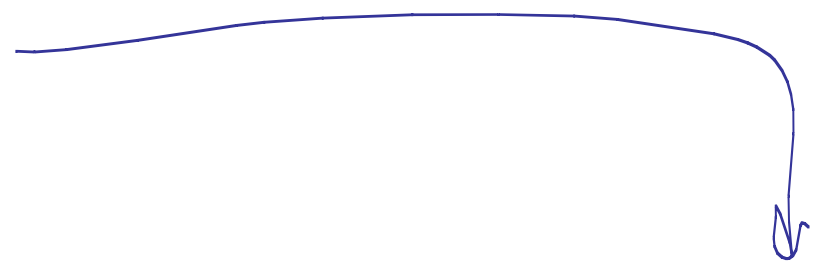
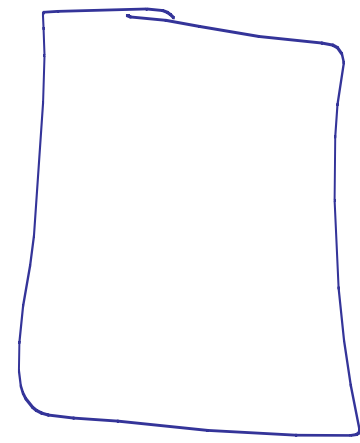
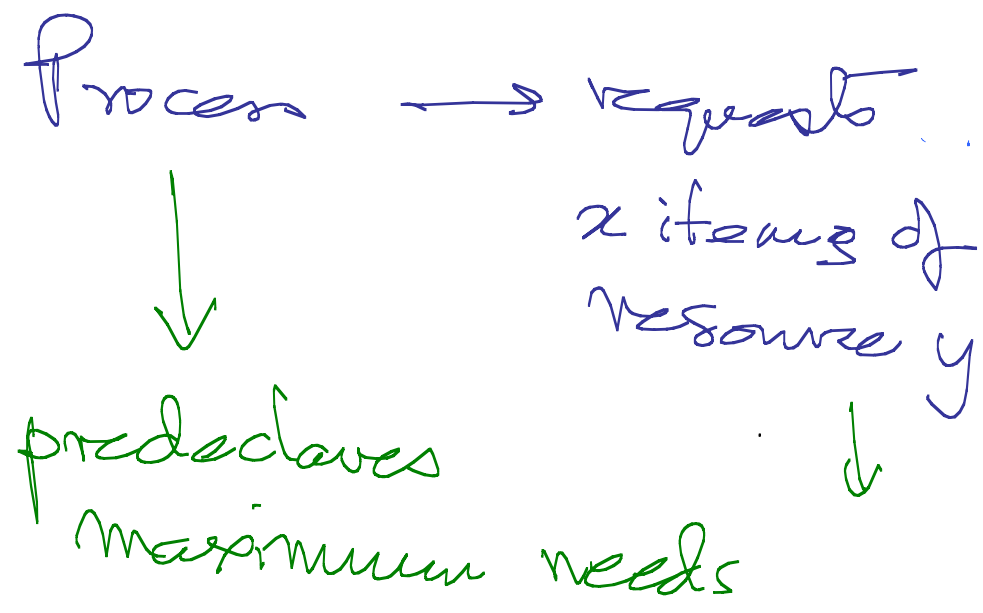


Banker's Algo



Request → enough resources
await

no → hold it

↓ yes

run the algo

unsafe → hold

↓ safe

grant

\$y\$ left in bank

Customer wants \$x\$ & $x < y$

→ pretend to allocate

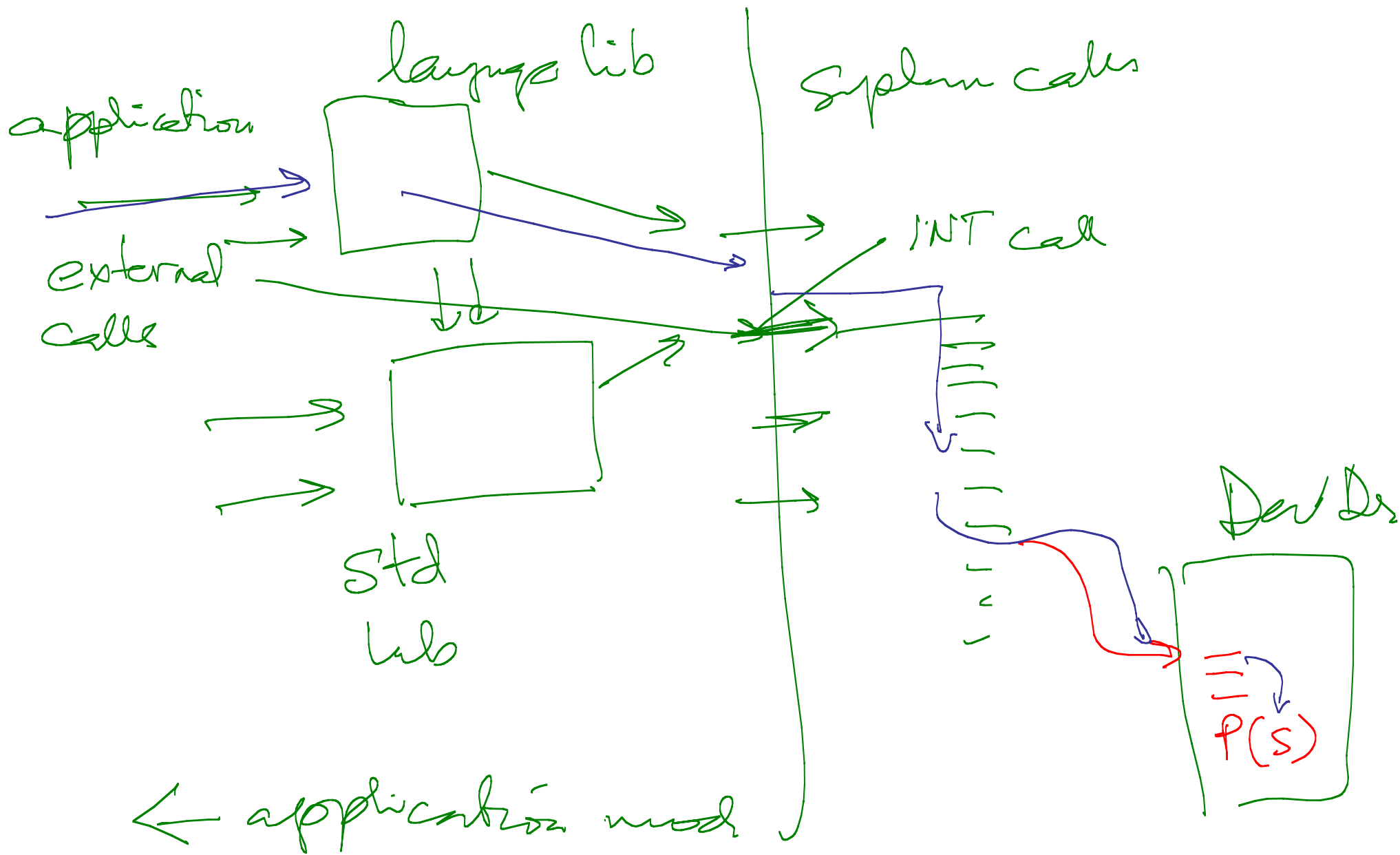
→ for all other customers

→ check if a customer need
can be satisfied

↓

yes → return all resources
held

all Done? → safe



$n \rightarrow$ set to number of processes

barrier()

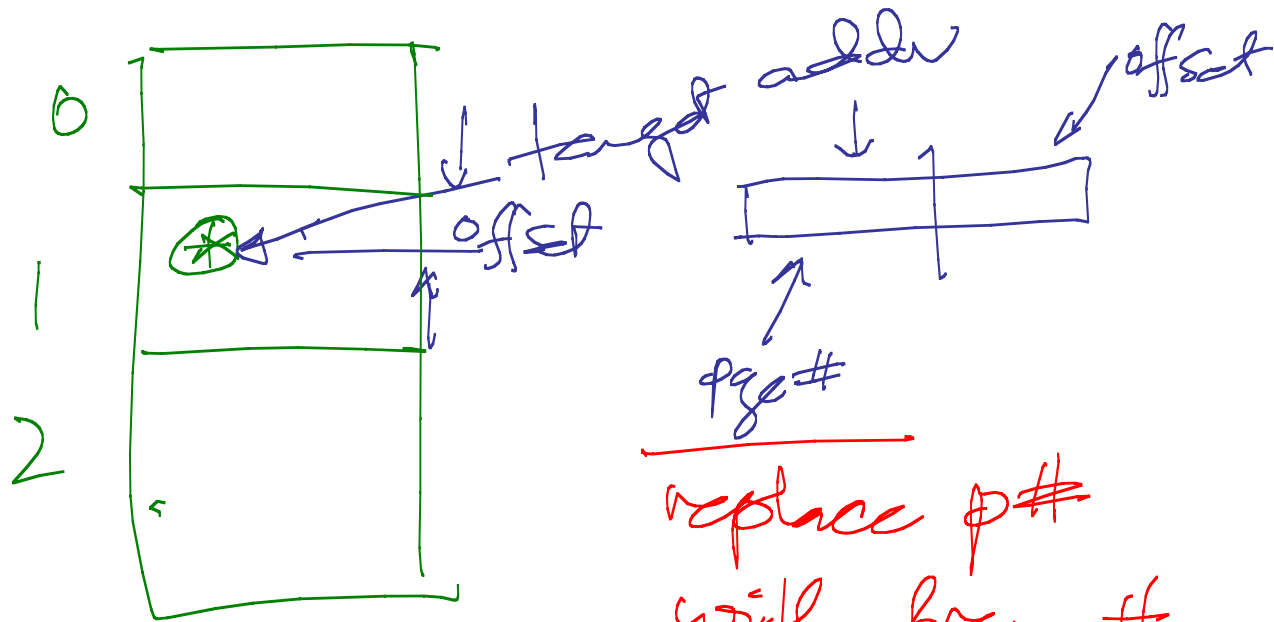
{ P(mutex)

$n--$

if $n > 0$ { V(mutex); P(sleep) }

else { for [n-1 times] V(sleep)
 { V(mutex)

logical \rightarrow physical mem

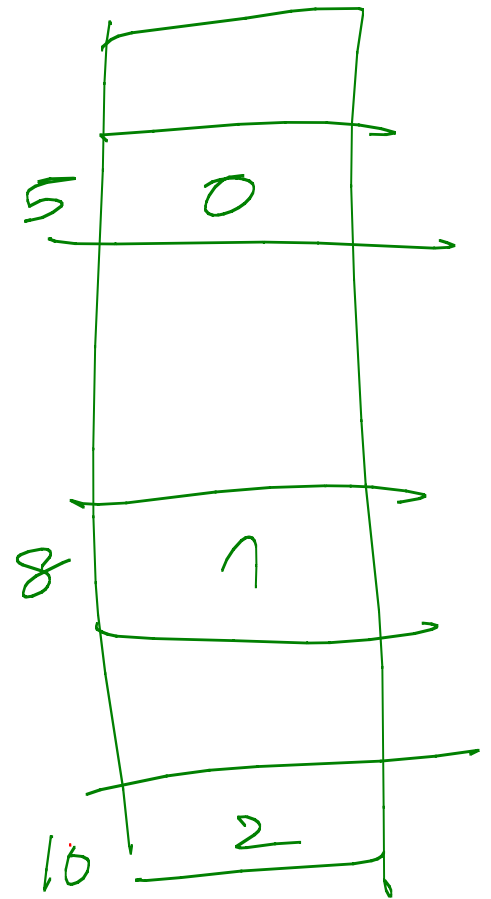


replace p#
with frame#

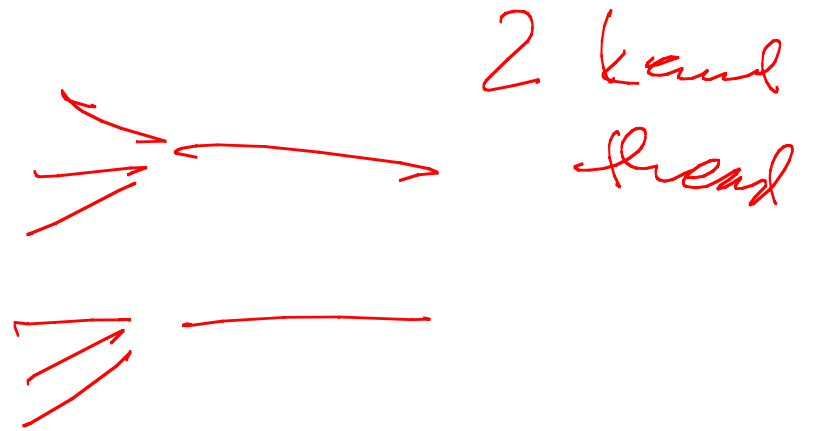
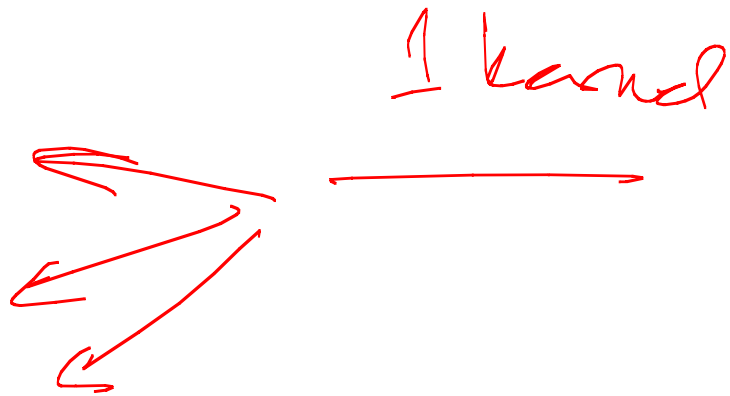
process

Page table

0	5
1	8
2	10



many
user
thr



M resources

n processes

max need of each process

$1 \rightarrow n$

max need of all processes

$m+n$

\rightarrow

m processes
 n resource

$(m+n-1)$ max total

x resources consumed

$m-x$ left

if m resources are allocated
— $(n-1)$ more to go → hence 1 process
needs no more

10 resources

2 processes (6 needed per process)

→ 12 total

A takes 5

deadlock

B " 5