CSE 330: Operating Systems

Class: 13 **Date:** 2/20

Spring 2018

Note Title

Semaphore programming

- mutual exclusion
- synchro

Producer consumer

multiple prod + multiple cons

put

P(full)

P(mutex)

buff[in] = item

in++

V(mutex)

V(empty)

init to 1

get

P(empty)

P(mutex)

item = buff[out]

out++

V(mutex)

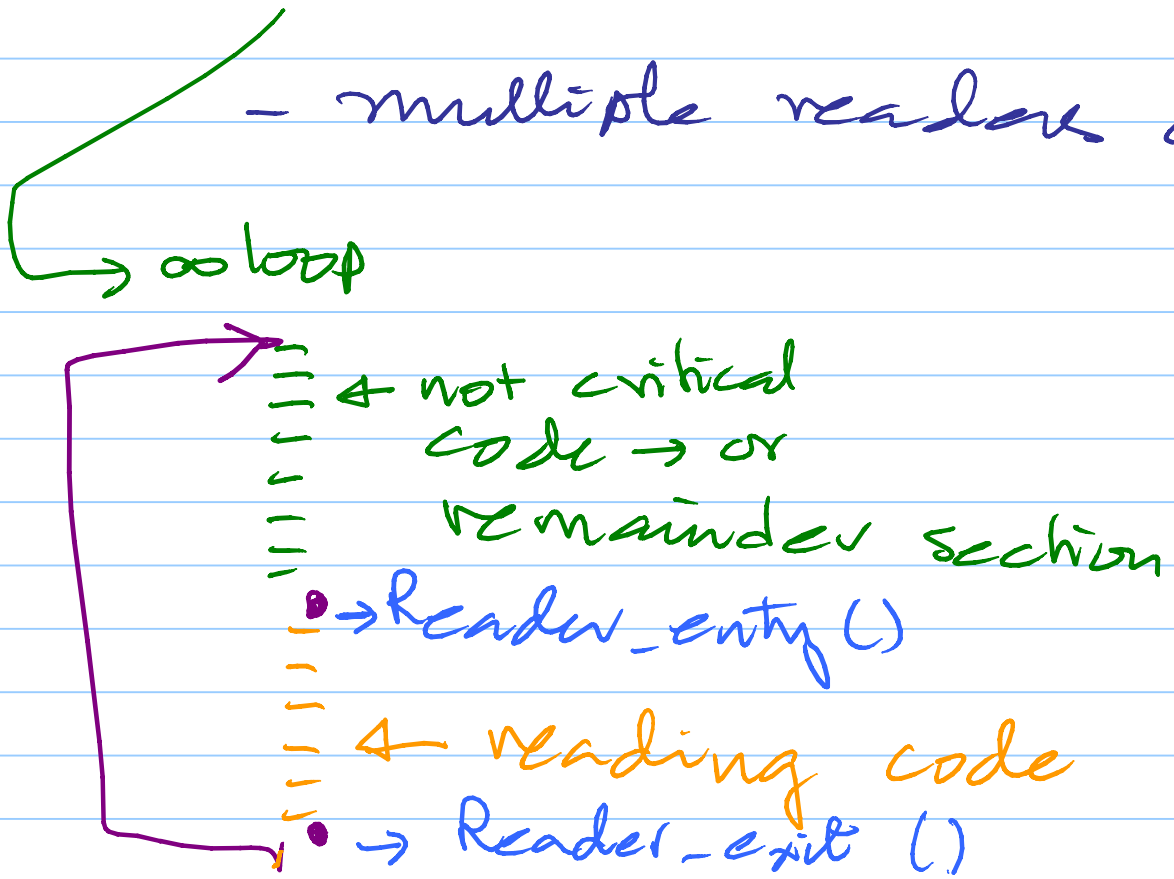
V(full)

do not use 2 semaphores if buffer is a linked list

mutex P for prod, mutex C for consumers

The readers & writers problem

- multiple readers & multiple writers



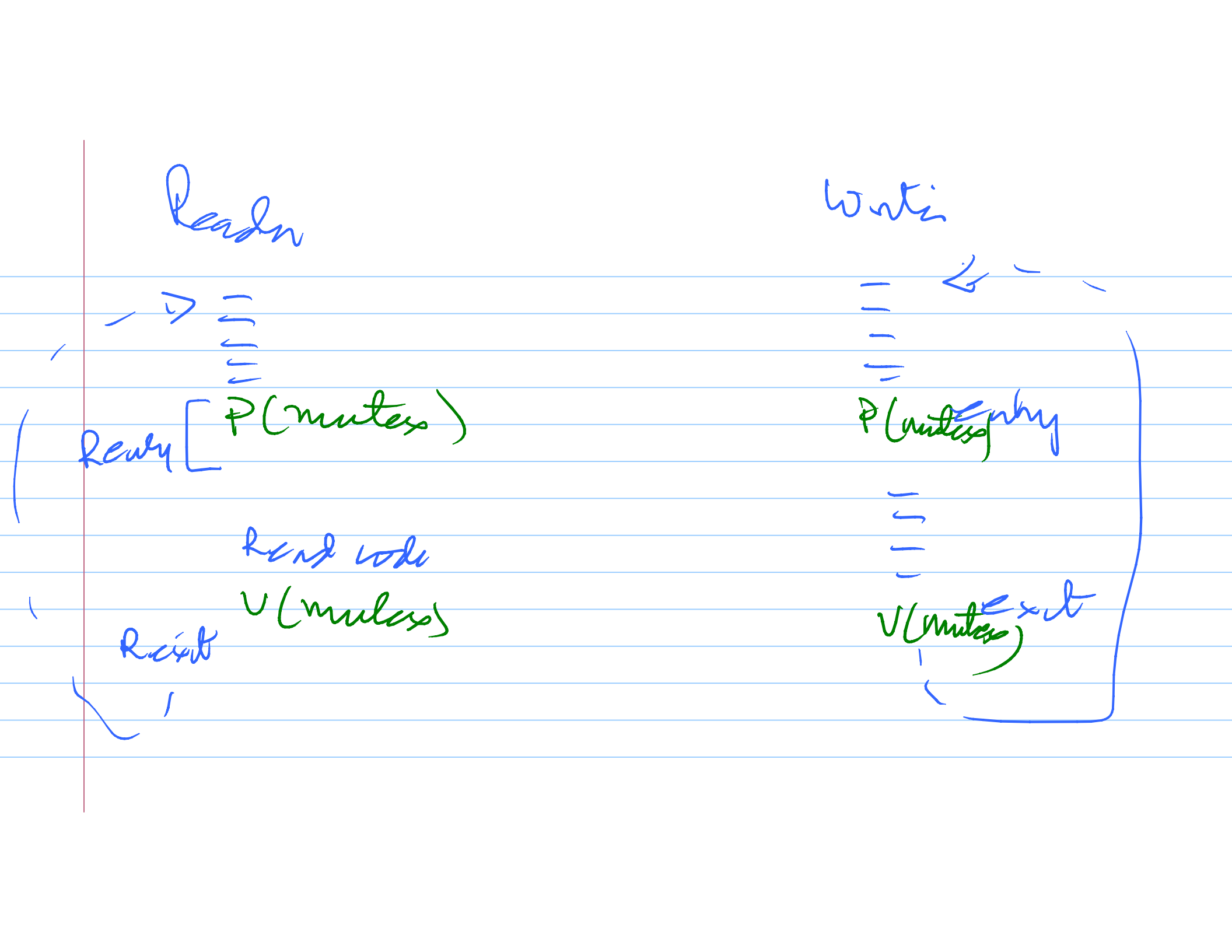
Writer

Same - replace Reader with writer

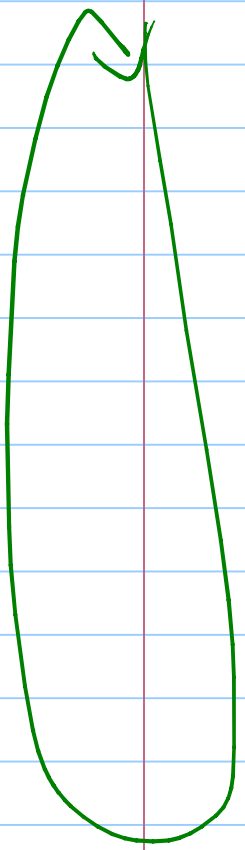
⇒ rule: multiple readers OR one writer

→ multiple threads can be in the read section

→ only 1 writer can be in the write section (and No readers)


Reader
P(mutex)
Read code
V(mutex)
Writer
P(mutex)
V(mutex)

Read



1

P(mutex)

int, init 0

if (rc == 0) P(write);

rc ++;

V(mutex)

=

=

P(mutex)

rc -- ; if (rc == 0) V(write)

V(mutex)

Write

entry

P(write)

V(write) // exit

write = 1

writer starvation

↳ how to fix it?

→ find a better algorithm.

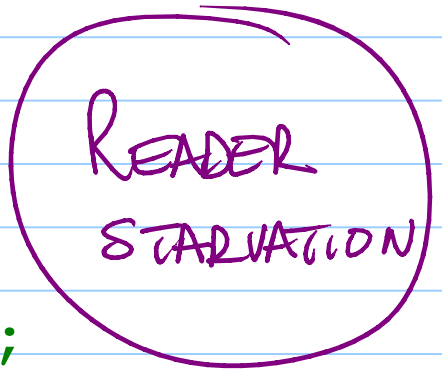
Reader Entry

```
P(rsem);
V(rsem)
P(rmutex);
  rc++
  if rc==1 P(wsem);
V(rmutex);
```

RRR ??

Reader Exit

```
P(rmutex);
  rc--;
  if rc==0 V(wsem);
```



Writer Entry

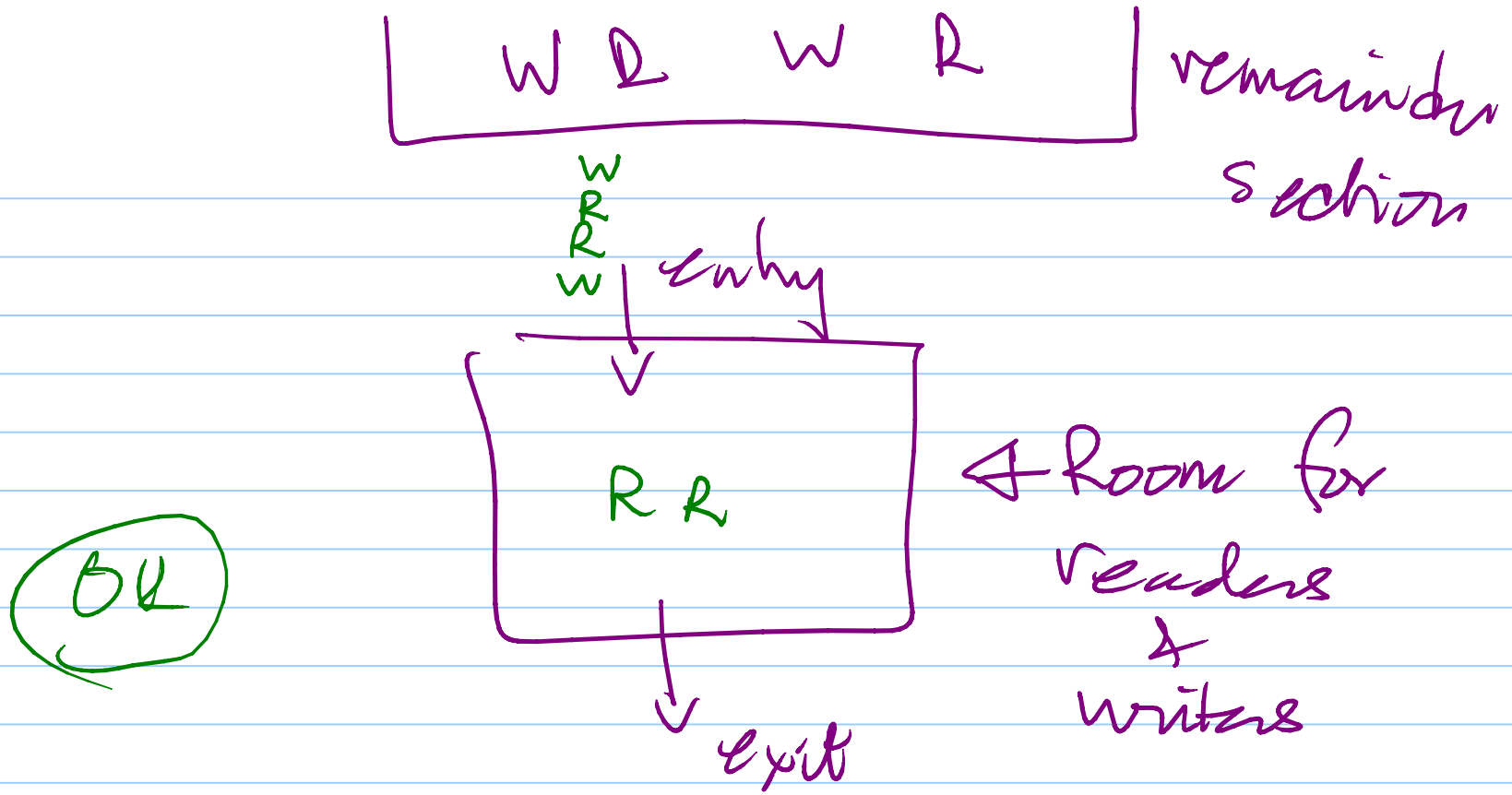
```
P(wmutex);
  wc++;
  if wc==1 P(rsem);
V(wmutex);
P(wsem);
```

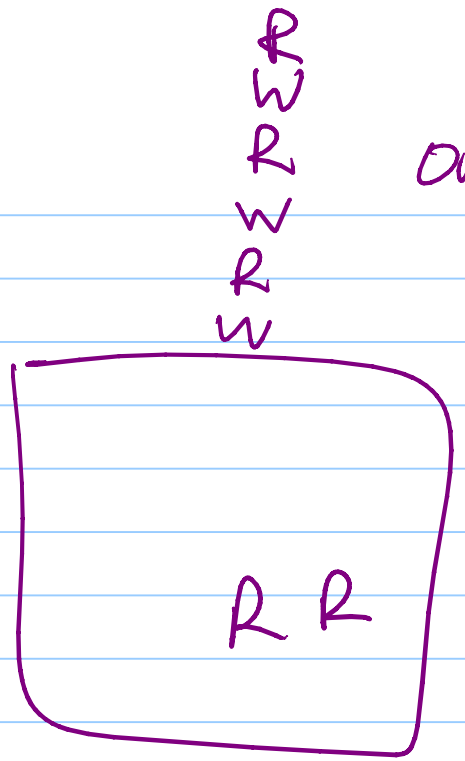
WWW

W

Writer Exit

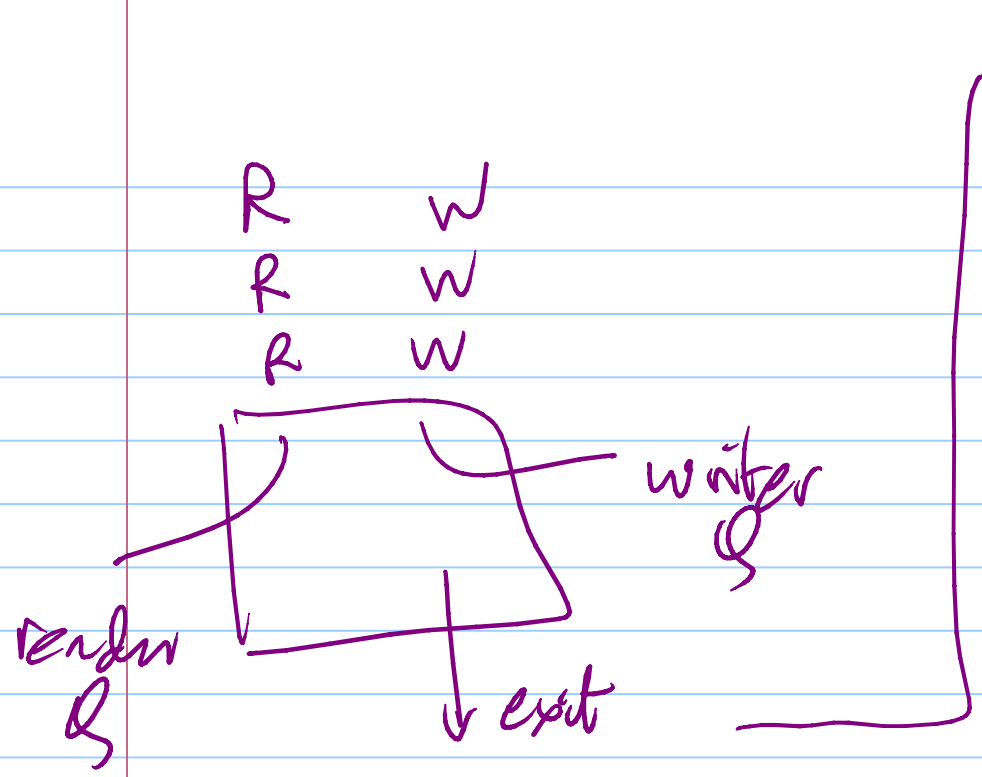
```
V(wsem);
P(wmutex);
   wc--;
   if wc==0 V(rsem);
```


W R W R
remainder section
W
R
R
W
entry
R R
Room for readers & writers
exit
OK



one by one

→ need more concurrency



→ last reader out

→ let one writer in if any

last (& only) writer out

→ let ALL readers in if any

else let 1 writer in if any

R-entry → no writer & no waiting writer → enter

W-entry → no R, no W → enter

Need 2 waiting queues

→ 2 semaphores → rsem, wsem (init 0)

Need 1 critical section → mutex // init 1

rc, wc → reader / writer count

rwc, wwc → reader / writer waiting count

Reader Entry

- get CS

check counters
& decide what to do

for CS

→ do it

Reader Entry

```
P(mutex);
   if (wwc>0) or (wc>0) {
              rwc++;
              V(mutex);
              P(rsem);
              P(mutex);
              rwc--;  };
  rc++;
V(mutex);
```

check w status

open CS

wait

get CS

update counters

Reader Exit

```
P(mutex);
  rc--;
  if (rc=0) && (wwc>0) V(wsem);
V(mutex);
```

Writer Entry

```
P(mutex);
if (rc>0)||(wc>0)||(rwc>0)||(wwc >0){
                wwc++;
                V(mutex);
                P(wsem);
                P(mutex);
                wwc--; }
 wc++;
 V(mutex);
```

Writer Exit

```
  P(mutex);
    wc-- ;
    if (rwc>0) then
       for (i=1; i<=rwc; i++)V(rsem)
```