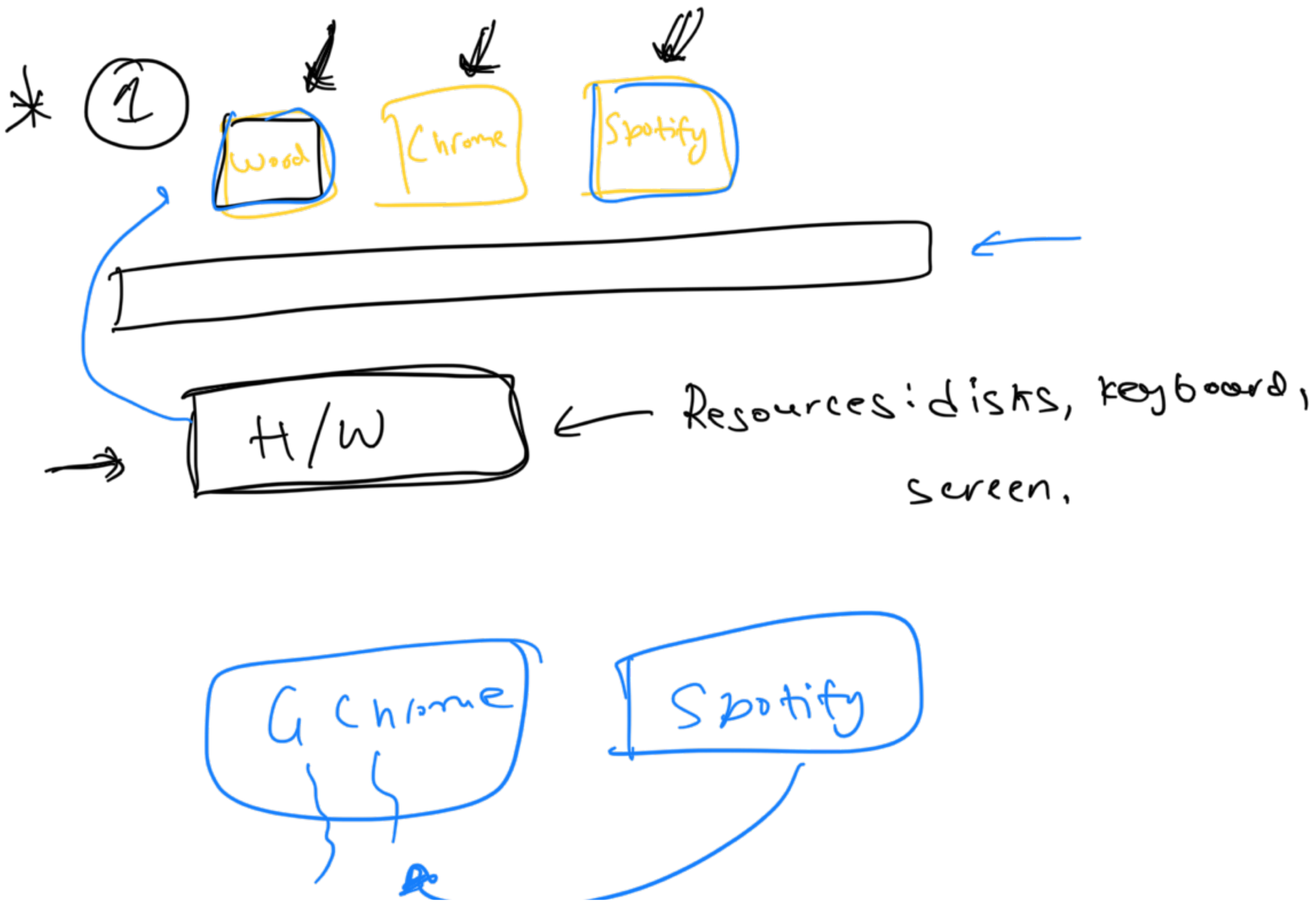
Operating systems - I



a) Resource management

b) APIs / syscalls

c) Process Isolation

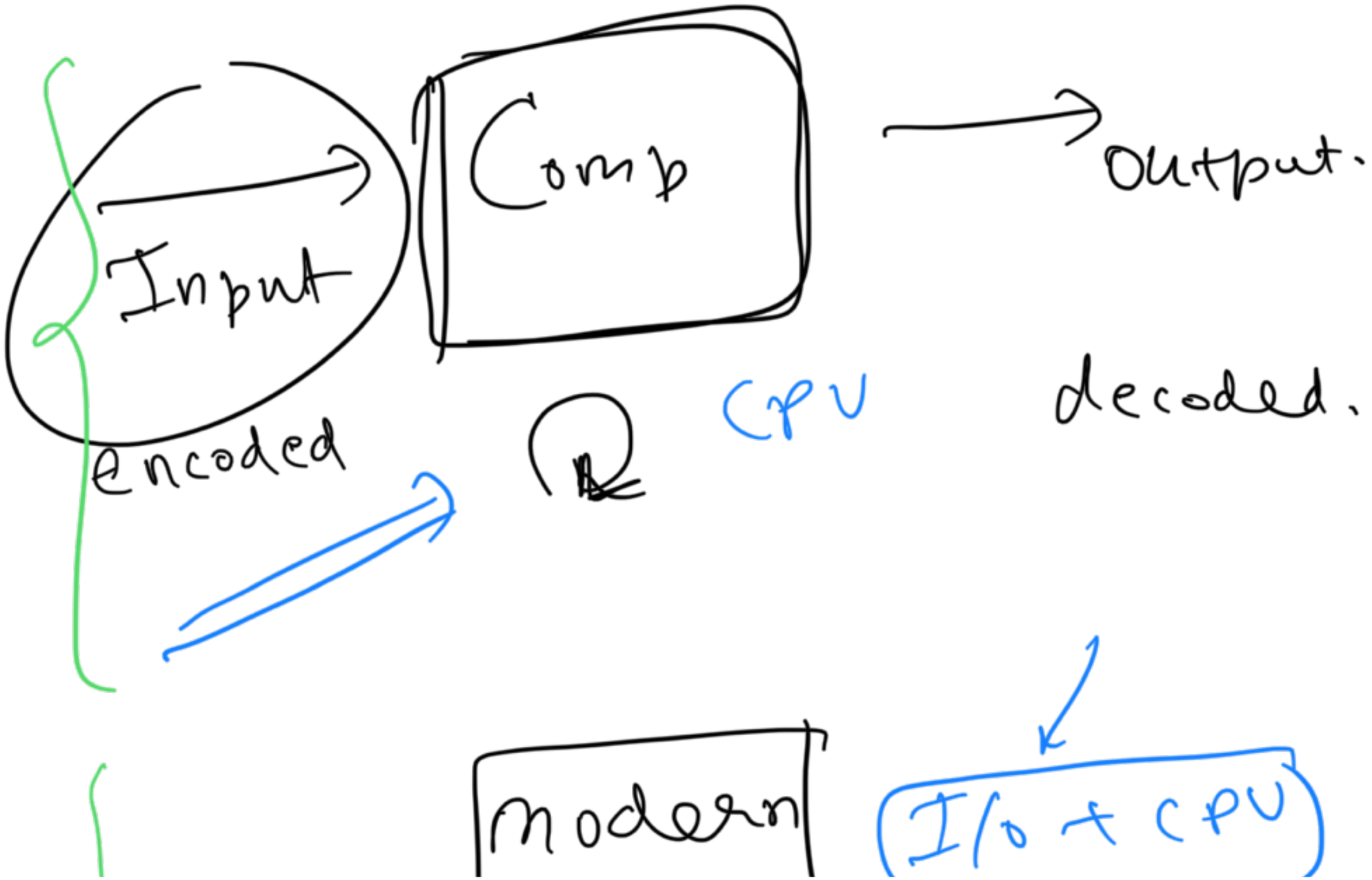
d) Security.

1 ⟶ Concurrency, mproc/mprog, Scheduling Algorithms, Process / Program, PCB

2 ⟶ Memory Isolation.

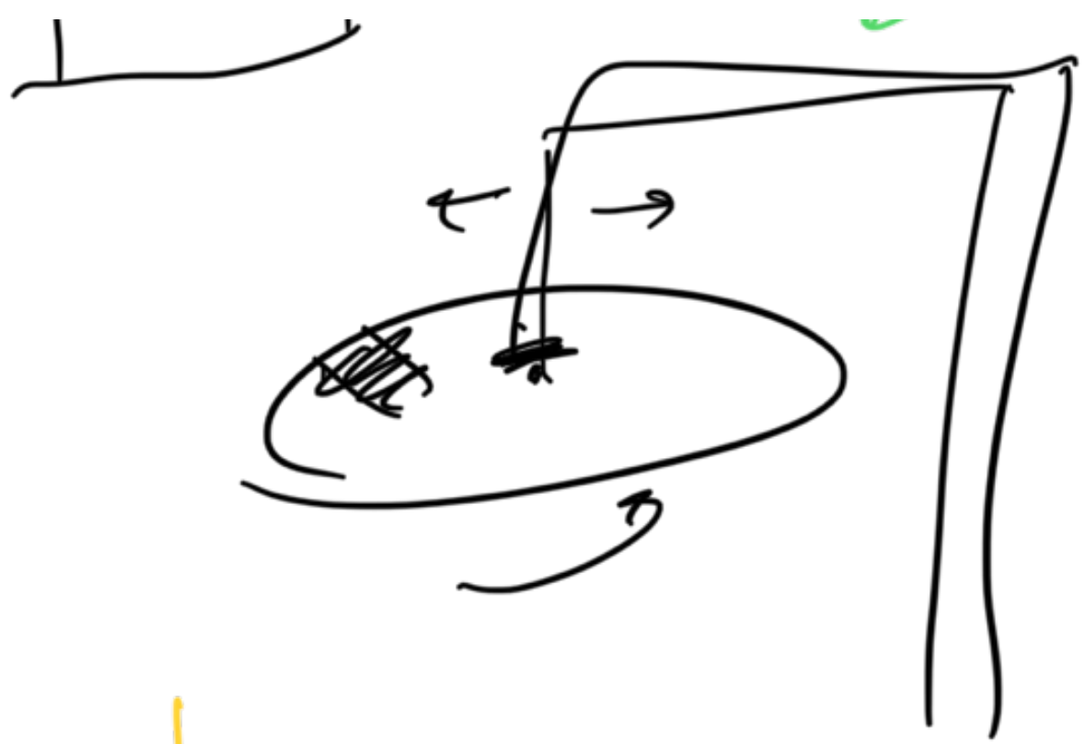
2 →

3 → Threads + Sync,



FS

disk



I/O bound

CPU bound

while(1) {

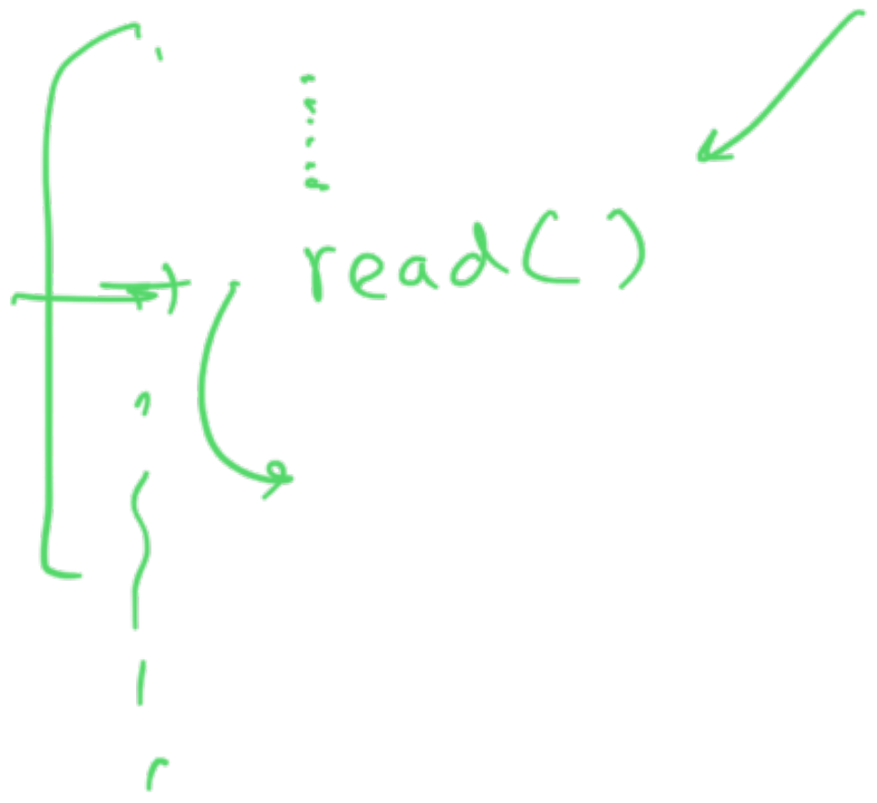
}

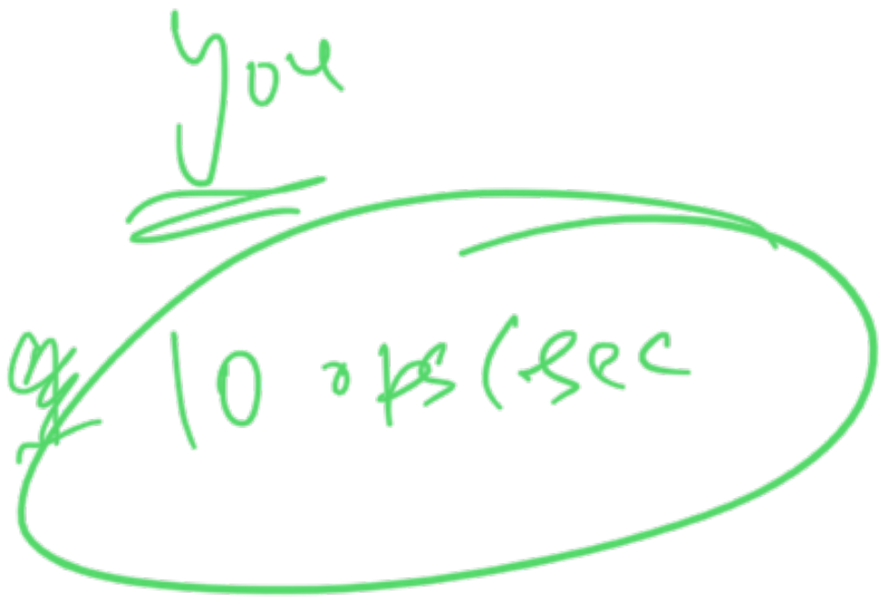
slow

FAST

Disk access

keyboard


read()


You
10 ops/sec


CPU
1 million /sec
ops

Being able to ↓ Schedule multiple programs on the same CPU.

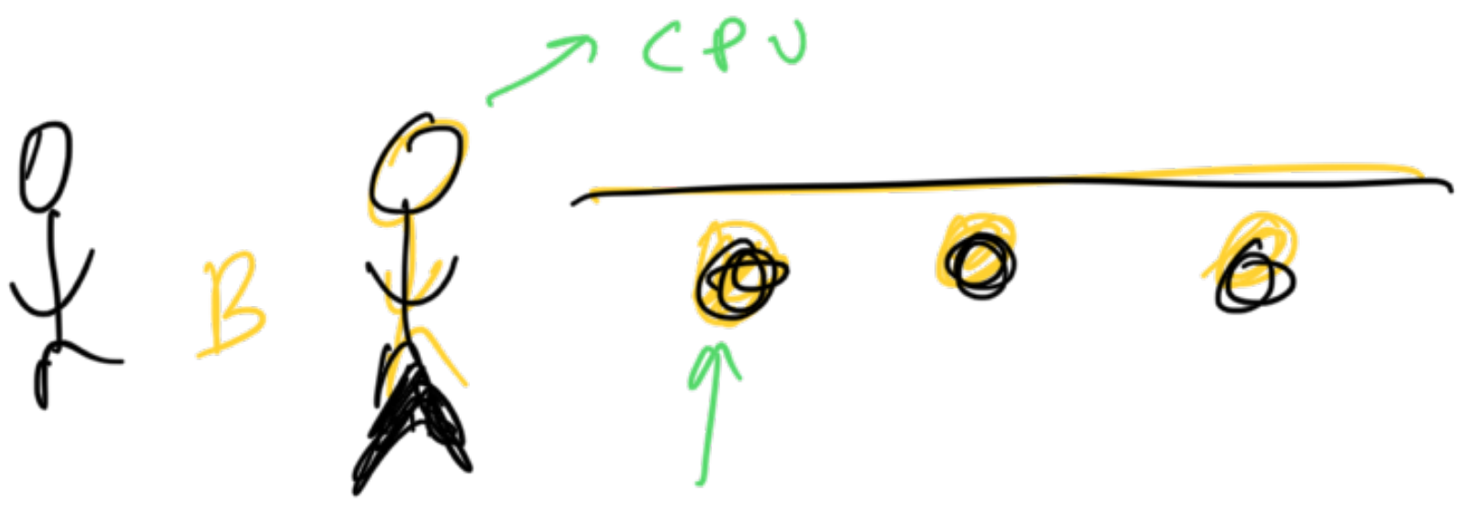
1 CPU / 2 core / 1 process

B —

A

Flash

Speed of light



Process

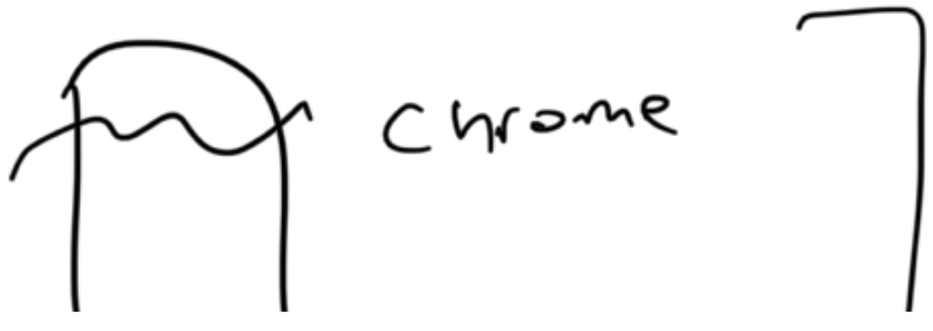
A

OS.

Context switch

Chromp

Spotify



Spot.

Chrome

Song

0.1 seconds

a.C

while(1){

}

Von Neumann Architecture

Program +

Data

a.c

a = 1

a ++

b = 4

compile

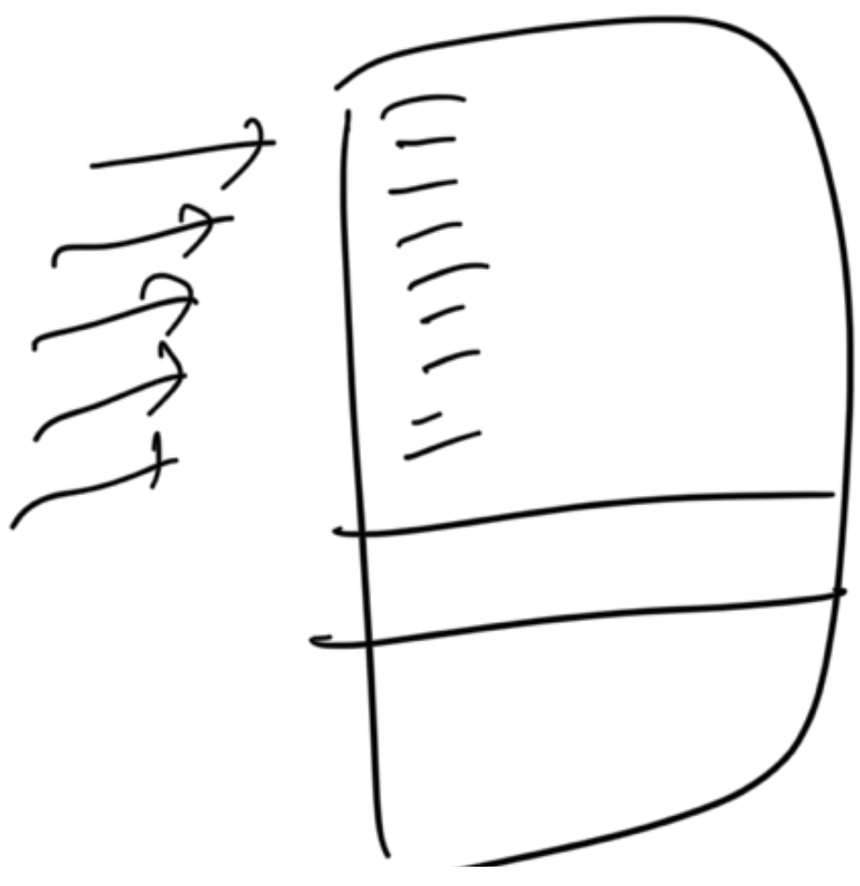
Instn's

b++
malloc

↓

disk

RAM



Assembly

Program → Set of instr^n's

Process → Running instance of program.

How do we schedule multiple Programs concurrently??

way to be able to

PAUSE / RESUME

a process.

P_1

P_2

PC →

registers →

list of open files

Process Control Block (PCB)

Priority

process id

① a++

② b++

→ ③ c++

PC → 2

Process = program + PCB

process

Swaps

Process Flow

Terminated

New

admission (PCB)

Scheduler

Ready

Running

Queue

Time based Interrupt

I/O completed

Waiting

I/O wait

① while(1) {

② read()

→ cin >> x

→ scanf( )

a.c

while(1){

}

(x)

Scheduler

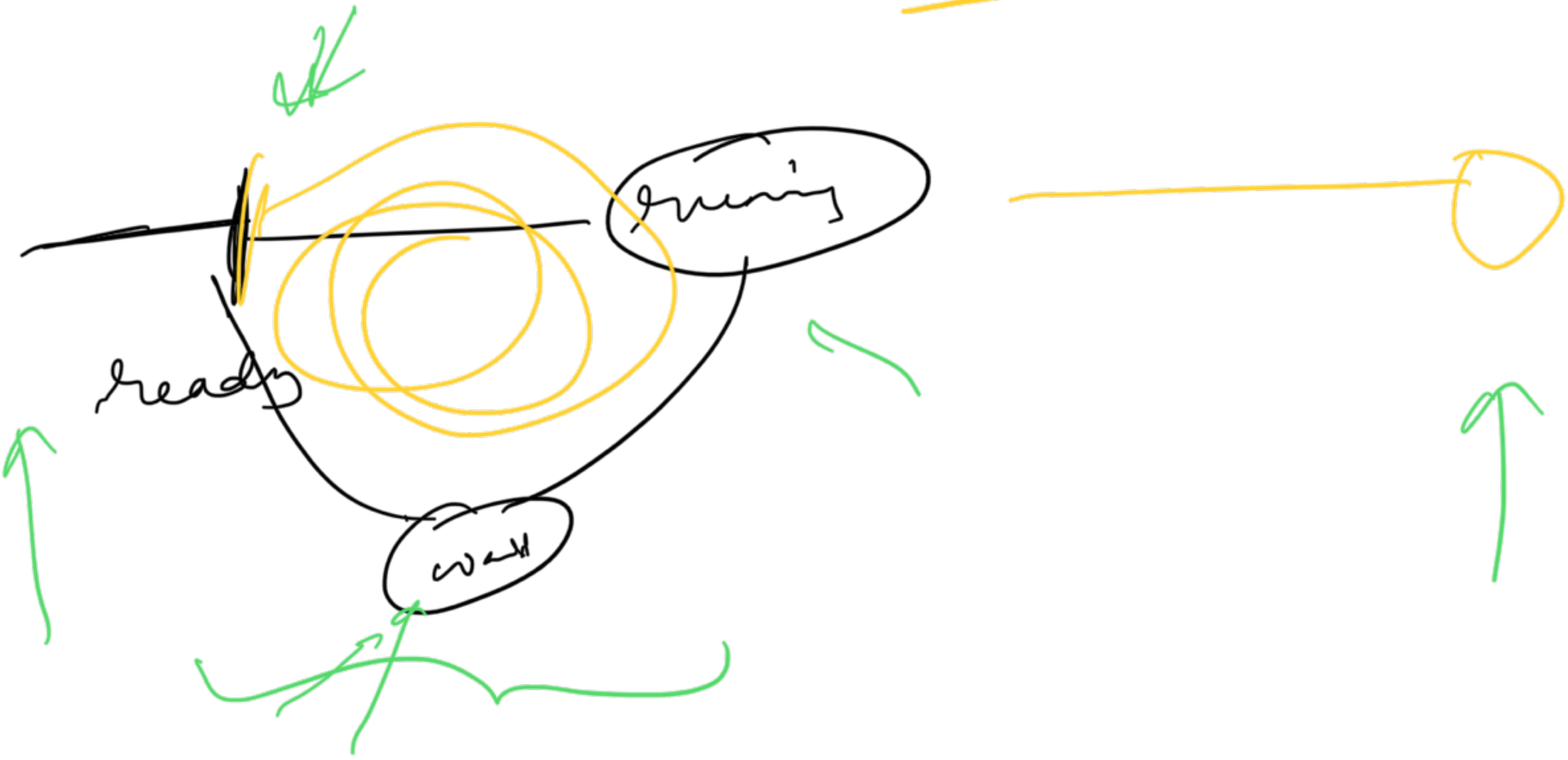
1 → define the performance of a scheduler.

Algo → How well does it work.

To ?.

QoS

metric → Throughput : # of processes completed / sec.



$$CT - AT = WT + \text{Burst Time}$$

amt of time a Process needs on the CPU to complete.

* Preemptive

Forcibly a process is scheduling another

* Non-Preemptive

Not forcing, waiting for process to complete & then

process.

scheduling the next process.

* FCFS

	AT	Burst Time	
P_1	$t=0$	5	
P_2	$t=2$	1	WT: 3
		1	WT: 0

P_3 $t = 6$ α

P_1 $t = 0$ 10000.

P_2 $t = 10$ 1

Shortest Job First

Preemptive

Non Preemptive

P_0 $t=0$ 10000

P_1 $t=0$ 2000

P_0 $t=0$ 10000 − 100

P_1 $t=100$ 1

P_2 $t=100$

P_0 100

P_0

P_1

P_0

100

P_1 $t=0$ Stayed.

P_2 $t=1$ 1

P_3 1

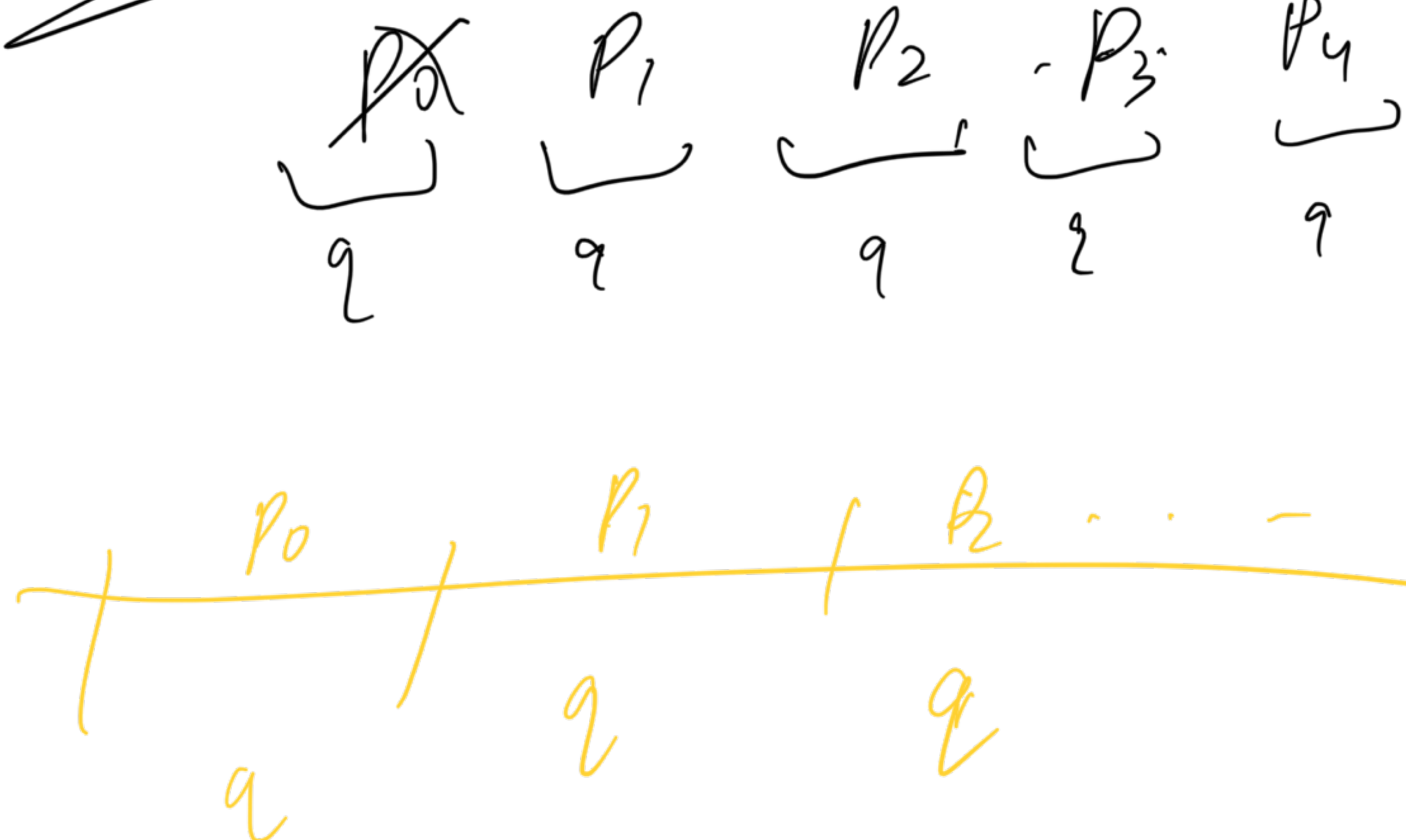
P_2

P10000

Worst time

Halting problem

Round Robin



→ multiple programs, single CPU.

"CONCURRENT"

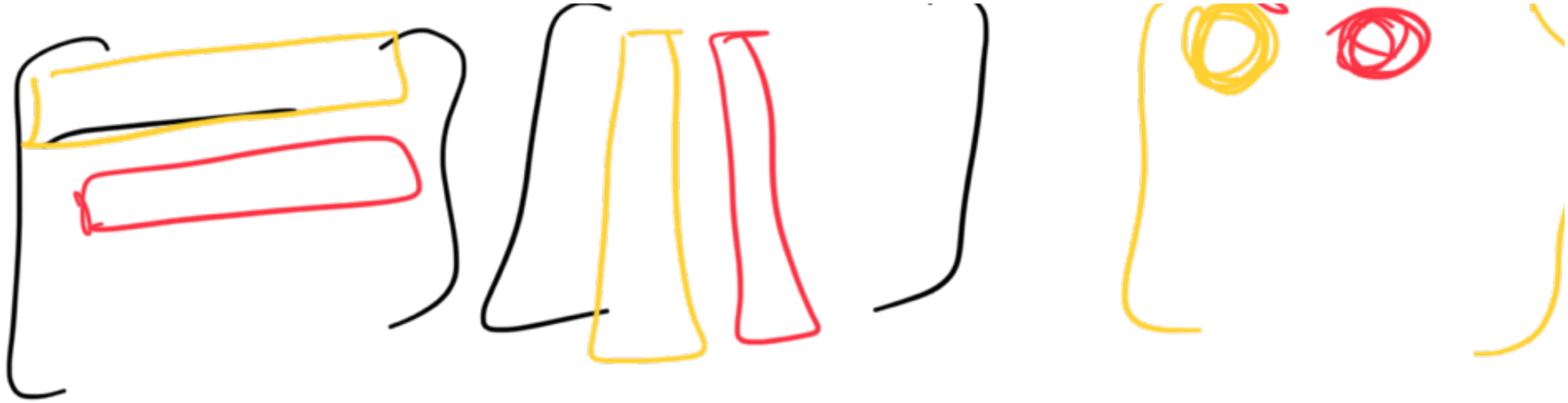
→ multiple CPUs, multiple programs

Parallel → 2 process

break

multiple CPUs.

matrix addition



Conc → multiple programs, 1 CPU

parallel → 1 program, multiple cpu

1 9 ' 4 t

while(){

}

10000 google.com

No. of cores

n cores

X4

Non preemptive SJF

AT

BT

P_0 $t = 0$ 100

$t = 1$

P_1 200

$t = 4$

P_2

NP SJF

	P_0	P_2	P_1
	100	—	—
	P_0	4 ✓	200 ✓

NP P0 100 — P1 — 12

200 4

FCFS: