Count ++

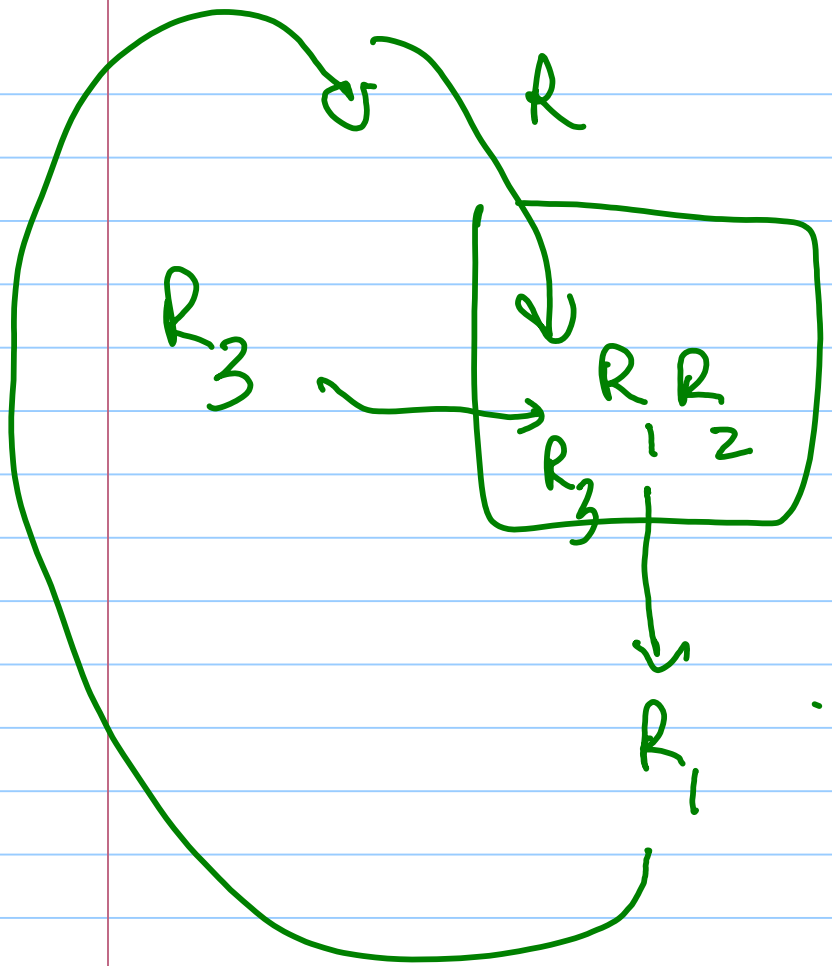
$V(S_2)$

Readers + Writers



multiple readers OR single writer


s = 1
Reader
w
P(s)
R - entry code
P(s)
R - exit code
V(s)
V(s)



W ← starvation

(1) if W is waiting new readers are not allowed to enter

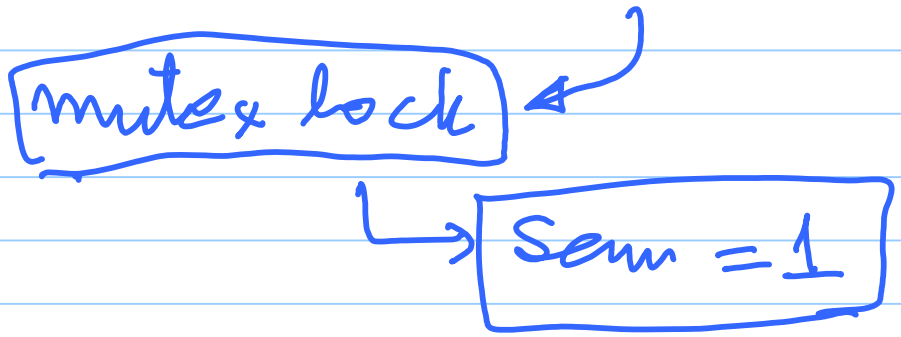
(2) last R out must give W preference

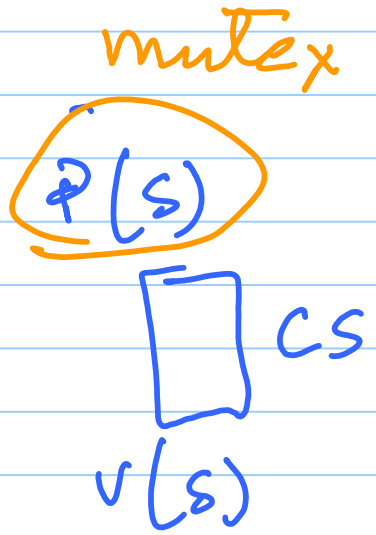
(3) W exit makes all waiting R enter at same time

Mutex

mutex → pthreads.

data struct





pthreads

create

clone

mutex

lock

unlock

futex

monitor

cond_wait

cond_sig