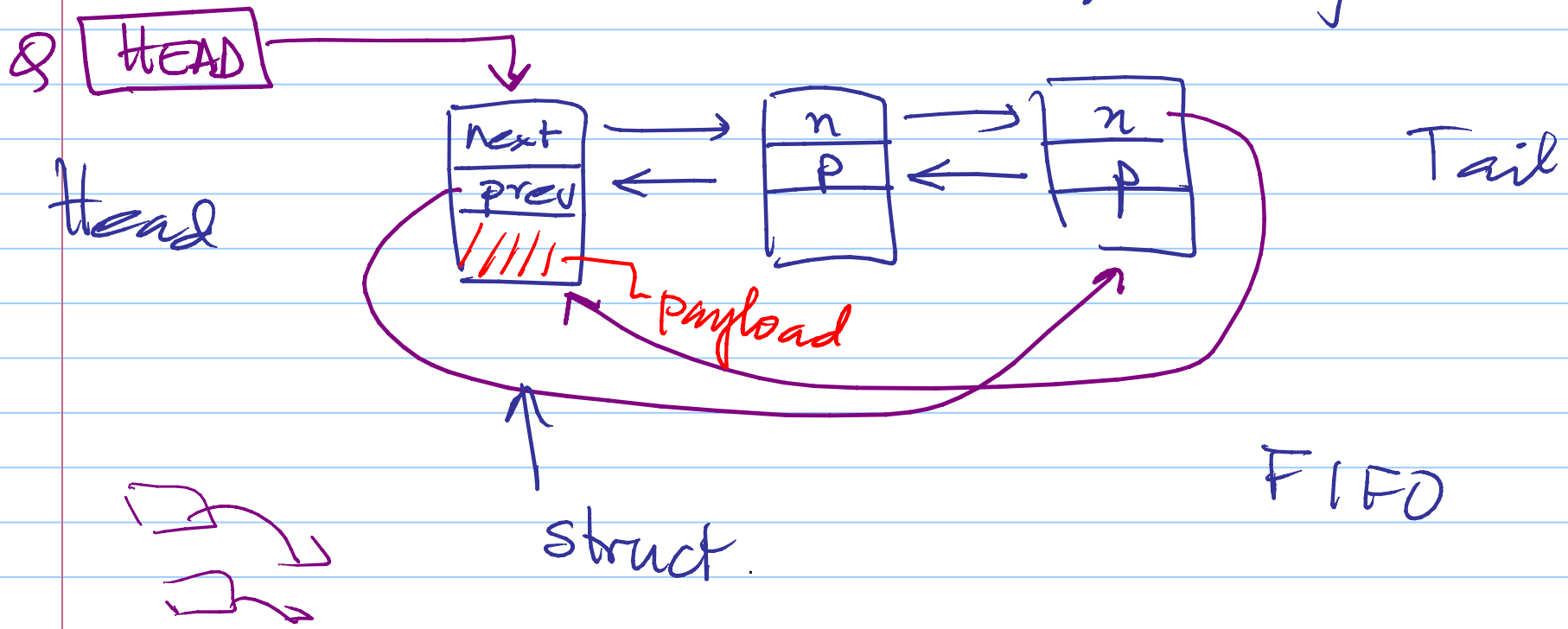
CSE 330: Operating Systems

Class: 10 **Date:** 2/8

Spring 2018

Note Title

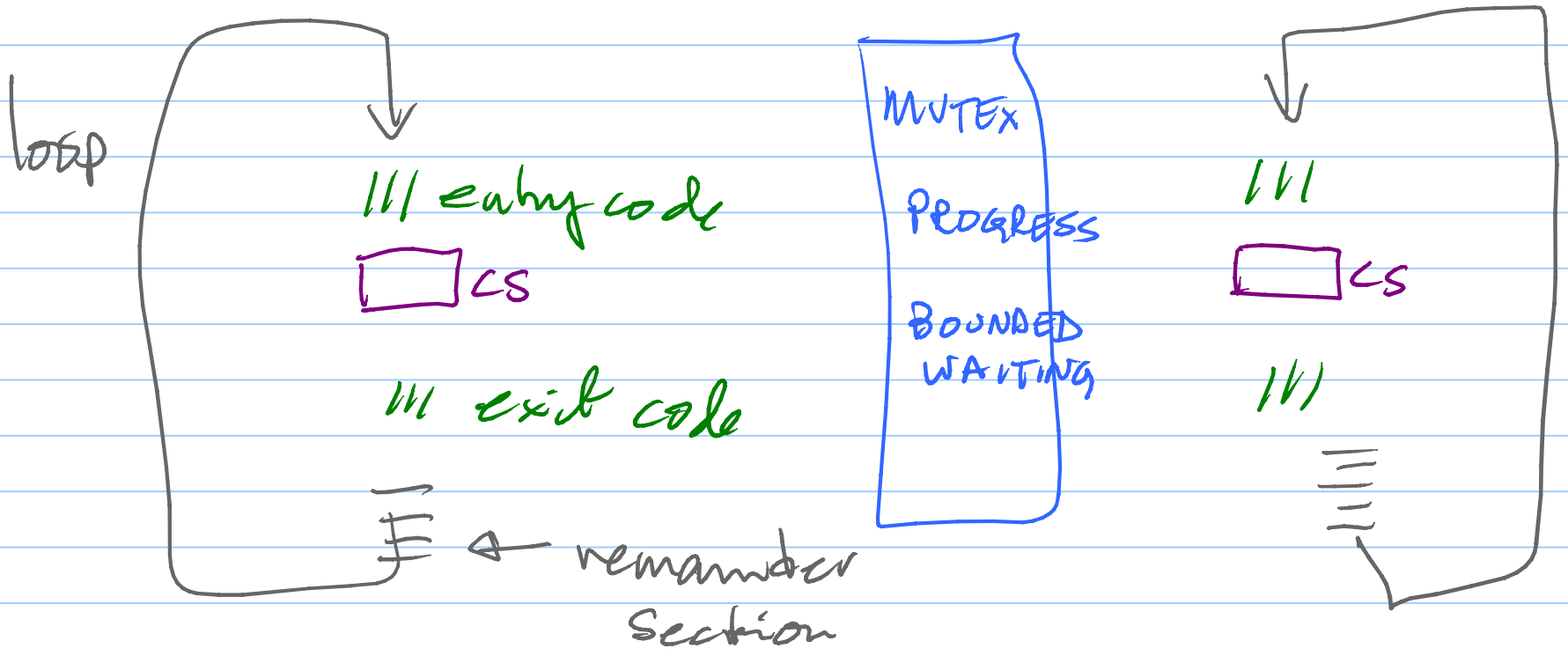
Queues / Doubly linked, circular, no header/dummy

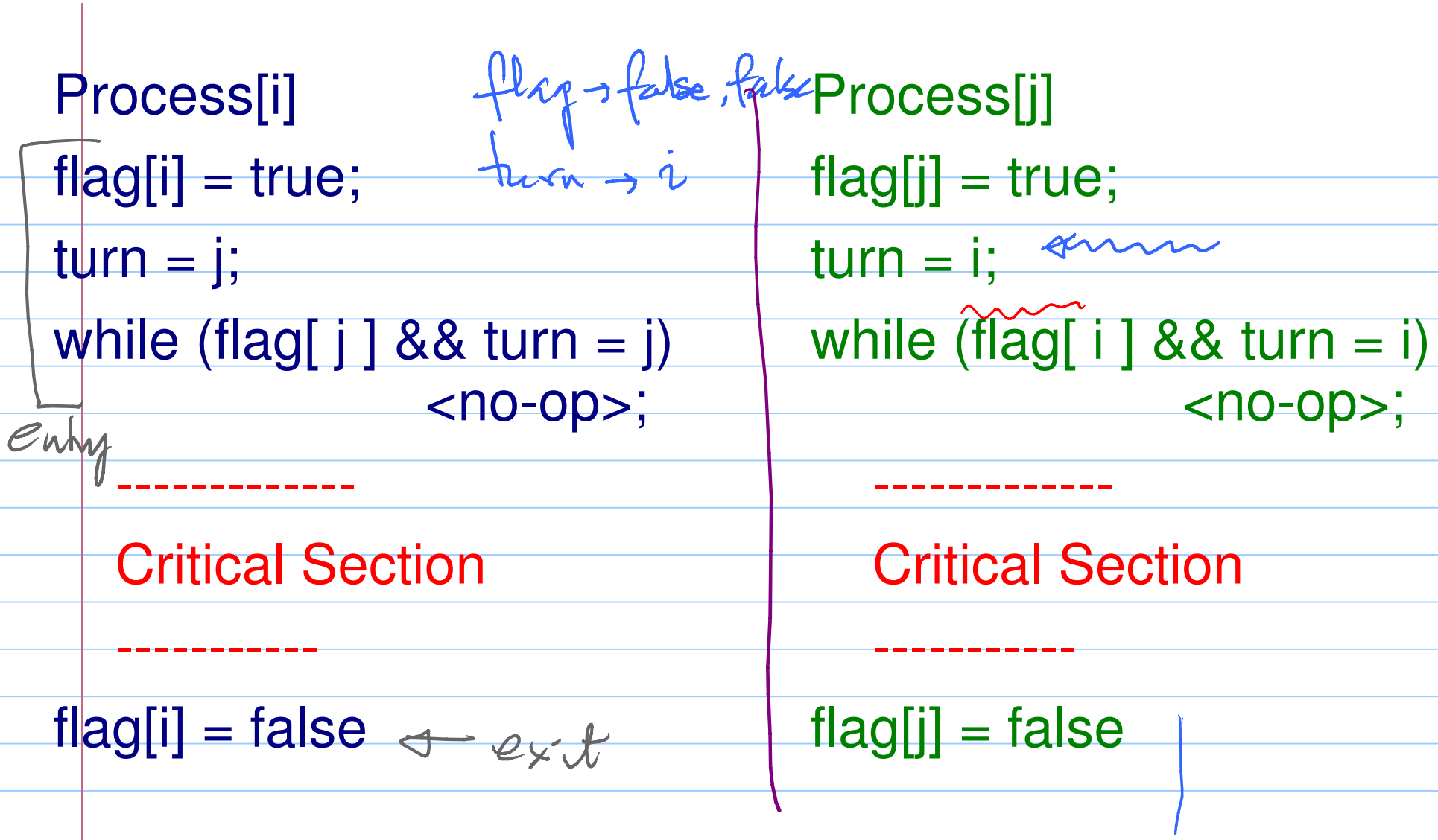


FIFO

Critical section problem

- 2 process, software, Peterson




Process[i]
flag[i] = true;
turn = j;
while (flag[j] && turn = j)
<no-op>;
entry
Critical Section
exit
flag[i] = false
flag → false, false
turn → i
Process[j]
flag[j] = true;
turn = i;
while (flag[i] && turn = i)
<no-op>;
Critical Section
flag[j] = false

more than 2 processes ...

↳ Lamport Bakery Algorithm

Bakery Algorithm → taking tickets at a Bakery

choosing[i] = true;

num[i] = max(num[0],num[1]...num[n-1])+1;

choosing[i] = false;

entry section for process i

for (j = 0; j<n; j++) {

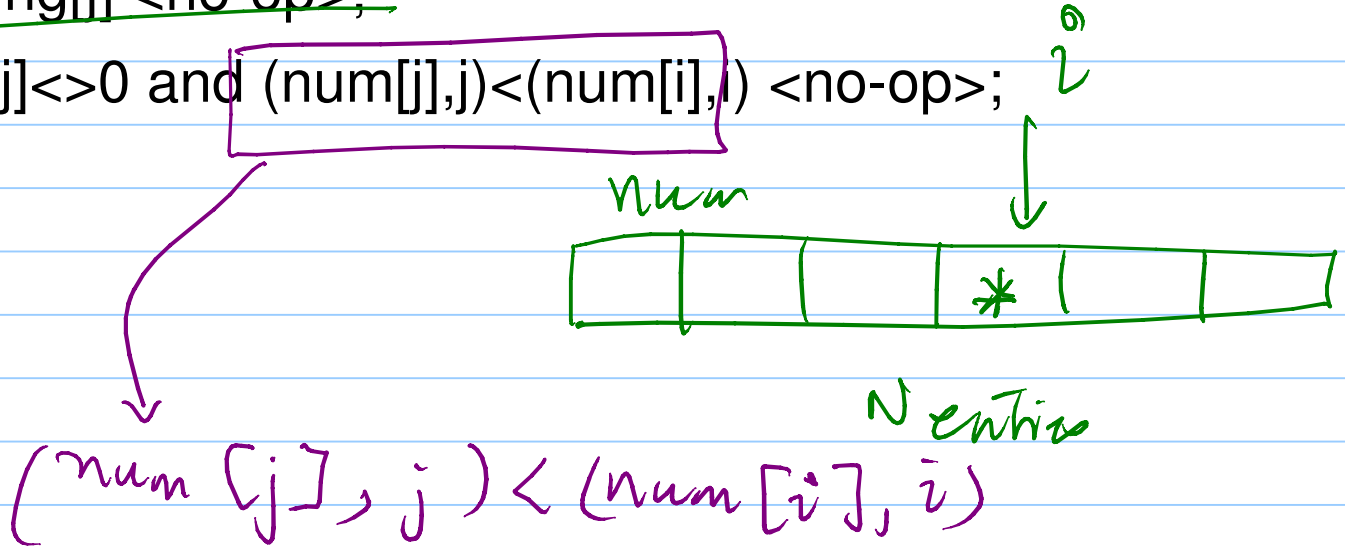
while choosing[j] <no-op>;

while (num[j]<>0 and (num[j],j)<(num[i],i) <no-op>;

};

Critical Section

num[i] = 0;



Peterson — 2 proc

Lamport — N proc

busy waiting

+ complex

Hardware Solutions

Works for single processor CPU

→ Disable all Interrupts

CS

Enable Interrupts

Disabling Interrupts

1. only works for kernel code (priv instruction)
2. the CS must be VERY SMALL
3. Does not work for multicores

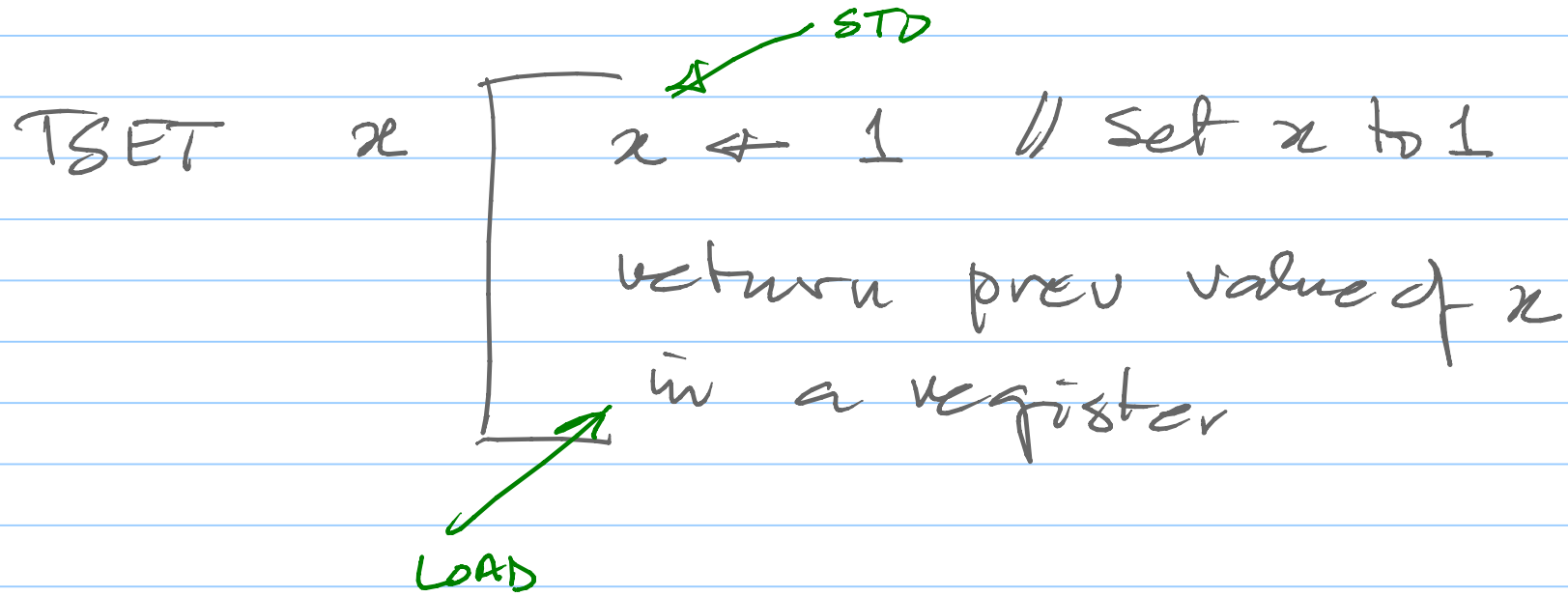
Atomic Instructions

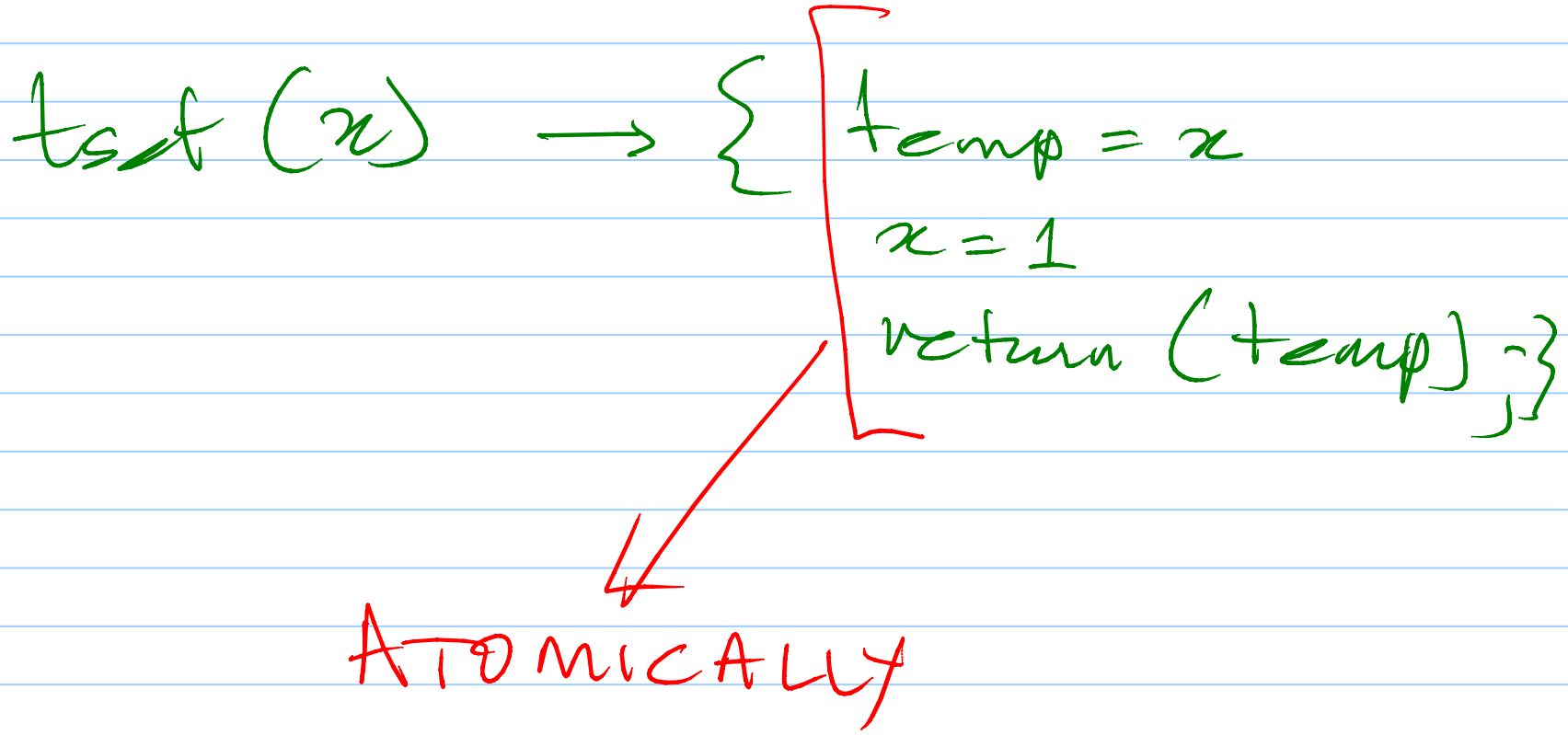
- test & set
- compare & swap
- exchange

test & set → used to teach

exchange → INTEL SYSTEM

[TEST&SET <mem address>




test (x) → {
temp = x
x = 1
return (temp) ;}
ATOMICALLY

lock = 0

while (testAndSet(lock));	while(testAndSet(lock));
CS	CS
lock = 0	lock = 0;